Advanced Cyber–Threat Intelligence, Detection, and Mitigation Platform for a Trusted Internet of Things

Grant Agreement: 786698

D5.1 State-of-the-art on proactive technologies

Work Package 5: Key proactive technologies and cyber-threat intelligence

Document Dissemination Level

PU	Public	X
CO	Confidential, only for members of the Consortium (including the Commission Services)	

Document Due Date: 28/02/2019
Document Submission Date: 28/02/2019

Co-funded by the Horizon 2020 Framework Programme of the European Union

Document information

Deliverable number:	D5.1
Deliverable title:	State-of-the-art on proactive technologies
Deliverable version:	1.0
Work Package number:	WP5
Work Package title:	Key proactive technologies and cyber-threat intelligence
Due Date of delivery:	28/02/2019
Actual date of delivery:	28/02/2019
Dissemination level:	PU
Editor(s):	Nicholas Kolokotronis (UOP)
Contributor(s):	Nicholas Kolokotronis, Costas Vassilakis, Spiros Skiadopoulos, Christos Tryfonopoulos, Nicholas Kalouptsidis, Konstantinos Ntemos, Christos-Minas Mathas, Athanasios Chantzios, Konstantinos-Panagiotis Grammatikakis, Paris Koloveas (UOP) Dimitrios Kavallieros, Vasiliki-Georgia Bilali, George Kokkinis (KEMEA) Stavros Shiaeles, Bogdan Ghita, Julian Ludlow, Salam Ketab, Muhammad Ali, Abdulrahman Alruban (CSCAN)
Reviewer(s):	Olga Gkotsopoulou (VUB) Gohar Sargsyan, Raymond Binnendijk (CGI)
Project name:	Advanced Cyber-Threat Intelligence, Detection, and Mitigation Platform for a Trusted Internet of Things
Project Acronym	Cyber-Trust
Project starting date:	01/05/2018
Project duration:	36 months
Rights:	Cyber-Trust Consortium

Version history

Version	Date	Beneficiary	Description
0.05	21/12/2018	UOP	Tentative ToC proposed
0.10	28/12/2018	UOP	Deliverable's ToC finalized
0.20	28/01/2019	UOP	First draft of chapter 4 on game theoretic security
0.30	02/02/2019	UOP	First draft of chapter 2 sections regarding crawling and sharing, of cyber-threat intelligence
0.40	07/02/2019	CSCAN	Contributions made to chapter 2 on data sources, chapter 3 on status-based aspects, and chapter 4 on cyber-defense objectives
0.50	12/02/2019	UOP	First draft of chapter 3 introductory sections and the underpinnings of trust/risk, as well as contributions made to chapter 2 on privacy-preserving aspects
0.60	15/02/2019	KEMEA	Contributions made to chapter 2 on CTI platforms, tools, and exchange formats
0.70	22/02/2019	UOP	First draft of chapter 3 is complete along with the executive summary, introduction, and conclusions
0.80	25/02/2019	UOP	Document has been reviewed (any abbreviations have been added) and properly formatted (incl. the references); has been sent to the reviewers
0.90	27/02/2019	VUB, CGI	Review comments received
1.00	28/02/2019	UOP	Accommodation of the reviewers' comments and submission of final version

Acronyms

ACRONYM	EXPLANATION
6BR	6LoWPAN Border Router
6LoWPAN	IPv6 over Low-power Wireless Personal Area Networks
AAA	Authentication, Authorization and Accounting
ACO	Ant Colony Optimization
AG	Attack Graph
aLADS	ALPHATECH Lightweight Autonomic Defense System
ANFIS	Adaptive Neuro-Fuzzy Inference System
APIs	Application Programming Interfaces
ATM	Author Topic Model
BAG	Bayesian Attack Graph
BFS	Breadth First Search
BMA	Bad-mouthing attacks
BoW	Bag-of-Words
BR	Backbone Router
BS	Base Station
BSA	Ballot-Stuffing Attacks
CAPTCHA	Completely Automated Public Turing test to tell Computers and Humans Apart
CERT	Cyber-Emergency Readiness Teams
CH	Cluster Heads
CIB-MPE	Common Information Based Markov Perfect Equilibrium
CoI	Community of interest
CPE	Common Platform Enumeration
CRF	Conditional Random Field
cRPL	classical-RPL
CTI	Cyber-Threat Intelligence
CTITC	Cyber-Threat Intelligence Technical Committee
CTP	Cloud Trust Protocol
CTV	Consolidated Trust Value
CVE	Common Vulnerabilities and Exposures
CVRF	Common Vulnerabilities Reporting Framework
CVSS	Common vulnerability scoring system
CybOX	Cyber-Observable eXpression

D2D	Device to Device
DaaS	Data-as-a-Service
DDoS	Distributed DoS
Dec-POMDP	Decentralized POMDP
DEs	Decision Entities
DFS	Depth first search
DG	Dependency Graph
DHT	Distributed Hash Table
DoS	Denial of Service
DTM	Dynamic Topic Model
EC	Energy Consumption
EEA	European Economic Area
EPFR	End-to-end Packet Forwarding Ratio
ERNT	Extended RPL Node Trustworthiness
ETL	Extraction–Transformation–Loading
EU	European Union
eVDB	enriched Vulnerability DataBase
EX	Experience
FFQ	Friend or Foe Q-Learning
GANA	Generic Autonomic Network Architecture
GDPR	General Data Protection Regulation
GEOINT	Geospatial Intelligence
GrSM	Graphical Security Model
GT	Game Theory
HBA's	Host-Bus Adapters
HiWE	Hidden Web Exposer
HTML	Hypertext Markup Language
HUMINT	Human I*ntelligence*
I2P	Invisible Internet Project
IDS	Intrusion Detection System
iIRS	intelligent Intrusion Response System
IMINT	Imagery Intelligence
INT	Intelligence
IoC	Indicators of Compromise
IODEF	Incident Object Description Exchange Format

IoT	Internet of Things
IR	Information Retrieval
JNI	Java Native Interface
KN	Knowledge
LCPD	Local conditional probability distribution
LDA	Latent Dirichlet Allocation
LLN	Low-power and Lossy Networks
LRA	Local Recoding Anonymization
MAD	Mutually Assured Destruction
MAPE-K	Monitor, Analyze, Plan, Execute, Knowledge
MASINT	Measurement and Signature Intelligence
MCUs	Microcontroller Units
MDP	Markov Decision Process
MEs	Managed Entities
MISP	*Malware Information Sharing Platform*
MPE	Markov Perfect Equilibrium
MUD	Manufacturer Usage Description
NDK	Native Development Kit
NE	Nash equilibrium
NER	Named Entity Recognition
NIS	Network and Information Security
NLP	Natural Language Processing
NN	Nearest-Neighbour
NVD	National Vulnerability Database
OEM	Original Equipment Manufacturer
OOA	On-off attacks
OpenIoC	Open Indicators of Compromise
OS	Operating System
OSA	Opportunistic Service Attacks
OSINT	Open Source Intelligence
OVAL	Open Vulnerability and Assessment Language
P2P	Peer-to-Peer
PBE	Perfect Bayesian Equilibria
PDR	Package Delivery Ratio
PESTLE	Political, Economic, Sociological, Technological, Legal, Environmental

PI	Policy Iteration
POIs	Places of Interest
POMDP	Partially Observed Markov Decision Process
POS	Part-of-Speech
POSG	Partially Observable Stochastic Game
PTO	Pre-Trusted Objects
Pub/Sub	Publish-Subscribe
PWLC	Piecewise Linear and Convex
QL	Q-Learning
QoS	Quality of Service
RA	Remote attestation
RC	Recommendation
RE	Relation Extraction
RFID	Radio-Frequency Identification
RL	Reinforcement Learning
RPL	Routing Protocol for LLNs
RRE	Response and Recovery Engine
rRPL	resilient-RPL
SaaS	Software as a Service
SCT	Stochastic Control Theory
SCUBA	Secure Code Update By Attestation
SG	Stochastic Game
SIGINT	Signal Intelligence
SIoT	Social IoT
SPA	Self-Promotion Attacks
SPBE	Structured Bayesian Perfect Equilibrium
STIX	Structured Threat Information eXpression
SVM	Support Vector Machines
SVO	Subject-Verb-Object
SWATT	SoftWare-based ATTestation
SWOT	Strength, Weakness, Opportunity, Threats
T-IDS	Trust-based IDS
TA	Trust Agent
TAXII	Trusted Automated eXchange of Intelligence Information
TCG	Trusted Computing Group

TDS	Top-Down Specialization
TM	Trust Manager
TMS	Trust management system
TOR	The Onion Router
TPM	Trusted Platform Module
tRPL	trust RPL
TTP	Techniques and Procedures
UEFI	*Unified Extensible Firmware Interface*
USMC	User-guided Social Media Crawling
VI	Value Iteration
VIP	Verification of Interaction Proof
VPA	Vertical Partitioning Anonymization

Table of contents

List of figures

List of tables

Executive summary

This report aims at providing the state-of-the-art in the areas covered by work package WP5 of Cyber-Trust project, which include *cyber-threat intelligence* (CTI) gathering and sharing techniques, trust establishment and risk assessment, as well as game-theoretic security.

Such methods and tools are responsible for the efficient collection of information that is available on various sources regarding emerging cyber-threats or possibly unknown vulnerabilities, their ranking in terms of various related criteria (e.g. popularity, impact), and their dissemination to the relevant authorities so as to increase awareness. More precisely, this report describes:

- How can sources with valuable cyber-security information be identified and efficiently crawled so as to retrieve relevant (time-varying) content on vulnerabilities, exploits, products, etc.?
- How can site-specific content be extracted (irrelevant be filtered out) from the retrieved data, like the *common vulnerabilities and exposures* (CVE), *common platform enumeration* (CPE), exploits, etc.?
- How can the gathered CTI be efficiently stored, indexed, etc. and disseminated amongst the cyber-security experts, to rate the quality of the CTI and increase awareness?

In addition, these methods and tools provide the means for designing and developing a reputation or *trust management system* (TMS). Such systems quantify the trust placed on a (possibly unknown) IoT device's hardware/software, for reliably performing the operations intended, and the risk that any communication with that device poses. The report reviews contemporary approaches for developing an advanced reputation system, which is robust to common attacks in this area, allowing a more granular characterization of an IoT device's state. In particular, the report describes how to incorporate into the TMS information about the following questions concerning a device's security state, behavior, and associated risk:

- Have critical operating system files or firmware been tampered with?
- Have the latest software patches been installed? (if this is possible to perform)
- Is the IoT device exposed to any known vulnerabilities and exploits?
- Is the network traffic (volume, type, etc.) generated by the device typical?

Finally, the review of the game-theoretic security that is given in this report, presents the means by which the information available from the rest of the proactive tools could be leveraged to allow devices responding to cyber-attacks in an intelligent and autonomous way. Given a system configuration, such games model the players (attackers and defenders) and their interactions, i.e. what each player knows about the system, what actions can be performed, what reward is obtained, etc. Key aspects described in this report include:

- How can game-theoretic frameworks for cyber-security be modeled accurately, what complexity and efficiency issues arise in each case, and how can these be effectively tackled?
- By confining to game-theoretic models for intrusion response systems, how could these be utilized with graphical security models, i.e. attack graphs, so as to enhance defenders' capabilities against sophisticated multi-stage cyber-attacks?
- What methods and algorithms can be used for solving the formulated problems, in a centralized or decentralized manner, to yield the optimal defender's response and minimize attacker's success?

The deliverable provides a thorough analysis of tools and methods for the areas covered by Cyber-Trust's key proactive technologies; it is therefore quite technical by nature. We believe that readers with a technical background will find the presentation quite comprehensive and the analysis accurate and complete. Non-technical readers might have to skip more technical parts, especially during the first reading.

1. Introduction

The Cyber–Trust project aims to develop an innovative cyber–threat intelligence gathering, detection, and mitigation platform to tackle the grand challenges towards securing the ecosystem of IoT devices. These challenges rest with the complex structure of the IoT ecosystem, which is comprised of heterogeneous connected devices –computers, laptops, smartphones, and tablets, as well as, embedded devices and sensors– communicating and exchanging large volumes of data. For example, security issues occur from embedded devices and other legacy hardware, whose flawed design or their poor configuration allows the cyber–criminals to compromise them in order to mount large-scale attacks. The project's interdisciplinary approach will capture different phases of such emerging attacks, before and after *known* (even years old) or *unknown* (zero-day) vulnerabilities have been widely exploited by cyber-criminals to launch the attack. To this end, this deliverable gives emphasis on proactive cyber-security technologies, such as

- *cyber-threat intelligence* (CTI) gathering and sharing in order to prevent the exploitation of zero-day vulnerabilities;
- utilization of *trust management systems* (TMS) to measure the trust placed on hardware, software, or network elements for reliably performing their (possibly mission-critical) operations; and
- *game theory* (GT) for providing an intelligent cyber-defense framework by altering in optimal ways the defending system's attack surface.

These technologies are utilized to provide the functionalities envisaged by the crawling service (A07), the eVDB admin module (A07) and eVDB sharing service (A09), the trust management system (A05) and trust DB admin module (A08), as well as the smart gateway/device iIRS (A13, A14). More details about Cyber-Trust's platform components are given in the deliverables D2.3 [384] and D4.1 [62].

1.1 Purpose of the document

This document aims at providing the state-of-the-art in the areas covered by work package WP5 (key proactive technologies and cyber-threat intelligence) of Cyber-Trust project, and in particular, it focuses on the three research tasks (T5.1–3), as illustrated in Figure 1.1.

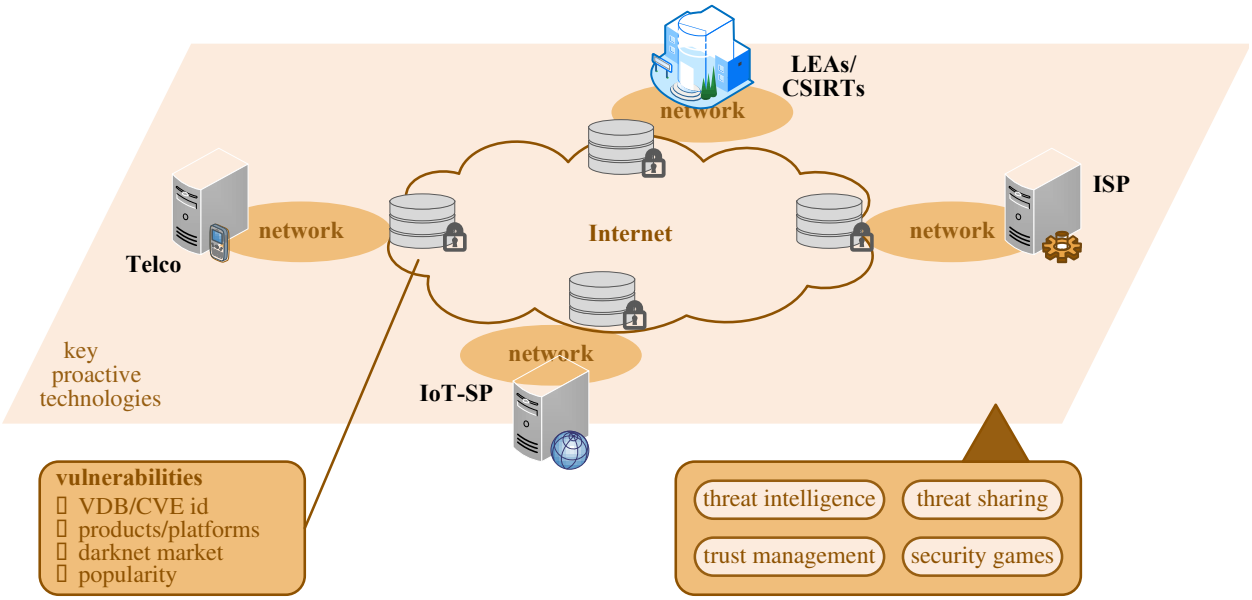


Figure 1.1: Overview of Cyber-Trust's key proactive technologies

The effectiveness of contemporary cyber-defense systems heavily depends on information gathered from various types of sources (e.g. internal, community, and external) in order to leverage this information into actionable *cyber-threat intelligence*. Although this task is necessary, given the evolving nature of current

cyber-threats, the collection, identification, mining, leveraging, and sharing of CTI has become increasingly complex. Therefore, the first chapter of the deliverable, focuses on the relevant technologies used for the efficient gathering of such information (by means of a crawler), its management/storage, and sharing. With respect to crawling, following a presentation of the different sources that are typically crawled, a number of architectures are outlined and the available solutions are classified based on the targeted content's type. The section devoted to management/storage deals with the need for processing, homogenizing, transforming, storing and indexing the information gathered from the clear, deep, and dark web. Then, the last part of this chapter deals with CTI sharing and the architectures of CTI sharing systems, associated tools and platforms (a short summary of D2.2 findings), while in addition it reviews solutions aiming at enhancing the quality of the CTI shared, by maintaining peer trust relationships, and preserving users' privacy; a number of privacy-preserving methods are presented ranging from generalization schemes to notions like *k*-anonymity and *t*-closeness.

The profound application of the CTI collected from external sources, particularly when it concerns previously unknown (0-day) vulnerabilities, is in managing the potential threats targeting at a device/system/network. Therefore, the next chapter deals with the notions of risk assessment/mitigation and trust establishment between a set of unknown (and hence untrusted) systems that need to interact for sharing data or using a service. Such models are well-suited with the application domains of Cyber-Trust, where either devices with weak security defenses need to interact (e.g. in the smart home domain), or devices that are completely unknown (e.g. in the mobile communications domain) — i.e. in both cases the surrounding environment is assumed to be adversarial. This chapter starts with an overview of the main foundations of trust and risk management (namely status-, behavioral-, and risk-based), and for each of which tools, methods, and other information sources that can be used for realizing trust and risk management in the relevant context are presented. Many models have been proposed in the literature for the computation of the trust score that should be associated with some party; these could either completely rely on a system's own measurements, or could consider third parties' recommendations. A large part of the chapter aims at presenting a number of *trust management models*, and for each of them, to review its resilience against faults and well-known attacks. Since trust management has not been treated so far, the chapter also reviews the available (open source) *trust management systems* (TMS) and their features, including their architecture that has a great impact on how the TMS components are deployed in a target network. Next the chapter is devoted to the ways in which both the trust and risk scores can be used in order to effectively respond to cyber-attacks (in the *long-term*, *short-term*, and *immediately*), and decide on the defense measures to be taken.

Such decision-making can be done in an optimal way (and this concerns either proactive defense mechanisms or reactive ones) by devising efficient cyber-defense strategies and accurately modelling attackers' behavior. These strategies are the result of game-theoretic models that allow us to outperform traditional solutions in security and privacy due to the theoretical guarantees they provide for a sound and coherent analysis. This chapter builds upon the game-theoretic methods for cyber-security that have been devised on attack graphs (a structure introduced in deliverable D2.5 and allows to capture more complex and multi-stage attacks, compared to traditional systems). The chapter presents a number of related works (on attack graphs) that deal with the design of an automated intelligent intrusion response system, as well as, a high-level overview of a number of efficient solution-finding algorithms.

1.2 Relations to other activities in the project

This document continues the work conducted in a number of tasks. More precisely, the structure and content of Chapter 2 builds upon the results obtained in Deliverable D2.2 [61], where issues pertaining to cyber-threat intelligence sharing have been thoroughly treated and MISP platform has been selected as the solution to be employed in Cyber-Trust. In addition, Chapters 3, 4 utilize the underlying structure, so-called as attack graphs, that has been shown in Deliverable D2.5 [60] to greatly facilitate the modelling and mitigation of sophisticated multi-stage cyber-attacks. This work also considered the developments and needs of tasks T6.1 (privacy-preserving profiling) and T3.1 (regulatory framework analysis), as Cyber-Trust's trust model utilizes information retained in a device's profile (component A17 in Deliverable D4.1 [62]) and considers privacy aspects while designing solutions (*see* e.g. Section 2.4.5). This work feeds all tasks of work package WP5 as

well as the task T6.3 (network attack detection and mitigation) which is the application area of all proactive technologies described hereinafter.

1.3 Structure of the document

This deliverable consists of six chapters, including the current introduction (chapter 1) and references (chapter 6). More precisely, the rest of the documents is structured as follows:

- Chapter 2 reviews the current state-of-the-art in cyber-threat intelligence gathering and sharing techniques. It includes aspects pertaining to the strategies used for gathering cyber-threat related information from the Clear, Deep, and Dark Web, the management of CTI and the heterogeneous sources from which information is collected along with ways to assess the quality of information gathered, as well as CTI sharing architectures, tools, formats, and privacy aspects.
- Chapter 3 reviews the current state-of-the-art in establishing trust and assessing risk in environments where a number of unknown parties need to interact with each other in order to use a service or to share data. The chapter starts by presenting the foundations of Cyber-Trust's three trust dimensions, which are associated with status-based, behavioral-based and risk assessment-based methods. Next, it presents an overview of the trust models and architectures that are typically used in the literature, and also reviews and compares a number of open source tools that implement a trust management system. Finally, ways in which trust and risk scores can be used in order to respond to cyber-attacks, and determine the defense measures to be taken, are also presented.
- Chapter 4 reviews the current state-of-the-art in the area of game-theoretic security that includes models and methods having been proposed to overcome traditional solutions' limitations and build an intelligent cyber-defense framework. The chapter presents the background of optimal decision-making, starting from single-agent problems, before moving on to multi-agent problems and define various game types (from static to dynamic ones). Finally, a number of decision-making frameworks that have been proposed for cyber-defense are presented.
- Finally, the main conclusions obtained are summarized in chapter 5.

2. Cyber-threat intelligence techniques

2.1 Introduction

Cyber-threat intelligence is the part of intelligence that relates to networks, computers and other types of information technology. Intelligence is the actionable information and/or knowledge gained about an adversary by utilizing observation and analysis. It is important to note that intelligence does not typically refer to data, but rather to information that has been analysed, leveraged and converted to a series of actions that may be followed upon (actionable).

Typically, cyber-threat intelligence can be categorized as tactical or strategic. *Tactical intelligence* is the most basic form of intelligence and is often used for machine-to-machine detection of threats; it is composed of low-level signals that describe what an organization needs to focus on when responding to incidents using the tools at their disposal. It includes concepts like *tactics, techniques and procedures* (TTP) and *indicators of compromise* (IoCs). Contrary, *strategic intelligence* refers to higher-level information on how an organization defends itself and improves its overall cybersecurity posture.

Over the years cyber-threats have increased in numbers and sophistication; adversaries now use a vast set of tools and tactics to attack their victims with their motivations ranging from intelligence collection to destruction or financial gain. Such tools can be found in Clear/Deep/Dark Web sites that altogether constitute the web as illustrated in Figure 2.1.

Figure 2.1. The pillars that constitute the web: clear, deep, and Dark Web [235]

Clear web. Clear/Surface Web or Clear/Surface Net is the part of web that can be indexed by a typical search engine like Google, Bing, and Yahoo. These web sites can be easily found through a search query within any search engine.

Deep Web. Deep Web/net is the part of web that a typical search engine cannot index. This part of the web is still accessible through standard browsers; sometimes special configuration required. However, it

might be protected, or hidden from the surface web using encrypted URLs, password protected pages, local/ internal networks, and direct IP address.

Dark Web. Dark Web/net is the part of the web, which is a subset of Deep Web, that is made intentionally hidden and/or made inaccessible through standard browsers or requires specific configuration to access through the standard browsers. Since search engines cannot index these sites, it is not possible to locate them in search results. Dark Web is a subset of Deep Web where criminal/illegal activities like purchase and sale of services/goods ranging from custom-made tools for cybercrimes to stolen data, digital fraud, counterfeits, drugs, weapons, and more take place. Dark Web comprises of various types of sites including marketplaces, forums, paste sites, IRC servers/channels, and chat rooms. Many of these sites are protected using passwords or other authentication mechanisms. There might be a vetting and vouching process in some cases, like passing a test or making a crypto-currency transaction, to get access to these sites. Also, most of these sites are only accessible through Tor[1], and some of others might have only an IP address, and most of these sites cannot be found through a search engine [235].

Given the evolving nature of current cyber-threats and the inherent distribution of the relevant information sources, the task of collecting, identifying, mining, leveraging, and sharing actionable cyber-threat intelligence has become increasingly complex. The current section provides an overview of different state-of-the-art tools and technologies that are currently utilized to achieve the aforementioned complex tasks.

2.2 Clear/Deep/Dark Web crawling

Web crawlers, typically also known as robots or spiders, are tightly connected to information gathering from online sources. They constitute an important part of search engines and they lie at the heart of most information gathering tasks that are performed online. Since the first crawler, called the *Wanderer* written by M. Gray in 1993 for collecting statistics about the growth of the web, a lot of progress has been made in the field of information gathering by crawling. In this section, we outline the state-of-the-art in (Clear, Deep, and Dark) web crawling and present different aspects of the crawling technology. Initially, we present the different online sources that are typically crawled and categorize them by typology, content dynamicity, and available crawling techniques. Subsequently, we outline typical architectural alternatives that fit the crawling task and categorize the available crawling solutions according to the type of the targeted content. Finally, in the last two sections we discuss techniques and tools pertaining to website accessibility and URL manipulation/exploration.

2.2.1 Crawler architectures and typology

Conceptually, the typical procedure followed by a web crawler is fundamentally simple: it visits a URL, downloads the webpage associated with it, extracts the URLs therein, compares them with a list of visited URLs and adds the non-visited ones to its frontier list (i.e., a list of URLs to be visited). This procedure is repeated until a (sub-)domain is fully crawled. Obviously, this simple procedure does not necessarily need a highly sophisticated architecture as it can nowadays be easily implemented by resorting to any of the high-level scripting languages. Figure 2.2 presents the main component of a typical crawler that include the page downloader module, the page storage module, the URL extraction module, and appropriate data structures storing the list of visited URLs and the crawl frontier (accompanied with appropriate metadata depending on the type of the crawler). However, harnessing the vast scale of online resources and dealing with varying types of content and information sources requires careful engineering and involves important architectural decisions.

[1] https://www.torproject.org/

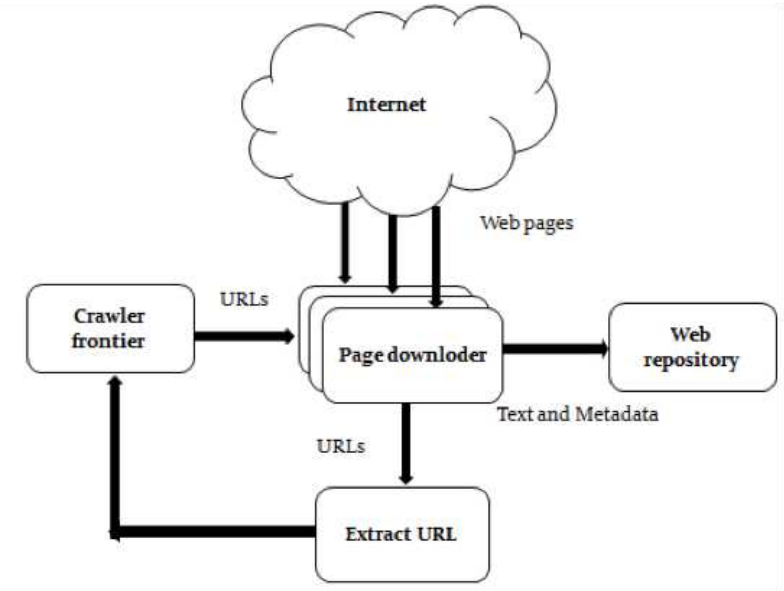


Figure 2.2. Typical crawler architecture

2.2.1.1 Architectural typology

Depending on the crawling application at hand, the available hardware, the desired scalability properties and the ability to scale up/out the existing infrastructure related literature provides a number of different architectural alternatives [221], [167], [14] for crawlers.

- **Centralized.** Typically, special-purpose or small-scale crawlers follow a centralized architecture [221]; the page downloading, the URL manipulation (i.e., the data structures maintaining the visited URLs and the frontier list), and the page storage modules, all resort in a single machine. This centralized architecture is, naturally, easier to implement, simpler to deploy, and straightforward to administer, but is limited to the capabilities of the hardware and thus cannot scale well. For this reason, the more sophisticated crawler designs put effort in scaling out, i.e., exploiting the inherently distributed nature of the web and adopt some form of decentralization.
- **Hybrid.** Hybrid crawler architectures (e.g., [333]) are the norm in the architectural typology as they aim for a conceptually simple design that involves distributing some of the processes, while keeping other centralized. In such architectures, the page downloading module is typically distributed, while URL management data structures and modules are maintained at a single machine for consistency reasons. Such designs aim at harnessing the control of a centralized architecture and the scalability of a distributed system, however the centralized component usually acts as a bottleneck for the crawling procedure and represents a single point of failure.
- **Parallel/distributed.** A parallel crawler [261], [100], [82], [63], [72] consists of multiple crawling processes (usually referred to as C-procs in crawler jargon), where each such process performs all the basic tasks of a crawler. To benefit from the parallelization of the crawling task the frontier is typically split among the different processes, while to minimize overlap in the crawled space links and other metadata are communicated between the processes. When all processes run on the same LAN then we refer to a (intra-site) parallel crawler, while when C-proc's run at geographically distributed locations connected by a WAN (or the Internet) we refer to a distributed crawler.
- **Peer-to-peer.** The advent of peer-to-peer computing almost two decades ago introduced peer-to-peer search engines like YaCy[2] and Minerva [301], [54] (to name but a few); this in turn gave rise to the concept of peer-to-peer crawlers [254], [30], [1], [36]. Peer-to-peer crawlers constitute a special

[2] https://yacy.net/en/index.html

form of distributed crawlers that are typically targeted to be run on machine at the edge of the Internet, as opposed to their distributed counterparts that are designed for clusters and server farms. To this end, peer-to-peer crawlers are lightweight processes that emphasize crawl personalization and demonstrate large-scale collaboration usually by means of an underlying conceptually global but physically distributed routing infrastructure (e.g., a DHT [152], [306] or a super-peer network [254]).

- **Cloud-based.** Lately, the requirement for more effective use of resources by means of elasticity gave rise to a new crawler paradigm: the cloud-based crawlers [207], [184], [350], which revived known machinery to a renewed scope, versatility and options. Such architectures use cloud computing features alongside big data solutions like Map/Reduce and NoSQL databases, to allow for resource adaptable web crawling and serve the modern the Data-as-a-Service (DaaS) concept.

2.2.1.2 Policy-based typology

Since the main task of a crawler is to traverse the webpage graph and collect information, crawlers are also categorized by the way the web graph is traversed. This graph traversal policy defines not only the type of the crawler at hand, but also the domain that the crawler is most suitable for.

- **Universal (general-purpose).** General-purpose crawlers are mainly meant to support search engines and typically aim at large-scale crawls; the typical universal crawler policy has to take into account trade-offs regarding thematic coverage, page freshness, and content/page bias. On the architectural side, universal crawlers are typically hybrid or highly parallelized, as they need to scale up to billions of webpages.
- **Incremental.** To date, the primary application of crawling is search engines and their effort to keep a good-enough replica of the web to their repository. However, given the dynamic and uncoordinated nature of the web, where URLs are constantly being created or destroyed and contents of webpages change, it becomes inherently complicated to keep up with this rate of change. Thus, to minimize the impact of the inconsistencies between the web and the contents of the repository, URLs should periodically be prioritized and revisited. The simplest way to do this is periodic crawling, but this implies a waste of resources by revisiting pages that have not changed. The informed process of prioritizing and revisiting URLs that are more likely to have changed (e.g., regarding changes the recent past can help predict the future [23]) since the last time they were crawled, is usually referred to as incremental crawling (see also [192] for a comprehensive survey).
- **Preferential (focused/topical).** Since many crawlers can download only a small subset of the Web, they need to carefully decide what page to download. By retrieving *important* or *thematically relevant* pages early, a focused crawler [261], [311], [297], [285], [256], may improve the quality of the downloaded pages. The studies in this category explore how a crawler can discover and identify important webpages or pages relevant to a topic [376] or user interest [211], [296], [285], [191], [321], [14], and propose various algorithms to achieve this goal.

2.2.1.3 Usage typology

Although the first web crawlers that set the pathway for the spidering technology were developed for the Clear (or Surface) Web, in the course of time specialized solutions aiming at the different facets of the web (web 2.0, Deep Web, Dark Web, etc) were gradually introduced. Below we organize crawlers in terms of their intended usage.

- **Clear/surface web.** Since the introduction of the first crawler in 1993, the majority of the research work on crawlers has focused on the crawling of the surface web, initially on behalf of search engines, and gradually also for other tasks. There is an abundance of work on Clear Web crawling; some insightful surveys on the topic include [192], [221].
- **Web 2.0.**The advent of the user-generated content philosophy and the participatory culture that was brought by Web 2.0 sites like blogs, forums and social media, formed a new generation of specialized crawlers that focused on forum [164], [349], [27], [164], [53], [179], [358], [277], [346],

blog/microblog [213], [257], [219], [250], [294], and social media [70], [377], [313], [295], [104], [106], [33] spidering. The need for specialized crawlers for these websites emerged from

  - the quality (i.e., typically structured and user-generated) and creation rate of content usually found in forums/blogs,
  - the well-defined structure that is inherent in forums/blogs that makes it possible to even develop frameworks for creating blog crawlers [283], [282], and
  - the implementation particularities (e.g., Javascript-generated URLs) that make other types of crawlers inappropriate or inefficient for the task.

- **Deep/Dark/Hidden web.** The amount of information and the inherent interest for data that reside out of reach of major search engines, hidden either behind special access websites (Deep Web) or anonymization networks like Tor[1] and I2P[3] (Dark Web), gave rise to specialized crawlers that are able to traverse these hidden regions of the web [131]. To this end, over the last ten years, a number of works related to Deep Web crawling have been published [137], [359], [110], [275], [200], [168], investigating also
  - different architectural [298], [177], [352], [71], [375] and automation [318] options;
  - quality issues such as sampling [166], [362], query-based exploration [165], [361], duplicate elimination [360], [200]; and
  - new application domains [308], [353], [348].

  Recently, interest has shifted towards Dark Web crawling [317], [46], with a number of tools and techniques that modify surface/Deep Web technologies to harvest information from the Dark Web.
- **Cloud.** The elasticity of resources and the popularity of cloud-based services inspired a relatively new line of line of research focusing on crawler-based service configuration [218], [315] and discovery in cloud environments [320], [181].

2.2.2 Collection and content typology

Focused crawling research studies are aimed at the indexing of various collections of pages such as social media, blogs, forums, and marketplaces by targeting specific topics that might exist within them. In the area of security, various research studies have been carried out focusing on such page collections.

Regarding the generic spectrum of focused web crawling, on the matter of security, there are studies which indicate crawling solutions that can seek and identify malicious websites efficiently [117], [32]. Hattori et al. in [117], have created an algorithm which can help a crawler focus on malicious webpages. The aforementioned functionality is achieved by determining the probability of webpages being malicious or harmless. This probability is calculated by taking under consideration all network-related attributes of the web server, deriving from the URL string. In [32], Singh et al. propose a focused web crawler, named "MalCrawler", which has been designed to crawl and search malicious websites efficiently. It starts the crawling procedure with an initial list of malicious seeds, which are more likely to contain hyperlinks to other malicious sites [199]. Then, it seeks pages with dynamic content, since such pages are more likely to contain malicious code, as server/client-side scripts contained within them might be vulnerable to hacks like cross-side scripting [309]. Finally, with smart filtering techniques [66], pages that need to be examined are reduced. In addition to the two focused web crawling solutions presented previously, in [262], Likarish et al. have focused research on a targeted web crawler which is able to build a malicious JavaScript collection. In order to do so, they use a machine learning classifier, which detects potentially malicious JavaScript to help the web crawler with targeting to domains that are more likely to host more malicious JavaScript.

[3] https://geti2p.net/

Regarding forum webpages, the crawling procedure differs from the one followed in a generic web crawler, due to the lack of a centralized index in the first [240]. For web forums, in addition to the standard crawling procedure, wrappers around information and metadata are required, in order to gather all practical data deriving from the dynamic content within it, such as authors, messages, timestamps, and so on. In other words, wrappers are important for getting an insight on data within forums and for achieving an incremental crawling of them (i.e., re-spidering only threads within forums which have been updated since the last indexing). In [240], Glance et al. propose a system for harvesting messages from web forums, which is consisted of the two aforementioned components; a crawler and a wrapper. In the same manner, Limanto et al. in [139] developed a web forums' information-extraction engine, which consisted of a crawler, a wrapper generator, and an information extractor for each generated wrapper. Moreover, there are numerous research studies, focusing on specific topics, such as online deals [336], USENET newsgroups topics [226], or even cooking recipes [351], based on focused web forums' crawling. Yih, Chang, and Kim in [336] created a web forum mining system, composed of a crawler and an information extractor for mining "hot deals" out of deal forums. There, participants would share information regarding deals and promotional events that were offered by e-shops. What is important about RecipeCrawler [351], is the fact that it uses tree edit distance scores between web pages, in order to rank them in the crawl space. Furthermore, Guo et al. in [347] proposed a board forum crawler, which is capable of traversing web forums, that follow the board architecture, in a hierarchical manner, similar to that preferred by actual users who manually browse the forum. All the aforementioned research studies on web forum crawlers, are focused on the surface (open) web. Lastly, there is only one research study, which proposes a focused web crawler for Dark Web forums [317]. It has a similar structure as in the solutions described in the previous studies, but it also focuses on the aspects of forum identification and forum accessibility, which are factors that make Deep Web forums' crawling challenging. The forum identification phase consists of three steps:

- Extremist groups' identification (from government sources, or open sources).
- Identification of forums that exist within extremist groups' webpages.
- Identification of forums hosted on major webpages (which are likely to be used by Dark Web groups).

Finally, the forum accessibility phase, includes three steps as well:

- Forums membership application (since many Dark Web forums do not allow anonymous access [367]).
- Identifying appropriate spidering parameters, such as number of connections permitted per webpage, download intervals, timeout, speed, and so on.
- Identifying appropriate proxies.

Concerning the social media, there have also been research studies, focusing on the crawling of such webpages' content. Firstly, Xu et al. in have developed SOUrCe [194]; a structure-oriented unsupervised crawler, which utilizes the social media page structures, in order to learn how to crawl it efficiently. It consists of two stages; the unsupervised learning phase and the information harvesting phase. During the unsupervised learning phase, it constructs a sitemap which clusters pages, based on the similarity of their structure and then generates a navigation table, which describes how all different types of pages in the site are interlinked. Then, during the information harvesting phase, SOUrCe uses the navigation table and a crawling policy to help guide the choice of which links should be crawled next. In another research study, the authors consider publicly available data published in open pages and groups on Facebook [105]. Erlandsson et al. [105] introduce the User-guided *Social Media Crawling* (USMC) method for efficiently and precisely crawling quality data from Facebook. What USMC does, is that it ranks posts based on metadata metrics, such as the number of likes, and then selects the highest ranked posts. Thus, USMC method adopts a quantitative data measuring strategy by regarding the quality of the crawled data as equivalent to the proportion of all available social interactions in the social media services.

There has also been work on blogs and micro-blogs crawling. The main differences between blogs and micro-blogs are: the crawled documents' length, and the fact that micro-blog posts usually contain a users'

commenting section, whilst there might exist numerous blogs without such a section, and hence simple blogs' crawled documents usually contain static content. For the purposes of blog crawling, one significant research study is RetriBlog [111], which follows the simple crawling architecture, but it contains a rich application services layer, which helps the crawling procedure by providing valuable insight of crawled documents. Such procedures that exist in the application services layer are tag parsing, text preprocessing, content extraction, and tag recommendation. These procedures can help the crawler focus on blogs that contain information on a specified topic. Another weblog crawler example is BlogForever Crawler [250], which as its core module uses Scrapy (1), an open-source framework for web crawling. The algorithms developed for BlogForever Crawler can extract blog post articles, as well as variations for authors, dates and comments. The crawling methodology in this study, works under two hypotheses:

- Blogs provide web feeds; structured and standardized views of the latest posts of a blog.
- Posts of the same blog, share a similar HTML structure.

For the content extraction, the idea is to use a training data, in this case pairs of HTML pages and target content, in order to build an extraction rule-set, capable of locating the target content on each HTML page. These rules, are XPath queries, derived from parsing of the HTML pages containing the blog posts. Finally, using techniques like string similarity, tree distance, and maximum weighted matching in a complete bipartite graph, allow us to extract authors, dates and comments expressed in various formats. Regarding micro-blogs, a research study by Agarwal et al. [293] presents a topical crawler for discovering hidden communities of extremist micro-bloggers on Tumblr; a micro-blogging and social networking website. This study proposes a crawler which, in a graph traversal, returns relevant nodes to a specific topic. The relevance of a node is defined by initially learning the characteristics and features of a given topic and then computing the extent of similarity against a set of exemplary documents. To collect the training examples, the authors performed an iterative search on Tumblr, using keyword-based flagging, where keywords are search tag like "jihad, anti-Islam, and hate". By using these tags as an initial seed, the crawler collects all textual posts, tags, and linked bloggers, who either posted, re-blogged a post, or liked one, removing duplicates. Following this approach in an iterative manner, the authors were able to extract more posts and linked bloggers, concerning the initial and additional tags that where similar to the ones initially submitted.

Concerning the last collection type, marketplaces, there is one existing research study to our knowledge, which demonstrates an XPath based crawling method, alongside with crowdsourcing, that is capable of focusing on the valuable information contained within marketplace webpages [173]. As stated by the authors, there are two critical issues faced when a traditional crawler is applied for gathering information from a targeted up-to-date online marketplace. The first, is that marketplaces include too many links, among which only a few are enough to navigate all web pages in the site. Next, the second issue is that most links are provided by JavaScript; not by the anchor tags, and hence they cannot be followed by the traditional web crawlers. The developed crawler in this study overcomes these issues, by following a method which can extract all necessary and sufficient links by adopting a crowdsourcing approach, and can also follow JavaScript links by using navigation paths represented by XPaths.

In regard to the content typology existing within the collections described previously, webpages contain a wide variety of indexable and multimedia files. Concerning the indexable files, content is divided in two main categories; static text files (e.g., HTML, Word, PDF documents, etc.), and dynamic text files (e.g., .asp, .jsp, .php, and so on). As pointed out previously, in order to crawl dynamic pages, systems require the generation of wrappers around content, to crawl only recent/unindexed content, whereas pages that contain static content, can be crawled with traditional crawlers efficiently. Additionally, in some cases of dynamic pages, systems contain methods which extract information from JavaScript blocks of code [262], [173]. Multimedia files on the other hand, which may include images, flash content, audio, and video files, are difficult to index and to collect accurately [39]. Moreover, multimedia file sizes are usually significantly larger than indexable files, and hence crawling them requires longer download times, which might result in timeouts. This statement was proven by Heydon and Najork [15], who designed a crawler that would fetch all multimedia files in addition to indexable files. They concluded that by collecting such files, increased the total crawling

time and doubled the average file size compared to fetching only HTML files. Therefore, most research studies ignore the fetching of multimedia files.

2.2.3 Website accessibility and crawling strategies

When it comes to how the Web can be accessed, two separate categories can be defined. The Clear Web and the Deep Web. On the Clear Web, a classic search engine or a focused crawler, such as BINGO! [310], is more than enough to access and discover new web pages that match a specific topic. On the other hand, search and direct access aren't as straightforward on the Deep Web as there is a wide range of techniques used to control access to Deep Web content [118]. **Error! Reference source not found.** describes some of those techniques.

Table 2.1: Deep Web access control techniques

Deep Web content	Access control techniques
Private Web	Private websites, such as intranets with public-facing webpages, typically require registration and login using password or other authentication mechanisms to gain access to the private, more protected, side of the website.
Contextual Web	Elements of value can be discovered through a history of navigation across websites with common contextual threads. For example, a unique semantic identity can be linked to an individual based on that person's online activity.
Dynamic content	A dynamic page appears as a hidden response to a specific query or through submission of a specific element or set of elements on a form. In either case, the resulting text fields are hard links to discover, much less navigate, without direct reference to a domain.
Limited access content	Many websites use a mechanism to limit access to or prevent duplication of their content. • The Robots Exclusion Standard or the Robot Standard serves to discourage searching. • A *completely automated public Turing test to tell computers and humans apart* (CAPTCHA) requires users to mimic a random code to assure they aren't robots. • A no-store directive serves to prohibit creating cached copies.
Non-HTML content	The World Wide Web depends on Hypertext Markup Language (HTML) to perpetuate linkages. Non-HTML pages are hard to discover and access, as the accepted links are absent. • Textual content encoded as images, video files, or similar visual file formats, not discoverable via search engines, are also difficult to isolate by other means. • Scripted content involves links produced by JavaScript or content that is dynamically downloaded from webservers via Flash or Ajax solutions. • Specialized software, like *The Onion Router* (TOR)[1] or the *Invisible Internet Project* (I2P)[3] anonymous peer-to-peer distributed communication network, are required to access content not otherwise discoverable on the WWW or Internet.

Unlinked content	Not all HTML pages contain external links. • Websites typically contain links from other websites, known as *backlinks* or in(bound) links. The number and quality of backlinks are a major factor in determining a website's ranking in search engine results. However, search engines don't usually detect all backlinks. In addition, not including backlinks can prevent search engines and web crawling programs from discovering and indexing website content. • Web archival services can reveal now-defunct webpages across their history. Websites such as the Wayback Machine (archive.org/web) allow viewing of old and often obsolete webpages that are often no longer accessible by today's search engines.

2.2.3.1 Form-based content

Dynamic content is accessed via form-based search interfaces that are designed primarily for human consumption. To access this type of content, a Deep Web crawler must be able to automatically parse, process and interact with such forms, but must also be able to provide input to them. To address these challenges the *Hidden Web Exposer* (HiWE) crawler was built at Stanford [305]. HiWE utilizes a semi-automatic, task-specific, human-assisted approach. With this approach, portions of the hidden Web are selectively crawled, to extract content based on the requirements of a particular application or task (topic). Then an archive is built in two steps:

- **Resource discovery:** Where sites and databases that are likely to be relevant to the task are identified.
- **Content extraction:** Where the crawler actually visits the identified sites to submit queries and extract hidden pages.

Human-assistance is very important to the content extraction step, because it's the only way to ensure that the crawler makes queries that are relevant to the particular task. For the system to be able to successfully understand a form, first the task-specific database (called Label Value Set-LVS in HiWE) must be populated. There are 4 mechanisms for adding entries to the database:

- **Explicit initialization**, where a user supplies HiWE with labels and associated value sets at startup time. Explicit initialization is particularly useful to equip the crawler with values for the labels that the crawler is most likely to encounter.
- **Built-in entries** for certain commonly used categories, such as dates, times, names of months, days of the week, etc., which are likely to be useful for a variety of tasks are in the LVS.
- **Wrapped data sources**. The LVS Manager can communicate and receive entries for the LVS table by querying various data sources (on the Web or elsewhere), through a well-defined interface. These data sources can either be task-specific or correspond to relevant portions of generic directories.
- **Crawling experience**. Finite domain form elements are elements where the set of valid values are already embedded in the page. Whenever HiWE encounters a finite domain form element, it extracts the label and domain values of that element and add the information to the LVS table.

When this process is completed, the labels of the LVS table have to be matched with the form labels. This is achieved with a process called Label Matching. There are two steps in this process First, all labels are normalized by converting to a common case, standard IR-style stemming and stop-word removal. Then, an approximate string-matching algorithm [74] is used to compute minimum edit distances, considering typing errors, as well as word reordering.

Finally, HiWE employs an aggregation function to compute a rank for each value assignment and uses the value assignments whose rank is higher than the minimum acceptable value assignment rank.

2.2.3.2 Registration-based content

A task-specific, human-assisted approach such as the one described above, is useful for Dark Web content as well, which is usually hidden behind registration-based systems, CAPTCHA and verification codes. In general, simple forms asking for name, e-mail address, and so on can be automated with standardized responses, but more complex questions require greater expert involvement. The semi-automatic approach allows varying human involvement, depending on the complexity of the accessibility issues faced. The crawling system proposed by Fu et *al.* [317], consists of an accessibility component that uses a human-assisted registration approach to gain access to Dark Web forums. It also utilizes multiple dynamic proxies and forum-specific spidering parameter settings to maintain forum access. Specifically, regarding forum accessibility, the steps needed to obtain access are the following:

- **Apply for membership.** Many Dark Web forums (~30–40%) do not allow anonymous access [366]. To access and collect information from those forums, one must create a user ID and password, send an application request to the Web master, and wait to get permission/registration to access the forum. In certain forums, Web masters are very selective. It can take a couple of rounds of e-mail to get access privilege. For such forums, human expertise is invaluable. In some cases, access cannot be attained. According to Fu et al. [317], approximately 10% of the Dark Web forums cannot be accessed at all.
- **Identify appropriate spidering parameters**. Spidering parameters such as number of connections, download intervals, timeout, speed, and so on, need to be set appropriately according to server and network limitations and the various forum-blocking mechanisms. One may also be blocked based on the IP address. Therefore, the use of proxies is needed to increase the chances of access, but also the researchers' anonymity.
- **Identify appropriate proxies**. Three types of proxy servers can be used for the task. *Transparent* proxy servers are those that provide anyone with your real IP address. *Translucent* proxy servers hide your IP address or modify it in some way to prevent the target server from knowing about it; however, they let anyone know that you are surfing through a proxy server. *Opaque* proxy servers (i.e., preferred) hide your IP address and do not let anyone know that you are surfing through a proxy server. There are several criteria for proxy server selection, including the latency (the smaller the better), reliability (the higher the better), and bandwidth (the faster the better). The list of proxy servers should be updated periodically and can originate from various sources, including free proxy providers[4,5]. Additionally, the crawler uses a Web browser user agent string and does not follow the robot exclusion protocol, though nearly none of the Dark Web forums have a robots.txt file.

In both form and registration-based the human-assisted approach could be coupled with open-source tools such as Formasaurus [144], which can classify form fields using machine learning. It can detect if a form is a login, search, registration, password recovery, "join mailing list", contact, etc. That way, the crawler would know which type of input it has to use, and if the different input data were categorized by type, human interaction could become minimal to non-existent. To detect HTML form types Formasaurus takes a `<form>` element and uses a linear classifier (Logistic Regression) to choose its type from a predefined set of types. Amongst others, features include:

- counts of form elements of different types,
- whether a form is POST or GET,
- text on submit buttons,
- names and char ngrams of CSS classes and IDs,
- input labels,

[4] www.xroxy.com
[5] www.proxy4free.com

- presence of certain substrings in URLs,

To detect form field types Formasaurus uses *Conditional Random Field* (CRF) model. All fields in an HTML form is a sequence where order matters; CRF allows to take field order into account. There are about 50 distinct field types. Amongst others, features include:

- form type predicted by a form type detector,
- field tag name,
- field value,
- text before and after field,
- field CSS class and ID,
- text of field <label> element,
- field title and placeholder attributes.

2.2.3.3 Collection update procedure

Another critical issue on Deep/Dark Web crawlers is the procedure of updating the collection. There are two major approaches for that task, periodic and incremental spidering. The periodic approach re-spiders the entire collection of targeted forums in a fixed time interval. This could be very time consuming and would make it difficult to keep the collection up-to-date. The incremental approach focuses on downloading only new content found in the targeted forums, so that the spidering process can be completed within a short time and even before the forums become aware of the spider [365]. Because of that, it seems like a more feasible solution for keeping the collection up-to-date.

While this is true in terms of spidering time and data redundancy, Fu et al. [317] suggest that a periodic crawling approach could be of use in the case of Dark Web forums, as the crawler gets multiple attempts to collect each page and subsequently improve overall collection recall. Because of the need to make multiple attempts to collect some pages, they propose a collection update procedure that consists of an incremental crawler and a recall-improvement mechanism (Incremental+RI).

With the Incremental+RI method, the incremental crawler fetches only new and updated threads and messages. A log file is sent to the recall-improvement component. The log shows the spidering status of each URL. A parser is used to determine the overall status for each URL (e.g., “download complete,” “connection timed out”). The parsed log is sent to the log analyzer, which evaluates all files that were not downloaded. It determines whether the URLs should be re-spidered. The recall-improvement phase also checks the file sizes of collected Web pages for partial/incomplete downloads.

Upon evaluation, the Incremental+RI method yielded higher Precision, Recall and F-measure percentages than the periodic approach. The periodic approach gets better scores with each iteration as previously uncollected pages are gathered, starting from a low score in the first iteration, to finally achieving a score close to the Incremental+RI. The difference is that the Incremental+RI method has this high score from the first iteration and it maintains it for the rest, making it more efficient. The simple incremental approach was faster than the rest, but the scores were low on all iterations, as it skips pages that it tried to crawl, even if it didn’t succeed. Overall, the Incremental+RI approach is the most efficient strategy.

2.2.4 URL manipulation

Since a typical web crawler follows a repeated procedure (that is visiting a URL, downloading the webpage associated with it, extracting the URLs therein, visiting the new URLs) and finally downloads just a fraction of all the available Web pages, it is of high interest to discuss issues like:

1. The features used to derive the crawl space; it is desirable that the downloaded (fraction of) Web pages contain the most relevant pages to the initial seed URL or topic.

2. The different available options of URL ordering, i.e., from a list of available URLs, which URL is visited first; this requires a metric of importance for prioritizing Web pages' links.
3. The different ways a web crawler updates the extracted collection of Web pages.

2.2.4.1 Crawl space derivation

Depending on the features used to extract the crawl space (i.e., the collection of Web pages to be visited), there have been noted four categories in the related literature [56].

- **Link-based crawling.** Starting from a seed URL, inlinks/backlinks, outlinks, and sibling links (i.e., ones with a shared parent in the link) have been considerably used in previous research [56], [126], [210], [20] in order to obtain new URLs to be visited. An interesting approach is presented in [210], where backlinks are derived for each seed URL and used to construct a multilayer context graph; these graphs are then, used to extract paths leading up to relevant nodes (i.e., target URLs) by enabling the crawler to predict the distance of a retrieved page from a target document and not to exclude irrelevant pages that lead to relevant ones from the crawl. Another strategy that gives credit to paths with delayed benefit, is the one used by the geospatial search engine that performs urban web crawling [147]. In contrast with the context graphs technique, which uses a static representation of the crawling environment, the geo-aware focused crawler implements an algorithm that dynamically expands the radius of permitted hops around relevant pages and narrows the crawling scope around the irrelevant ones.
- **Page text-based crawling.** Towards a different approach, focused/topical crawlers may use bag-of-words (BOW) found in the Web page text to derive the crawl space [56, 126, 135, 136], as for example [268] that used BOW for biomedical-text categorization in their focused crawler. Notice that, although page text features can be very effective, they are also language-dependent and can be harder to apply in situations where the collection is composed of pages in different languages.
- **URL and/or anchor text-based crawling.** In a similar approach, other studies use tokens found within the URL and/or anchor text to control the crawl space [211, 56]. An interesting approach is presented in [336], where the crawler presented, coined Hot Deal Crawler, is based on thread dates and performs date comparison to extract the collection of Web pages to be visited. In [305], Hidden Web Exposer crawls dynamic search forms and manages to stay within the target sites using only the information found in URL tokens.
- **Page level-based crawling.** The last used feature for obtaining the crawl space is page levels, where text classifiers are trained to categorize Web pages at various levels away from the target URL [210]. This information can then, be used to build path models and attain target pages; a potential path model can use the number of slashes "/" or levels from the domain as an indicator of URL importance and may consider only the pages one or two levels away from a target, known as tunneling [211], as pages closer to the main page are likely to be of greater importance.

2.2.4.2 Update procedures

For updating the collection of the crawled Web pages, the two most popular approaches used in the literature are periodic and incremental crawling [156]. Periodic crawling is the simple approach of re-spidering all forum pages (as for example [346]), which eventually entails to update the whole URL collection. This approach is commonly used since it is easier to re-fetch the whole crawl space than figuring out which Web pages have been refreshed, especially when the if-modified-since header does not provide any information about which pages within a Web forum have been updated. Although periodic crawling is simpler from a crawler design/development perspective, it makes the collection process time consuming and inefficient. Alternatively, gathering multiple versions of a collection may improve overall recall. To this end, incremental crawlers gather new and updated content by fetching only those boards and threads that have been updated since the forum was last collected [336, 240, 351]. To this end, incremental crawlers require the use of a wrapper that can parse out the "last updated" dates for boards and threads [336]; this information is typically contained in the page's body text.

2.2.4.3 URL ordering

Having decided the information used to derive the crawl space, one has to resolve the way URL ordering is performed. Most web crawlers use an algorithm that estimates a score of importance for every extracted URL in order to evaluate the relevance of the page, pointed by the seed URL, to the initial topic. The estimated score defines then, the priority of the links in the crawl frontier. In this way, in each iteration of the process, the crawler downloads the page (pointed by the seed URL) with the highest score.

Previous research has typically used *breadth*, *depth*, and *best-first* search for URL ordering. Depth first search (DFS) has been used in crawling systems such as Fish Search [64]. Although breadth first search (BFS) is one of the simplest known strategies, it has worked fairly well in comparison with more sophisticated best-first search strategies on certain spidering tasks [155, 220]. However, BFS is typically not employed by focused crawlers that are concerned with identifying topic-specific Web pages using the aforementioned URL features (i.e., page text, URL, or anchor text). For focused crawling, BFS has been out-performed by various best-first strategies [108].

Best-first search uses some criterion for ranking URLs in the crawl space, such as link analysis, text analysis, or a combination of the two [108]. Considering link-analysis techniques, [126] used Page Rank to decide on URL ordering, [155] evaluated the effectiveness of both Page Rank and back-link counts on URL ordering, while [56] used the number of relevant siblings considering that pages with a higher percentage of relevant siblings are more likely to also be relevant. [311] used the HITS algorithm to compute authority scores, whereas [297] used a modified version of HITS to decide on URL ordering. As an alternative, [209] used a Hopfield net crawler that collected pages related to the medical domain based on link weights. On the other hand, text-analysis methods include similarity scoring approaches and machine learning algorithms. For deciding on the similarity among URLS, [56] used similarity equations with page content and URL tokens, while [108, 126, 268] have used the vector space model and cosine similarity measure for the same purpose. In a similar spirit, the work in [311] was based on the text of the visited page and used support vector machines (SVM) with BOW for document classification, while [268] used BOW and link structures together with a neural net for ordering URLs based on the prevalence of biomedical content. Towards a different approach, [135] and [136] used a genetic algorithm to order the URL crawl space for the collection of topic-specific Web pages based on BOW representations of pages. A machine learning technique is used in [296]; an apprentice learner is incorporated and used to evaluate the utility of each outlink by comparing its HTML source code against prior training instances. Finally, recent studies (e.g., [124] and [125]) have compared the effectiveness of various machine learning classification algorithms, such as Naïve Bayes, SVM, and Neural Networks, for focused crawling.

Overall, most previous research on focused crawling has used BOW, link, or URL token features coupled with a best-first search strategy for crawl space URL ordering. Furthermore, most prior research has also, ignored the multilingual dimension, by using text-based features and collecting content in a single language (usually English).

2.3 Data management for CTI

Cyber-Threat Intelligence (CTI) is gathered by crawling and searching a variety of sources (Surface, Deep and Dark Web) and comes in a various of types and formats. Gathered information should be processed, homogenized, transformed and then stored and indexed in appropriate data management systems (see Figure 2.3).

The above procedures follow the general flowchart of the Extraction–Transformation–Loading (ETL) processes that is responsible for the extraction of data from several sources, their cleansing, customization and insertion into a data warehouse [271, 270, 269]. ETL processes are supported by major database vendors [153, 237, 255] and support:

- The identification of relevant information.
- The extraction of this information.

- The customization and integration of the information coming from multiple sources into a common format.
- The cleaning of the resulting data set and its propagation to the data warehouse.

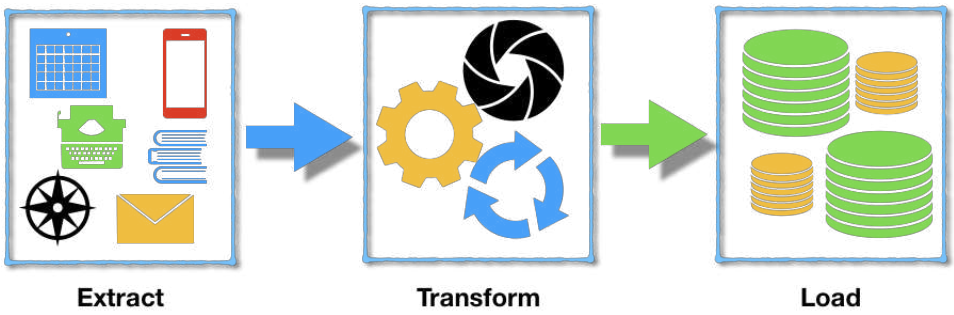


Figure 2.3. CTI management

The case of CTI has many similarities and several differences with ETL. In the following sections we will further elaborate and discuss the above tasks. We will start by discussing the relevant CTI sources and collection techniques (Section 2.3.1) and the related data quality aspects (Section 2.3.2). Then, we elaborate on preprocessing of the collected data (Section 2.3.3). Finally, we conclude by discussing data mining and data enrichment methods (Section 2.3.4).

2.3.1 Data sources and collection techniques

Information security analysts usually focus on scientific methods for data collection and analysis that are testable and reproducible, but there is art and science behind cyber-threat intelligence. Art includes the analysis and the interpretation of data about the attackers and how to disseminate that information to the audience in a more natural way in order to provide incentives to act. This will help security analysts to understand a thinking, evolving and reacting adversary. The security analysts are equipped with new technologies to identify malicious activity on networks and are able to trace that maliciousness back to the attacker. Over the years, these technologies have been developed, aiming at increased detectability of the malicious activity on networks (DPI, NAC and network security intelligence appliances etc., which are based on familiar concepts with new applications).

An effective way to capture potential threats in a proactive manner is by monitoring Internet activities to employ passive monitoring using sensors or traps such as darknet [49]. This section focuses on threat intelligence techniques including quality issues such as quality modelling and assessment for sources, feeds, indicators, data of Dark Web.

2.3.1.1 Commonly used techniques

As already mentioned in D2.2 [61], *intelligence* (INT) sources are most often centered on the INTs, which describe where the data is collected from, namely HUMINT (human INT), SIGINT (signal INT), OSINT (open source INT), IMINT (imagery INT), MASINT (measurement and signature INT), and GEOINT (geospatial INT). These sources are complemented by special tools and techniques to explore the deep and Dark Web. Some of them are similar to or closely related to those sites to explore the Clear Web. Depending on one's overall goals, different tools and techniques will help reach different depths. For most users, there are generally two different but related approaches to access the Deep and Dark Web:

- Use special search engines accessed from regular browsers such as Internet Explorer, Firefox, Chrome, Safari, etc.
- Use special search engines that can be accessed only from a TOR browser.

The research community and those familiar with technology can go even deeper by developing a custom-built crawling program using link crawling techniques and API programming skills. One easy way to gain access to the Deep Web is to use alternative/special search engines that are designed specifically for the purpose. These alternative search engines are designed to access different parts of the Deep Web (as shown in Table 2.2), but the challenge is that all search engines developed so far only crawl or index a small part of the Deep Web. Therefore, it is still necessary to visit the right online directory or hidden web site listings[6]. Since these websites are not indexed, they will not be found using normal search tools. However, their URLs can be found using other means and, once the URL is known, one can then access some of these sites on the Deep Web using regular browsers [162].

Table 2.2. Alternative search engines/tools to access the Deep Web using browsers [162]

General search engines and databases	**DeepDyve:** One of the newest search engines specifically targeted at exploring the Deep Web, available after signing up for a free membership.
	The Scout Archives: This database is the culmination of nine years' worth of compiling the best of the Internet.
	Silobreaker: This tool shows how the news impacts the global culture with current news stories, corresponding maps, graphs of trends, networks of related people or topics, and fact sheets.
	OAIster: Search for digital items with this tool that provides 12 million resources from more than 800 repositories.
	Dogpile: Dogpile searches rely on several top search engines for the results then removes duplicates and strives to present only relevant results.
	SurfWax: This search engine works very well for reaching deep into the web for information.
	Mamma: Click on the Power Search option to customize the search experience with this meta-search engine.
Semantic search tools and databases	**Zotero:** Firefox users will like this add-on that helps organize research material by collecting, managing, and citing any references from the Internet.
	Freebase: This community-powered database includes information on millions of topics.
	Gnod: When searching for books, music, movies and people on this search engine, it remembers specified interests and focuses the search results in that direction.
	DBpedia: This semantic program allows users to ask complex questions and get results from within Wikipedia.

In addition to these search engines, there are black markets in which hackers are presenting zero-day vulnerabilities and exploits for sale. The markets for cybercrime tools and stolen data are quite advanced. Cybercrime markets are rapidly growing, maturing, and continuously innovating. They are full of increasingly sophisticated organizations, people, products, and methods for communicating and conducting business transactions. They are resilient in the face of takedowns and constantly adapting to new tactics and techniques of law enforcement and computer security vendors [132].

[6] https://sites.google.com/site/howtoaccessthedeepnet/working-links-to-the-deep-web

2.3.1.2 Dark Web data collection techniques

Extracting structured data from Deep Web pages is a challenging problem due to the underlying intricate structures of such pages. The problem of Web data extraction has received a lot of attention in recent years and most of the proposed solutions are based on analysing the HTML source code or the tag trees of the Web. These solutions have the following main limitations. First, they are dependent on HTML. As most Web pages are written in HTML, it is not surprising that all previous solutions are based on analysing the HTML source code of Web pages. However, HTML itself is still evolving (from version 2.0 to the current version 5.2) and when new versions or new tags are introduced, the previous works will have to be amended repeatedly to adapt to new versions or new tags.

Furthermore, HTML is no longer the exclusive Web page language, and other languages have been introduced, such as XHTML and XML (combined with XSLT and CSS). The previous solutions now face the following dilemma: Should they be significantly revised or abandoned? Or should other approaches be proposed to accommodate the new languages? Also, they are incapable of handling the ever-increasing complexity of HTML source code of Web pages. Most previous works have not considered the scripts, such as JavaScript and CSS, in the HTML files. In order to make Web pages vivid and colourful, Web page designers are using more and more complex JavaScript and CSS. In a large number of real Web pages, especially Deep Web pages, the underlying structure is more complicated than ever and is far different from their layouts on Web browsers. This makes it more difficult for existing solutions to infer the regularity of the structure of Web pages by only analysing the tag structures [335].

A number of approaches have been reported in the literature for extracting information from Web pages. Informative surveys about previous works on Web data extraction can be found in [12]. Main Dark Web data extraction approaches are as follows.

Manual approaches: The earliest approaches are the manual approaches in which languages were designed to assist programmer in constructing wrappers to identify and extract all the desired data items/fields. Some of the best-known tools that adopt manual approaches are Minerva [328] and Web-OQL [123]. Obviously, they have low efficiency and are not scalable.

Semi-automatic approaches: Semi-automatic techniques can be classified into sequence-based and tree-based. The former, such as WIEN [241], represents documents as sequences of tokens or characters, and generates delimiter-based extraction rules through a set of training examples. These approaches require manual efforts, for example, labelling some sample pages, which is expensive and time-consuming.

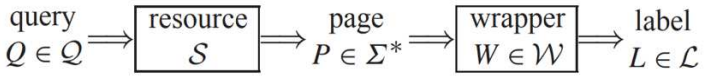


Figure 2.4. An information extraction model [241]

Automatic approaches: In order to improve the efficiency and reduce manual efforts, most recent researches focus on automatic approaches instead of manual or semi-automatic ones. Some representative automatic approach is Omini [65]. Figure 2.5 presents the Omini system's architecture.

Some of these approaches perform only data record extraction but not data item extraction, such as Omini do not generate wrappers, i.e., they identify patterns and perform extraction for each Web page directly without using previously derived extraction rules. The techniques of these works have been discussed and compared in [65]. Note that all of them mainly depend on analysing the source code of Web pages. In addition, there are several works (DeLa [174], DEPTA, and the method in [355]) on data item extraction, which is a preparation step for holistic data annotation, i.e., assigning meaningful labels to data items. DeLa utilizes HTML tag information to construct regular expression wrapper and extract data items into a table. Similar to DeLa, DEPTA also operates on HTML tag tree structures to first align data items in a pair of data records that can be matched with certainty. The remaining data items are then incrementally added. However, both data alignment techniques are mainly based on HTML tag tree structures, not visual information. The automatic data alignment method in [355] proposes a clustering approach to perform

alignment based on five features of data items, including font of text. However, this approach is primarily text-based and tag-structure-based [335].

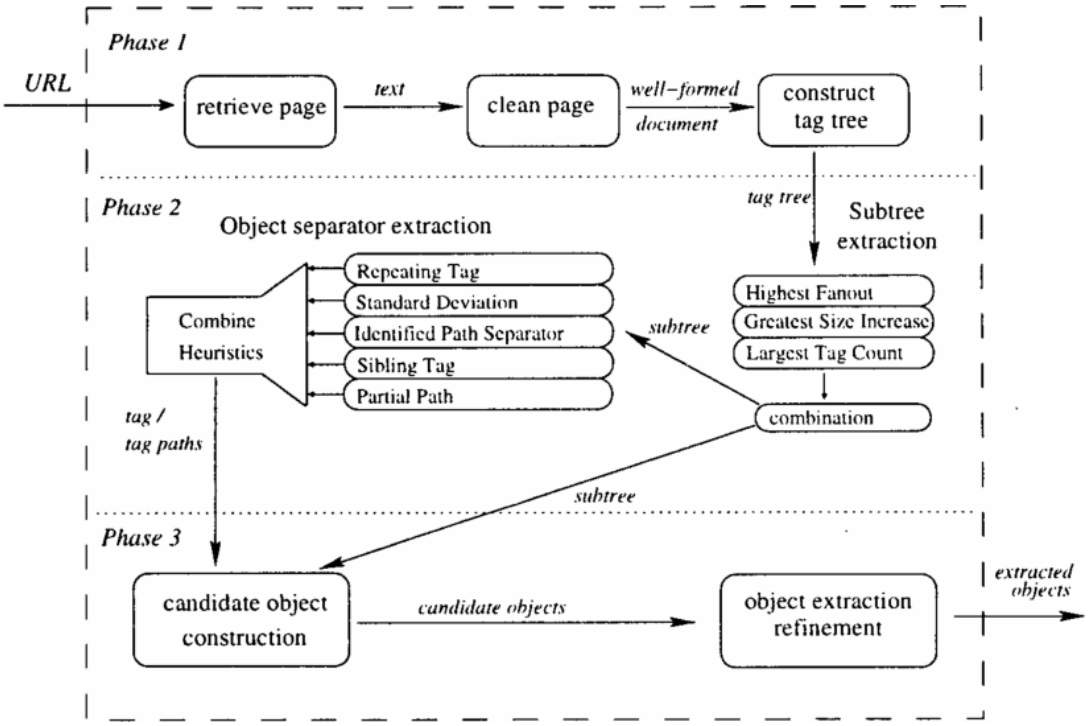


Figure 2.5. Omini system architecture. Source [65]

2.3.2 Data quality aspects

The aggregation of cyber-threat related information, which is gathered from many sources, is a challenging task since the vast volumes of heterogeneous collected data might not be complete or accurate. Therefore, the credibility of the primary source of information is an important aspect for determining the quality of the generated CTI. The quality can be attributed either to the credibility of the website/forum where such information resides, the particular user/member that posts the information, or both.

2.3.2.1 Source ranking factors

Important factors towards assessing the credibility of the website or forum where cyber-threat related information resides include the following.

Top websites ranking overall ranking: Some Dark Web sites, forums, blogs are visual to the Websites ranking engines such as Alexa.com and *SimilarWeb*. In these sites, ranking is based on a combination of metrics of page views and unique site users and creates a list of most popular websites based on this ranking (time-averaged over three-month periods). Only the site's highest-level domain is recorded, aggregating any subdomains. *SimilarWeb* ranks websites based on a panel of millions of Internet users, International/U.S. internet service providers, direct measurement of web traffic from data from thousands of websites and web crawlers scanning public websites.

Number of populations: If the site is a web forum, the number of registered users can indicate how big is the community that sits behind such sites. In which, this factor be used when assessing and comparing such Dark Web sites for the purpose of cyber-threat intelligence.

Number of posts per day: It quantifies the number of posts published per day in a social network site including blogs and forums shows how active such site is. For example:

- https://hackforums.net/index.phh
- http://offensivecommunity.net/

- https://sinister.ly/
- https://www.hellboundhackers.org/

However, accessing these sites might harm users' computer and steal their personal data.

Number of CVEs published per day: More than three-quarters of vulnerabilities are publicly reported online before *National Vulnerability Database* (NVD) publication [50]. News sites, blogs and social media pages as well as more remote areas of the web including the Dark Web, paste sites, and criminal forums first published bugs more often than NIST's centralised NVD. Data taken from [37] (beginning of 2016) and an analysis of more than 12,500 security bugs, showed that the median lag was seven days between a CVE being revealed to ultimately being published on the NIST's NVD. The number of CVEs that are daily, weekly and monthly published in those Dark Web sites can be used as an indicator to rank such malicious sits.

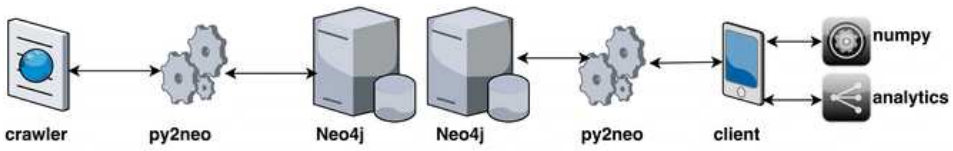
2.3.2.2 User-level influence ranking

Ranking account influence constitutes an important challenge in social media analysis. Until recently, influence ranking relied solely on the structural properties of the underlying social graph, in particular on connectivity patterns. Currently, there has been a notable shift to the next logical step where network functionality is considered, as online social media such as Reddit and Twitter are renowned primarily for their functionality. However, contrary to structural rankings, functional ones are bound to be network-specific since each social platform offers unique interaction possibilities [114].

Abbasi et al. [2] proposed a framework to identify expert hackers in web forums, based on content-mining. First, the authors represented each user with three categories of features: cybercriminal assets, specialty lexicons, and forum involvement. Then, they profiled the users into four groups based on their specialties: black market activists, founding members, technical enthusiasts, and average users. Analysing the interactions among hackers, they noted that average users (86% of the total) were participants that did not engage in the community enough, while the other three groups constituted the key-hackers.

Later, Zhang et al. [343] also used a content-mining approach in a hacker forum to analyse post orientations regarding knowledge transfer. Knowledge acquisition and knowledge provision were noted as the patterns to construct user profiles, classified by the authors into four ordinal types: guru, casual, learning, and novice hackers. They found that guru hackers act as key knowledgeable and respectable members in the communities, increasingly acting as knowledge providers.

In a sequence study, Fang et al. [371] developed a framework with a set of topic models for extracting popular topics, tracking topic evolution and identifying key-hackers with their specialties. Using Latent Dirichlet Allocation (LDA), Dynamic Topic Model (DTM) and Author Topic Model (ATM), they identified five major popular topics, trends related to new communication channels, and key-hackers in each expertise area.



(a) Twitter subgraph creation. (b) Analytics operation and data flow.

Figure 2.6. System architecture [114]

In [114], a twitter account influence metric is introduced utilising the system architecture presented in Figure 2.6. A general algebraic scheme for deriving higher-order Twitter influence rankings from first-order rankings was proposed. A convex combination of the score of a given account and those of its followers lies at its heart. The analysis indicates that among the seven higher-order metrics derived by the proposed scheme, namely *μC, μM, μE, μP, μA, μS, μF1*, and *μI* (see [114] for the details). The *μI* metric performs well compared to both *μZ* and *μR*, reflecting the fact that it combines structural and functional features, although it is among

the more demanding metrics. Another interesting finding was that μE, μM, and μC had similar behavior to μR. The fact that μI outperforms the other digital influence rankings can be attributed to the diverse factors it combines, which capture a major part of Twitter activity.

The activities in a Web forum are usually following a power law. That means a relatively fewer number of users are actively participating in the forum and leading the discussion in the community. On the other hand, a large number of users are relatively less active in their participations and mainly following the messages led by the forum leaders or the influential users. However, these influential users cannot be simply detected by the frequency of participations.

The proposed user rank algorithm was developed on the basis of PageRank which is an algorithm that is developed for ranking web pages. PageRank score is representing the authority conferring by the authors of other Web pages which is explicitly presented by the hyperlinks. If a Web page receives a hyperlink from another Web page with a high PageRank score, it is receiving authority from a high authoritative page. The higher authority a Web page received from other Web pages, the higher the ranking of this Web page is. In PageRank algorithm, every Web page is initialized with a small value and computed iteratively until convergence. PageRank has also been investigated to rank the expert level of users in forum settings. The interactions between users in a forum are represented as a social network. The nodes correspond to the actors participating in the forum and the directed edges are corresponding the interaction that one actor is offering an answer to the question raised by another actor in the forum. The experts are those who are offering advice to others. The more advice a user offers, especially offering to other expert users, the higher the PageRank score this user receives [42].

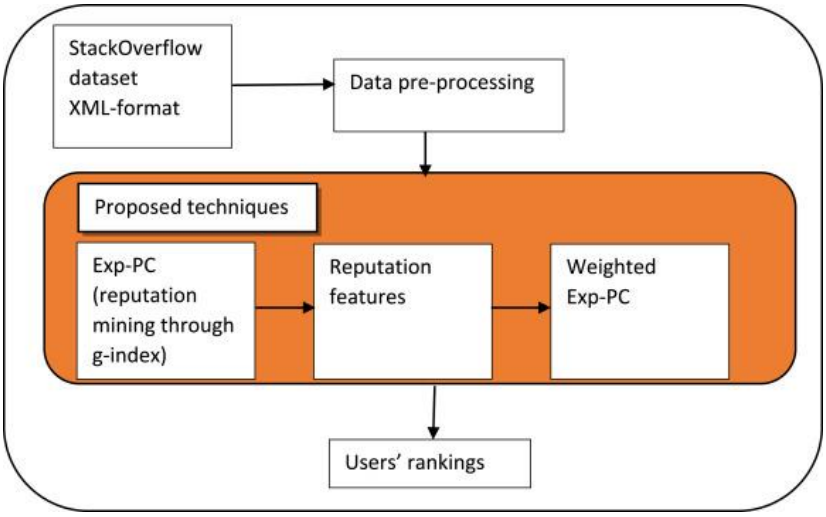


Figure 2.7. Architecture of proposed expert ranking system [225]

Another expert-ranking technique purposed by [225], consists of the application of *g*-index approach to various reputation features to identify experts on CQA sites for programming language problems, in this case SO-forum. In that work, expert-ranking techniques, i.e., Exp-PC and Weighted Exp-PC, are discussed in detail. A generic overview of the proposed work is given in Figure 2.7. First, the expert ranking problem is formulated as follows: Let RF be the rated-forum containing a set of questions Q = $\{q_1, q_2, \ldots, q_m\}$, where a question q_i has answers $A_i = \{a_{i1}, a_{i2}, \ldots, a_{in}\}$ by users U. Expert users from U need to be found whose performance is consistent and whose reputation is high. Performance consistency is measured through a bibliometric *g*-index. The reputation of the users is computed from their voter reputation, tag quality and participant's reputation.

2.3.3 Data pre-processing, storage and indexing

Cyber-Threat Intelligence (CTI) is gathered by crawling and searching a variety of sources (Surface, Deep and Dark Web). After the relevant web pages are collected, they are temporary stored for further processing. In the general case, the collected pages can be viewed as text files. This textual information should be further

processed in order to extract the relevant CTI information. This is a complex process that involves (see also Section 2.3.2)

- natural language processing,
- text classification,
- entity extraction and resolution and
- ontology integration and mapping.

In order to support the above operations, the collected data should be pre-processed. This involves:

- **Parsing documents into tokens**. It is useful to identify various classes of tokens, e.g., numbers, words, complex words, email addresses, so that they can be processed differently. In principle token classes depend on the specific application, but for most purposes it is adequate to use a predefined set of classes. This step is can be performed by a parser customizable for CTI needs.
- **Converting tokens into lexemes.** A lexeme is a string, just like a token, but it has been normalized so that different forms of the same word are made alike. For example, normalization almost always includes folding upper-case letters to lower-case, and often involves removal of suffixes (such as s or es in English). This allows searches to find variant forms of the same word, without tediously entering all the possible variants. Also, this step typically eliminates stop words, which are words that are so common that they are useless for searching. In short, then, tokens are raw fragments of the document text, while lexemes are words that are believed useful for indexing and searching. Typically, this step is performed using dictionaries.
- **Storing preprocessed documents optimized for searching.** For example, each document can be represented as a sorted array of normalized lexemes. Along with the lexemes it is often desirable to store positional information to use for proximity ranking, so that a document that contains a denser region of query words is assigned a higher rank than one with scattered query words.

Note, that in many cases, the textual information of the collected pages, is annotated with additional meta-data that can be incorporated to assist processing. Such annotations include special tags that are not displayed but can be used in machine parsing. For instance, it might include information about the author, the content, the type of the web page, but also classification and ontological information. Even presentational information may be helpful to assist content extraction. For instance, important information is commonly highlighted in special typesetting environments (e.g., using italics or bold typefaces).

2.3.3.1 Data indexing

Textual information is an eminent example of Big Data. To support (a) the effective storage and maintenance of the information (b) the efficient searching, several indexing structures where introduced. The most popular data structure used in document retrieval systems is the *inverted index*. It is commonly used in large scale paradigms like search engines.

An inverted index is a database structure that withstand a mapping from the content to its location in the dataset (see Figure 2.8). For instance, in the case of documents and textual information, an inverted index stores a mapping of the contents of the documents (e.g., words and numbers) to their corresponding locations. The purpose of an inverted index is to allow fast full-text searches, at a cost of increased processing when a document is added to the database.

Inverted index schemes can be combined with finite state transducers that is commonly used in NLP [201]. For other data type, e.g., to store meta-data information or other data, we can employ traditional indexing structures like B-trees, hash-based indexes, KD-trees offered by most relational database systems. The above indexes are incorporated and used in the text management systems presented in Section 2.3.3.3.

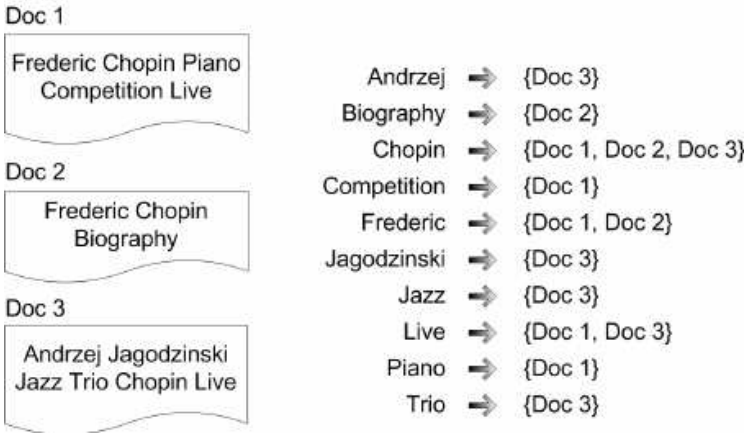


Figure 2.8. Inverted index example

2.3.3.2 Data storage

There is a plethora of available systems able to store and index textual information. Key functionalities of such systems include full-text search, hit highlighting, faceted search, real-time indexing, result clustering, database integration and rich document handing. In more details:

Full-text search refers to methods for searching the full content of stored documents. Full text search considers documents as first-class citizens (and not blog data). Thus, it differentiates from searches based on metadata or on parts of the document (such as authors, titles, abstracts, selected sections, or bibliographical references) that can be easily stored in a relational database system.

Hit highlighting is the searching feature that points out (usually by highlighting using different color or typeset) the words or phrases that helped match that document to the search.

Faceted search is a technique which enhance traditional search techniques with a faceted navigation system. This allows users to narrow down search results by applying multiple filters based on faceted classification of the searched items. Such classification system categorizes each information element along multiple explicit dimensions (termed facets) that enable the classifications to be accessed and ordered in multiple ways rather than in a single, predetermined, taxonomic order.

Real-time indexing is the ability of the system to adapt to fast document changes. Typically documents indexes are built in an off-line fashion and require time to adapt to document insertions, deletions and updates. Real-time indexing offers mechanisms that improve indexing restructuring and support the online maintenance of indexes.

Result clustering is another search feature that automatically discover groups of related search documents and assign human-readable labels to these groups.

Database integration refers to the ability of the text storing system to cope with common relational data. For instance, to be able to handle the meta-date of a document (e.g., authors, titles, abstracts, selected sections, or bibliographical references) or other structured information.

Rich document handing is the ability of the system to handle additional textual formats like XML, Word and PDF.

Free and open sources systems that support textual information include Apache Solr [35] that stems from the Apache Lucene project. Solr supports full-text search, hit highlighting, faceted search, real-time indexing, result clustering, database integration, NoSQL features and rich document handling. Additionally, Solr offers distributed search and index replication focusing on scalability and fault tolerance. It also has a very active development community and regular releases. Due to its features, Solr is widely used for enterprise search and analytics use cases.

BaseX [141] is an XML database management system supported by an XQuery processor and full test extensions. It is specialized in storing, querying, and visualizing XML documents and collections. Using BaseX is is possible to store, query and process large corpora of textual data (XML, JSON, CSV, many others). BaseX is developed as a community project on GitHub. BaseX is platform-independent and distributed under a permissive free software license.

Elasticsearch [145] is another search engine based on the Apache Lucene library. It provides a distributed, multitenant-capable full-text search engine with an HTTP web interface and schema-free JSON documents. Elasticsearch is developed in Java and is released as open source under the terms of the Apache License.

Searchdaimon [314] is an open source enterprise search engine for full text search of structured and unstructured data. Its major features include hit highlighting, faceted search, dynamic clustering, database integration, rich document (e.g., Word, PDF) handling and full document level security.

MongoDB is a cross-platform document-oriented database system. Classified as a NoSQL database, MongoDB supports JSON-like documents with schemata.

PostgreSQL is an is an open source object-relational database management system that also provides full text search capabilities [272].

Microsoft Azure Search [236] is a component of the Microsoft Azure Cloud Platform providing indexing and querying capabilities for data uploaded to Microsoft servers. Azure Search as a service framework is intended to provide developers with complex search capabilities for mobile and web development while hiding infrastructure requirements and search algorithm complexities.

SAP HANA [146] is an in-memory, column-oriented, relational database management system developed and marketed by SAP. It supports data storage and retrieval and advanced analytics (predictive analytics, spatial data processing, text analytics, text search, streaming analytics, graph data processing).

dtSearch [142] can instantly search terabytes across an Internet or Intranet site, network, desktop or mobile device. It also allows publishing, with instant text searching, large data collections to Web sites or portable media.

2.3.4 Data mining and data enrichment/leveraging

There are many different text mining approaches to consider, and they all depend on the source and the type of data that the crawler collects. Indicative approaches are next described where important features are highlighted.

2.3.4.1 Indicative approaches

TTPDrill. On the first case that we are exploring, TTPDrill [119], where the crawler collects CTI reports written in unstructured text, such as Symantec reports, and then maps them to attack patterns and techniques (TTPs) of a Threat-Action ontology, based on MITRE's CAPEC and ATT&CK threat repository. This was achieved by combining concepts of Natural Language Processing (NLP) and Information Retrieval (IR). To filter out which of the crawled pages contained useful information and discard advertisement, help, contact pages, etc., an SVM document classifier was built, using the following features:

- **Number of words**. Articles that contain TTPs are almost always longer because they communicate detailed descriptions about threats, their actions, and targets. Other articles, such as ads or news on security tools are considerably shorter.
- **SecurityAction-word density**. All verbs were extracted from various highly-reputable publicly-available information security standards, namely, ATT&CK, CAPEC, CWE, and CVE using the part-of-speech (POS) NLP technique. The Stanford Dependency Parser [234] was used to label verbs in these texts. Then, the SecurityAction-word density was calculated by computing the percentage of times these verbs appear in an article compared to the total number of words in it.
- **SecurityTarget-word density**. TTP-containing articles describe threats, therefore, they contain more security nouns (e.g., registry, vulnerability) than their counterparts of non-TTP containing articles.

Similar to the SecurityAction-word density, all noun phrases were extracted from the same information security standards and were used to calculate the SecurityTarget-word density, by computing the percentage of times the noun-phrase appears in an article compared to the total number of words in it.

This classification method works for free-text, but some special terms used in threat reports confuse NLP tools. For example, they have difficulty parsing the sentence: “Malware connects to 192.168.1.1” correctly because of the periods in the IP address. For that reason, a set of regular expressions for common objects in the ontology such as IP address, port number, domain, etc. was built. When a regex matches a string, it replaces it with a generic name to prepare it for further processing. For example, the string “fil_1.exe” in the sentence “create fil_1.exe” gets captured by a regex and replaced with the words “executable file”, so the sentence becomes “create executable file”, which is easy to parse with the NLP tools. Of course, the replaced strings don’t get discarded, they are kept for later use, (e.g., specific indicators at STIX threat reports). Also, synonyms such as “logs” and “records”, are checked using WordNet, Thesaurus, and Watson Synonym to improve the text understanding.

TTPDrill then has to identify candidate threat actions from the threat reports and map them to the ontology based on a similarity score. A threat action consists of a grammatical structure of Subject-Verb-Object (SVO), where the subject is the name of a malware, the verb is the action, and the object is the target of the action. The Stanford Dependency Parser is used to identify and extract such structures, which are then marked as the candidate threat actions.

To map each extracted candidate action in the ontology, TTPDrill measures the similarity between the candidate actions and known threat actions in the ontology by using the TF-IDF method with the enhanced BM25 weighting function. This function ranks the ontology entries based on their similarity to a given action extracted from a CTI report. Both the candidate threat actions and the known ones from the ontology are represented as a document of a “bag of words”. TTPDrill uses a cut-of threshold to eliminate threat actions with low similarity scores. Each candidate threat action is mapped to the threat action in the ontology with the highest score.

iACE. A data-mining task on another project called iACE [338] was to identify semantic elements (called Named Entity Recognition or NER) and extract relations between them (called Relation Extraction, or RE). The type of data that they wanted to collect were Indicators of Compromise or IOC.

Automatic collection of IOCs from natural-language texts is challenging. Simple approaches like finding IP, MD5 and other IOC like strings in an article, as today’s IOC providers do, do not work well in practice, which brings in false positives, mistaking non-IOCs for IOCs.

It was observed that the open intelligence sources that produce high-quality IOCs, describe IOCs in a simple and straightforward manner, using a fixed set of context terms (e.g., “download”, “attachment”, “registry”) which are related to the iocterms used in the OpenIOC format to label the types of IOCs [326]. Also, the grammatical connections between such terms and their corresponding IOCs are also quite stable. This allows iACE to leverage relations to identify an IOC and its context tokens, combining the NER and RE steps together.

iACE also utilizes a set of regular expressions and common context terms extracted from iocterms to locate the sentences within the articles that contain putative IOC tokens, such as IP, MD5-like string. Within each of the sentences, the approach attempts to establish a relation between the IOC token and the context term: it converts the sentence into a Dependency Graph (DG) to describe its grammatical structure and extracts the shortest paths linking each pair of a context token and the putative IOC token; over each path, a graph mining technique is applied to analyze the relation between the tokens, which cannot be handled by existing RE techniques, for the purpose of determining whether they indeed include an IOC and its context. This approach leverages the context and regex terms to locate potential IOC-carrying sentences and the predictable relations among them to capture IOCs, avoiding the complexity of direct application of existing NLP techniques, which needs to solve the NER first to identify an IOC token before addressing the RE problem to find its context.

2.3.4.2 *Source code classification - topic extraction*

Another type of data that can be extracted from Dark Web forums is source code. Understanding source code is a non-trivial task compared to free-text data. There are two sub-areas worth exploring, source code classification and source code topic extraction. The general strategy to classify code is to identify a set of target classes for classification (databases, games, communications, etc.) and develop a training set with sample source code from each of those classes [6], [89]. The training set typically contains features which are unique to the classes which they are representing. This particular strategy is useful when the programming language of the code being classified is the same. Many classifiers have been tested for this task, but Support Vector Machine (SVM) consistently has the highest performance [89], [69].
If the programming language of the source code is varied, a different approach must be followed. By collecting sample source code files in various programming languages to develop a feature set, a classifier can be trained to classify source code files into their appropriate languages. Once again, SVM classifiers appear to be the most effective in this type of source code classification [260].

However, classifying the programming language does not identify the function or purpose the code serves, unless it is already in a pre-defined category, which is rarely the case. If the source code is complete, it can be executed to identify its function or purpose. If the source code is incomplete, then topic modeling techniques must be used.

The primary method to extract topics from source code is *Latent Dirichlet Allocation* (LDA), also known as topic modeling. LDA is typically used to text documents, but it has been adapted for source code [260], [5], [312]. LDA is used in source code analysis when the target categories (i.e., games, databases, etc.) are unknown. To process source code postings for LDA, the identifiers found in the source code must be split to meaningful sub-words which can be better interpreted. For example, the identifier DATA_AUTH_RESPONSE is split into DATA, AUTH and RESPONSE. After some normalization (stemming, stop-words), LDA is run using the Mallet toolkit and it clusters the source code into topic categories. These topics are then manually labeled and tabulated. After the categories have been defined, the source code that has been classified by programming language, can be classified by topic as well.

2.4 CTI sharing

Cyber-security[7] is the practice of protecting systems, networks, and programs from digital attacks. These cyber-attacks usually aim at accessing, changing, or destroying sensitive information; extorting money from users; or interrupting normal business processes. Implementing effective cyber-security measures is particularly challenging today because of the vast number of internet-connected devices. In addition, attackers are becoming more resourceful. In response to the above, the cyber-security domain evolved substantial to suppress cyber-crime and prevent malicious attacks. As a result, CTI tools and dedicated repositories are created, to assist the security endeavors to advance cyber-security for the Internet of Things (IoT) ecosystem.

A number of datasets from various sources are stored and analyzed to produce meaningful CTI recommendations for the IoT devices. Furthermore, the expanding number of the devices and their evolving characteristics/capabilities, functionalities, interoperability with other devices and existing systems, connectivity, etc. is driving the IoT device repositories and their analysis systems to handle a vast volume of data and manage the intelligence sharing to IoT industry stakeholders.

Based on [94], CTI sharing is the process that enables the exchange of a variety of network and information security related information, such as risks, a) vulnerabilities, b) threats and c) internal security issues as well as d) good practices. IoT intelligence sharing has various objectives and factors. Based on [94], [95], [83], [48] these factors can be categorized as shown in Table 2.3.

[7] https://www.cisco.com/c/en/us/products/security/what-is-cybersecurity.html

Table 2.3. IoT intelligence sharing objectives and factors

Factor	Objectives
Security/Threat intelligence	• Improved security through shared best practices • Increased efficiencies in risk assessments • Empower decision-makers to reduce risks • Better defensive agility with shift from reactive to proactive strategies • Development of shared situational threat awareness • Enhanced threat understanding • Access to relevant, actionable intelligence
Economic factors	• Reduce remediation cost from Data Breaches • Cost savings through elimination of duplicated effort
Legal factors	• Network and Information Security (NIS) Directive[8] • Regulation (EU) 2016/679 – General Data Protection Regulation (GDPR)[9]
Generic factors	• Implementation of sharing models without new legislation • More rapid customer notification of potentially impactful incidents • Correlation of seemingly unrelated indicators • Improved, more timely decision making via shared, real-time information • Identification of trends and indicators of compromise

Using CTI is highly beneficial for advancing the security standards of an organization. The security policy and associated tools, techniques, and management strategy will become more effective if CTI is integrated with the organization's security operations. Having the relevant information available in a timely manner, will enable security personnel and incident response teams to determine which alerts or identified incidents are active threats, required further investigation, and priorities the response actions.

Especially, the use of real time or near real time CTI provides a clear and comprehensive picture of active threats and ongoing security incidents. In the case of new technologies like IoT this will assist security teams to take immediate actions and implement mitigation plans in response to real time intelligence. Other benefits of having access to real time CTI in the IoT ecosystem include:

- Immediate insight into threats concerning the IoT devices and the associated risks that might impact the IoT connected devices.
- Provision of a holistic view of the IoT security that will assist security personnel to decide which vulnerabilities should be addressed first – and how should be addressed.
- Active monitoring of social media and online channels for mentions on IoT devices.
- Active intelligence for identifying and preventing IoT related security breaches.
- Gaining insight into how vulnerable specific IoT devices and equipment is.
- The ability to track the ongoing activities of cyber-criminals and hacktivists specific to IoT industry.
- Access to information needed for IoT devices risk management.

[8] https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=uriserv:OJ.L_.2016.194.01.0001.01.ENG&toc=OJ:L:2016:194:TOC

[9] https://eur-lex.europa.eu/legal-content/EN/ALL/?uri=CELEX%3A32016R0679

- Monitoring of online communications channels for evidence of current cyber-criminal activity, or intended campaigns pertaining to IoT ecosystem.

The Information Sharing Actors is a cornerstone in CTI concerning the IoT ecosystem. The information sharing actors are the enablers of the communication flow between various IoT stakeholders. The flow of information is more than essential as it is the primary source of generating intelligence in IoT. Following the social networks paradigm, IoT-centric social networks would allow sharing of devices between users that would provide useful information captured by sensor devices or giving ways to make remote actions on user devices. Information Sharing Actors pertaining the IoT ecosystem are:

- IoT device owners
- Critical and non-critical infrastructure owners
- IoT device manufacturers and relevant standardization bodies
- IoT specialized security Firms
- Security Researchers

CTI sharing in the framework of Cyber-Trust allows information gathered through various sources to be shared (e.g., exploits, cyber-attacks, attackers' methodologies, etc.). Gathering, sharing and storing that information are processes that are linked with Cyber-Trust platform's architecture. The threat sharing services of the platform have been thoroughly explained in the deliverable D2.2 [61], where it was concluded that the Cyber-Trust platform will be mainly based on the MISP platform by following a detailed research analysis. Such information is shared through Cyber-Trust platform via the *enriched vulnerability database* (eVDB) sharing services.

2.4.1 Sharing architectures

CTI sharing communities follow two main sharing architecture designs; the centralized or the decentralized (P2P). Frequently, these are combined, resulting in a third sharing architecture; a hybrid solution of the aforementioned. In this subsection, the three types of CTI sharing architectures are presented, alongside with their advantages and disadvantages.

2.4.1.1 Centralized architecture

In this model, there is a central unit acting as an information repository, receiving the information from peripheral units, such as participating members or other sources. Information encompassed in this central unit is distributed to all interested community members, either directly (forwarded-as-received), or processed-and-delivered. In the latter, the information processing might involve procedures such as:

- the amalgamation of information that is derived from different sources, by correlating similar incidents,
- the normalization of information,
- the enrichment with supplementary information, by taking under consideration additional context, such as the qualitative and credibility aspects of the sources.

The benefits of using a centralized architecture mainly depend on the services and features provided by the central unit of the CTI sharing community. For example, a central unit processes the information in an aggregating manner and hence is able to enhance its content. In addition to that, centralized architectures that use open standards for the data structuring, and transport protocols, minimize the need for the community to adapt to different formats and protocols in order to exchange CTI. Moreover, by adopting these common data formats and transport protocols, community members can easily automate procedures which structure and then forward the information respectively. Finally, due to the form of the centralized architecture, community members do not have to maintain many connections, by the time they have established a connection with the central unit, in order to share CTI.

On the other hand, one disadvantage of following the centralized sharing paradigm, is that all community's CTI exchanging procedures rely on the proper functioning of the central unit. Hence, in the events of central unit malfunctioning or high network congestion all community members are affected. Finally, a known central unit that serves the purposes of CTI sharing within a community, draws the attention of attackers.

2.4.1.2 Decentralized (P2P) architecture

Participating members of a decentralized CTI sharing community, exchange information directly with each other. So, in this case, in order to enhance the information, rather than simply delivering it, each participant should act as a central unit (as described previously).

Although the tasks of each participant seem to get more complex than in the centralized paradigm, P2P sharing architecture comes with great benefits. First of all, by following the standard P2P model, it allows CTI to be shared in a fast manner amongst community members. In addition to that, due to the autonomy provided by decentralized architectures, to each participant, it is easier to choose the CTI sources, on an individual level. Finally, this architectural paradigm provides great malleability, since the information is exchanged through various channels, thus multiplying the points of failure, and hence minimizing the possibility of a total community CTI sharing failure due to a system's malfunction or a potential attack. However, a P2P architecture has the following drawbacks:

- P2P sharing communities that do not adopt common data formats and transport protocols, have difficulty scaling, since each individual community member should adapt to all existing structures followed by the rest.
- In high scales, decentralized architectures tend to become less efficient, since for a number of new peers added to the community, the amount of connections required, and hence the effort needed to establish trust, between all new members and the rest and exchange information, grows exponentially.

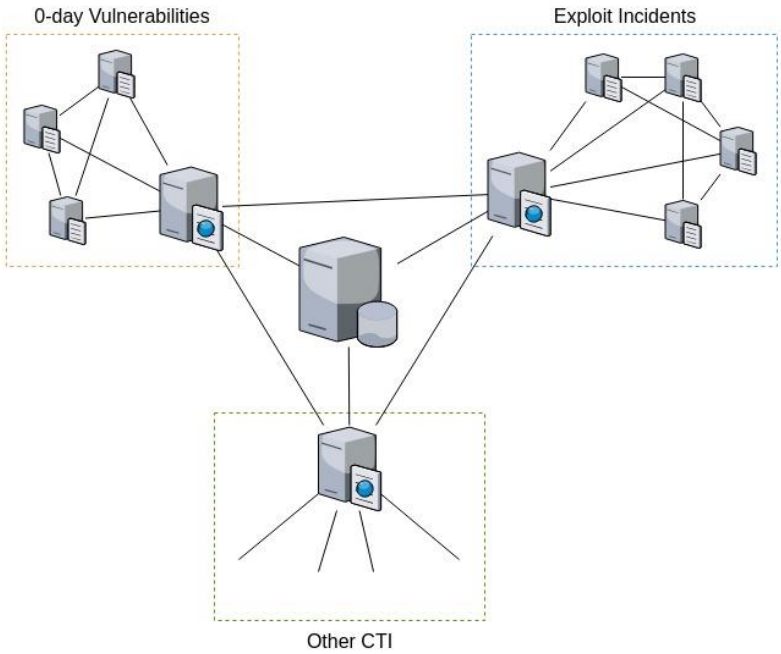


Figure 2.9. A use case of a hybrid CTI sharing architecture

2.4.1.3 Hybrid architecture

By using hybrid architectures, communities exploit benefits from both centralized and P2P sharing architectures, in order to combine them for a better outcome. In a community that follows the hybrid architecture for example, one could see the grouping of CTI sharing (by different topics), via smaller intra-

communities following the P2P sharing architecture, where each community member is not only connected with each other but is also connected with the intra-community's central unit. In addition to that, all these intra-communities' central units might as well be connected with each other and finally connect to the entire community's central unit (as shown in Figure 2.9).

In this case, the intra-communities take advantage of the decentralized architecture's rapid distribution of information, sharing low-level CTI amongst them. Simultaneously, information delivered to the intra-community's central unit, can be processed, enriched and delivered to the rest intra-communities, and hence eliminate the burden of these procedures being executed on an individual level. Nevertheless, following a hybrid solution can increase the operating cost of connections management, and makes the implementation more challenging.

2.4.2 Tools and platforms

In Cyber-Trust project, information sharing will be achieved by employing the *Malware Information Sharing Platform* (MISP) platform, which is a mature and flexible solution. To this extend, Table 2.4 presents various other platforms that have similar functionality with MISP (see deliverable D2.2 [61] for more details). It is important to highlight that MISP can be extended and tailored based on end-user's needs.

Table 2.4. Cyber-threat sharing platforms overview [61]

Platforms	Standards	Highlights
MISP	STIX/TAXII, OPENIoC, others	MISP platform is fully organized and the range of individuals that could utilize it could be developers or even simple users, providing materials for stand-alone learning, is very flexible and extended, automation supported. The information in the database can be extended by external sources while its functionality can be extended by integrating with third-party tools; It Is both human and machine-readable making correlations between observables and attributes. Exceptional characteristic consists the series of data models created by MISP community.
GOSINT	STIX, OPENIoC, CYBOX, IODEF, IDMEF	GOSINT has an organized repository a managing system and exporting data. Also can be extend as source (database) by external sources (URL, TEXT, ADHOC). Platform has community that applying research from third parties to your event data to identify similar, or identical, indicators of malicious behavior, automation supported. It is both human and machine readable.
OPENTPX	STIX	OPENTPX has an organized repository, is very flexible and extended, automation supported, we can extend the threat observables and make them as complex as you want for this reason we define them as loose (extension of data capabilities). provides a comprehensive threat-scoring framework that allows security analysts, threat researchers, network security operations and incident responders to make relevant threat mitigation decisions straight forward, while efficiently automating those decisions (threat scoring framework).
YETI	STIX/TAXII	YETI has an organized repository, is very flexible and extended, automation supported. Extended as source(database). It Is both human and machine readable. Yeti, goals is to turn it into a self-sustainable project, where not only the core developers but the whole community helps out when the community needs help (they don't have achieve it yet). The communication handled centrally for this reason only in GitHub, GitHub issues for all communication.

OPENTAXII	STIX/TAXII, CAPEC, IODEF, IDMEF, others	It is assumed that OPENTAXII has an organized repository and managing system, also can mimic already known cases and threats. It is flexible and extendable since it is providing machine-readable threat intelligence, possibility of layer extension, source intelligent extension and APIs extension provides automation.
CIF	STIX	CIF has an organized repository and managing system and exporting data. Provides combination of malicious threats and utilize that information for identification (incident response), detection (IDS) and mitigation (null route). Extended as source (database) indicators of malicious behavior, automation supported, human machine readable. It Is both human and machine readable.

MISP supports a vast number of repositories of cyber-threat information, adhering to many frameworks and standards for formatting and disseminating CTI information towards Cyber-Trust's components. Such components include the Profiling Service (A17), the cyber-defense service (A04), and the devices of the smart-home network (A03g, A11, A05g, A08g, A04g) [62]. The interconnection of systems with MISP can be achieved using a variety of protocols, modules, tools and plugins, the selection of which depends on the system to be utilized (e.g., file sharing over a Windows network could be done with the SMB protocol[10]), *application programming interfaces* (APIs). MISP platform possesses sharing modules as well as, the python library PyMISP, which is developed by CIRCL, that allows easy access to APIs. There are three types of MISP modules[11,12]:

- Import modules: Import new data into MISP.
- Export modules: Export existing data from MISP.
- Expansion modules: Enrich the data which found in MISP, such as hover type (shows the expanded values directly on the attributes) and expansion type (shows the adding the expanded values via a proposal form).

Within the Cyber-Trust platform, the information gathered are being disseminated towards users and devices as described, below:

- The information of MISP's sources are updated with the information gathered from the crawling service (A10) and stored in the eVDB (A07) to enhance its current information; moreover, such information is matched against the device profiles in the profile repository (A17) so as to notify the users through the pub/sub service (A09) depending on their preferences.
- The information stored in the eVDB is also disseminated to the smart home gateway IDS system allowing to Cyber-Trust subsystems to recalculate the risk of communicating with nearby devices (note that MISP standard core format supports Snort, Suricata, and Bro – now renamed to Zeek).

[10] https://www.misp-project.org/galaxy.html

[11] https://github.com/MISP/misp-modules/tree/master/misp_modules/modules

[12] https://www.circl.lu/assets/files/misp-training/switch2016/2-misp-modules.pdf

2.4.3 Exchange formats

The *structured threat information expression* (STIX)[13] and the *trusted automated exchange of intelligence information* (TAXII)[14] constitute MISP's main CTI formats [61]; they cooperate excellently with each other as they are both developed by the OASIS Cyber-Threat Intelligence Technical Committee (CTITC). Below, we will be more detailed about project's primary standards for formatting. STIX is a serial language for describing and exchanging CTI in a way that can be stored and analyzed in a consistent manner. TAXII is a data-exchange mechanism and has two primary services (collection & channel) to support a variety CTI sharing models – preferably STIX data.

- Collection is an interface to a logical repository of CTI objects provided by a TAXII Server that allows a producer to host a set of CTI data that can be requested by consumers: TAXII Clients and Servers exchange information in a request-response model
- Channel is maintained by a TAXII Server and allows producers to push data to many consumers and consumers to receive data from many producers: TAXII Clients exchange information with other TAXII Clients in a publish-subscribe (pub/sub) model.

Other CTI sharing formats include the following:

- *Cyber-Observable eXpression (CybOX)*[15] is a standardized language for encoding and communicating high-fidelity information about cyber-observables, whether dynamic events or stateful measures that are observable in the operational cyber-domain.
- *Open Indicators of Compromise (OpenIoC)* is a standardized language that communicate forensic information (e.g., on describing malware artifacts, indicators and attacker TTPs). It is suitable for descriptions of technical characteristics that identify a known threat, an attacker's methodology, or other evidence of compromise [97].
- *Incident Object Description Exchange Format (IODEF)* defines the framework for exchanging operational and statistical security incident information. The data model allows the encoding of information about hosts, networks, services, attacks methodologies and forensic data. IODEF was designed for information exchange between CERTs.
- *Common Vulnerabilities Reporting Framework (CVRF)* is a data exchange format designed to support the automation of software vulnerability data reporting and consumption. A CVRF document describes the whole vulnerability handling lifecycle, from the discovery of the vulnerability to shipping a patched version of vulnerable software [61].
- *Open Vulnerability and Assessment Language (OVAL)* is a language for specifying automated tests of system configurations and defines the format for the results of such assessments. Vendors include OVAL specifications in vulnerability advisories to share information about vulnerabilities and misconfigurations in a machine-readable format. OVAL may also be used to distribute descriptions of threat indicators [61].

2.4.4 Information sharing quality

A number of challenges that exist for production and consumption of high-quality cyber-threat information are the following (derived from [247]):

- *Establishing trust.* Trust relationships create the basics for reliable information sharing, but their maintenance requires significant effort.

[13] https://oasis-open.github.io/cti-documentation/stix/intro

[14] https://oasis-open.github.io/cti-documentation/taxii/intro

[15] https://cybox.mitre.org/about/

- *Achieving interoperability and automation.* The use of standardized data formats is an important building block for interoperability because it allows organizations and repositories to easily exchange threat information in a standardized way. However, the adoption of specific formats may require significant time and resources.
- *Securing sensitive information.* Publishing sensitive information such as controlled unclassified data and personally identifiable information may result in violating sharing agreements, and loss of reputation, or even financial loss. Organizations should implement policies and technical controls to manage the risks of disclosure of sensitive data.
- *Enabling information consumption and publication.* All the organizations that are willing to publish and consume threat intelligence information should have the necessary tools and well-trained personnel to do so. Organizations unable to support automated indicator exchange formats for best practices should either use manual exchange or summary indicator information.

There are more challenges related to circumstances that need to be tackled where an organization should use CTI. For example, how to access external sources and incorporate actionable CTI, how to estimate the quality of the received CTI and how to provide CTI (e.g., how to comply with policies or requirements pertaining to privacy and the limitation of attribution) [247].

The growth of cyber-threat has increased the likelihood that signals of impending attacks will be visible in the open public data sources. Cyber-attackers exploit vulnerabilities using tools, techniques, and tradecraft. Therefore, to conduct an attack, malicious actors typically have to:

- Identify vulnerabilities
- Acquire the necessary expertise and tools to use them
- Choose targets
- Recruit participants
- Plan and execute the attack.

Other actors—system administrators, security analysts, and even victims— may discuss vulnerabilities, threats, or coordinate defences against exploits. These discussions are often conducted in online forums, including blogs and social media networks, thereby creating potential signals to identify an upcoming attack or a new vulnerability [28]. However, such sources vary in their activeness, population, content volume, language. These aspects are related to the quality of the CTI shared, and the issues that have already been discussed in section 2.3.2, also apply in this setting.

2.4.5 Privacy-preservation issues and techniques

2.4.5.1 Overview of privacy policies

A privacy policy identifies the ways a party collects, stores, manages, uses and discloses information about individuals. The focus is to fulfill the legal requirement to protect personal information and ensure the privacy of individuals. Personal information can be anything that can potentially identify an individual, expanding but not limited to: name, address, date of birth, family status, contact information, ID information and personal records, e.g., financial, medical, travel, shopping. A privacy policy informs individuals about the specific information that is collected. Moreover, it specifies if the collected information is confidentially stored, if it is shared with (external or internal) partners and if it is sold to others.

The specific content of a privacy policy depends on the legal jurisdictions. Most countries have their own legislation and guidelines. In European Union (EU) data protection laws apply to government operations,

private enterprises and commercial transactions and cover all European citizens. The 2018 reform of EU's data protection rules is commonly referenced as the General Data Protection Regulation (GDPR)[16].

GDPR is EU's regulation law on data protection and privacy for all individuals within the European Union (EU) and the European Economic Area (EEA). It also addresses the export of personal data outside the EU and EEA areas. The GDPR simplifies the regulatory environment for international business by unifying the regulation within the EU. Its focus is to provide to individuals control over their personal data. To this end, processes handling data should consider GDPR principles in order to provide data privacy (for instance, using pseudonymization or full anonymization techniques). The main requirements of GDPR include:

- Data protection by design and by default (Article 25).
- Right to access
- Right to erasure

Enforcing GDPR is essentially achieved using pseudonymization or full anonymization techniques [113, 300]. This is a challenging research and engineering task that highly depends on type and the volume of the involved data. In the following, we will review some popular anonymization techniques for a variety of data types.

2.4.5.2 Anonymization methods and techniques

The first and obvious set towards data privacy protection is to explicitly remove anything that can disclose the identity of an individual (e.g., name, social security number). Unfortunately, even if we maintain datasets without disclosing user's unique identifiers, the potential risks for revealing the identity of users are not dismissed. For instance, a malicious user may hold partial knowledge about an individual (taken for instance from publicly available datasets like census datasets) and use it over a set of attributes, called quasi identifiers (QI), to identify the person's record. This type of privacy breach is called *identity disclosure*. Another type of privacy breach, called *attribute disclosure* or *attribute linkage* happens when an attacker can link a person to some of the persons sensitive information (e.g., income, HIV results).

Relational datasets. A plethora of models have been proposed in the context of protecting relational datasets from identity and attribute disclosure. Sweeney [204] showed that identity disclosure can be performed by linking the released dataset to external datasets, containing identifying information, based on demographics and also proposed the k-anonymity principle to prevent this threat. k-anonymity requires that there are at least k records sharing the same set of demographics. Thus, the probability that a user is re-identified, when an attacker knows all demographics in the patient's record, is limited by 1/k. This group of at least k similar records is called an equivalence class. Most of the methods protecting from identity disclosure are based on generalization and suppression. This hides specific details from the set of quasi identifiers and protects the dataset.

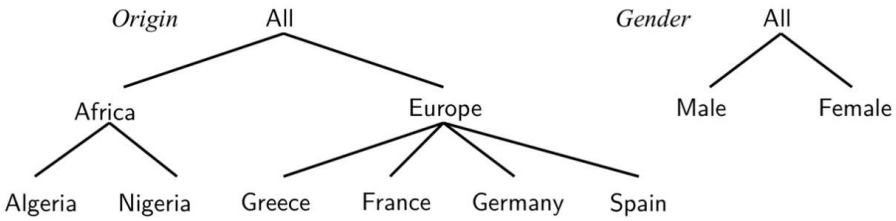


Figure 2.10. Hierarchy trees and generalization

The way that generalization is performed depends on attributes type. The values of a relational attribute can either be categorical (like gender) or numerical (like age). Generalizing a categorical value, simply replaces this value with a more general but semantically close one, using a taxonomy (e.g., a hierarchy tree). For

[16] https://ec.europa.eu/commission/priorities/justice-and-fundamental-rights/data-protection/2018-reform-eu-data-protection-rules_en

instance, according to the hierarchy tree of Figure 2.10, country Greece can be generalized to Europe or All. Similarly, a numerical value can be replaced with an interval having a number of consecutive values, where the original value is one of them. For instance, the original value of age=21 can be replaced by [19:22]. Other methods to protect a published dataset employ suppression which simply removes the value from the dataset, replacing it with a special character or number, denoting that this value was suppressed. Suppression can be viewed as a form of generalization, where each value is replaced by a value All (i.e., a person can have any value in an attribute). Overall, generalization is more general than suppression and typically incurs lower information loss.

Generalization can be employed using various schemes. These schemes are categorized as follows:

- Global recoding schemes, where, when an instance of a value is generalized to a more general value, all its instances are generalized to the same value.
- Local recoding schemes, where an instance of a value can be generalized, while other instances can remain original or generalized in different levels of the hierarchy.

Full-domain generalization [186, 266, 204] is a global recoding scheme, where the same level of a taxonomy tree is used for all the values of an attribute. For example, using the hierarchies of Figure 2.10, if Origins' value Algeria is generalized to Africa, then all Origin's values are generalized to the continent level (i.e., Africa or Europe). This scheme has the smallest search space for a possible solution, as an algorithm has to climb the hierarchy tree level by level. On the negative side, this scheme entails great distortion of data.

Examples of full domain generalization include Incognito [186], MinGen [204] and Binary Search [266]. Incognito [186] generates all possible k-anonymous full-domain generalizations, and chooses the one which creates the most equivalence classes. MinGen [204] searches the entire solution space, examining every full-domain generalization in order to find the optimal k-anonymous solution. This makes MinGen impractical even for medium size datasets. Finally, Binary Search [266] on the other hand initially finds the minimal generalizations that derive to an anonymous datasets and then choose the best generalization based on a penalty defined over every generalization or suppression.

Another global recoding scheme is *subtree generalization* [291, 41, 40]. Under this scheme, at a non leaf node, either all values are generalized to their parent, or none of them becomes generalized. In more detail, assuming the hierarchy of Figure 2.10, if an instance of Algeria is generalized to Africa, then also every instance of Nigeria has to be generalized to Africa as well, but instances of Greece, France, Germany and Spain may remain intact. This scheme has an increased search space for a solution, but reduces the distortion of the data compared to full domain generalization. In this context, Bayardo et al. proposed K-Optimize [291] that uses a sets enumeration tree to effectively prune the search space. Each node of the tree contains a possible solution. The algorithm examines every solution in the tree according to a top-down manner and discards a node if it's children cannot be a global optimum solution. Fung et al. also proposed a method employing subtree generalization, namely Top-Down Specialization (TDS) [41, 40]. TDS initially generalizes all values to their most general value, and then applies specialization. Specialization is the opposite function of generalization, traversing the hierarchy tree from top to bottom specializing certain values. If a specialization violates the k-anonymity principle is discarded. TDS evaluates every specialization in terms of information gain and privacy loss. Finally, Iyengar et al. proposed Genetic, which encodes each state of generalization as a "chromosome" and uses a fitness function to encode the amount of generalization. Then, it searches for the fittest chromosome using genetic evolution functions. Iyengar's experiments show that it can effectively preservce utility, but with a higher computational cost, making it impractical for large datasets.

Multidimensional generalization is a local recoding scheme that enables two groups of records, having the same value on a quasi-identifier, to be generalized independently into different levels of the taxonomy. For example, let as assume two groups of records. The first group has two records, both sharing the same origin Nigeria. The second group has also two records, having origin values Nigeria and Algeria respectively. The first group is already 2-anonymous with respect to attribute origin, and thus no generalization is applied. On the other hand, in the second group, we generalize both origin values to Africa in order to make it 2-anonymous. This scheme enables us to generalize only the groups of records that violate the anonymity

principle, producing less distorted data than the full-domain and subtree generalization schemes. For instance, LeFevre et al. proposed Mondrian [187], a greedy top-down specialization algorithm, using a local recoding scheme. Mondrian employs specialization separately at each group (i.e., equivalence class), as it examines the possible specializations of a group, and continues if the respective specialized subgroups has sizes at least k. Mondrian does not use a hierarchy tree, but generalizes each attribute according to it's values inside the group it belongs.

The above described methods, although sufficient to protect from identity disclosure, cannot protect from attribute disclosure. Stronger models have been proposed in this context, like l-diversity [18] and t-closeness [243]. l-diversity guarantees that inside a group of records there at least l distinct values of the sensitive attribute. Thus, the probability of linking a person to a sensitive attribute is at most 1/l. To satisfy this principle, Machanavajjhala et al. proposed l-Diversity Incognito [18]. This algorithm considers the sensitive values of an attribute and satisfies the l-diversity principle by employing bottom-up generalization. The intuition is that larger groups can be diverse, as they will contain more and potentially different values of the sensitive attribute. Mondrian was also extended by LeFevre et al., in order to satisfy l-diversity. This version of Mondrian is called InfoGain Mondrian [188]. The idea behind InfoGain Mondrian is similar to l-Diversity Incognito. Larger groups are more probable to be diverse, thus, specialization can result in a fast and accurate solution. Using a different approach, Xiao et al. proposed Anatomy [341] that does not publish a sanitized dataset, but instead it separates the records from their sensitive attributes, releasing two tables. The first table holds the quasi identifiers in a non-generalized (original) form, while the second holds sensitive attribute values. These two tables share a common attribute, called groupId. Through this attribute a group of records is assigned to a group of l-diverse sensitive values. This function of separating several attributes from records and releasing it as a different table is formally known as disassociation.

t-closeness [243] is another method protecting from attribute disclosure, overcoming the inability of l-diversity to sufficiently protect datasets holding non-uniformly distributed sensitive attributes. For example, let us assume a dataset holding the HIV results of users, where the 99% of the results are negative. While the probability of a user having HIV inside the original dataset is only 1%, inside a 2-diverse anonymous group holding two records, the probability will be 50%. t-closeness [243] addresses this issue, by requiring that the distribution of a sensitive attribute inside an anonymous group will be close to the overall distribution of the attribute in the dataset (i.e., the maximum allowable distance between the two distribution is t). The authors also presented t-Closeness Incognito [243], a version of Incognito that measures the distance of the sensitive attribute distribution inside an anonymous group and inside the whole dataset, in order to find a solution satisfying the t-closeness principle.

Transaction datasets. Privacy-preserving transaction data publishing requires different privacy models and algorithms, due to the high dimensionality and sparsity of transaction data [116, 231, 364]. Also, it is difficult to enforce k-anonymity in transaction datasets, as identifying the set of quasi identifiers is a controversial task [231]. To address this issue, various principles have been employed, making different assumptions about attackers' knowledge. Complete k-anonymity [109] protects from attackers holding all the items of a person. This principle requires that for any person in the dataset, there are at least k−1 other persons sharing the same items. The downside of complete k-anonymity is that it considers a stringent setting where attackers know all the items of a person. Higher utility may be achieved by exploiting the fact that only certain combinations of items are potentially linkable and can be used as quasi identifiers. This is the basis of a principle called privacy constrained anonymity [11, 121]. Each combination of potentially linkable items is called a privacy constraint p, where the set of all privacy constraints formulates a privacy constraint set P. P is satisfied if for every privacy constraint p in P (a) there are at least k users supporting p, or (b) there are no users supporting p and for every combination of the items in p there are either 0 or k users supporting it. This principle achieves better utility than complete k-anonymity, but, it requires from data publishers to define the set of privacy constraints, in order to anonymize a dataset. To balance the previous principles, Terrovitis et al. [231] proposed km-anonymity. This principle requires that for any subset of m items of a person, there are at least k−1 other persons with the same m items. Thus, limiting the attackers maximum knowledge for a person, increases the data utility, as fewer generalizations are used to anonymize a dataset. km-anonymity makes a realistic assumption about the attackers knowledge.

To protect transaction datasets, several methods have been proposed based on generalization [11, 121, 231], suppression [363, 364], bucketization [115] or disassociation [230].

Methods based on generalization. Terrovitis et al. [231] presented the Apriori algorithm to enforce km-anonymity. Apriori works in a bottom-up fashion, initially identifying and fixing problematic itemsets of size 1 and moving up to incrementally larger problematic itemsets. In each iteration, the proposed algorithm uses a full-subtree, global generalization model. The same authors also proposed two partitioning methods, namely Vertical Partitioning Anonymization (VPA) and Local Recoding Anonymization (LRA) respectively [231]. VPA partitions the items domain into sets and then generalizes the items inside each set. Next, VPA merges the generalized items in order to make them km-anonymous. LRA on the other hand, uses a horizontal partitioning scheme, in order to create clusters holding semantically close items. Then, LRA anonymizes separately each cluster, using the Apriori algorithm, in a local recoding generalization scheme. Loukides et al. presented COAT [121] and Gkoulalas-Divanis et al. PCTA [11]. Both of these methods guarantee the privacy constrained anonymity principle described earlier. These methods assume that different data publishers have different privacy requirements, modeled as privacy constraints. Also, they assume that specific item combinations have to be preserved, in order to make anonymous data suitable for specific analysis. These combinations of items are modeled as utility constraints. COAT and PCTA employ generalization to satisfy the set of privacy and utility constraints. If certain items cannot satisfy their respective utility constraints, they are suppressed from the dataset. PCTA outperforms COAT, as it constructs generalized items “on the fly” to achieve km-anonymity with lower information loss [11]. In more detail, PCTA considers singleton itemsets, each corresponding to an item, which are then merged to create clusters that represent generalized items. Cluster merging is performed in a hierarchical fashion, until the dataset, constructed by replacing each item in the original dataset with its corresponding cluster, satisfies km-anonymity.

Methods based on suppression. In addition to generalization, privacy can be achieved using suppression [364, 363]. For example, Xu et al. [364] applies suppression focusing on preserving the frequent itemsets. Authors define “borders”, which hold the frequent itemsets and the itemsets violating the privacy principle. Then they apply the minimum amount of suppression, in order to eliminate problematic itemsets and preserve as many frequent itemsets as possible.

Methods based on bucketization. Bucketization focus on creating groups of similar records. For instance, Ghinita et al. [115] proposed a method, bucketizing similar records, using approximate nearest-neighbour (NN) search in high-dimensional spaces. Additionally, they also proposed two data transformations that capture the correlation between records, (a) reduction to a band matrix and (b) Gray encoding-based sorting. Then, the records inside each bucket are anonymized using generalization and perturbation.

Methods based on disassociation. Terrovitis et. al also proposed a method enforcing km-anonymity, based on disassociation [230]. In more detail, instead of eliminating identifying information by not publishing many original terms, either by suppressing or generalizing them, the authors partition the records so that the coexistence of certain terms in a record is obscured. Their proposed transformation partitions the original records into smaller and disassociated subrecords. Their objective is to hide infrequent term combinations in the original records by scattering terms in disassociated subrecords. Although the aforementioned problems tackle identity disclosure, they do not guarantee protection against attribute disclosure. To thwart both identity and attribute disclosure in transaction data publishing, [231] proposes lm-diversity. lm-diversity guarantees that any person cannot be associated with less than l sensitive values. In more detail, lm-diversity practically leads to the creation of equivalence classes, where each set of m quasi identifiers is associated with at least l different sensitive values.

Trajectories anonymization. k-anonymity has been considered in the context of publishing user trajectories, leading to several trajectory anonymization methods [103]. These methods operate by anonymizing either entire trajectories [249, 248, 244], or parts of trajectories (i.e., sequences of locations) that may lead to identity disclosure [229, 290].

Clustering and perturbation. Methods based on clustering and perturbation are applied to time-stamped trajectories. They operate by grouping original trajectories into clusters (cylindrical tubes) of at least k

trajectories, in a way that each trajectory within a cluster becomes indistinguishable from the other trajectories in the cluster. One such method, called NWA [249], introduces the anonymity principle of (k, δ)-anonymity, to anonymize user trajectories. This principle generates cylindrical volumes of radius δ, and guarantees that every cylindrical volume contains at least k trajectories. Each trajectory that belongs to an anonymity group (cylinder), generated by NWA, is protected from identity disclosure, due to the other trajectories that appear in the same group. To produce the cylindrical volumes, the algorithm in [249] identifies trajectories that lie close to each other in time and employs space translation.

Trujillo-Rasua and Domingo-Ferrer [289] performed a rigorous analysis of the (k,δ)-anonymity model, which shows that this model is not able to hide an original trajectory within a set of k-indistinguishable, anonymized trajectories. Thus, the algorithms in [249, 248] may not provide meaningful privacy guarantees, in practice. An effective algorithm for enforcing k-anonymity on trajectory data was recently proposed by Domingo-Ferrer et al. [157]. The algorithm, called SwapLocations, creates trajectory clusters using microaggregation and then permutes the locations in each cluster to enforce privacy. The experimental evaluation in [157] demonstrates that SwapLocations is significantly more effective at preserving data utility than NWA [249].

Finally, Lin et al. [73] guarantees k-anonymity of published data, under the assumption that road-network data are public information. Their method uses clustering-based anonymization, protecting from identity disclosure.

Generalization and suppression. This category of methods considers attackers with background knowledge on ordered sequences of places of interest (POIs) visited by specific individuals. Terrovitis et al. [229] proposed an approach to prohibit multiple attackers, each knowing a different set of POIs, from associating these POIs to fewer than k individuals in the published dataset. To achieve this, the authors developed a suppression-based method that aims at removing the least number of POIs from user trajectories, so that the remaining trajectories are k-anonymous with respect to the knowledge of each attacker.

Yarovoy et al. [290] proposed a k-anonymity based approach for publishing user trajectories by considering time as a quasi-identifier and supporting privacy personalization. Unlike previous approaches that assumed that all users share a common quasi-identifier, [290] assumes that each user has a different set of POIs and times requiring protection, thereby enabling each trajectory to be protected differently. This approach uses generalization and creates anonymization groups, of size at least k, that are not necessarily disjoint, in order to protect users' data.

A recent approach proposed by Monreale et al. [19] extends the l-diversity principle to trajectories by assuming that each location is either nonsensitive (acting as a QI) or sensitive. This approach then proposes an anonymity principle, called c-safety to prevent attackers from linking sensitive locations to trajectories. In more details, c-safety guarantees that an attacker cannot link a sensitive locations to a user trajectory with a probability greater than c, while a user's trajectory is indistinguishable among at least k − 1 other trajectories. To enforce c-safety, the proposed algorithm applies generalization to replace original POIs with generalized ones based on a location taxonomy. If generalization alone cannot enforce c-safety, suppression is used.

Assuming that each record in a dataset is comprised of a user's trajectory and user's sensitive attributes, Chen et al. [278] propose the (K, C)L-privacy model. This model assumes that an attacker can know at most L locations of a user's trajectory. (K, C)L-privacy guarantees that a user is indistinguishable from at least K−1 other users, while the probability of linking a user to its sensitive values is at most C. Authors satisfy (K,C)L-privacy by employing local suppression.

Differential privacy. Differential privacy is a privacy model, which ensures that the presence or absence of information about an individual in a dataset does not significantly affect the outcome of analysis applied to the dataset [44]. Thus, an attacker cannot learn "more" about an individual, from what the attacker already knows, because all inferences that the attacker can make about an individual will be (approximately) independent of whether the individual's information is contained in the dataset or not. Proposed algorithms offering differential privacy are using either interactive or non-interactive approaches. Interactive approaches enable data miners to issue count queries and get noisy results of the original data through a private mechanism. In non-interactive approaches, data owners release a sanitized version of the original dataset for public use.

A method to publish noisy summaries of the original dataset, offering differential privacy, is proposed by Bonomi et al. [196]. This method uses a prefix-tree to generate candidate patterns, used in the construction of the data summary. In more detail, this method considers the statistical properties of the dataset in order to construct a model-based prefix tree. This tree is used to mine prefixes and a candidate set of substring patterns. Next, the frequency of the substring patterns is further refined by transforming the original data to reduce the perturbation noise. Another method, DiffPart [280 initially creates clusters of semantically close records. It then applies specialization inside each cluster. Finally, noise is added to the clusters in order to produce differentially private count statistics. The authors state that the specialization step increases the cell counts and, thus, less noise has to be added. Privelet [342] proposes a wavelet-transformation-based approach that lowers the magnitude of noise needed to ensure differential privacy. Privelet initially applies a wavelet transform on the data and then it adds Laplace noise to each wavelet coefficient in order to ensure differential privacy. Hay et al. [212] proposed a method to publish differentially private histograms for single-dimensional datasets. The proposed approach takes as input a set of count queries and it then exploits consistency constraints that should hold over the noisy output. Thus, the noisy output can provide more accurate answers on the aforementioned count queries. On a similar context, SHARE [178] takes as input a dataset, a privacy budget defined by the data owner, and outputs aggregated statistics (e.g., means, histograms) with differential privacy guarantees. SHARE operates on interactive mode, thus it does not release an anonymous version of the original dataset. Chen et al. proposed NGrams [279], a method generating summaries that permit highly accurate count query answering. This method works in three steps. First, it truncates the original trajectory dataset by keeping only the first lmax locations of each trajectory, where lmax is a parameter specified by data publishers. Larger lmax values improve efficiency but deteriorate the quality of the frequencies, calculated during the next step. Second, it uses the truncated dataset to compute the frequency of n-grams (i.e., all possible contiguous parts of trajectories that are comprised of 1, 2, ... , *n* locations). Third, this method constructs a differentially private summary by inserting calibrated Laplace noise to the frequencies of n-grams. Mohammed et al. [245] presented a generalization-based algorithm for differentially private data release, supporting datasets holding both relational and transaction attributes. This method initially generalizes the data in order to create groups of records with the same generalized values. Then, it adds noise to these groups and publishes their noisy counts.

2.4.5.3 Compliance checking and monitoring

In general, compliance monitoring observes business process execution and reports violations of specific laws, regulations or contracts. Several commercial products exist that focus on compliance with information security regulation and standards such as ISO 27001 [7] and Sarbens-Oxley [267]. However, to our knowledge there are no equivalent products focused on CTI sharing.

2.5 Recommendations

The review on the state-of-the-art regarding the collection, identification, mining, leveraging, and sharing of cyber-threat intelligence has revealed a number of tools and technologies related to the aforementioned tasks, but has also helped us identify issues and challenges that are involved in the design of key technologies to be used for pre-reconnaissance cyber-threat intelligence. In the following we summarize our key findings, outline interesting research directions, and discuss possible solutions.

Regarding the collection and identification of cyber-threat intelligence from the clear, social, deep, and dark web, there exist a number of technical challenges involving the architectural decisions, the identification and access to relevant content, and the efficient traversal of the web graph. To this end the most promising research avenue regarding the crawling for content lies in the exploitation of topic-specific crawlers coupled with best-first strategies that are able to identify related, high-quality links. Technical issues regarding access to content, freshness of crawled information, and duplicate elimination are also interesting research directions, especially in the context of the hidden (deep/dark) web. Finally, architectural recommendations for the task at hand typically aim at centralized or hybrid solutions that balance effectiveness and efficiency.

Data management issues that were identified during the literature review include source identification and ranking, data manipulation (usually pre-processing and storage), and data mining/data enrichment.

Interesting research directions in this topic include named entity recognition techniques that are able to identify different types of actionable intelligence within unstructured textual content, topic extraction for the classification of both textual content and code segments, and efficient storage solutions that may include the latest advances in NoSQL datastores. Recommendations in this topic include the utilization of user-level data for the ranking of sources, the employment of machine-learning algorithms for text processing and concept identification/leveraging, and the exploration of NoSQL solutions to tackle efficiency issues.

Finally, in the context of cyber-threat intelligence sharing, our findings are in-line with those already outlined in previous deliverables. MISP is the dominating platform in the field, presenting properties such as maturity and extensibility that make it ideal for the Cyber-Trust ecosystem, while STIX/TAXII are the de facto CTI exchange formats that are also seamlessly adopted by MISP. Interesting research directions in this area lie mainly in privacy-related issues that are raised during the sharing of cyber-threat intelligence.

3. Trust establishment and risk assessment

3.1 Introduction

In the context of computing, parties interact with each other to access services and information. Traditionally, access control mechanisms are employed to safeguard such accesses: authentication mechanisms provide the necessary guarantees about the identities of the interacting parties (i.e., that either the service/information requestor or the server are indeed who they claim they are), whereas authorization mechanisms enforce information/service access policies, ensuring that only authorized clients can access the information/service resources provided by the servers. While this approach is adequate for a number of information system use cases, and predominantly in client-server systems where a closed set of clients or client groups interact with a limited set of servers that are known *a priori*, modern internet-scale computing necessitates the interaction between unknown parties, with each party being able both to request and offer services and/or information. In such an environment, traditional access control mechanisms face considerable limitations, since interacting parties are highly likely to be unknown to each other before the beginning of the interaction. In this respect, a different approach is needed to allow interacting parties to decide:

1. Whether the requestor is entitled to access the service/information requested and
2. Whether the provider is trusted as a source of the particular service/information.

The concept of Trust management has emerged as a solution to this issue. Trust management is defined in [208] as an aid the automated verification of actions against security policies. According to this definition, an action is allowed if the credentials presented are deemed sufficient, without the need to state or verify the actual identity of the interacting party; in this respect, symbolic representation of trust is separated from the actual person (or the person's digital agent). Later research has replaced the examination of credentials (which could be considered as *pseudonymized identities*, limiting hence the benefits of introducing trust management [55]) to the examination of a *set of properties*, which can be proven by an interacting party through the presentation of a set of digital certificates [258, 185, 242, 170, 325]. Under this scheme, the original set of trust management system elements identified in [55] is modified as follows:

1. *Security policies*, which are a set of trust assertions that are considered "ground truth" and are trusted unconditionally;
2. *Trust-related properties*, which represent aspects of interacting parties that are relevant to the application of security policies; typically, such properties are examined as antecedents of rules that comprise a security policy. Trust-related policies are safeguarded through digital signatures or other prominent means.
3. *Trust relationships*, which are a special kind security policy.

While the scheme presented above explicitly lists two interacting parties, i.e., the service/information requestor and server, trust establishment may involve more parties, resulting in a highly decentralized model: firstly, trust-related properties may be (and typically are) provided and testified for by third parties. Secondly, *trust relationships* may designate other trust management system entities with which a trust management system instance liaises to exchange any of the system elements listed above (security policies, trust-related properties or trust relationships), including also trust assessments that can be taken into account when a trust management system instance assesses the trust level of an interacting party.

The computation of the trust level for an interaction peer may involve all observable aspects for this peer: this spans across (a) the *security aspects of the interaction peer*, including the current assessment of the peer's integrity status (known tampering of firmware, operating system, critical files; patching level; etc.) and the security controls that are known to be in place for the device (firewalls; IDS/IPS; and so forth [52]) and (b) *behavioral aspects of the interaction peer*, mainly focused on whether the interaction peer (i) operates according to is predefined usage description and (ii) deviates from its normal behavior.

Services, information and resources that need to be protected through trust management or other relevant approaches are ultimately *assets*, and each asset has a *value* for its owner. Furthermore, assets are exposed to a number of threats, as detailed in D2.1 [52]; each such threat constitutes a risk for demotion of the respective assets' value. To this end, effective asset protection entails the assessment of the risk posed by each interaction and deciding on the defensive actions that possibly need to be taken on the grounds of this assessment. This is in line with the procedure described in the ISO/IEC 27001 standard [180] for addressing risks, which encompasses the steps of:

(a) information security risk assessment, which is further refined in (i) establishment and maintenance of information security risk criteria that include the risk acceptance criteria (ii) identification of information risks and (iii) analysis of information security risks and (iv) evaluation of information security risks and

(b) information security risk treatment, where (i) appropriate information security risk treatment options are selected, taking account of the risk assessment results, (ii) controls that are necessary to implement the information security risk treatment are determined and (iii) the information security risk treatment plan is validated, including the acceptance of the residual information security risks.

Trust and risk assessment are closely linked, since analysis of information security risks entails the assessment of the realistic likelihood that the identified risks will occur [180], and this probability in turn depends on the level of trust placed on systems that could potentially be threat agents: this is reflected on the definition of trust listed in [287] according to which "Trust is the willingness of a party to be vulnerable to the action of another party based on the expectation that the other will perform a particular action important to the trustor, irrespective to the ability to monitor or control that other party"; similarly [43] defines trust as "An attitude of confident expectation in an online situation of risk that one's vulnerabilities will not be exploited", therefore we can conclude that trust moderates the level of risk, through the belief that a trusted system is not bound to effectively function as a threat agent. Under this viewpoint, the trust assessment for a system is a critical parameter to be taken into account when performing risk assessment.

Finally, contemporary attack methods entail complex multi-stage, multi-host attack paths, where each path represents a chain of exploits used by an attacker to break into a network [339]. Attack graphs enable the comprehensive risk analysis within a network, considering cause-consequence relationships between different network states; furthermore, the likelihood that such relationships would be exploited can be also taken into account [246].

The Cyber-Trust project will employ a trust- and risk-based approach to security, within which trust establishment and risk assessment will encompass all the above listed aspects, providing thus a holistic trust management and risk assessment view. In the following subsections we firstly overview the underpinnings of trust and risk management, and proceed with reviewing trust management models and trust management systems, with the focus on the latter being on open source implementations. Afterwards, we survey existing approaches to reaction modelling and computation, since reaction computation and application is an integral part of a trust-based security approach, and finally we provide recommendations for the implementation tasks of work package 5.

3.2 Underpinnings of trust and risk management

In this section we will overview the three main foundations of trust and risk management identified in subsection 3.1, namely (a) *behavioral-based methods*, focusing on the observed interactions of the devices, (b) *status-based methods*, focusing on the devices' security aspects and (c) *risk assessment-oriented methods*, focusing on the quantification of the risk associated with the devices and operations. For each of the three foundations, we present methods, tools and information sources that can be employed for realizing trust and risk management in the relevant context.

3.2.1 Behavioral aspects

The behavior of a device can be monitored and used in the process of trust and risk assessment. The term "behavior" in this context refers to the observable activities performed by the device, and this predominantly includes network traffic directed towards other nodes. This network traffic can be:

- *Compared against a predefined static model of behavior* that has been specified for the device and prescribes the operation of a benign instance of the device. Deviations from the prescribed behavior are then treated as indications of malicious behavior and demote the trust level, increasing correspondingly the risk level. Manufacturer Usage Description Specification files [88] are the main tool in this area.
- *Compared against a dynamically built model of behavior for the device*; under this approach, the behavior of the device instance is profiled at a state that is known to be benign, and further behavior is compared against the baseline within the profile. Deviations from the baseline are flagged as anomalies, reducing the trust level and increasing the associated risk. Provisions for dynamic evolution of the profile can be made.
- *Matched against a known set of malicious requests*. Under this approach, the network traffic emanating from the device is matched against a malicious requests signature database, to identify whether the device is the source of attacks to other devices; if so, it can be concluded that the device has been compromised, and consequently trust and risk assessments are adjusted accordingly.

Another aspect that can be taken into account at this point concerns the observable consequences of information flows, rather than the information flows themselves. Under this viewpoint, information that has leaked from a device (e.g., user passwords or personal data) constitutes evidence that the device does not provide an adequate level of security (including the case that it discloses information to entities that should not be trusted), and on these grounds the trust level to this device is reduced.

In the following paragraphs, we will elaborate on the aforementioned approaches to using behavioral aspects in trust and risk assessment.

3.2.1.1 Manufacturer usage description specification

Manufacturer usage description (MUD) specification is an activity towards designating intended (and therefore accepted) behavior for devices that are not meant to be used for general purpose computing tasks, but rather perform a specific, limited set of operations [88]; this is contrasted to general-purpose computers and laptops, the behavior of which spans across a wide spectrum of activities and associated protocols, which cannot be efficiently described and controlled through static specifications. The following important aspects are stressed in [88], which should be taken into account throughout the context of the use of MUDs:

1. The notion of "manufacturer" does not necessarily refer to the entity that physically produces the device, but is used as a reference to an entity or entity or organization that will state how a device is intended to be used. This may extend to include an integrator, or any other link within the supply chain of the particular device that will assume the task of informing the network about the proper usage of the device (through the compilation of the MUD specification).
2. The MUD is intended to help addressing threats to the device, however its use within the scope "device as a threat" is limited, i.e., it is not designed to assist in tackling threats from compromised devices. In the latter case, MUD could offer some level of protection depending on the MUD-URL is communicated, and how devices and their communications are authenticated.
3. MUD use is more effective when the endpoints that the particular device will communicate with are known and specified. This indicates that it is not possible to exploit the benefits of MUD to their full extent by simply using a generic MUD template that is provided by a manufacturer: the MUD template, when provided, should be tailored to the particular installation of the product, to match the security policy that applies to the network. This requirement, combined with the fact that generic MUD templates are not readily available from device manufacturers, considerably limits the

applicability of MUD solutions. As pointed out in [198], "*the promising MUD standard must be finalized as a draft document, and manufacturers of IoT devices and routers must then embrace the standard. If that happens, though issues will always remain, enterprises will have all the advantages of an IoT that is significantly safer and more secure*". Hence, at this stage, MUD file exploitation for leveraging security can mostly be accommodated in design specifications, rather than as an operational solution. Principles however from the MUD approach can be adopted in the Cyber-Trust solution.

Figure 3.1 presents a usage scenario where a new IOT device (termed "IOT Thing") is installed within the premises of a network [259]. The device communicates the MUD URI embedded into its firmware/operating system to the network's authentication, authorization, and accounting (AAA) server; in this scenario, the MUD is embedded in the device's X509 certificate, and thus originates from the device itself, although different methods can be used for determining the MUD URI (e.g., it can be securely specified by the network administrator through the DHCP protocol). Then, the AAA server, after validating the identity of the request, extracts the MUD URI and passes it to a newly introduced component, the MUD controller, which is responsible for fetching the MUD file from the appropriate repository. The MUD controller then extracts the ACLs contained within the MUD file, and delivers them to the AAA server, which arranges for installing these ACLs onto the network access control devices (e.g., firewalls, IPSs) within the enterprise network.

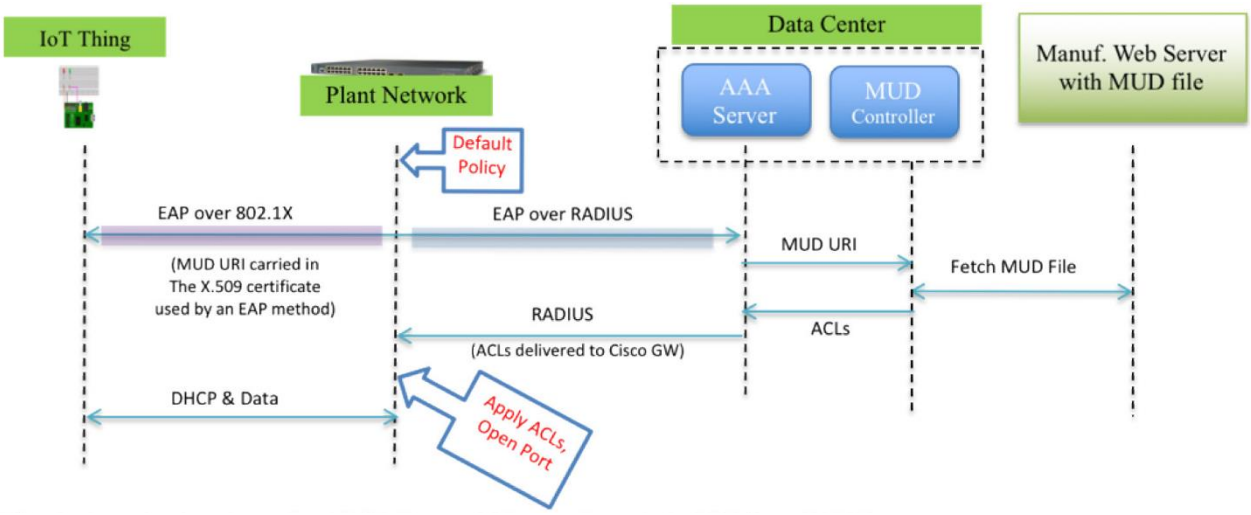


Figure 3.1. MUD usage scenario

ACLs in MUDs are resembling to ACLs used in firewalls, albeit they used different vendor-neutral syntax. ACLs are structured in two parts, with the first part regulating the traffic emerging from the device (*from-device-policy*) and the second part controlling the traffic directed towards the device (*to-device-policy*). Each ACL element may be applied either to IPv4 or IPv6 traffic and contains a matching condition and an action; the condition may specify source and destination identity elements (DNS name, IP address and port). Actions may be either "accept" or "drop", and provisionally "reject" can be specified, which may be interpreted as "drop".

The primary usage of MUD ACLs is to be converted to network defense appliance-specific ACLs, such as firewall-specific ACLs or IPS-specific ACLs, and be installed on respective network defense appliances within the network architecture, allowing thus for effective network traffic regulation. The work in [13] has demonstrated that the use of MUDs towards this end can provide effective protection for IoT devices, rendering the compromising of IoT devices a non-trivial matter for attackers. However, MUDs can also provide the basis for "golden standard" for measuring device-specific traffic flow compliance: the traffic portion of the traffic addressed to/emerging from the device that matches rules having "accept" actions is compliant to the intended use of the device, whereas traffic addressed to/emerging from the device that matches rules having "drop"/"reject" actions is non-compliant to the intended use of the device. The ratio

$$\mathrm{ct_r} = \frac{\text{compliant traffic}}{\text{total traffic}}$$

can then be used in the computations of trust and risk levels for the device, with trust being an increasing function of ct_r and risk being a decreasing function of ct_r. The device's outgoing traffic can be examined separately and be given higher importance, since non-compliant emanating traffic may be an indication of a compromise (followed by information leak or by launching of attacks to other devices). Finally, special characteristics of the traffic, such as source/destination addresses and ports, protocol and payload information can be also taken into account besides the network data volume: for instance let us consider the case that a real-time camera is compromised and used for launching attacks to other devices: even if the ratio of the compliant outgoing traffic (volume of the video stream) to the total traffic (size of payload of attacks plus the volume of the video stream) is high, clearly the compliance metric for the device is low. The nominal data flow volume per protocol or port or data type can be used in the calculation of trust / risk levels to normalize the magnitudes of different data flow volumes and provide more accurate measures of compliance.

Usage of MUDs involves low impact threats for personal data protection. Traffic is monitored only regarding its flows, and even in the case that protocol rules compliance are verified by the defense mechanisms, the payload of the traffic need not (and should not) be examined. Computation and collection of statistical measures however about data flows may entail threats for personal data protection, since these statistical measures may reflect habits or time patterns followed by the user; for instance, control commands sent to a smart lightbulb may provide indications on the hours during which the user is present in a room, or control commands sent to a smart lock on a door may uncover patterns in the user's entry/exit hours. In this respect, statistical measures should be appropriately safeguarded.

3.2.1.2 Dynamic behavior model-based approaches

Dynamic behavior model-based approaches create behavior baselines that represent normal behavior of users, hosts, or network connections. Subsequently, anomaly detectors process traffic and characterize activity as "legitimate" or not, by detecting deviations from baseline behavior which is defined as "normal"; deviant behaviors indicate potential anomalies [17]. The whole operation cycle of dynamic behavior model-based approaches involves three stages (a) *parametrization*, in which the observed instances of the target system are captured and modelled (b) the *training stage*, where normal (or abnormal) behaviors of the system are tagged and the required models are created and (c) *detection stage*: having the model for the system at hand, this model is compared with the extracted features of the observed traffic and a deviation metric is computed. If the deviation metric meets some predefined threshold, an alarm is flagged [92]. Figure 3.2 depicts the workflow for dynamic behavior model-based approaches.

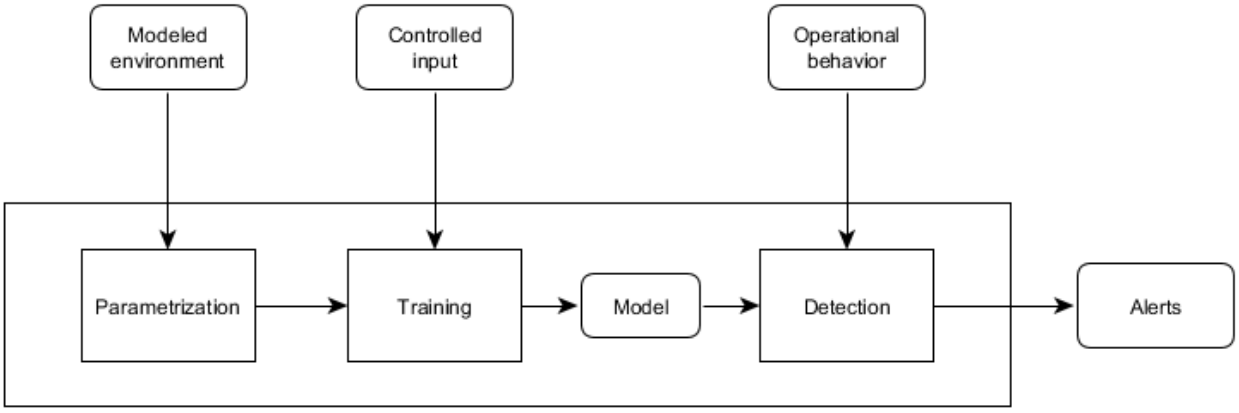


Figure 3.2. Workflow for dynamic behavior model-based approaches

Dynamic behavior model-based approaches offer several advantages. Firstly, they are able to detect attacks that are unknown, contrary to the approach of signature-based scanning, where some properties of the attack need to be known a-priori. Secondly, they are able to detect attacks coming from within the system: through the monitoring of the whole spectrum of activities within the target system and comparing it against the relevant baselines, they are able to detect attacks launched from any point within the system. This can be contrasted with the firewall and IPS techniques where checks are made only at the target system ingress points, offering thus high levels of protection from outsider attacks but being inefficient at protecting from attacks launched from inside the target system perimeter, either due to a deliberate insider attack or due to systems being compromised as an effect of Trojan [98], watering hole [98], [96] or other malware infection. Thirdly, baseline behaviors are dynamic and internal to the system, intruders cannot formulate an evasion strategy, since they are not aware of the patterns that will be considered as intrusion signs by the anomaly detection mechanisms [17].

On the negative side, dynamic behavior model-based approaches need a period of operation to gather adequate data to build profiles, a phase termed as the training stage [130], [24]; during this period no effective protection can be offered, while it should be also guaranteed that the operation is attack-free, or at least attacks should be limited to a minimum. Additionally, they are prone to the creation of false positives, i.e., misclassifications of unusual, but otherwise legitimate behavior, as attack-related: this is due to the fact that unusual events deviate from the normal behavior, and this is the sole criterion exercised by dynamic behavior model-based approaches to flag activities as normal or suspicious [17].

According to [130], the methods used for detecting deviant behaviors are classified in three major categories, namely *statistical-based, knowledge-based* and *machine-learning based*; each of these categories can be further refined into subcategories depending on the techniques employed.

In statistical based methods, the entity activities are captured and used to create a model representing the stochastic behavior of the entity. Features considered in this process include connection rate, traffic rate, per-protocol data volumes and number of packets, number of different IP addresses involved in the communications and so forth. The trained model data are then compared against the current model data, with the latter being determined according to the recent entity activity; if the comparison indicates significant deviation (i.e., a deviation surpassing a certain threshold), then an anomaly is flagged.

Methods falling in the knowledge-based category employ a set of specifications that perform classification of captured behavior traces into “legitimate” and “suspicious”/”malicious”. Specifications can take the form of rules in the expert systems subcategory, whereas for FSM and description logic-based subcategories other tools can be used, including UML and N-grammars. For expert systems in particular, the rules may be crafted manually by an expert, or (semi-)automatically through the analysis of training behavior, extraction of features and classes and –finally- derivation of rules. Similarly, for FSMs and description languages, expert intervention is required. Once the specification set has been compiled, it can be used to classify observed activities and raise alerts where appropriate.

Finally, machine learning-based methods employ models according to which the analyzed behavioral patterns are classified as “legitimate” and “suspicious”/”malicious”. The models are constructed during the training phase, according to behavioral data that has been labeled; this requirement makes the model construction phase resource consuming. While the machine learning-based methods are based on features, similarly to those of the statistical methods, machine learning-based methods can accommodate self-evolution, in the sense that the method adjusts its decision making criteria according to the stream of behavioral events captured and processed. Apache Spot [34] is a newly emerged system that employs machine learning techniques for anomaly detection within network traffic. Figure 3.3 illustrates a classification scheme for methods used to identify deviant behaviors.

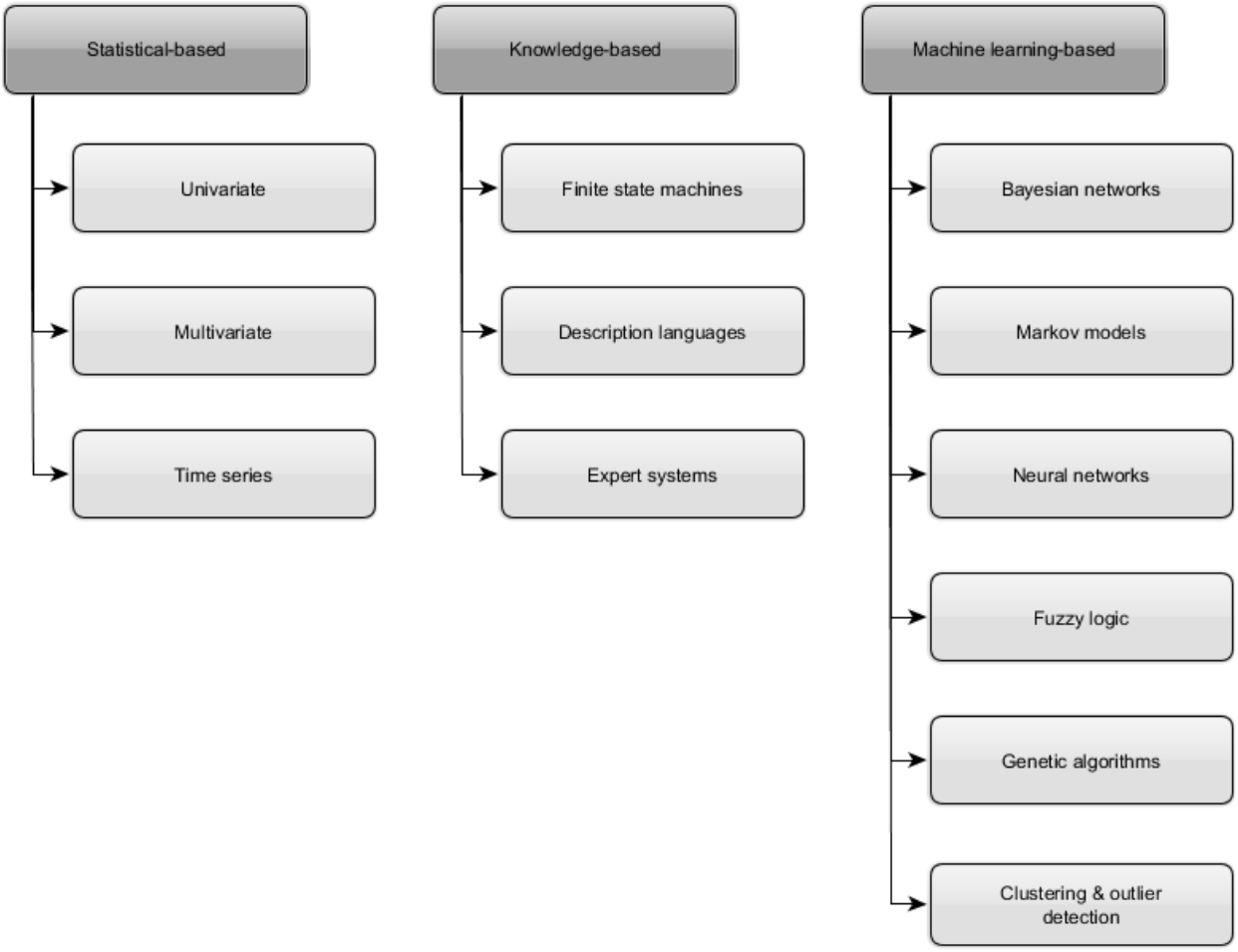


Figure 3.3. Classification of methods used for identifying deviant behaviors

The results of deviant behavior identification can be directly used in trust and risk computation. Entities for which deviant behaviors are identified will have their trust scores demoted and the associated risk assessments being raised. The magnitude of the trust and risk score adjustments will depend on (a) the number of deviations observed either as an absolute number or as a ratio against the overall number of observed behaviors and (b) the type of recorded deviations, in the sense that e.g., deviations in incoming traffic may designate an attack in progress, while deviations in outgoing traffic may more strongly designate that some attack has been successful.

Usage of dynamic behavior model-based approaches entails low impact threats for personal data protection. Traffic is monitored only regarding its flows, and payloads are not examined: this is true even for the cases that the defense mechanisms examine whether packet flows are compliant with protocol rules. Approaches however that collect and process statistical data regarding the traffic flows, pose the same threats to user privacy as the corresponding case in the context of MUD use: the statistical measures may reflect habits or time patterns followed by the user. For example, control commands sent to a smart lightbulb may provide indications on the hours during which the user is present in a room, or control commands sent to a smart lock on a door may uncover patterns in the user's entry/exit hours. In this respect, statistical measures should be appropriately safeguarded.

3.2.1.3 Signature-based scanning

The most common scanning technique used for detecting malicious activities in a network or a single device is *signature-based scanning.* Each signature is a unique feature of a file, a payload or a pattern in general, also known as the fingerprint [370]. Signature-based scanning systems use a predefined set of signatures or rules and scan files and network traffic against them, and if a match is found, an attack is flagged and required

measures are taken. Signature-based scanning can be used to scan for malicious activities from the narrow scope of a single device to the wider scope of a network. The single-device scanning is usually done from an anti-virus program to search for internal threats and a firewall or intrusion detection system (IDS) for detecting external threats. IDSs are widely used in networks monitoring, two prime examples are Snort and Suricata [60]. When any device sends data inside the network, this data first passes from a dedicated server running the IDS, where the packet is decoded and checked against the corresponding ruleset. If the packet is determined to be part of an attack (i.e., it matches a known attach signature) it is discarded, and the sender marked as malicious/compromised; if not, the packet is forwarded to its destination.

Main advantages of signature-based scanning include the fact that because of the uniqueness of each signature the false positive rate is negligible, provided that signatures are crafted in enough detail to match only packets that are actually part of attacks. Furthermore, it is very simple to put into production, as opposed to dynamic behavior-based scanning models which need a training period as discussed in Section 3.2.1.2. On the other hand, signature-based scanning has a number of disadvantages. The most important disadvantage is the fact that signature-based scanning is not able to detect new, unknown attacks (zero-day attacks) or even variants of known attacks; it can only detect attacks whose signatures are known to the scan engine. Furthermore, there are several malware development techniques used for evading signature-based systems, which include: [370]

- *Polymorphic strategy*: Malware that uses polymorphic strategy usually encrypts itself with the use of an encryption algorithm. Furthermore, in different infection instances different decryption keys can be generated thus making the instance unique. Another extension of this technique includes malware that uses a number of different encryption algorithms, thus avoiding detection.
- *Metamorphic strategy*: Malware that implement the metamorphic strategy are very complex and hard to detect. For every infection instance the malware changes itself in a way that it has no resemblance to the original one, e.g., by changing its source code.

Traditional signature-based techniques are based on malware signatures blacklisting, but as this has been deemed as insufficient, an alternative approach called *whitelisting* has become relatively popular. This is an approach used to manage the software that is installed on a device which involves allowing only approved software to be installed and run. However, the rules of this technique can be very strict and eventually create a rigid environment which is not user-friendly. Additionally, whitelisted applications that are vulnerable pose a critical security threat and can provide an attack surface for unauthorized access to a system that remains undetected, e.g., a browser that is whitelisted could easily become the target of an attack. [26]

In [223], a hybrid signature and trust-based IDS for WSNs is discussed, which is well-suited for the scope of our Cyber-Trust. The trust evaluation part of this model proposes that each node calculates trust values of its neighbors from direct observations based on functional reputation metrics. These direct observations are exchanged between the nodes of the network and the consolidated trust metric is calculated by combining direct and indirect trust evaluations. This work considers a hierarchical cluster-based WSN where each network is divided into sensors, cluster heads (CH), and a base station (BS). Control packets are sent to the cluster heads by each sensor containing its neighbor's IDs and the corresponding trust values. However, if a CH is deemed as suspicious, sensors can send their packets directly to the BS by alternative routes. The cluster heads are responsible for aggregating the data received from the sensors. The base station is responsible for monitoring the whole network, including detecting attacks and informing the nodes.

This work uses a probabilistic model for computing reputation, which is called the beta reputation system as defined in [154]. The nodes take specific values into consideration when computing the reputation of another node, namely, reliability in communication, in sensed data, in reporting the data timely, event report rate, and in forwarding data. The indirect trust is formed by recommendations which are weighted based on the recommendation receiver's trust towards the recommender. The consolidated trust value (CTV) is formed as a weighted sum of direct and indirect trust.

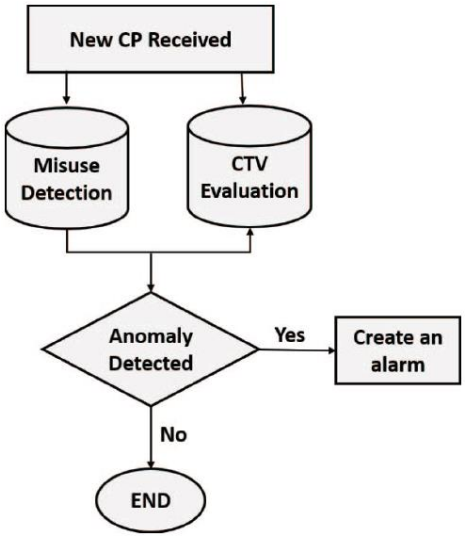


Figure 3.4. Intrusion detection at the BS

The base station uses both misuse detection and consolidated trust evaluations for detecting malicious activities as shown in Figure 3.4.

- Misuse detection uses a set of rules:
  - *Reception and Delay rule*: All cluster heads must send control packets before data packets and a defined time out. Failure to respect this rule can mean marking a node as malicious while combined with further observations it can lead to the detection of DoS attacks, selective forwarding, and black hole attacks.
  - *Members of a CH rule*: When a cluster head reports its cluster members there should be no missing nodes.
  - *Frequency of CPs*: The same CP can be retransmitted only a limited number of times.
- Consolidated trust values (CTV) evaluations
  - A base station keeps a trust evaluation table for cluster heads and sensors that don't belong to any cluster. There are attacks and malicious behaviors that cannot be detected by the predefined set of rules. Consolidated trust evaluation is used to detect malicious nodes to evaluate node's behaviors. The base station sets a threshold that is shared with the CHs and sensors, so that when a node whose trust evaluation is found to be lower than this threshold it is reported as malicious. For example, in the case of a sensor detecting a malicious cluster head, it encrypts the data using a pre-shared key with the base-station and send encrypted data to the base station indicating that the CH maybe compromised.

This work proposes that when a node is found to be malicious is simply disabled. The simulations comparing the energy consumption and network lifetime with and without using the proposed IDS show that the IDS really the network lifetime while a little more energy consumption is observed which is expected.

Signature-based scanning entails a demotion of communication privacy, since the actual payload of communications is scanned. Even encrypted sessions could be broken through the use of SSL/TLS termination proxies or through the disclosure of the connection endpoint decryption keys to the IDS/IPS [345]. Organizations should document that proportionality of the demotion of the involved entities' right to secrecy against their legitimate interest to protect their assets and refrain from permanently logging captured traffic, except for forensic evidence [143].

3.2.1.4 *Consequence-based assessment*

Information systems and devices handle data and information that need to be kept safe within the system and device boundaries, such as user passwords. Moreover, within the networked world, data and information need to be exchanged across system or even organizational boundaries, in the context of multi-peer transactions. Within such transactions, receiving systems are entrusted to safeguard the data and information they receive, according to pre-agreed information usage policies; typically, these policies state that the data and information should be used only for the purpose communicated for, it should not be disclosed to unauthorized third parties. Honest entities are the ones that observe this obligation, whereas dishonest entities are those which disregard their obligations and leak information to unauthorized parties [227]. We note here that within the duties of entities that receive data and information in the context of transactions is to make every reasonable effort to prevent accidental disclosure to unauthorized parties, however data may leak from honest entities, in the cases they are compromised or do not apply adequate measures to authenticate their communicating parties and/or encrypt their communications.

The work in [227] presents a trust management system that estimates the trustworthiness (honesty) of an entity. In this work, an entity is assigned a trust score which reflects the recipient's ability to meet its obligations; intentional or non-intentional leakages are not discriminated and the measure of the trust score assigned to an entity is computed as

$$trustScore_{entity} = 1 - \frac{\#objectsLeaked_{entity}}{\#totalObjects_{entity}}$$

with $\#objectsLeaked_{entity}$ being the number of objects that have leaked by the entity and *#totalObjects* being the total number of objects entrusted to the entity. Time is quantized into intervals and the trust score for entities is updated at the beginning of each interval.

This approach requires the following underpinnings:

- a scheme for quantizing the data/information entrusted to entities into objects;
- one or more methods for monitoring information leakage, so that dishonest parties and parties that non-intentionally fail to maintain information confidentiality are identified and punished (in terms of their trust score). The work in [227] does not prescribe any mechanisms, indicating that information leakage from an entity can be measured using either audit logs, interrogating other entities or via some domain-specific leakage detection mechanism. The presence of a lag between the time point of information leakage and its detection is acknowledged, and addressed through the introduction of a scheme that provides incentives for honest entities to self-report information leakage and warn thus their peer entities about their own demoted trust.
- algorithms that support trust threshold-aware encryption and decryption, i.e., algorithms that allow the sending entity to encrypt data in a way that the receiving entity can decrypt the key only if its trust score is above a sender-specified threshold. [227] proposes some algorithms that satisfy this requirement, however the requirement can be relaxed in contexts that the sending entity refrains from communicating data to entities not meeting its trust threshold.

Consequence-based approaches can be directly incorporated into the Cyber-Trust trust management algorithms. Clearly, the Cyber-Trust solution should be generic and domain-agnostic, hence a single domain-specific method for monitoring leakage is not a viable approach: provision of multiple, domain-specific plugins could be a method for tackling this issue; additionally, manual specification of leakage metrics could be adopted in the absence of adequate or accurate domain-specific modules.

3.2.2 Status-based approaches

Status-based approaches to trust and risk assessment examine the current state of the interacting device, regarding its security aspects. The goal is to determine whether (a) a breach has already been made to the device, having resulted in tampering of either software or its configuration and (b) how prone the device is

to breaches, in the sense that known vulnerabilities have not been appropriately and timely handled through installation of patches. The security controls that apply to the device, are also taken into account since they moderate the device's vulnerability levels. In more detail, the following aspects are considered in status-based approaches:

- Have critical files been tampered with? Relevant validations span across:
  - the device's firmware;
  - the operating system and other s/w;
  - the system/network config files;
  - the audit and event logs.
- Have the latest patches been installed? Missing patches increase the vulnerability level of the device and therefore demote the trust level.
- Which security controls are in effect to protect the device?

In the following subsections, we review methods that can be used in the context of status-based approaches to trust and risk assessment.

3.2.2.1 Remote attestation

Remote attestation (RA) is a method by which a host (client) authenticates its hardware and software configuration to a remote host (server). The goal of remote attestation is to enable a remote system (challenger) to determine the level of trust in the integrity of platform of another system (attestator). The architecture for remote attestation consists of two major components: Integrity measurement architecture and remote attestation protocol [109].

Many RA techniques with various assumptions, security features and complexities have been proposed. Most of them can be divided into three approaches: hardware-based, software-based, and hybrid [182, 197].

- **Hardware-based approaches** typically rely on security provided by a separate and dedicated secure hardware component, such as a Trusted Platform Module (TPM). Despite resisting all, except physical attacks, hardware-based approaches are unsuitable for low-end and legacy embedded devices due to its added complexity and various cost factors.
- **Software-based RA techniques** offer a very low-cost alternative. These techniques rely on a one-time special checksum function that covers memory in an unpredictable (rather than contiguous) fashion. Any interference with, or emulation of, the computation of this checksum is detectable by extra latency incurred by self-relocating malware moving itself (in parts) while trying to avoid being "caught" by the checksum. Unfortunately, the security level offered by this approach is uncertain after several attacks on software-based RA schemes. Another problem with the software-based approach is that it requires strong assumptions about adversarial capabilities, which are unrealistic in many real settings. However, this is the only RA option for legacy devices.
- **Hybrid (software-hardware)** RA co-designs attempt to overcome limitations of purely software-based techniques while minimizing hardware requirements. Hybrid RA is also especially suitable for mid-range and low-end embedded devices, which usually lack, or cannot accommodate, a secure hardware component, such as a TPM. SMART is the first hybrid RA architecture with minimal hardware modifications to existing microcontroller units (MCUs). SMART requires uninterruptible and atomic execution of non-malleable ROM-resident attestation code which has exclusive access to attestation key(s); this is enforced by hard-wired MCU access control rules. Actual attestation is performed by a prover (Prv) computing a cryptographic checksum over a specific memory region and returning the result to a verifier (Vrf). Notably, SMART's requirement for atomic execution of attestation code was motivated by the need to mitigate code-reuse, e.g., ROP attacks.

3.2.2.2 *Remote attestation protocol*

Figure 3.5 illustrates a generic architecture of a remote attestation protocol. The challenger generates a challenge and sends it to the sensor node to be attested. Upon receiving this challenge, the sensor node will check the corresponding firmware, construct the attestation response and return the response, which is associated with the challenge content. The sink can verify this response based on the challenge it generated before and the expected firmware content [109]. The attacker might keep a copy of the correct firmware simply to answer the challenge from the sink (i.e., step 3 in Figure 2) and make the malware practically run on the sensor nodes. Therefore, most software-based attestation protocols such as SoftWare-based ATTestation (SWATT), Secure Code Update By Attestation (SCUBA) and software abstraction approach set a timer right after it sends out the challenge. If there is a timeout before a response is received, the sink will suspect that the sensor node to be attested has been compromised. A recent research has demonstrated that these secure time-based attestation schemes (namely SWATT and SCUBA) are very difficult to design and implement correctly in practice [182].

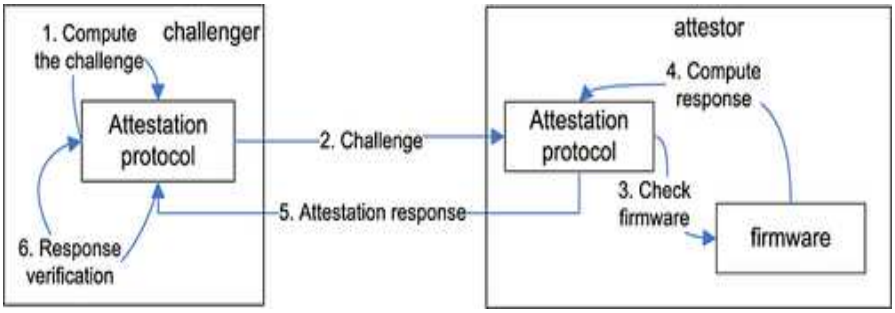


Figure 3.5. Remote attestation protocol

3.2.2.3 *Unified Extensible Firmware Interface*

Unified Extensible Firmware Interface (UEFI) is a specification for a software program that connects a computer's firmware to its operating system (OS). UEFI is expected to eventually replace BIOS. UEFI is installed at the time of manufacturing and is the first program that runs when a computer is turned on. It checks to see what hardware components the computing device has, wakes the components up and hands them over to the operating system. The new specification addresses several limitations of BIOS, including restrictions on hard disk partition size and the amount of time BIOS takes to perform its tasks.

Because UEFI is programmable, original equipment manufacturer (OEM) developers can add applications and drivers, allowing UEFI to function as a lightweight operating system.

The Unified Extensible Firmware Interface is managed by a group of chipset, hardware, system, firmware, and operating system vendors called the UEFI Forum (https://uefi.org/). The specification is most often pronounced by naming the letters U-E-F-I.

The salient details of the UEFI specification includes the definition of ANSI C-callable API's that are published by the system board firmware implementation. The figure below describes various facets of UEFI, including the dynamically loadable UEFI drivers that can be installed on host-bus adapters (HBA's). This allows for moving beyond the limitations of PC/AT BIOS wherein option ROM's had to reside in a fixed location. The UEFI Specification defines a set of mandatory capabilities, including boot and runtime services. These services allow for managing memory, timers, events, UEFI variables, capsules, and loading code.

Figure 3.6 also highlights other capabilities of UEFI, including the 'UEFI Secure Boot' feature. The gist of the feature is to allow some administrative control of the extensible image loading found in UEFI. And what is potentially of interest to the cloud community includes the integrated networking capability of UEFI, along with a shell. In the spirit of the design-by-interface methodology of UEFI, these API's for all of the 6 underlying networking interfaces, from the IHV-supplied lowest level wired networking driver up to the software-visible API's that correspond to the OSI stack. Correspondingly, the UEFI Shell has a specification hosted on [UEFI],

too, alongside the main UEFI Specification. This mapping to documentation allows for interoperability and investment protection of applications that consume the network stack, shell script, and shell [UEFI-SHELL] applications, respectively.

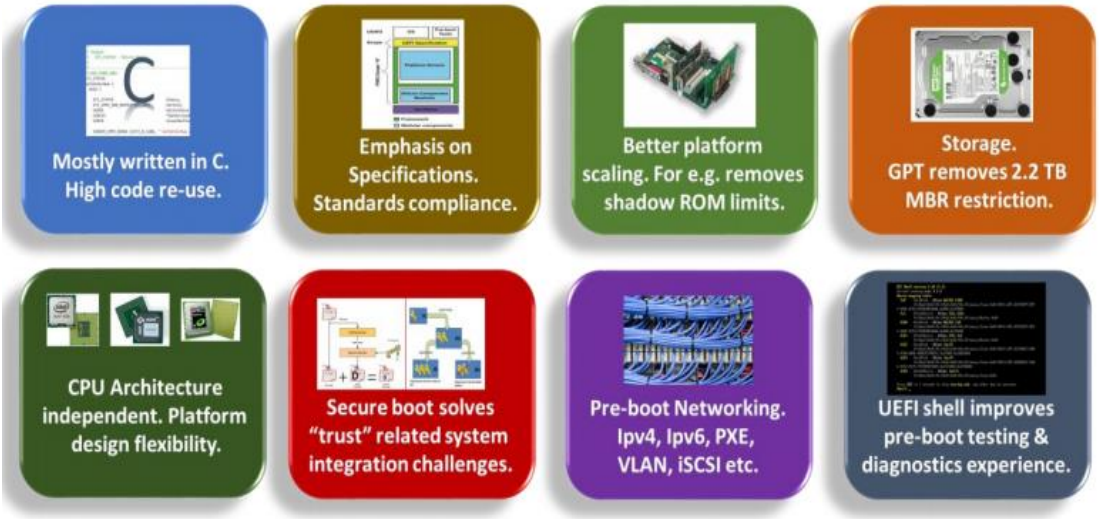


Figure 3.6. UEFI specifications

3.2.2.4 Trusted computing group

The Trusted Computing Group (TCG - https://trustedcomputinggroup.org/) is a not-for-profit organization that was formed in 2003 to define, develop and promote security specifications for computers and networks. These standards help protect data, hardware and other resources from compromise, damage or theft by malicious entities without adversely impacting the rights of individuals or businesses who participate.

Trust and security are fundamental requirements for commercial and private usage of modern information and communication technology. Users, enterprises, and governments are using digital processes through Internet of Things (IoT) every day in mission critical operations like trading, banking, the operation of critical infrastructures, and many others. Therefore, TCG has been aiming to define the necessary components to improve trust and security in computing systems. The main goal is to establish trust, which means that you get assurance that the system is always acting in the expected way. The specifications of TCG are:

- **Security:** TCG-enabled components should achieve controlled access to designated critical secured data and should reliably measure and report the system's security properties. The reporting mechanism should be fully under the owner's control.
- **Privacy:** TCG-enabled components should be designed and implemented with privacy in mind and adhere to the letter and spirit of all relevant guidelines, laws, and regulations. This includes, but is not limited to, the OECD Guidelines[17], the Fair Information Practices[18] and the European Union Data Protection Directive[19].
- **Interoperability:** Implementations and deployments of TCG specifications should facilitate interoperability. Furthermore, implementations and deployments of TCG specifications should not introduce any new interoperability obstacles that are not for the purpose of security.

[17] http://www.oecd.org/sti/ieconomy/privacy.htm

[18] https://www.justice.gov/opcl/docs/rec-com-rights.pdf and relevant updates

[19] https://ec.europa.eu/info/law/law-topic/data-protection/data-protection-eu_en

- **Portability of data:** Deployment should support established principles and practices of data ownership.

3.2.3 Risk assessment

As technology continues to permeate modern-day society, the security of, and trust that we place in, these systems becomes an increasingly significant concern. This is particularly true given the plethora of attacks being launched that target organizations, governments and society [171]. The traditional approach to address such challenges has been to conduct cybersecurity risk assessments that seek to identify critical assets, the threats they face, the likelihood of a successful attack, and the harm that may be caused. Only in this way, and after the identified risks have been prioritized, would appropriate approaches be selected to effectively address them [171].

Risk assessment is a process of identifying, estimating and prioritizing risks to the organizational assets and operations. This is a critical activity within risk management, as it provides the foundation for the identified risks to be treated. Treatment options include: risk acceptance for cases where the risk is at an acceptable level considering the organization's risk management policy; risk mitigation using security controls; risk transfer through the purchase of cyber-insurance; or risk avoidance by removing the affected asset. There are several core concepts that feature within risk assessment, such as assets, vulnerabilities, threats, attack likelihood, and impact or cyber-harm [232].

Assets can be defined as any items of value to the organization, and can have various different properties. For instance, assets can be tangible (e.g., technical infrastructure) or intangible (reputation or a business process), or they can be small components within a system or be the system themselves. Vulnerabilities are the ways in which assets can be exploited, and define weaknesses in assets or in the risk controls put in place to protect them. A threat is the action that could adversely impact an asset, and typically involves exploiting a vulnerability. Such actions may be deliberate (e.g., stealing corporate data) or accidental (e.g., being the victim of a social engineering attack). Cyber-risk is the combination of these concepts, and considers the likelihood of a successful threat or attack occurring, and the harms that may result to assets[20].

3.2.3.1 Risk identification

In order to assess any risk, it is important to identify the risks that exist within the systems. Therefore, the risk identification phase tries to create a comprehensive list of events that may prevent, degrade or delay the achievement of the system objectives. Comprehensive identification is critical because a risk that is not identified at this stage will not be included in the risk analysis phase. Although there are numerous tools and techniques that can be used to facilitate the identification and analysis of risks, it is recommended that a multidisciplinary workshop discussion be used. The workshop should include the business and service owners (or their nominated delegates) and subject matter experts from both the business and ICT [172].

The following provides an overview of the techniques that should be used to ensure that comprehensive lists of relevant risk are identified [205]:

- People with the appropriate knowledge should be involved in identification of risks. Discussions must include the business owner and subject matter experts who can provide relevant and up-to-date information during the process; and
- Group discussions and workshops to facilitate the identification and discussion of the risks that may affect the businesses objectives.

There are different factors (assets, threats, vulnerabilities, consequences and incident scenarios) that will be used to identify and evaluate the risks [172]:

- System scope delimitation: Determining the scope included in the risk assessment and its boundaries.

[20] https://www.thebalancesmb.com/assets-definition-2947887

- Asset identification: Identifying any type of item that has value to the organization and that could cause damage if it is involved in an incident.
- Threat analysis: identifying all agents, either natural or human made, accidental or intentional, internal or external, that could pose a threat to the organization.
- Vulnerability analysis: Identifying all potential weakness in the organization that could facilitate a successful attack and cause damage to the assets.
- Consequence determination: Identifying the possible consequences that different events could have on the organization.
- Incident scenario identification: Determining the possible events that could have an impact on the organization and that will serve as a base to identify the risks.

There is no scientific method that provides guarantees that all risks are identified. Some additional good practice approaches to identify all risks and relevant information sources are listed in the following table:

Table 3.1. Approaches to risk identification

Approach /Sources	Description
SWOT analysis (Strength, Weakness, Opportunity Threats)	Commonly used as a planning tool for analyzing a business, its resources and its environment by looking at internal strengths and weaknesses; and opportunities and threats in the external environment.
PESTLE (Political, Economic, Sociological, Technological, Legal, Environmental)	Commonly used as a planning tool to identify and categorize threats in the external environment (political, economic, social, technological, legal, environmental).
Brainstorming	Creative technique to gather risks spontaneously by group members. Group members verbally identify risks in a 'no wrong answer' environment. This technique provides the opportunity for group members to build on each other's ideas.
Scenario analysis	Uses possible (often extreme) future events to anticipate how threats and opportunities might develop.
Surveys/Questionnaires	Gather data on risks. Surveys rely on the questions asked.
One-on-one interviews	Discussions with stakeholders to identify/explore risk areas and detailed or sensitive information about the risk.
Stakeholder analysis	Process of identifying individuals or groups who have a vested interest in the objectives and ascertaining how to engage with them to better understand the objective and its associated uncertainties.
Working groups	Useful to surface detailed information about the risks i.e., source, causes, consequences, stakeholder impacted, existing controls.
Corporate knowledge	History of risks provide insight into future threats or opportunities through: • Experiential knowledge – collection of information that a person has obtained through their experience. • Documented knowledge – collection of information or data that has been documented about a particular subject. • Lessons learned – knowledge that has been organized into information that may be relevant to the different areas within the organization.

Process analysis	An approach that helps improve the performance of business activities by analyzing current processes and making decisions on new improvements.
Other jurisdictions	Issues experienced and risks identified by other jurisdictions should be identified and evaluated. If it can happen to them, it can happen here.

3.2.3.2 Risk analysis

Once the relevant risks have been identified the likelihood and impact of them eventuating must be assessed and rated. Typically, the likelihood and impact of a risk eventuating are rated using a qualitative scale.

Impact assessment. Assess the impact of the risk eventuating with no controls in place. This will inform the gross risk rating and enable the effectiveness of any current controls that reduce the impact of a risk event that occurs to be assessed. Although there may be multiple impact statements documented for a risk, only one impact rating can be assigned to the risk. As a result, the highest rated impact statement should be used to determine the impact rating of a risk [327].

Likelihood assessment. Assess the likelihood of the risk eventuating with no controls in place. This will inform the gross risk rating and enable the effectiveness of any current controls that reduce the likelihood of a risk event occurring to be assessed. Where historic information is available about the frequency of an incident's occurrence it should be used to help determine the likelihood of the risk eventuating. However, it must be noted that the absence of such information does not necessarily mean that the likelihood of the risk eventuating is low. It may merely indicate that there are no controls in place to detect that it has occurred [169].

Table 3.2. Likelihood scale

Rating	Description	Meaning
5	Almost Certain	It is easy for the threat to exploit the vulnerability without any specialist skills or resources or it is expected to occur within 1 – 6 months.
4	Highly Likely	It is feasible for the threat to exploit the vulnerability with minimal skills or resources or it is expected to occur within 6 – 12 months.
3	Possible	It is feasible for the threat to exploit the vulnerability with moderate skills or resources or it is expected to occur within 12 – 36 months.
2	Possible but Unlikely	It is feasible but would require significant skills or resources for the threat to exploit the vulnerability or it is expected to occur within 3 – 5 years.
1	Almost Never	It is difficult for the threat to exploit the vulnerability or it is not expected to occur within 5 years.

Risk rating. The risk rating is evaluated using a risk matrix. Example Risk Scales and Matrix also presents a risk matrix that can be used to map the likelihood with the impact rating, the overall risk rating being the point where the two ratings intersect. For example [195]:

- A risk with likelihood of Almost Never, and impact rating of Moderate would result in an overall risk rating of 6;
- A risk with a likelihood rating of Possible, and an impact rating of Severe would result in an overall risk rating of 22; and
- A risk with a likelihood rating of Almost Certain, and an impact rating of Minor would result in an overall risk rating of 16.

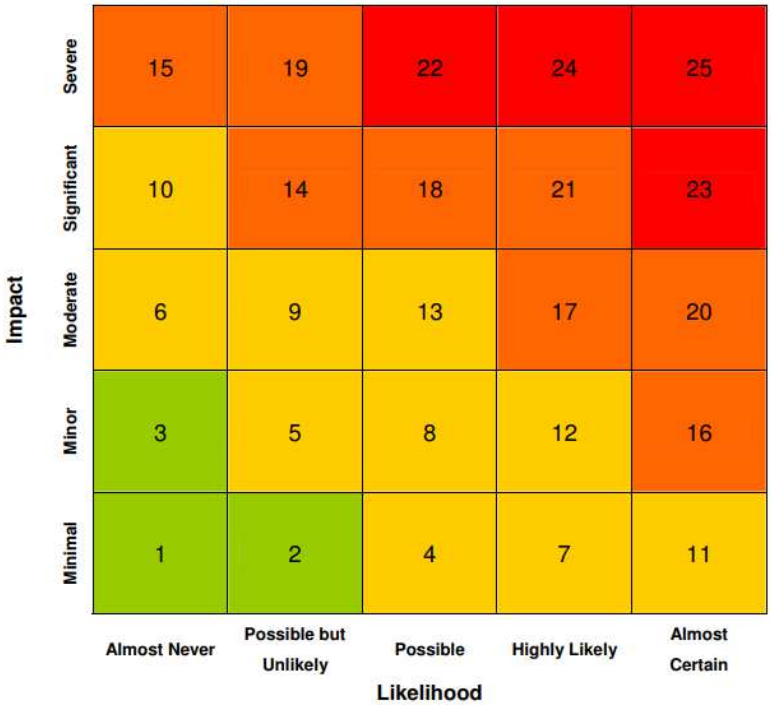


Figure 3.7. Risk matrix

The risk rating without any controls in place have been assessed is called the gross risk. Typically risks that are assessed as being 1 to 3 on the rating scale without any controls in place are considered acceptable to the business and may not require the implementation of any controls to manage them. However, because risk is rarely static they should be added to the agency's risk register so that they can be monitored and re-assessed on a regular basis to ensure that the likelihood and/or impact do not change.

3.3 Trust management models

Trust management models target at enabling nodes that participate in the trust management system to determine a trust metric value for other nodes within the system. Approaches to how trust models approach trust computation vary regarding numerous aspects, including the input used to compute trust, the way that trust values are updated, the consensus sought for trust value computation, the scale at which trust is measured, their resilience against attacks and so forth. Furthermore, trust management models vary with respect to architectural paradigm they follow, i.e., the way that the components participating in the trust management system are deployed in the target network, the relationships between the components and the information flows.

In the following subsections we survey existing trust models and their architectures, commenting on their merits and demerits.

3.3.1 Review of existing trust models

This section overviews the trust models that have been proposed by the literature trying to find an effective and efficient trust computation method. In service-oriented networks, an IoT device acting as a service requester needs a way of evaluating which of its peers can be trusted to provide it with the requested service, while taking into consideration the energy demands of carrying out such evaluation. This is the challenge that trust management models are aiming to solve. We present trust management models as seen in the literature and we categorize each model by trust dimensions, resiliency against certain attacks and qualitative characteristics.

3.3.1.1 Trust dimensions

Trust models are composed of several trust dimensions which can vary between them depending on the approach followed. In this section we present the five most essential trust dimensions, namely, trust composition, trust propagation, trust aggregation, trust update and trust formation. [161]

Trust composition. Refers to what components the given model takes into consideration. The components include Quality of Service (QoS) and Social trust.

- QoS trust refers to the evaluation of a node based on its capability to deliver the requested service. It is considered as the "objective" evaluation of trust. In order to compute QoS trust, models use various trust properties including competence, cooperativeness, reliability, task completion etc.
- Social trust refers to the social relationship between owners of IoT devices. Social trust is used in systems where IoT devices must not be evaluated only on a QoS basis but also on a social basis, which is the device's commitment and willingness to cooperate. It can also be derived from similarity of devices. Social trust properties include connectivity, honesty, unselfishness etc.

Trust propagation. Refers to the way trust values are disseminated between entities. In general, there are two approaches, namely distributed and centralized.

- In distributed trust propagation each device acts autonomously by storing trust values and disseminating them as recommendations to other devices as needed.
- In centralized trust propagation a central entity exists, which is responsible for storing trust values of the monitored network and disseminating them as needed.

Trust aggregation. Refers to the computation techniques used by a model to combine trust obtained from direct observation with indirect trust coming from recommendations. Main aggregation techniques include weighted sum, Dempster-Shafer theory, Bayesian inference, fuzzy logic and regression analysis.

- Weighted sum is a technique where weights are assigned on the participating values either statically either dynamically. For example, one model could use a trust property, e.g., competence, in order to assign higher or lower weights.
- Dempster-Shafer theory is based on belief function and plausible reasoning. It is a framework for reasoning with uncertainty related to other frameworks like probability, possibility and imprecise probability theories.
- Bayesian inference considers trust to be a random variable which follows a probability distribution. It is a simple and statistically sound model.
- Fuzzy logic uses approximate reasoning meaning that it doesn't use a binary evaluation variable but rather a variable whose values range between 0 and 1 for example, or even linguistic limits like High and Low which are translated using a membership function.
- Regression analysis is basically a prediction model which predicts the probability of an event happening or not happening (binary). In trust computing it is used to estimate relationships between environmental conditions, e.g., how much computing resources a node needs, which are treated as variables and used to predict the trustworthiness of an object.

Trust update. Describes when trust values are updated. There are two approaches: event-driven and time-driven.

- Event-driven is the approach in which trust values are updated when an event occurs.
- Time-driven is the approach in which trust values are update periodically.

Trust formation. Refers to how the overall trust is formed out of the trust properties considered. Trust can be formed by considering only one trust property (Single-trust) or many properties (Multi-trust).

- Single-trust is when only one property is taken into consideration when computing trust and it is usually a property of QoS. It is considered as a narrow approach because trust is multi-dimensional, but it is useful in cases with limited resources.
- Multi-trust is the multi-dimensional approach in computing trust, because it uses more than one trust properties to form the overall trust evaluation of a device.

We also used the following properties to classify the trust managements models: [356]

Trust scaling. Trust is represented by either discrete or continuous numerical values.

- *Binary discrete values*: Represented with 0 or 1, distrust or trust respectively.
- *Multinomial discrete values*: Sometimes binary values are not sufficient, so more scaled metrics are used, e.g., “very trust”, “trust”, “distrust”, and “very distrust”.
- *Continuous value*: For example [0,1].
- *Interval*: Instead of using one value to represent trust, an interval is used in order to introduce the uncertainty property of trust.

Semantic meaning. In different models and scenarios trust can have various semantic meanings. Some semantic meanings include:

- *Evidence- or experience-based trust*: Trustors build their trust based on their own observations and past interactions. This can be done using probabilities, mean average, mode average or difference.
- *Application-specific behavior-based trust*: This means that trust is calculated based on specific monitored behaviors.
- *Similarity-based trust*: This approach assumes that devices that are similar to each other, will probably trust each other.
- *Reputation*: Reputation is a type of trust which isn’t relative to the trust between two specific devices but instead each device has a trust value representing how much it is trusted by the whole community.
- *Fuzzy logic-based trust*: Trust is considered to be nondeterministic and because of this, fuzzy logic is suitable for evaluating it.
- *Comprehensive trust*: Many approaches take into consideration trust as seen in human relationships. In this case, trust is seen as a sum of complex human interactions. On this basis, social metrics are introduced in trust evaluation, like social similarity, social contact, friendship, etc.

Trust inference. In IoT networks, nodes are not always directly connected with another and in these cases trust evaluation cannot be done by direct observation. Therefore, trust recommendations are introduced. There are two operators to be considered for trust inference: transitivity and aggregation:

- Transitivity operator refers to the way the recommendations are combined by building a transitivity “chain” to the trustee node and it is based on the transitive relation from mathematics.
- Aggregation operator refers to the way the recommendations are combined to calculate the overall indirect trust.

Table 3.3. Overview of different trust models

Model	Composition		Propagation		Aggregation			Update	Formation	
	QoS	Social	Distrib	Central	Weigh	Fuzzy	Bayes	E/T	Sin	Mul
[101] [149]	X	X	X		X			E/T		X

[239] [107]										
[102] [150]	X	X	X		X		X	E/T	X	
[67]	X		X		X	X		T	X	
[264]	X		X		X	X		T	X	
[330]	X		X		X			E	X	
[302]	X			X	X			T	X	
[222]	X	X	X	X	X			E		X
[368]	X			X	X			E/T	X	
[369]	X			X	X			E	X	
[344]	X		X		X			T	X	
[303]	X		X	X				T	X	
[340]	X		X	X	X			E	X	
[175]		X	X	X	X	X		E		X

3.3.1.2 Trust-based attacks

Inside an IoT network, every node wants to have a high trust value. A high trust value means the node will be selected more times over nodes with lower trust value, thus increasing their gains and influence over the network. Malicious nodes will try a variety of attacks in order to gain more trust among their peers. There are a lot of attacks that can be executed in an IoT network, such as, jamming attacks, replay attacks, eavesdropping attacks, DoS attacks, etc. However, there are some attacks that are especially used to disrupt trust and reputation systems. These attacks fit better into the scope of this work and the most common ones are shortly presented below. [217]

- **Self-promotion attacks (SPA) [161].** The malicious node provides good recommendations for itself.
- **Bad-mouthing attacks (BMA) [161].** A malicious node provides bad recommendations for a "good" node in order to decrease its trust value and probability of being chosen as a service provider.
- **Ballot-stuffing attacks (BSA) [161].** A malicious node boosts the trust of another malicious node in order to increase the possibility of the malicious node being chosen as a service provider.
- **Opportunistic service attacks (OSA) [161].** When the trust of a malicious node starts dropping, it starts acting as a "good" node in order to regain its trust.
- **On-off attacks (OOA) [161].** A malicious node is behaving randomly, sometimes performs well sometimes bad, so that it won't get labeled as malicious.
- **Whitewashing attacks [356].** When a malicious node has very low trust, it discards its identity by leaving the network and re-entering it.
- **Discriminatory attacks [149].** A malicious node attacks non-friends or nodes without strong social ties.
- **Sybil-Mobile attacks [107].** A malicious node creates one or more fake identities in order to manipulate recommendations, promote itself and gain influence over the network.
- **Selective Behavior attacks [369].** A malicious node is behaving well and bad between different services. For example, well for simple services, but bad for more complex ones.

Table 3.4. Overview of trust-based attacks

Attack Resiliency	SPA	BMA	BSA	OSA	OOA	White-washing	Discrimi-natory	Sybil-Mobile	Selective Behavior
[101]	X	X	X						
[149]	X	X	X			X	X		
[239]	X	X	X						
[107]	X	X	X					X	
[102]	X	X	X	X					
[150]	X	X	X	X					
[67]	X								
[264]									
[330]					X				
[302]									
[222]	X	X	X	X					
[368]									
[369]	X	X	X		X				X
[344]									
[303]									
[340]		X							
[175]									

3.3.1.3 Trust management models

In this section we survey the different trust models proposed in the literature. For each model, the approach adopted for trust computation is presented, while salient features of the models are summarized in Table 3.5, within subsection 3.3.1.4 below.

Bao, 2012 [101]. This model considers Community of interest (CoI) based social IoT (SIoT) systems. Devices have owners and owners have many devices. Each owner keeps a friends list. Nodes belonging to similar communities are more likely to have similar interests or capabilities. Both QoS and Social trust composition are considered, including three trust properties: honesty (QoS), cooperativeness (QoS) and community-interest (Social); please refer to Table 3.5 for further details. The trust value is a real number in the range [0,1] where 1 indicates complete trust, 0.5 ignorance, and 0 distrust. The trust values can occur from direct observations and when such observations are not available, from recommendations. It follows a distributed scheme, while for trust aggregation the weighted sum technique is used. It is worth mentioning that the weights that were used for past experiences can be dynamically adjusted when new evidence occurs to rebalance the trust convergence rate and trust fluctuation rate. In the simulation results, the effect that changing weights have is observed, but a way to dynamically adjust them is not mentioned.

Chen, 2016a [149]. This model is very similar to Bao, 2012. Main differences include: 1. A general approach for the computation of overall trust is not discussed. Instead, overall trust computation for specific scenarios is discussed. 2. The friends (nodes) lists exchanged between nodes upon interaction are encrypted with a one-way function in a way that nodes can identify only common friends. Hashing is cost-efficient. 3. The model is tested in two real-world scenarios, namely, “Smart City Air Pollution Detection” and “Augmented Map Travel Assistance”.

Bao, 2013 [102]. This model considers Community of interest (CoI) based social IoT (SIoT) systems. Devices have owners and owners have many devices. Each owner keeps a friends list. Nodes belonging to similar communities are more likely to have similar interests or capabilities. Both QoS and Social trust composition are considered. The trust value is a real number in the range [0,1] where 1 indicates complete trust, 0.5 ignorance, and 0 distrust. The trust properties considered are honesty, cooperativeness and community-interest; please refer to Table 3.5 for more details. The trust propagation is distributed. For trust aggregation, Bayesian inference is used for direct trust and weighted sums are used to aggregate recommendations into indirect trust. Most importantly, this model introduces an efficient storage management strategy suitable for large-scale IoT systems.

Chen, 2016b [150]. This model is an extension of Bao, 2013 [102]. Extensions include: 1. In the evaluation of recommenders, it introduces two additional properties, namely, friendship and social contact, which are further analyzed in Table 3.5. 2. In trust aggregation it combines the direct with the indirect trust to form the overall trust. 3. Its simulations outperform EigenTrust [183] and PeerTrust [76] in trust convergence, accuracy, and attacks resiliency.

Chen, 2011 [67]. This model considers only QoS metrics for evaluating trust, namely, end-to-end packet forwarding ratio (EPFR), energy consumption (EC), and package delivery ratio (PDR). Each node maintains a data forwarding transaction table which includes the values: 1. Source: the trust and evaluation evaluating nodes, 2. Destination: the evaluated destination nodes, 3. $RF_{i,j}$: the times of successful transactions made between nodes i and j, and 4. $F_{i,j}$: positive transactions. It follows a distributed scheme in terms of trust propagation. In trust aggregation, a fuzzy trust model is used, and the overall trust is formed using a weighted sum of direct and indirect trust based on recommendations. The direct trust is computed by first aggregating the aforementioned QoS metrics, then labeling the results as a positive or negative experience based on a threshold and then a fuzzy membership function computes the direct trust based on the number of positive and negative experiences. Additionally, the model was tested on simulations and achieved better performance from BTRM-WSN [206] and DRBTS [122] in both packet delivery ratio and detection probability of malicious nodes.

Mahalle, 2013 [264]. This model considers three QoS metrics: Experience (EX), Knowledge (KN) and Recommendation (RC) ratings. It follows a distributed scheme, as every device considers the ratings of its neighbors for the calculation of the trust score. Trust is calculated periodically using Mamdani-type fuzzy rules (representing If-Then relationships between their input variables) from the linguistic values of the three aforementioned metrics. Trust scores (as linguistic values) are then mapped to a set of access control permissions. Experience (EX) is the weighted sum of a number of previous interaction ratings between two devices (+1 for a successful interaction and -1 for an unsuccessful interaction), Knowledge (KN) is the weighted sum of direct and indirect knowledge ratings, and Recommendation (RC) is the weighted sum of RC ratings from a number of devices about the device to be trusted. The three metrics are mapped to their linguistic variables using predefined numeric (crisp) ranges. The model was tested in a simulated environment of wireless sensors with communication between sensors being controlled by trust ratings, resulting in more energy efficient communications, and proving to be scalable.

Prajapati, 2013 [344]. This model considers the service satisfaction at a given time from a specific service provided by a node (a QoS property). Trust is defined as: the Direct Trust value, the Recommended Trust value if the node calculating the trust value had no interaction with the rated service/node and thus the Direct Trust value can't be calculated, or as a predefined Ignorance Value if the rated node is joining the cloud environment for the first time. Direct Trust is defined as the weighted sum of the rated service satisfaction ratings over time (with the weights decreasing over time, thus favoring newer ratings). Recommended Trust is defined as the weighted sum of the Direct Trust values of the other nodes. The weights assigned for each Direct Trust value are calculated using the number of positive interactions between the node calculating the trust value and the node whose rating is considered in the weight calculation, and the Satisfaction Level –a factor dependent on availability, recovery time, connectivity and peak-load performance as defined in the service agreement. All nodes maintain a Direct Trust Table and a Recommended Trust Table containing the respective trust values with both tables being updated periodically. This model follows a distributed model as in the case of Recommended Trust, the trust values of all network nodes are considered.

Saied, 2013 [369]. This model considers ratings given to a specific node and service at a given time while also taking into consideration its state (e.g., age, resource capacity, etc.) It follows a centralized scheme with a Trust Manager (TM) node receiving reports from the network and calculating the trust values on demand. This leads to reduced communication overheads –since trust values are calculated and transmitted on demand, less memory usage for each node –since the trust values can be requested again from TM, and thus being energy efficient. The model operates in five phases: 1) TM receives reports from the network nodes, 2) TM calculates the trust values of a number of candidate nodes and sends a list of trustworthy nodes to the requesting node, 3) the requesting node receives the list and interacts with a chosen trustworthy node, 4) the requesting node rates the service provided by the chosen trustworthy node and sends the rating to the TM, and finally 5) TM updates its trust values accordingly. Trust is calculated as the weighted average of the scores given to a node while taking into consideration the reputation of the node providing the score, the contextual similarity of all the reports concerning the same node, and the age of the report –favoring the most recent reports. Contextual similarity is calculated from the node capabilities between two nodes –to locate similar nodes, and/or from the difference of required resources between two services –to locate nodes able to run a similar service. Initially all nodes of the network are deemed trustworthy.

Mendoza, 2015 [330]. This model is a distributed version of the model proposed by Saied et al. [369]. It is noted that centralized schemes may not be suitable for IoT systems as server installation and server costs may be prohibitive. The model considers ratings given to a specific node and service. The model operates in three phases: 1) every node announces its presence to its neighbors while also keeping a record of its neighbors, 2) a node requests a service from a neighboring node and rates the response as positive or negative, and 3) the node calculates and stores the trust value of its neighbor. The response rating is defined as the fixed value of the provided service weighted by an adjusting factor, with the negative response rating being equal to two times the positive response rating. The provided service value is proportional to the processing requirements of the service, as more processing power or energy is required to run a service the higher the service value will be. The trust value of a node is calculated as the sum of all interaction ratings. The model was tested against On-Off Attacks (OOA) and it is noted that a large number of neighbors can cause delays in the assignment of the maximum distrust score to the malicious nodes.

Namal, 2015 [302]. This model considers four parameters: availability of resources to its users, reliability of produced information, response time irregularities, and capacity. It follows a centralized scheme with a Trust Manager (TM) module, hosted on the cloud, receiving filtered data from Trust Agents (TA) distributed on the network which in turn receive raw data and monitor the state of the network nodes. The TM implements a Monitor, Analyze, Plan, Execute, Knowledge (MAPE-K) feedback control loop and calculates the trust using the weighted sum of the trust parameters for all parameters considered. The trust parameter is also a weighted sum of the current value and the previous value calculated. This model shows advantages in: availability and accessibility –as the TMS is hosted on the cloud and is accessible from the internet, scalability –as the TMS utilizes TAs filtering the raw data, and flexibility –as the TAs can be deployed in a flexible manner.

Khan, 2017 [368]. This model considers ratings given to a node by its neighbors, these ratings are the combination of three variables: belief, disbelief and uncertainty –as defined in Jøsang's Subjective Logic. This model is proposed as part of an extension of the RPL routing protocol utilizing the proposed model to isolate malicious nodes. It follows a centralized scheme with a central node (e.g., RPL border router or cluster-head) calculating trust values for all network nodes and deciding to isolate malicious nodes. Each node of the network is assumed to be able to detect and therefore rate the performance of its neighboring nodes; each of the three aforementioned variables is defined as follows: belief is the number of positive interactions divided by the total number of interactions & a constant k, disbelief is defined similarly but instead of the positive interactions the number of negative interactions is used, and uncertainty is also defined similarity but with the constant k used instead of the number of positive/negative interactions. The central node calculates the trust value of each network node by combination of the trust values regarding the node to be trusted and using a threshold the central node isolates malicious nodes from the network.

Djedjig, 2017b [154]. This model considers two QoS parameters: selfishness and energy, and one social parameter: honesty as ratings given about a node from its neighbors. This model is a proposed extension of the RPL routing protocol, as in Khan et al. [67], to isolate malicious nodes. It follows a distributed scheme

with each node calculating the trust values of its one-hop neighbors while also considering the trust values of its one-hop neighbors. Trust calculation is performed as follows: 1) each node calculates the direct trust values of its one-hop neighbors as a weighted sum of the honesty, energy and unselfishness metrics (definitions of which are not discussed in detail) with each metric being the weighted sum of the current value of the metric and the previous value of the metric, 2) each node receives the direct trust values calculated by its one-hop neighbors concerning the node to be rated, and 3) the indirect trust is then calculated by each node as the average of the direct trust calculated by the node itself and its neighbors. All nodes are assumed to be equipped with Trusted Platform Module (TPM) chips.

Medjek, 2017 [161]. This model is based on the one proposed by Djedjig et al. [154] with the difference in the metrics considered: honesty, energy and mobility. The main difference is the network architecture as this model applies to RPL networks consisting of a Backbone Router (BR) that federates multiple 6LoWPAN networks, each consisting of a 6LoWPAN Border Router (6BR) connected to the BR and the rest of the network nodes. This model follows a distributed scheme with each network node calculating the trust of its one-hop neighbors, as in [154], with the added steps of notifying its 6BR if a node is found to be untrustworthy and with the 6BR in turn notifying the BR of the malicious node. All nodes are assumed to be equipped with a Trusted Platform Module (TPM) and all nodes are registered with the BR at installation time, with every node having a unique ID assigned by the BR. Several lists are maintained by the various network nodes; the BR maintains two lists: one of potential malicious nodes and one of all nodes and their states; the 6BR maintains three lists: one of all 6BR area nodes, one of all the mobile nodes, and one of the potential malicious nodes; finally the remaining nodes also maintain three lists: one of potential malicious nodes, one of suspicious nodes and a copy of the mobile node list from the 6BR. Three modules operate on the various network nodes: IdentityMod controls access to the network and ensures that every node has a unique ID, MobilityMod ensures that both the BR and the 6BRs are aware of mobile nodes and of their status, and IDSMod is responsible for attack detection and mitigation. Trust is calculated in a similar fashion to [154] with the values of the honesty metric supplied by the IDSMod and the values of the mobility metric supplied by the MobilityMod; the three metrics are not discussed in detail.

Nitti, 2014 [222]. The two proposed models, subjective and objective, consider seven parameters: service ratings, number of transactions per node –to detect nodes with an abnormal number of transactions, node credibility, transaction factor –separating important transactions to avoid trust to be built solely by many small transactions, computation capacity –as “smarter” nodes can be better suited to become malicious, relationship factor –the type of relation between two nodes, and finally the notion of centrality –as a node with many connections or involved in many transactions takes a central role in the network.

The subjective model follows a distributed scheme where each node stores the necessary information to calculate the trust values locally. Two situations are covered relating to the social relationship between nodes: when the rating node has a social relationship with the rated node and when the two nodes have no direct social relationship. In the first situation trust depends: on the centrality of the rated node in relation to the rating node –by count of the common friends out of all the neighboring nodes, the direct experience of the rating node –further defined as the weighted sum of both short-term and long-term opinions, and the indirect experience of the rating node’s friends –defined as the weighted average of the trust values assigned to the rated node by the rating node’s friends, weighted by their credibility. In the second situation trust depends: on the opinions of the chain of common friends connecting the two nodes, again weighted by their credibility. Generally, after each transaction a rating (positive/negative) is given to the node providing the service and to the nodes whose opinion was considered in calculating the trust value. Negative recommendation ratings are given to both malicious nodes and to nodes in their neighborhood, thus isolating the malicious nodes and their influence further.

The objective model follows a more centralized scheme where each node reports its feedback to special nodes, referred to as Pre-Trusted Objects (PTO), responsible solely for maintaining the distributed storage system, in this case a Distributed Hash Table (DHT) and more specifically one following the Chord architecture. Trust is calculated in a similar fashion as in the subjective model; node centrality is defined as the total number of transactions performed by the node to provide a service divided by the total number of transactions performed to either provide or request a service, and both short-term and long-term opinions

consider the ratings of every network node weighted by their credibility. Nodes with few social relations, high computation capabilities and nodes involved in a large number of transactions between them are assigned low credibility, as they are more likely to become malicious.

Wu, 2017 [303]. The system model consists of four entities with three trust relationships among them. The four entities are defined: RFID tags, RFID readers, authentication centers and one administration center, with the first three being grouped in domains. A domain has multiple RFID readers connected with the domain authentication center which authorizes the readers to interact with the RFID tags, and the domain authentication centers are connected with the administration center. The trust relationships of this system model are defined as: intra-domain trust –trust relationship between RFID tags and readers of the same domain, inter-domain trust –trust relationship between authentication centers, and cross-domain trust – trust relationship between RFID tags and readers belonging to different domains.

The trust management model consists of two layers: the authentication center trust layer –a centralized trust management system managing the trustworthiness of authentication centers, and the reader trust layer – two proposed trust management schemes managing the trustworthiness of RFID readers. The RFID tags are always assumed to be trusted.

The first reader trust management layer scheme proposed uses the Dempster-Shafer evidence theory and consists of four steps: 1) the interaction of an RFID reader is recorded by its neighbors, 2) the neighbors calculate the local trust values which are then transmitted to the authentication center, 3) the authentication center calculates the global trust of the RFID reader by using the Dempster knowledge rule, and finally 4) it the RFID reader is malicious or malfunctioning the administration center is notified. Possible RFID reader interaction events are identified and marked as: malicious behavior, malfunctioning behavior and normal behavior by the neighboring RFID readers, each counting the number of events within a specified time frame. Using the number of recorded events the neighboring RFID readers can calculate the local trust value for each type of interaction events as: the number of events marked as malicious/malfunctioning/normal divided by the total number of recorded events. The final value of the local trust value is then chosen from the event-specific local trust values using a threshold. The authentication center calculates the global trust of the RFID reader by aggregating the event-specific local trust scores calculated by the neighboring RFID readers and then choosing the final integrated event-specific score using a threshold.

The second reader trust management layer scheme proposed considers the fact that events may not be detected by neighbors of the RFID reader and thus the first reader trust management layer scheme may not be applicable to certain situations. Each RFID tag keeps record of the last interaction with an RFID reader, more specifically the RFID reader ID, a timestamp and the rating assigned to the RFID reader by the tag. This record is sent at the next time the RFID tag interacts with any RFID reader (and is then deleted from the RFID tag), with the RFID reader forwarding the record to its authentication center which checks for abnormalities and if any problem arises, it notifies the administration center as well as the authentication center the previous RFID reader belongs.

The proposed authentication center trust layer scheme considers abnormal event reports by RFID readers and affects the trust value of the domain authentication center the readers are part of. Calculation of trust in this case can be performed by either of the two methods proposed for the reader trust management schemes.

Mahmud, 2018 [175]. This model considers three social trust metrics for a pair of nodes, namely: relative frequency of interaction, intimacy and honesty, and the deviations of generated data from the historical data of the node that generated the trust metric and its neighbors. Two trust dimensions are defined: node behavioral trust and data trust; both calculated by combination of direct (from the rating node) and indirect (from the rating node's neighbors) interactions, with indirect interactions being weighted by the distance of the neighbor to the rated node. Node behavioral trust is calculated using an Adaptive Neuro-Fuzzy Inference System (ANFIS), a fuzzy system using back propagation to tune itself. The three inputs to ANFIS are defined as: relative frequency of interaction is defined as the ratio of interactions with the rating node out of all interactions of the rated node in a given time period, intimacy is defined as the ratio of time amount spent interacting with the rating node out of the total time spent interacting with all nodes except the rating node,

and honesty is defined as the ratio of successful interactions out of the total number of interactions of the rated node with its rating node. Three linguistic terms are used in ANFIS for each of the three inputs: Low, Medium and High. Deviations of generated data, used to calculate the data trust, are defined as follows: direct data trust is defined as the deviation of instantaneous data from the historical data generated by the rated node, and indirect data trust is defined as the deviation of instantaneous data from the historical data from the historical data generated by the rated node's neighbors.

Arabsorkhi, 2016 [3]. The work of Arabsorkhi et al. presents the general principle behind many proposed trust management models considering ratings given to network nodes for the quality of the services provided over a specific time period. If the rating node has enough information to determine the trust value from its own ratings over the specified time period (by direct observation) it can proceed to calculate the trust value of the node to be rated. If not, then the rating node can query the rest of the network and aggregate the trust values assigned by the other network nodes to the rated node.

Yuan, 2018 [340]. This model considers ratings given after node interaction for the quality of provided services. The network model consists of IoT edge nodes being part of a domain federated by an edge broker node, which in turn contact a central cloud server responsible for the final calculation of trust values. Three trust values are calculated: the direct trust about a device to another device (D2D direct trust), the feedback trust about a node by an edge broker (B-to-D feedback trust), and the overall trust (the final trust value) about a device. D-to-D direct trust is updated and based on the history of direct interaction between nodes, it is defined as the ratio of positive interactions and the number of total interactions between the two nodes. B-to-D feedback trust is updated by the edge broker periodically and is based on all the D-to-D direct trust values concerning an edge node (except self-ratings); the edge broker aggregates the D-to-D direct trust values using weights derived by use of object information entropy theory, overcoming the limitations of assigning the weights manually. The overall trust value is calculated as the weighted sum of the D-to-D direct trust and the B-to-D feedback trust, thus considering the opinion of the rating node as well as the opinion of the whole network about the rated node.

3.3.1.4 Qualitative characteristics

Table 3.5 summarizes the qualitative characteristics of the surveyed trust models. The following characteristics are included in this summary:

- *Inference:* which mechanisms are employed for inferring trust values based on recommendations?
- *Trust scaling*: which is the range of the trust computation function?
- *Advantages*: which are the strong points of the model?
- *Complexity*: comments on space, time, processing, memory and communication complexity of the model.
- *Limitations*: aspects that constrain the effectiveness or the applicability of the model.
- *Monitored behavior*: which activities and evidence are collected to support the calculation of the trust metric?
- *Trust metric*: lists the dimensions expressed within the trust metric, such as honesty, reputation etc.
- *Context*: refers to the environment for which the model has been developed for.
- *Semantic meaning:* lists how the approach to trust computation is interpreted at a high level of abstraction. For instance, some could be experience-based or reputation-based, while some others could be application-specific or application-agnostic. Note that multiple orthogonal dimensions can be involved here.

Table 3.5. Overview of qualitative trust characteristics

Model	Inference	Trust Scaling	Advantages	Complexity	Limitations	Monitored behavior	Trust metric	Context	Semantic meaning
[101]	Multiplication for transitivity and weighted sum of trust values for aggregation.	Continuous [0,1].	Its trust-based service composition outperforms random service composition and approaches the maximum achievable performance from ground truth.	Node storage needed to keep trust values.	Hostility is considered to be increasing only over time in the simulations. When ground trust changes dynamically, recommendations don't contribute to convergence speed.	Honesty: estimated by keeping a count of suspicious dishonest experiences observed over a time interval using a set of anomaly detection rules such as high recommendation discrepancy as well as interval, retransmission, repetition, and delay rules. Cooperativeness trust: of node i towards node j is the ratio of the number of common friends over the number of node i's friends. Community-interest trust: of node i towards node j is the ratio of the number of common community/group interests over the number of node i's community/group interests.	Honesty, Cooperativeness and community-interest.	Community of interest (CoI) based social IoT (SIoT) systems. Devices have owners and owners have many devices. Each owner keeps a friends list. Nodes belonging to similar communities are more likely to have similar interests or capabilities.	Comprehensive
[149]	Multiplication for transitivity and	Continuous [0,1].	Its trust-based service composition outperforms	Node storage needed to	The storage needs can be	Honesty: estimated by keeping a count of	Honesty, Cooperativene	Community of interest (CoI)	Comprehensive. Although a

	weighted sum of trust values for aggregation.		random service composition and approaches the maximum achievable performance from ground truth.	keep trust values. The storage cost per node is $O(N_T N_X)$, where N_T is the number of IoT devices and N_X is the number of trust properties.	excessive for IoT devices with limited memory space.	suspicious dishonest experiences observed over a time interval using a set of anomaly detection rules such as high recommendation discrepancy as well as interval, retransmission, repetition, and delay rules. Cooperativeness trust: of node i towards node j is the ratio of the number of common friends over the number of node i's friends. Community-interest trust: of node *i* towards node *j* is the ratio of the number of common community/group interests over the number of node i's community/group interests.	ss and Community-interest.	based social IoT (SIoT) systems. Devices have owners and owners have many devices. Each owner keeps a friends list. Nodes belonging to similar communities are more likely to have similar interests or capabilities.	general approach for overall trust formation is not discussed.
[102]	Multiplication for transitivity and weighted sum of trust values for aggregation.	Continuous [0,1].	It introduces a storage management strategy suitable for large-scale IoT systems. Newly joining nodes can build their trust very quickly through available recommendations.	The storage management strategy is very efficient, "find medium, maximum and minimum operations have a	Trust recommendations can be biased when the recommender is from a different CoI.	Honesty: estimated by keeping a count of suspicious dishonest experiences observed over a time interval using a set of anomaly detection rules such as high recommendation discrepancy as well as	Honesty, Cooperativeness, Community-interest	Community of interest (CoI) based social IoT (SIoT) systems. Devices have owners and owners have many devices. Each owner	Comprehensive.

			The simulations considering the limited storage method, where storage management was used, achieved similar performance level with the unlimited space simulation and even better trust convergence time.	complexity of O(1) by using the max-min-median heap and all other operations (find, insert, delete) can be performed in O(logn) time.	The case in which a new node joins when the systems hasn't converged yet is not tested.	interval, retransmission, repetition, and delay rules. Cooperativeness trust: of node *i* towards node *j* is the ratio of the number of common friends over the number of node i's friends. Community-interest trust: of node *i* towards node *j* is the ratio of the number of common community/group interests over the number of node i's community/group interests.		keeps a friends list. Nodes belonging to similar communities are more likely to have similar interests or capabilities.	
[150]	Multiplication for transitivity and weighted sum of trust values for aggregation.	Continuous [0,1].	It introduces a storage management strategy suitable for large-scale IoT systems. The weights for combining social similarities are adjusted dynamically and this leads to credible trust feedback and minimized trust bias. It outperforms EigenTrust [183] and PeerTrust [76] in trust convergence, accuracy, attacks resiliency.	The storage management strategy is very efficient, "find medium, maximum and minimum operations have a complexity of O(1) by using the max-min-median heap and all other operations (find, insert,	Only persistent attack patterns considered, i.e., malicious nodes perform attacks with a probability of 1 or whenever there is a chance. The determination of the optimal trust decay parameter by means of	User feedback (binary: satisfied/not satisfied). Friendship similarity: the cosine similarity of the two users' friends lists. Social contact similarity: cosine similarity of the two users' locations lists. Community of interest similarity (CoI): cosine similarity of the two users' devices lists.	User satisfaction based on service completion, Friendship, Social- contact, Community-interest.	Service oriented architecture (SOA) based social IoT (SIoT) systems. Devices have owners and owners have many devices. Each owner keeps a friends list. Nodes belonging to similar communities are more likely to have similar	Comprehensive.

			Simulations considering limited storage had same performance as the ones with unlimited storage.	delete) can be performed in O(logn) time.	convergence and accuracy tradeoff based on environment conditions is left for future work.			interests or capabilities.	
[67]	Multiplication for transitivity and weighted sum of trust values for aggregation.	Continuous [0,1].	Fast convergence. The model reduces energy consumption caused by the presence of malicious nodes. Better performance from BTRM-WSN [206] and DRBTS [122] in both packet delivery ratio and detection probability of malicious nodes.	-	-	End-to-end packet forwarding (EPFR): the ratio between the numbers of packets received by the destination nodes to the number of packets sent by the source node. Energy consumption (AEC): the nodes' energy consumption ratio. Package delivery ratio (PDR) calculated by packet loss and packet retransmissions.	End-to-end packet forwarding (EPFR), Energy consumption (AEC), Package delivery ratio (PDR).	Wireless sensor networks of IoT and cyber-physical systems (CPS). Highly dynamic topology.	Fuzzy logic based trust.
[264]	Multiplication for transitivity and fuzzy membership function mapped to three linguistic	Continuous [-1,1] mapped to three linguistic values "Low",	The framework is scalable in terms of number of nodes and number of trust linguistic terms, without affecting performance.	-	-	Experience (EX) metric calculation is based on past interactions. When interaction is successful it has a value of +1 and a value of –1 otherwise. The	Experience (EX), Knowledge (KN), Recommendation (RC)	A fuzzy trust based access control (FTBAC) framework for IoT is discussed. It focuses on	Fuzzy logic based trust.

	values which are in turn used to create rules that form overall trust	"Average" and "Good"	The simulations show that energy consumption is less in access control with FTBAC than without. Furthermore, residual energy is higher in access control with FTBAC than without.			final value is relative to past interactions values sum. Knowledge (KN) is calculated based on "direct and indirect knowledge" but the monitored behavior to obtain these values is not discussed. Recommendation (RC) metric is calculated with the use of RC values from other devices for the trustee.		permissions that are assigned to a device based on the service provider's trust towards this device.	
[344]	Multiplication for transitivity with no aggregation between direct and indirect trust.	Continuous [0,1].	-	-	-	Service satisfaction for direct trust is not discussed in detail. For recommended trust a satisfaction level is defined which depends on availability, processing capacity, recovery time. Connectivity and peak-load performance.	Service satisfaction.	This trust model is defined in the context of Software as a Service (SaaS) in cloud environments. It is perceived, that a consumer will ensure the trustworthiness of the relevant service providers before accessing a service.	Evidence, experience and reputation based trust.
[369]	Multiplication for transitivity and weighted	Continuous [-1,1].	Its context-aware multi-service approach introduces a high level of	-	-	Service satisfaction: the service for which the node was evaluated, the	Service satisfaction	A trust management system for the	Evidence, experience, application-

	average for aggregation but with centralised propagation (only recommendations are used).		sophistication in trust management.			resources-based capability of the node at the time of the service request, the time at which the service was requested. How the service satisfaction is evaluated depends on the service.	(negative/positive).	IoT which takes into consideration that an IoT network can contain different kinds of devices providing different kinds of services. On this basis it proposes a context-aware and multi-service approach.	specific and similarity-based trust.
[330]	Only direct trust.	Continuous [-1,1].	A multi-service approach is followed.	-	The model convergence time is relative to the number of nodes and simulations show it doesn't scale well.	Service satisfaction is relative to the service for which the node was evaluated. Services are valued based on their processing and energy requirements. More demanding services have a higher weight value.	Service satisfaction (negative/positive). The positive/negative follows an award/punishment logic of the trustor is weighted based on the above.	A trust management system for the IoT which takes into consideration that an IoT network can contain different kinds of devices providing different kinds of services. On this basis it proposes a multi-service approach.	Evidence, experience, and application specific based trust.
[302]	No recommendations. Calculations	Continuous [-1,1].	The MAPE-K control feedback loop improves trust level consistence	-	-	The IoT sensors send raw data that they collect, and the representation of the	Availability: availability of resources,	This model proposes a framework for	Evidence and experience based trust.

	are done on the central TMS which receives raw data from sensors.		over time in a highly dynamic environment as opposed to the simulations run without feedback.			data differs based in the trust metric they are referring to. Examples include: A sensor that senses availability would send the number of successful ping requests. A sensor that senses reliability would send the Bit Error Rate (BER) of the target environment. Response time could be evaluated based on the round-trip time and capacity based on the current sessions of a device and maximum number of connections to a device.	Reliability: a reliable system always produces correct information, Response time: irregularities in response time can mean a device is compromised, Capacity: accessibility and scalability	integrating cloud and IoT in order to develop a cloud-based autonomic TMS which evaluates the level of trust in an IoT cloud ecosystem.	
[368]	Multiplication for transitivity and weighted mean of evidence for aggregation	Continuous [0,1].	He achieves better performance than both resilient-RPL (rRPL) and classical-RPL (cRPL) in various tests. When the network size increases the number of bad paths is reduced as opposed to the other implementations where it increases. In the simulation where the number of bad nodes varied the proposed trust	-	A small false positive rate is associated with bad nodes detection.	Nodes monitor the activity of their neighbors. So, a node *x* sends a packet to a node *y* to be forwarded. If the *y* forwards the packet correctly and timely, *x* increases the value of positive experiences, otherwise the value of negative experiences.	Trust metrics are belief, disbelief and uncertainty. They are relevant to the number of positive and negative experiences.	This model proposes a trust-based extension of the RPL routing protocol.	Evidence and experience based trust.

			RPL (tRPL) has less bad paths than the other two. Better packet delivery ratio. tRPL can detect 80% of bad nodes successfully.						
[154]	Indirect trust is not weighted and aggregation is done through average of direct and indirect sum.	Continuous [0,1].	Simulations show that it avoids malicious paths better than classical RPL.	-	This model uses additional hardware embedded in every device for security computations and processing.	For the ERNT metric computations nodes monitor their neighbors for selfishness, energy, and honesty.	Extended RPL Node Trustworthiness (ERNT)	This model proposes an alternative scheme for the RPL protocol.	Comprehensive.
[161]	Indirect trust is not discussed.	Continuous [0,1].	It achieves high levels of security.	The use of T-IDS is resource-demanding in both storage and communication overhead.	This model uses additional hardware embedded in every device for security computations and processing.	For the ERNT metric nodes are monitored for honesty, energy and mobility.	Extended RPL Node Trustworthiness (ERNT)	This model considers the work of [154] and extends it by proposing a trust-based IDS (T-IDS).	Comprehensive.
[222]	Subjective: Multiplication for transitivity and weighted sum for aggregation. Objective: The computation is	Continuous [0,1].	In the simulations with Class 1 malicious objects, both Subjective and Objective models outeperform TVM/DTC [206] and TidalTrust [160] models.	-	-	Subjective: Feedback: each node evaluates the service received with a value in [0,1].	Subjective/Objective: Feedback, Total number of transactions, Credibility, Transaction factor,	Trustworthiness for the social IoT. Two separate models are proposed, namely, Subjective and Objective	Comprehensive.

	done by a dedicated node based on feedback by the other nodes and nodes retrieve trust values from it. The feedbacks are weighted based on the credibility and transaction factor metrics.					Total number of transactions between two nodes. Credibility: the credibility of the recommender is based on the direct trust of the recommendation receiver towards the recommender and the centrality of the recommender. Transaction factor: the relevance of a transaction considered between two nodes to discriminate relevant from irrelevant ones. Relationship factor: based on the nature of the relationship between two nodes different values are assigned. The relationships include but are not limited to ownership object relationship and co-location object relationship. Notion of centrality is based on the sequence	Relationship factor, Notion of centrality, Computation capability.	Trustworthiness.	

						of social links that form the path between the two nodes. Computation capability: Objects with greater computation capabilities are considered as more capable of malicious activities. Objects are divided in two classes. Class 1 includes objects with great computational capabilities, such as smartphones. Class 2 includes objects with only sensing capabilities, such as a sensor. Objective: Feedback: same as in Subjective. Total number of transactions: same as in Subjective. Credibility: depends on relationship factor, computation capabilities and total			

						number of transactions. Transaction factor: same as in Subjective. Relationship factor: same as in Subjective. Notion of centrality: is based on the number of times the node requested a service, the number of times it acted as an intermediate node in a transaction, and how many times is has provided a service. Computation capability: same as in Subjective.			
[303]	Dempster-shafer: based on an algorithm. Verification od interaction proof (VIP): based on a ratio of the positive/negative interactions a reader had.	Dempster-Shafer: {Trusted, Malfunctioning, Malicious} Verification of interaction proof: Continuous [0,1].	Fast trust convergence. Able to support large scale RFID applications. Both Dempster-Shafer and VIP outperform the Bayes-based scheme in convergence speed, malicious event detection rate.	-	-	Behavior: Discarding data, Tampering with data, Replaying or forging data. Positive interactions: RFID tag rates a reader with 0 for negative and 1 for positive.	Dempster-Shafer: Behavior. Verification of interaction proof (VIP): Ratio of positive interactions.	This model proposes a trust management system for multi-domain RFID systems. The RFID system model consists of one or more domains and each domain	Evidence, experience and reputation based.

								includes RFID tags, RFID readers, authentication centers and an administration center. A centralized trust propagation approach is followed.	
[175]	Adaptive neuro-fuzzy inference.	Not clarified. (pg. 8-9)	The proposed model TMM outperformed TRM [67] in both packet forwarding ratio and energy efficiency. It also outperformed AODV and trusted-AODV in throughput. TMM has higher accuracy and f-measure than a model using a fuzzy inference system instead of the adaptive neuro-fuzzy inference system.	-	-	Behavioral trust: Relative frequency interaction: relative to the number of interactions between the nodes and the total number of interactions with other nodes over the same period of time. Intimacy: relative to the time of interaction between two nodes and the cumulative time of interactions with other nodes. Honesty: based on the numbers of successful and unsuccessful interactions.	Behavioral trust: Relative frequency interaction, Intimacy, Honesty. Data trust: Direct, Indirect.	This model proposes a Neuro-Fuzzy based Brain-inspired trust management model for cloud based IoT architectures security and data reliability.	Comprehensive.

						Data trust: deviation of node's current data from the historical data of the node. In both direct and indirect evaluations.			
[340]	Recommendations are computed and provided in a centralized manner. Objective information entropy theory is used for transitivity. Weighted sum for aggregation.	Continuous [0,1].	It achieves better global convergence time and task failure ratio than PSM and DRM [337]. It's lightweight in terms of complexity.	Space complexity (communication overhead): $3*m*n*\delta$, m: number of clusters, n: size of clusters, δ: the maximum number of trust computing for a given Δt. Time complexity: the total time complexity of overall trust evaluation is $O(n^2)$.	-	After each transaction, each participating node evaluates the other node based on service completion and sends the value to the broker.	Service satisfaction.	This model proposes a trust computing mechanism specifically designed for IoT edge computing.	Evidence, experience and reputation based trust.

3.3.2 Trust management architectures

In section 3.3.1 we have reviewed the existing trust management systems and their features. In this section, we survey the relevant trust management architectures, which dictate how the TMS components are deployed in the target network, the relationships between the components and the information flows. A trust management system involves a number of different components that are involved in the various activities taking place within the system; taking into account the aspects of trust models identified in section 3.3.1, we can identify the following types of components:

1. *Data collection components*. These components collect the necessary data for performing trust assessment, which range from consumer satisfaction, QoS aspects, suspicious/dishonest behavior and so forth.
2. *Data storage components*. These components store the data collected by data collection components and make them available for trust calculations.
3. *Trust calculation components*, which extract the data from data storage components and calculate trust. In this process, they may query other trust calculation components regarding their trust assessments and use the replies in their computation.
4. *Trust consumers*, which query trust calculation components regarding trust assessments and use the obtained values for implementing security policies.

While data collection and storage components as well as trust consumers are typically dispersed across the network, trust calculation components are laid out according to different paradigms, and these layouts characterize the trust management system architecture. Overall, the following categories are identified for trust management architectures: (a) centralized, (b) hierarchical, (c) distributed/Peer to peer. In the following subsections we elaborate on each of the categories, presenting its features and prominent application cases.

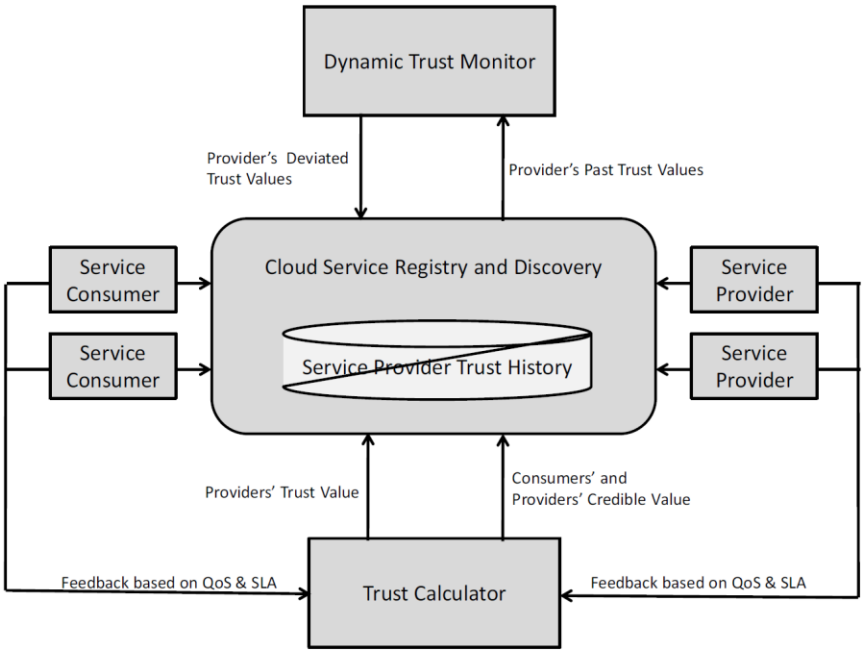


Figure 3.8. A centralized trust management system architecture

3.3.2.1 Centralized

The centralized architecture paradigm involves a unique trust management authority, which collects all the information necessary for trust calculation and computes the trust score for entities. Then, interested parties can query the trust score of entities, subject to suitable authorization.

Centralized systems are known to have scalability and reliability issues, hence only few systems have been reported in the literature to follow this paradigm. Figure 3.8 presents a reference centralized TMS architecture from [214]. In this architecture, a single trust calculator collects all the metrics related to trust computation and computes the trust metric for entities. Both service providers and consumers are assigned a trust score (credibility): the trust score assigned to service consumers moderates the weight of the trust metrics they contribute to the system.

3.3.2.2 Hierarchical

The hierarchical architecture paradigm identifies clusters of nodes, where each cluster elects a coordinator. Nodes within a cluster liaise with the coordinator, exchanging observations, metrics and trust values; the coordinator is responsible for synthesizing the trust assessments of the nodes within the cluster it coordinates into a comprehensive trust score and for communicating with other coordinators to exchange trust values. Hierarchical architectures are well-suited for IoT infrastructures, where nodes with limited resources are placed under the coordination of the corresponding gateway node, which is more resource-rich and can host resource-intensive operations. Respectively, trust assessments are generally performed having available more detailed local data (measurements and observations obtained and collected at cluster level) whereas inter-cluster communications are limited to the exchange of either trust assessments or data summaries, rather than detailed data.

Nodes in hierarchical systems may be organized across multiple levels of hierarchy. In this line, [331] describes an architecture where nodes are clustered into autonomic nodes, and autonomic node contains multiple autonomic Decision Entities (DEs). In turn, A DE is introduced in the Generic Autonomic Network Architecture (GANA) designed to follow hierarchical, sibling and peering relationships with other DEs within a node or network. It collects information from peering DEs or sibling DEs, makes decisions and manages Managed Entities (MEs) at a lower level i.e., the level of abstracted networking functions. DE is the element that drives the control-loop over the MEs and implements the self-* functionalities e.g., self-configuration, self-monitoring, self-healing.

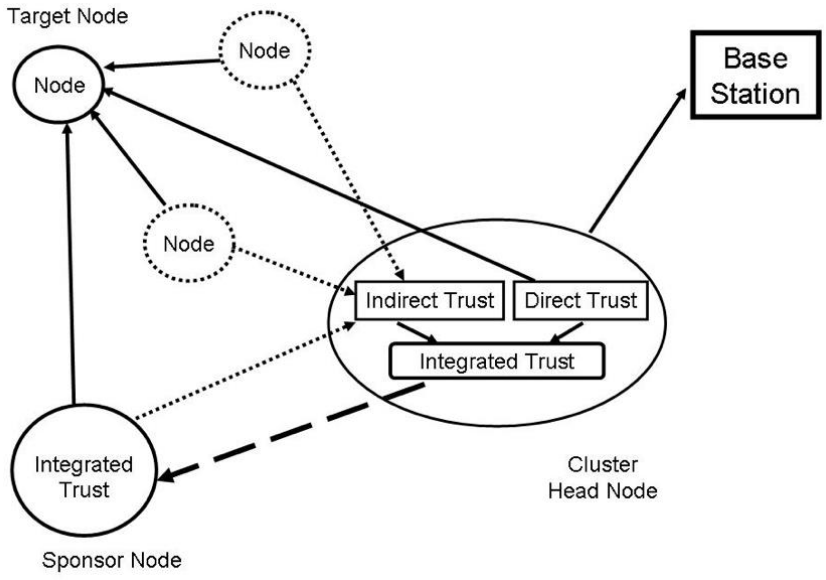


Figure 3.9. Internal structure of a cluster [176]

Figure 3.9 presents the internal structure of a cluster, as per the design reported in [176], whereas Figure 3.10 depicts the organization of multiple clusters into a hierarchical trust management system [331]. [120] is also an example of a hierarchical architecture, applied in peer-to-peer nodes having super-peers.

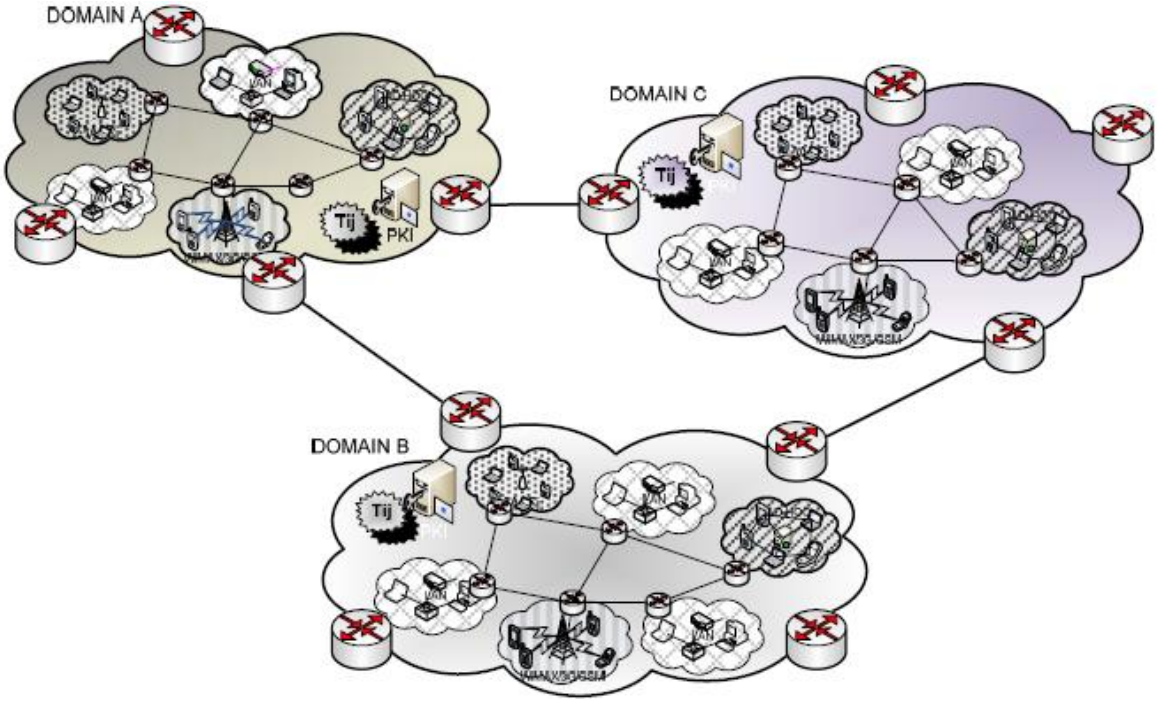


Figure 3.10. Integration of multiple clusters into a hierarchical trust management system [331]

3.3.2.3 Distributed/Peer to peer

In this architectural paradigm trust management components are dispersed across the network and operate autonomously. Each node makes its own observations and measurements, and maintains them into a local database. Nodes may also request from other peer nodes either detailed measurements and observations or synopses of measurements of measurements and observations, or trust assessments; then, they compute a trust score for other entities, synthesizing their own data and the data they have received.

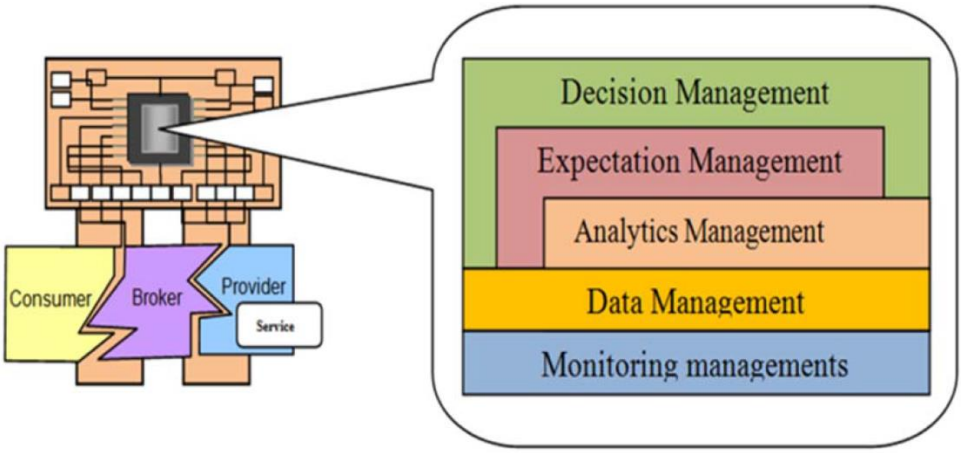


Figure 3.11. Components within a peer node participating in a TMS [138]

Figure 3.11 illustrates the components within a peer node participating in a TMS. The node itself responds to requests from other nodes for trust assessments (and provisionally other data), while it can become itself a client to other nodes, requesting trust assessments (and provisionally other data).

The distributed/peer to peer paradigm is the most widely used in the literature: [138], [86], [129], [128], [284], [151] are typical cases where this paradigm is used.

3.4 Trust management systems

In the previous subsections, we have surveyed of trust management methods, protocols, algorithms and architectures; in this section, we examine trust management implementations, focusing on the open source implementations, which could be used as a basis for the implementation of the Cyber-Trust TMS. To this end, we have performed extensive searches in the major open source repositories, namely GitHub[21] and SourceForge[22]. For each of the trust management systems located, we performed an initial assessment, dropping those repositories that were incomplete or not adequately populated (i.e., contained only a few files or only documentation with no concrete implementations). Subsequently for the remaining implementations, we considered the following aspects:

1. *Domain of use*. The intended domain of use for the software was assessed, examining whether the system was oriented to the computer security trust domain or other trust domains with different concepts and requirements; in particular, many systems were oriented towards *financial trust*, not being thus suitable for use within the Cyber-Trust project.
2. *Functionality*. The functionality offered by the system was analyzed, considering whether REST/web service/remotely invokable APIs are offered, the existence of UIs and in particular web-based UIs and the algorithms implemented.
3. *Extensibility, modifiability, active support and documentation*. These properties are required for adaptation of a system to the needs to Cyber-Trust to be accommodated. Extensibility and modifiability are contextualized for the Cyber-Trust project, considering the expertise of the consortium in the implementation language and environment.
4. *Deployability to Cyber-Trust target platforms*. The Cyber-Trust system specifications dictate that TMS instances will be running on data centers, smart gateways and smartphones; each deployment target has its own runtime environments and resource capabilities and TMS implementations should be able to run efficiently on top of all these deployment targets.

In the following paragraphs, we present the open source TMS implementations surveyed. Systems designed for use in other domains, and therefore not being useful for the context of Cyber-Trust are briefly described in subsection 3.4.8.

3.4.1 Soutei

Soutei (https://sourceforge.net/projects/soutei/) is a trust-management system for access control in distributed systems. Soutei policies and credentials are written in a declarative logic-based language. Soutei policies are modular, concise, readable, supporting conditional delegation. Policies in Soutei support verification, and, despite the simplicity of the language, express role- and attribute-based access control lists, and conditional delegation. They support policy verification, and, despite the simplicity of the language, express role- and attribute-based access control lists, and conditional delegation.

Soutei provides a number of interesting concepts, and may model, among others, role-based access control, capabilities, policies predicated on time and lists, trees, organizational charts & partial orders. It provides a TCP server hence functionalities can be remotely invokable. Its documentation however is limited, hindering installation, configuration and maintainability. The Cyber-Trust consortium does not have adequate experience with the Haskell language, hence the extension and maintenance of the software will be further hindered. The provided implementation is almost 10 years old and in many aspects it is incompatible with the recent developments of the Haskell language; Soutei does not compile and run successfully under the recent versions of the Haskell compiler; it has been found to compile successfully under version 6.8.3 of the Haskell compiler, which is severely outdated and may contain functionality or security issues. Finally, running

[21] https://github.com/

[22] https://sourceforge.net/

Haskell on Android devices, which is a significant target for the Cyber-Trust project, has only been reported in 2018, and requires the use of low-level techniques, such as Java Native Interface (JNI) or the Native Development Kit (NDK) [140] which introduces an additional set of required programming skills and another level of mapping which increases the probability of errors. The requirement to use version 6.8.3 of the Haskell compiler will probably introduce additional compatibility issues with the Android platform.

Considering the above, Soutei is not a prospective candidate for use in the Cyber-Trust project.

3.4.2 Trust guard

Trust Guard (https://github.com/blatyo/trust_gaurd) is an implementation of an algorithm for countering vulnerabilities in reputation management for decentralized overlay networks, introduced in [228]. This implementation dates back 9 years and is reported by the author as "untested". It includes some code on how trust values are computed/updated according to newly arriving information and old estimates decay with time in favor of new information. The implementations are simple and can be directly derived from the reference paper [228]. No server is provided, hence no remote invocation is possible; only running through the command-line is supported. Obviously, the implementations can be wrapped within web service containers, however considering the simple nature of the implementations and the fact that the project is untested, the benefits from using the Trust Guard are minimal and additionally introduce the need for testing and binding to the Ruby platform.

Considering the above, Trust Guard is not a prospective candidate for use in the Cyber-Trust project.

3.4.3 pyKeynote/keynote library

pyKeynote (https://github.com/argp/pykeynote) is a Python extension module for the KeyNote trust management system [208]. It provides a high-level object-oriented interface to the KeyNote trust management API. The implementation is very outdated, with its last update dating back 12 years, and it relies on the keynote library (http://www1.cs.columbia.edu/~angelos/keynote.html) which dates back at an even older timepoint (2000). Some concepts of the keynote library –which is written in C-, including assertions and grants are usable. The C library accommodates provisions for local invocations, it could however be wrapped under remotely invokable containers.

Considering the above, the concepts of the Keynote library will be examined for inclusion in the Cyber-Trust TMS implementation, and possibly some code can be ported to the TMS implementation language.

3.4.4 SAFE

SAFE (https://github.com/wowmsi/safe) is an integrated system for managing trust using a logic-based declarative language. Logical trust systems authorize each request by constructing a proof from a context---a set of authenticated logic statements representing credentials and policies issued by various principals in a networked system. Two informal publications ([334] and [273]) describe the theoretical and practical basis of the system. SAFE aims to address the problem of managing proof contexts: identifying, validating, and assembling the credentials and policies that are relevant to each trust decision. The approach of SAFE to managing proof contexts is using context linking and caching. Credentials and policies are stored as certified logic sets named by secure identifiers in a shared key-value store. SAFE offers language constructs to build and modify logic sets, link sets to form unions, pass them by reference, and add them to proof contexts. SAFE fetches and validates credential sets on demand and caches them in the authorizer. We evaluate and discuss our experience using SAFE to build secure services based on case studies drawn from practice: a secure name service resolver, a secure proxy shim for a key value store, and an authorization module for a networked infrastructure-as-a-service system with a federated trust structure [334], [273].

The SAFE implementation is distributed under the Apache 2.0 license, which is permissive, so it can be reused, either as a whole or at the level of selected portions. Scala implementations can be run on Android (https://scala-android.org/) with a small footprint, however more extensive tests should be made to determine the footprint of the particular implementation.

The code implementing SAFE has some high-level documentation regarding the architecture, however the documentation on compiling and running the code is lacking. No executable commands or relevant sources are present in the default distribution, hence execution procedures cannot be determined. The code implementing the SAFE TMS lacks comments, therefore code modifiability and extensibility is low.

Considering the above, the concepts of the SAFE TMS will be examined for inclusion in the Cyber-Trust TMS implementation, and possibly some tools can be accommodated in the Cyber-Trust TMS implementation, the code base however will not be used on an “as-is” basis.

3.4.5 TMLib

The *tmlib* system (https://github.com/pchapin/tmlib) is a library of functions that allow applications to support trust management style distributed authorization. The TMlib library is reported to provide an administrative application that can be used to create and manually verify certificates in multiple certificate formats. In addition this library provides functions for performing a proof of compliance computation that can be used in any application that wishes to use trust management services.

TMLib assumes that the participating nodes specify local policies and encrypts the communication between nodes. Node identity is proved by means of certificates. Fundamentally TMLib is a library of functions that can be called by an application that is interested in trust management services. However, there are a number of administrative tasks that any node must support in order for the system to be usable. Accordingly, TMLib comes with an administrative application that allows its user to perform certificate creation, managing of public key and policy databases, as well as executing test queries and setting policy dissemination rules.

TMLib has an undocumented dependency on an ACO project (presumably *ant colony optimization*) written in ADA; however, no such open-source project or relevant files could be located. Hence. It was not possible to compile and test the project. Furthermore, consortium expertise with ADA is very limited, hindering thus modifiability and extensibility. Considering additionally the lack of documentation, TMLib is not a prospective candidate for basing the Cyber-Trust TMS implementation.

3.4.6 Cloud trust protocol daemon

The Cloud Trust Protocol Daemon (https://github.com/CloudSecurityAlliancePublic/ctpd) is a prototype server implementing the Cloud Security Alliance's Cloud Trust Protocol. The Cloud Trust Protocol (CTP) is designed to be a mechanism by which cloud service customers can ask for and receive information related to the security of the services they use in the cloud, promoting transparency and trust. This prototype called *ctpd* is a Unix-style server written in Go with *mongodb* as a database backend. It has been tested on Ubundtu/Debian Linux and Mac OS X. The code of *ctpd* is reported to be still in 'beta' stage and is mainly intended for testing and research purposes.

ctpd aims to fully implement the CTP data model and API [58], as well as the non-official CTP 'back office' API [59]. This API includes provisions for managing the concepts of:

- *service views*: represents a service offered to a specific customer under the responsibility of a single provider. This service is usually described in a SLA or service interface. A service-view encompasses a set of assets.
- *Assets*: used to represent any tangible or intangible element of a cloud information system, such as for example simple API URLs, storage, processor cores or compute instances, databases, full blown platforms, etc. A set of attributes is attached to an asset.
- *Attributes:* used to represent characteristic of an asset that can be evaluated quantitatively or qualitatively and is identified with a distinct name (e.g., “availability”, “incident response”, etc.). Associating a value with a security attribute requires the specification of a measurement.
- *Metrics:* a standard of measurement, which will be referenced in measurements. A metric is typically specified in an external document that describes in human readable form the conditions and the rules for performing the measurement and for understanding the results of a measurement.

- *Measurement,* which describes how a specific attribute is evaluated, using a specific metric.
- *Triggers*, which enable cloud service customers to receive notifications when specific conditions are met. A trigger is a conditional expression on the measured value of a security attribute.
- *Log entries*, which are generated by triggers.
- *Dependencies*, which are used to describe relationships between different cloud services that are associated together to form a cloud supply chain.

Notably, the Cloud Trust Protocol Daemon does not compute any trust or risk metric: its purpose is to manage the concepts listed above, which can be used (among others) in the formulation of trust scores, the delivery of notifications (through triggers) and the creation of persistent log entries. However, it should be noted that triggers are not fully implemented (trigger deletion is lacking), and XMPP-based notifications (through which notifications raised by triggers are delivered) are not implemented at all (c.f. [57]).

Under this view, the utility of CPTD for the implementation of the Cyber-Trust TMS is limited, and will not be further considered towards this direction.

3.4.7 Retrust

Retrust (https://github.com/liamzebedee/retrust) is a work-in-progress protocol for decentralized reputation/trust, based on Evidence-Based Subjective Logic (EBSL) [38]. The model is based on capturing interactions between nodes in the form *(source, target, value)*, with *value* being > 0 for trusted interactions and < 0 for negative interactions. In this model, trusted friends/seeds need not be specified or explicitly maintained, since this information is automatically derived from interactions. An application-agnostic mode is also considered, in which reputation is a subjective-logic opinion of (belief, disbelief, uncertainty) that can model any quality of reliability in interaction. Naturally, the implementation should provide implementations of the methods computing the reputation of entities: the reputation for an entity is *perspective-specific*, i.e. a single entity may be assigned multiple reputation scores, depending on the perspective under which it is evaluated. The perspective may be a self-view, the individual view of another entity or the view of a group of other entities (e.g. all entities belonging to a specific entity category such as servers within the demilitarized zone, or entities bound together with any arbitrary criterion).

The implementation is command-line based, therefore it is oriented towards single runs that retrieve interactions/evidence for entities, compute the results and display them and/or generate graphical representations for them. This means that substantial development effort should be put into modifying the code so as to provide server-type operation, i.e. daemonize the code and render it capable to responding to REST calls as well as receiving from third parties and maintaining trusted notifications regarding evidence, on which reputation computation will be based. Considerations also exist regarding the efficiency of the code: a simple simulation involving 10 “good” nodes, 20 “bad” nodes and 20 “Sybil” nodes, and having few interactions between nodes, 15.5 seconds were needed to run it on a Linux server with one 6-core Xeon E5-2420@1.90GHz CPU and 8GB of memory, and requiring a virtual memory size of 1.15GB (albeit the resident set was only 80MB; the required size of virtual memory can pose problems in mobile or –more generally- resource-constrained devices).

The concepts used in *Retrust* and the code implementing these concepts will be further examined for potential exploitation in the Cyber-Trust TMS implementation.

3.4.8 Systems in other domains of use

The *Linux SGX Trust Management Framework* (https://github.com/IBM/sgx-trust-management) is a system for supporting the Software Guard Extension technology, available in Skylake and later processors. SGX technology supports the creation of *enclaves*, i.e., secure memory regions that are protecteded with hardware encryption in the system-on-chip (SoC). In more detail, according to the SGX framework, the data exists in unencrypted format only inside the processor. Before being written to the main memory it is encrypted by the SoC and then decrypted by the SoC when fetched from the main memory.

TrustApp (https://github.com/dedicatedvivek/TrustApp) is a financial risk assessment application, with its core model involving banks and expenses. Furthermore, it lacks documentation and the code does not readily run on Unix due to some non-portable convensions.

TrustFeatures (https://github.com/hashinclude-co-in/kamban.org) is oriented towards NGO members and volunteer management; to this end it includes features such as contact management, survey management, event management, inventory management and task management. These aspects do not intersect with the functionalities needed in the context of Cyber-Trust.

Imob (https://github.com/zeqing-guo/imob) claims to implement "an identity and trust relationship management on blockchain for IoT", however no relevant functionalities or documentation are implemented in the code.

Trust Management System (https://github.com/shiwenbo/Trust-Management-System) accommodates the concept of nodes that are verified by credentials and assume roles, however there is no notion of trust and risk metric computation.

The *Tennessee Risk Management Trust* (https://github.com/lindseyemaddox/tnrmt) is oriented towards economic insurance, and its core concepts are loss control, property and liability, tort liability etc. In this respect, its scope does not intersect with the functionalities needed in the context of Cyber-Trust.

The *CA* system (https://github.com/pontiflex/trustme/tree/master/CA) is a web-based application, appearing to suite the management of certificates. Its functionality is limited, with a very limited overlap with the functionalities needed in the context of Cyber-Trust, while issues exist in the setup process and no updates have been provided for 6 years.

Trust composer (https://github.com/ricktobacco/trust-composer) is a web application for demonstrating a secure trust composer on blockchain using Hyperledger composer. The model realized by *Trust composer* involves claims for *service/resource accesses* issued by *users*; these claims are supported by *proofs*, whereas *assessors* are a specific subclass of users that provide guarantees to support a user's claims. Finally, *services* maintain a trust balance of users they manage and consider claims, together with associated proofs and assessor-issued guarantees to accept or deny requests. In more detail:

- *Users* are the super class of all other participants. They have a name and may create claim request assets, which may be further used to build trust for exchange against access balances with services.
- *Issuers* issue claim receipts based on users' requests, upon examination of proofs therein. They are the super class of services, subclass of assessors, and have a list of assessors operating on their behalf.
- *Services* upload resources with their associated access costs, and maintain a list of balances for the users that have requested access and also been granted trusts by assessors operating on behalf of the service.
- *Assessors* operate on behalf of services and issuers to package claim receipts, and assign levels of trust according to their own weights for various claim definitions (based on their verification specializations).

The example usage listed in the *TrustComposer* distribution is oriented towards the economics domain, persons submitting claims for damages they sustained from disasters or hospitals (as service providers) considers level of guarantee from assessors (e.g., insurance companies) to grant resources, some concepts might be used in the Cyber-Trust TMS; however, implementations that are closer to the Cyber-Trust domain, like SAFE and TMlib, provide more direct analogies, hence *TrustComposer* will not be further considered.

3.5 Trust and risk aware defense

Trust and risk metrics can be exploited in the context of defending against attacks. This exploitation may span across the whole defense activity spectrum, i.e.:

1. *Response to simple, one-step attacks*. Many attacks are simple, one-off attempts to exploit known vulnerabilities of a system/subsystem, with a goal of obtaining or elevating access to the particular system/subsystem. In these cases, defense actions typically involve the rejection of the offending activities, while responses may also escalate to blacklisting the attacker IP for a period of time; the scope of blacklisting may be (i) the specific service that sustained the attack, (b) the machine that hosts the service (e.g. fail2ban]) (c) the subnet within which the target machine resides or (d) the whole enterprise network. Forensic information collection can be also performed in the context of the response and alerts to incident response teams can be issued.
2. *Response to complex, multi-step attacks*. Contemporary attacks may include highly sophisticated sequences of malicious activities, comprising multiple steps that form *attack paths* [29]. In these cases, the goal of the attacker lies beyond the initially attacked service or system (probably at a more valuable asset), and the defense strategy should be crafted towards exploiting the information extracted from the attack patterns and the trust and risk scores to safeguard the most important assets of the organization. Attack and defense graphs [91], [323] are the predominant tools for modeling these types of attacks and deriving the appropriate defense actions.
3. *Performing risk analysis and prioritizing proactive risk mitigation actions*. In this context, trust and risk scores are used to determine the overall risk that existing vulnerabilities pose to the organizational assets; this information can be then exploited to prioritize defense actions, so that the available security budget can be directed to mitigation actions that will minimize the overall residual risk, or determine the security budget that should be afforded to confine the residual risk to acceptable levels.

These three response categories roughly correspond to the classification of responses to *immediate*, *short-term* and *long-term reactions* listed in [323]. In the subsections paragraphs we examine in more detail the exploitation of trust and risk scores for the cases of responding to simple, one-step attacks (subsection 3.5.1) and for determining proactive defense measures (subsection 3.5.2). The case of responding to complex-multi-step attacks is discussed in detail in Section 4, “Game-theoretic cyber-defense framework”.

3.5.1 Use of trust and risk for simple attack mitigation

As discussed in section 3.2, attacks may be flagged due to different observations: (i) activities performed by a device can be determined to be non-compliant as compared to a “ground truth”/”golden standard” behavior (e.g. as prescribed by a MUD file [88]); (ii) activities or an activity stream executed by the device can be found to be deviant from a dynamically built model; or (iii) activities performed by a device can be found to match known attack signature patterns.

When mitigating an attack flagged to originate from a specific device, the trust and risk assessment of the particular device can be expected to play a role of varying weight, depending on the method that has been used to flag the attack. More specifically:

i. If the attack has been flagged due to a deviation from a “ground truth”/”golden standard” behavior, then by virtue of the baseline’s nature, the flagging is known to be accurate; therefore, it is expected that of activities associated with the attack will be rejected (e.g. network packets will be dropped), regardless of the trust level that was known to be effective for the offending device prior to the attack detection. The device trust level as well as the risk level associated with the attack may however moderate the escalation of defense measures: for example a device with low trust may be blacklisted for a longer period of time, or at a wider scope than a device with a high trust level. Similarly, attacks that pose high risks may result to prolonged or wider-range blacklistings than low-risk attacks. Analogous provisions can be made for incident response team alerting. The way that risk level moderates escalation is preferably defined at a policy level, since policies guarantee easier comprehensibility, modifiability validation and consistency checking [379], whereas suitable policy instantiation engines can render the policy specifications actionable [380].

ii. If the attack has been flagged due to a deviation from a dynamically built model, then the trust level known to be in effect for the offending device prior to the attack can play a more significant role in

determining how the attack will be handled. This is due to the fact that dynamic model-based detection techniques are known to be prone to false positives, hence we can consider than the probability that the detection is due to a false positive is higher when the offending device is trusted to be benign, being correspondingly lower when the offending device is untrusted. This can be formally expressed as

$$trust(D_1) > trust(D_2) \Rightarrow p\big(falsePositive\ (a, D_1)\big) > p\big(falsePositive\ (a, D_1)\big)$$

where D_1 and D_2 are devices and a is a specific attack flagging. This directly stems from the definitions of trust, such as the ones in [43], where trust is defined as "An attitude of confident expectation in an online situation of risk that one's vulnerabilities will not be exploited", hence the confidence placed on a highly trusted device that it will not exploit the vulnerabilities of another device is still considered in the presence of *indications*, as is the case of behavior-based flaggings, as contrasted to the case of "*hard evidence*", such as the deviations from a "golden standard" behavior. In the cases that an attack flagging a based on behavior-based detection can be associated with a confidence *conf(a)* [378], we can compute a trust-aware confidence metric for the attack flagging, $conf_t(a) = f(conf(a), trust(D_1))$, where D_1 is the device flagged as the attack source and f is the combination function. Should the $conf_t(a)$ metric surpass some policy-specified threshold, then the attack-related packets will be dropped. Analogously, the risk metrics and relevant activities may be moderated by the offending device trust assessment, since they affect the belief that the attack has actually took place.

It has to be stressed here that adopting a security policy dictating that the device trust level moderates the moderates attack flagging procedure presupposes that a strong identity mechanism is employed in the context of attack source and device trust determination, since the attacker may try to spoof the identity of a trusted device in order to evade attack flagging and successfully proceed with the attack.

iii. If the attack has been flagged due to attack pattern matching, then the extent to which the trust level of the offending device will be taken into account depends on the level of confidence assigned to the recommendations of the signature-based scanner that flagged the attack. While attack signature-based detection is known to have a lower false-positive detection rate than dynamic model-based ones, especially if additional flagging criteria are added to simple pattern matching, such as stateful pattern matching and protocol decode-based analysis [381], in all cases a tradeoff exists between false positive and false negative rates, as shown in Figure 3.12 (adapted from [382]). Generally however, we do not expect that the detection threshold is set to a high value, since this would allow many actual attacks to evade detection. Therefore, similarly to the case of dynamically built models presented above, we can consider that when an attack is flagged to originate from a device with a high trust level it is more probable to be a false positive, rather than when a similar attack flagging relates to a device with a low trust level. Again, a strong identity mechanism should be employed in the context of attack source and device trust determination, to mitigate attacker efforts to spoof the identities of trusted devices. Analogously, the risk metrics and relevant activities may be moderated by the offending device trust assessment, since they affect the belief that the attack has actually took place.

Using the confidence level in attack flagging procedures can tackle some deficiencies of standard false positive elimination techniques, which ultimately resolve to deactivating certain detection patterns [383]: indeed, if –according to the standard practices– a detection criterion is deactivated to accommodate access for a device that is known to be benign, the attacks related to the deactivated patterns can be launched by any device, either deemed as benign or not. On the other hand, trusted devices that are compromised but have not yet been detected to be so, can be allowed to launch attacks, since the belief that an attack is actually launched will be lowered, due to the fact that the device trust level is (incorrectly) high. To this end, repetitive flaggings related to device with high trust can be set to automatically demote the device trust level or at least be examined to determine whether they actually correspond to attacks or not.

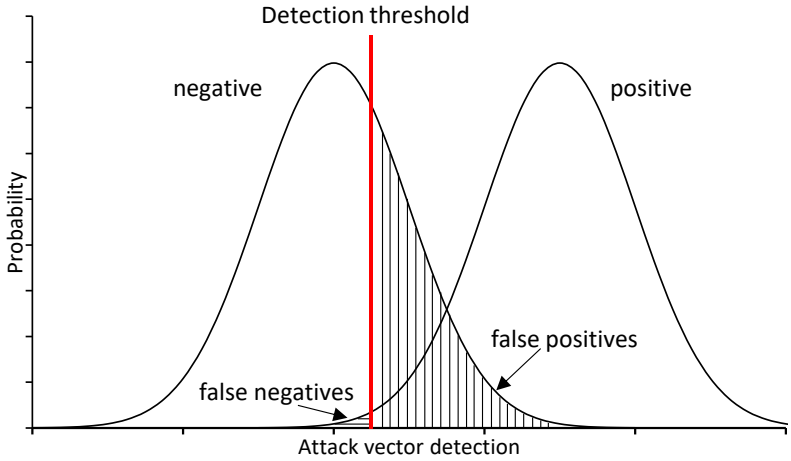


Figure 3.12. Detection threshold effect on false positive and false negative detection rate.

3.5.2 Use of trust and risk for proactive defense

Deliverable D2.5 [60, §6.2] has already made an introduction to dynamic risk modeling as a way to deal with the drawbacks associated with the conventional risk models. Such risk management frameworks are based on *graphical security models* (GrSM), and in particular on attack graphs –commonly *Bayesian attack graphs* (BAG) as in Figure 3.13– and probabilistic techniques to model and assess the identified risks.

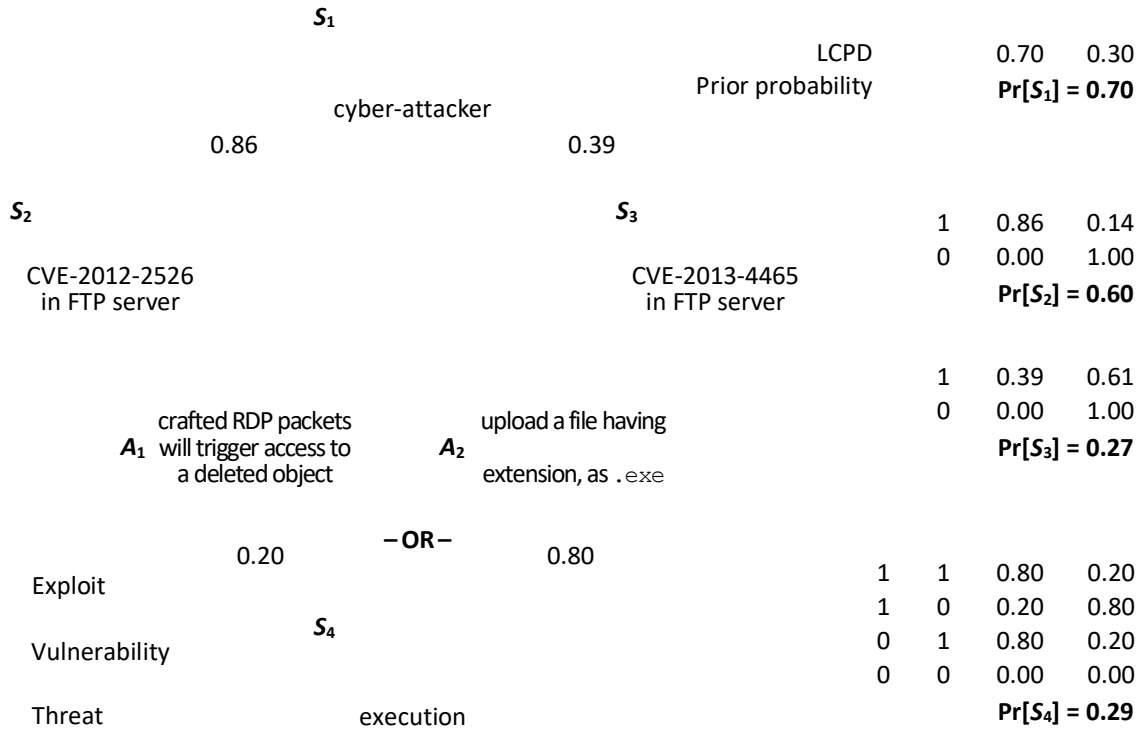


Figure 3.13. An example attack graph with the probability of successful attacks as labels

In order to perform risk analysis in BAGs a number of system attributes $S = \{S_1, S_2, \ldots, S_n\}$ are identified, which are defined based on the so-called *attribute-templates* [246]: these are a combination of generic properties, such as system vulnerabilities, insecure system/network properties, access privileges, etc. Each attribute S_i is binary-valued, `True` (1) or `False` (0), and if seen as a Bernoulli random variable it is also given a probability $\Pr[S_i]$ for the attribute S_i to hold true. Moreover, a number of *atomic* attacks $A = \{A_1, A_2, \ldots, A_m\}$ are defined as mappings $A_i : S \times S \longrightarrow [0,1]$; each attack A_i allows the attacker to compromise (with a nonzero success

probability) an attribute $S_{\text{post}} \in S$ (that depends on i), called post-condition, starting from a pre-condition $S_{\text{pre}} \in S$ (again depending on i). The realization of A_i is associated with the use of an exploit e_i and thus

$$A_i(S_{\text{pre}}, S_{\text{post}}) = \Pr[e_i]$$

where $\Pr[e_i]$ is the probability of successful exploitation; by convention we have that $\Pr[e_i] = 0$ whenever $S_{\text{pre}} = 0$. Various ways to assign a value to $\Pr[e_i]$, which are based on information obtained from the CVSS, have been reviewed in deliverable D2.5 [60]. The attribute S_{pre} is called *parent node* of S_{post} and is denoted as $S_{\text{pre}} = \text{Pa}(S_{\text{post}})$; note that an attribute S_i may have multiple parent nodes that are combined according to $d_i \in \{\text{AND}, \text{OR}\}$.

Formally, the Bayesian attack graph is the tuple $\text{BAG} = (S, \tau, \varepsilon, P)$, where $\tau \subseteq S \times S$ defines the underlying graph structure, ε is the set of tuples of the form $\langle S_i, d_i \rangle$ with the variable d_i defining how the parent nodes of S_i are combined, and P is a set of the discrete *local conditional probability distribution* (LCPD) functions representing the values of $\Pr[S_i|\text{Pa}(S_i)]$. The computations of the LCPD for the example attack graph are also shown in Figure 3.13. These are computed according to the following equations [246]:

$$\Pr[S_i|\text{Pa}(S_i)] = \begin{cases} \prod_{S_j \in \text{Pa}(S_i)} \Pr[e_j], & \text{if } d_i = \text{AND}, \\ 1 - \prod_{S_j \in \text{Pa}(S_i)} (1 - \Pr[e_j]), & \text{if } d_i = \text{OR}. \end{cases}$$

In the former case (the AND operator), the LCPD equals zero if $S_j = 0$ for at least one $S_j \in \text{Pa}(S_i)$, whereas in the latter case (the OR operator), the LCPD is equal to zero if $S_j = 0$ for all $S_j \in \text{Pa}(S_i)$, since $S_j = 0$ implies $\Pr[e_j] = 0$.

In order to utilize the above framework to proactively defend a network/system against the possible cyber-attacks, a security administrator should devise a security plan that selects appropriate *security controls* in the set $C = \{C_1, C_2, \ldots, C_k\}$ to mitigate the identified vulnerabilities and to minimize the expected probability of successful exploitation; clearly, for each security control $C_j \in C$ there should exist $S_i \in S$ such that it holds

$$\Pr[S_i|\text{Pa}(S_i), C_j = 1] < \Pr[S_i|\text{Pa}(S_i), C_j = 0].$$

If the above inequality holds, then the attribute S_i is in the *coverage* of the control C_j, and this is denoted by $(S_i, C_j) \in \Lambda \subseteq S \times C$, i.e. Λ represents the new edges between the controls and the attributes of the graph. Each security control $C_j \in C$ has an associated *implementation cost* $V_j > 0$ and can be included or not to the security plan; this is represented by the binary variable $M_j \in \{0,1\}$. Hence, the *overall cost* of the security plan is given by the expression

$$V = \sum_{j=1}^{k} M_j V_j .$$

A business-oriented approach to modelling the *impact* $U_i \in \mathbb{R}$ that an attribute S_i has to an organization is given in [246], by considering the potential loss/damage L_i that has to be paid when S_i gets compromised, and the potential gain G_i if S_i is not compromised; overall, the impact is defined as

$$U_i = (1 - \Pr[S_i])\, G_i - \Pr[S_i]\, L_i$$

where $\Pr[S_i]$ is the unconditional probability (and actually computed as $\Pr[S_i|C_j]$ if $(S_i, C_j) \in \Lambda$ depending on the security plan). Clearly the above definition of the impact can be well-adjusted to utilize measurable security properties, e.g. coming from the CVSS standard, that would allow making the computation of the whole risk mitigation problem more automated and objective. From the above we derive *overall impact* is

$$U = \sum_{i=1}^{n} U_i \, .$$

As a result, a security administrator can subsequently formulate an optimization problem that maximizes the objective function or reward that is defined as $R = wU - (1 - w)V$, for a design parameter $w \in [0,1]$ that signifies the importance of each control factor. Algorithms for solving the problem

$$\boldsymbol{M}^* = \text{argmax}_{\boldsymbol{M} \in \{0,1\}^k} R$$

where $R := R(\boldsymbol{M})$ and $\boldsymbol{M} \in \{0,1\}^k$ is the vector $\boldsymbol{M} = (M_1, M_2, \dots, M_k)$ that determines the security controls being included in the security pan, belong to the class of linear programming techniques, or more-generally (non)convex optimization techniques; however, iterative (and possibly greedy) algorithms can also be used, which not always determine the optimal solution (but rather yield suboptimal solutions).

The risk mitigation optimization problem could also be extended to incorporate information about the trust T_l placed on a device l, e.g. by replacing the potential loss/damage L_i in the computation of U_i with a function $f(L_i, T_l)$. Other approaches for risk mitigation are also presented in deliverable D2.5 [60].

3.6 Recommendations

In this section we have reviewed the aspects regarding trust management and risk assessment, which will be taken into account in the development of the Cyber-Trust TMS component. Our review has determined that behavioral aspects and status assessments can be exploited in determining the trust status of individual devices, while trust levels, combined with asset values and environmental aspects (e.g. threat agents and security controls) can derive risk levels. Furthermore, we have reviewed the state-of-the-art trust models, which describe how independent entities can operate in the context of a distributed system to exchange, synthesize, propagate and update trust metrics. In the context of a distributed system however malicious nodes exist which may launch different types of attacks to illegitimately affect trust metrics – either for themselves or for other nodes: in this respect, we have examined the level of resiliency offered by trust models against the various attacks types. Subsequently, we have identified existing implementations of trust management systems and analyzed their utility for the implementation of the Cyber-Trust TMS. Finally, we have studied how trust and risk assessments can be exploited in the context of the defense strategy, both reactive and pro-active, resulting in trust- and risk-aware defense. In the following we summarize our main findings, identify research directions, and discuss possible solutions.

Regarding the behavioral aspects that can be used to compute trust, while MUDs are a promising approach, their widespread is substantially limited and this constitutes a major impediment for their usage. However, the relevant behavior assessment components should be designed so that MUD usage can be accom-modated. Dynamic models and signature-based scanning are far more mature technologies and employed in many modern IDSs, still being challenged with non-negligible false positive and false negative rates.

Considering status-based approaches, while solutions such as the Trusted Platform Module (TPM) provide an adequate level of security and ensure accurate reporting of the platform's integrity status, IoT devices typically do not include such modules and it is not foreseeable that the use of TPMs in IoT devices will increase. Tuning and efficiently implementing software-based remote attestation techniques or identifying minimal hardware additions that can be implemented in a cost-effective function to leverage the accuracy of remote integrity checks, is a major challenge in this area.

A number of contemporary trust management models have been examined: some of them are specifically designed for the IoT domain, however a number of them have not be proven to be resilient to attacks [175], [302] or have been shown to resist only very few attacks [67], [330], [340]. The trust models described in [222] and [369] have been shown to be capable of withstanding most attacks: [369] includes a centralized propagation aspect which could introduce bottleneck problems in a highly populated network, albeit it could fit the domain of a smart home, or an enterprise. From the non IoT-specific models, the one proposed by

Chen, [149] encompasses many defenses against attacks and has been designed in a generic fashion; however, it should be tested in the context of large-scale IoT.

Regarding trust management system implementations, none of the identified implementations is directly usable in the context of Cyber-Trust: many of them are out of context (pertaining to other domains), while other implementations present different challenges, including lack of features, implementation languages that are not appropriate for mobile platforms or for which the consortium lacks experience, large footprints or performance issues. Concepts however from Keynote [208], SAFE [334], [273] and ReTrust can be exploited in the context of Cyber-Trust.

Finally, considering trust- and risk-aware defense, incorporation of trust metrics in the process of computing the attack existence probability can leverage the precision of the attack detection, yet backing mechanisms should be in effect to tackle the case that devices of high trust levels are compromised –but still have not been detected to be so- and exploit their high trust level to evade attack detection. Analysis and experimentation is needed to identify the optimal methods for providing trust-aware reactions. In the context of proactive risk management, graphical security models have been identified to provide adequate expressive power and solving capacity to realize risk prioritization; as an extension, representative optimization problems can be formulated to assist security administrators in selecting the security controls that can be implemented in their systems to maximize security and minimize the residual risk.

4. Game-theoretic cyber-defense framework

4.1 Introduction

Cyber-attacks constitute a great threat for modern networks with high socio-economic impact [93]. For this reason, much research effort has been devoted to their study during the past years [51], [304], [374], [265], with the researchers aiming at accurately modelling the attackers' behavior, as well as devising efficient defense strategies against sophisticated intelligent attacks for cyber-systems. Towards this end, various models have been proposed based on *Stochastic Control Theory* (SCT) and *Game Theory* (GT). Such approaches overcome traditional solutions to cyber-security and network privacy due to the theoretical guarantees they provide for a sound and coherent analysis; more precisely, the advantages include the following:

- *Proven mathematics:* Most conventional security solutions, which are implemented either in preventive devices (e.g., firewall) or in reactive devices (e.g., anti-virus programs), rely only on heuristics. However, game theory can investigate security decisions in a methodical manner with proven mathematics.
- *Reliable defense:* Relying on analytical outcome from the game, researchers can design defense mechanisms for robust and reliable cyber-systems against selfish behaviors (or attacks) by malicious users/nodes.
- *Timely action:* While adoption of the traditional security solution is rather slow due to the lack of incentives for participants, game-theoretic approaches advocate for defenders by using underlying incentive mechanisms to allocate limited resources to balance perceived risks.
- *Distributed solutions:* Most conventional defense mechanisms make decisions in a centralized manner rather than in an individualized (or distributed) manner. In a network security game, the centralized manner is almost an impossible solution due to the lack of a coordinator in autonomous system. Using appropriate game models, security solutions will be implemented in a distributed manner.

Cyber-security studies vary in a wide area of applications, such as (D)DoS attacks [324], physical layer security [372], intrusion detection [79], selfish behavior in packet-forwarding [190] and information sharing [189], to name a few. In this section we review some fundamental works on cyber-security models based on SCT and GT with focus on *state-based* approaches that model the attacker-defender interactions using some type of Attack Graphs (AGs); *see* deliverable D2.5 for a discussion over the various state-of-the-art AG models. In such models, the attacker aims at exploiting system vulnerabilities for progressing his attack on a cyber-system aiming at reaching some *goal*, while the defender aims at preventing the attacker's progression. Such works aim at developing efficient *automated intrusion response systems* (IRSs) that are capable of automatically responding to intrusions without the need for a human operator to intervene [90]. The reason for our focus on such models is due to their generic nature and as a result wide applicability to a variety of cyber-attack problems, which is in accordance with the ambitions envisaged by the Cyber-Trust project for developing an effective multiple-purpose intelligent Intrusion Response System (iIRS). For comprehensive survey on IRS-related literature, the interested can refer to [374], [265].

4.1.1 Cyber-security needs

Global networks continue to undergo dramatic changes resulting in ever-increasing network size, interconnectivity, accessibility, and a consequent increase in its vulnerability. Several recent Federal policy documents have emphasized the importance of cyber-security to the welfare of modern society. The President's National Strategy to Secure Cyber-Space describes the priorities for response, reduction of threats and vulnerabilities, awareness and training, and national security and international cooperation. Cyber-Security: A Crisis of Prioritization describes the need for certain technologies for cyber-security. Security should be an integral part of advanced hardware and software from the beginning, as described by

[25]. Next generation information infrastructure must robustly provide end-to-end connectivity among computers, mobile devices, wireless sensors, instruments, etc. Cyber-security is an essential component of information and telecommunications, which impacts all of the other critical producing secure and reliable software. NSA has an effort on high-assurance computing platforms. The Trusted Computing Group has an ongoing effort. Microsoft has an effort on next-generation secure computing.

In future warfare, cyberspace will play a major role where no one is guaranteed to have information dominance in terms of intelligence and accessibility. As a result, a game-theoretic approach of collaboration (carrot) and compelling (counter) moves (stick) need to be played efficiently. This notion is not unlike the *mutually assured destruction* (MAD) of nuclear warfare. The question then becomes: How do we construct such a game theoretic approach in cyberspace? In general, a game-theoretic approach works with at least two players. A player's success in making choices depends on the choices of others. In game theory, players are pitted against each other taking turns sequentially to maximize their gain in an attempt to achieve their ultimate goal. In the field of cyber-security, game theory has been used to capture the nature of cyber-conflict. The attacker's decision strategies are closely related to those by the defender and vice versa. Cyber-security then is modelled by at least two intelligent agents interacting in an attempt to maximize their intended objectives [25]. Different techniques available in game theory can be utilized to perform tactical analysis of the options of cyber-threat produced either by a single attacker or by an organized group. A key concept of game theory is the ability to examine the huge number of possible threat scenarios in the cyber-system. Game theory can also provide methods for suggesting several probable actions along with the predicted outcome to control future threats. Computers can analyse all of the combinations and permutations to find exceptions in general rules, in contrast to humans who are very prone to overlooking possibilities. This approach allows identification of the what-if scenarios, which the human analyst may not have considered. The use of game theory in modelling good and evil has also appeared in several other areas of research. For example, in military and information warfare, the enemy is modelled as an evil player performing actions and adopting strategies to disrupt the defence networks [10].

4.1.2 Emerging challenges

There are significant advances in information technology and infrastructures offering new opportunities. In many cases though, the employed security solutions are ad hoc and lack a quantitative decision framework. While they are effective in solving the particular problems they are designed for, they generally fail to respond well in a dynamically changing scenario. To this end, the game theory can provide huge potential to place such an approach on a solid analytical setting. Currently, game theoretic has been an important concept in various security situations and has found great application in cyber-security. Furthermore, recent research works have seen game theory being applied to network security, web security and lots more. Games can be designed and analysed, optimal moves of players (e.g., firewalls) are used to determine how to best approach security in the cyber-world. One *key challenge* with game theory is the ability to come up with feasible and computationally efficient mathematical solutions to the particular problem at hand.

4.1.3 Cyber-defense objectives

The basic design of the World Wide Web makes to many kinds of threats such as DoS/DDoS, Brut force, SQL injection and etc. Researchers have over the years been exploring the possibility of applying game theoretic approaches to deal with cyber security problems and some of these methods have been fruitful. However, the cyber world continues in complexity over the years and has become more sophisticated, moreover, the cyber crimes have grown in complexity [9]. The game theoretic model is one distinguished approach, which ideally allocates cyber security resources such as administrators' time across different tasks [4].

Currently, research studies are focusing on bringing working network security solutions to organizations; one of such approach is utilising theories that suitable into real life scenarios to create mitigation methods. Therefore, the studies are examining the game theory as it is one of such approaches that shows a competitive activity used to model the behaviour of attackers and defenders of a network.

Safeguarding an organization's cyber assets from impositions and breaches caused by attacks that implemented by malicious actors is an increasingly dangerous, challenging and complex issue. Several current major breaches have highlighted this challenge which have caused serious damage, for instance the Equifax breach in 2017 and Yahoo in 2016 [148].

Recently, protecting an organization's cyber-assets from intrusions and breaches due to attacks by malicious actors is an increasingly challenging and complex problem. This challenge is highlighted by several recent major breaches which have caused severe damage, such as the Equifax breach in 2017 and Yahoo in 2016. To protect from cyber-breaches, companies and organizations employ the use of anti-virus software, *Intrusion and Detection Systems* (IDS), and *Cyber-Emergency Readiness Teams* (CERT) composed of cyber-analysts tasked with the general protection of an organization's network and cyber-assets. Modern day cyber-adversaries are persistent, targeted and sophisticated. This highlights the tremendous need for organizations to protect against such attacks and model these adversaries for optimizing the responses of the network defender's in order to protect targets and systems across the enterprise network [148]. The cyber-kill chain encapsulates the necessary steps that an adversary must complete to successfully breach the defender's enterprise network (see Figure 4.1). In the first phase of the cyber-kill chain the adversary spends a significant amount of time completing reconnaissance of the defender's enterprise network to learn about the vulnerabilities present and potential points of compromise. After recon, the adversary's next phases consist of weaponizing his exploit or malware, delivering it to the network through some medium and then exploiting a vulnerability in a system connected to the defender's network. To finish out his attack, the adversary installs additional malware to ensure persistence and then establishes a command and control channel to allow meeting his objectives, i.e., exfiltrate sensitive information, and complete his ultimate goal from breaching the network [9].

Figure 4.1. The cyber-kill chain developed by Lockheed Martin

These sort of attacks, can target anyone from persons to companies or government agencies. For instance, in 2011, a massive attack that forced Canada's Finance Department and Treasury Board to disconnect from the Internet. Three years later, the news were led for some time by the hacking of Sony Pictures [51].

Therefore, rising and developing of cyber security and privacy concerns need effective defence methods to face these threats that are being developed on daily bases by the criminals. Game theory is the practical answer of the question that says how the defender will react to the attacker, and in contrast, in cyber security [10].

4.1.4 Simple game types

Each game is characterized by a number of parameters, including: (a) the number and type of players; (b) the order of moves, i.e., when does each player gets to move (sequentially or concurrently) in the game; (c) the actions being available to each player; (d) the knowledge that each player has about the opponent; and (e) how each player is rewarded for each action. Based on the above, a number of simple game types, which are commonly used to model interactions between defenders and attackers, is next presented.

Perfect information games. Each player in this game is aware about the moves of all other players that have already taken place in the game. For instance, chess, tic-tac-toe, and go. On the other hand, if at least one player is not aware of the moves of at least one other player that have taken place, then the game is called an imperfect information game [9].

Bayesian games. "A game in which information about the strategies and payoff for other players is incomplete and a player assigns a 'type' to other players at the onset of the game". Because of the use of Bayesian analysis in predicting the outcome, such games are called Bayesian games [9].

Static/strategic games. "A one-shot game in which each player chooses his plan of action and all players' decisions are made simultaneously". This means when choosing a plan of action each player is not informed of the plan of action chosen by any other player [9}.

Dynamic/extensive games. "A game with more than one stages in each of which the players can consider their action". It can be considered as a sequential structure of the decision making problems encountered by the players in a static game [9].

Stochastic games. ''A game that involves probabilistic transitions through several states of the system". A sequence of states shapes the progresses of the game. A start state entails the game; the players choose actions and receives a payoff that relay on the present state of the game, and then the game transitions into a new state with a probability based upon players' actions and the present state [77].

4.2 Background on optimal decision-making

Before proceeding to the presentation of state-of-the-art works on IRSs, we will present some fundamental background needed to comprehend the proposed IRSs' operation. The study of optimal decision-making has a long history [252]. Under the assumption of *rationality*, the agents make decisions that will maximize their expected utility. When multiple rational agents interact with each other, Game Theory [68] is an appropriate tool for modelling their interactions and analyzing their *strategic* decision-making process so that their behavior can be explained and predicted. Formal mathematical frameworks from optimal decision-making study have recently been applied to cyber-security for explaining elaborate attack behaviors and devising optimal defense strategies for dynamic systems. For this reason, in the sequel we present some basic concepts for optimal control in dynamic systems, so that their application in cyber-security can be understood.

4.2.1 Dynamic processes for single-agent problems

The task of *sequential decision-making* under uncertainty, where a decision-maker has to plan a sequence of actions, in a dynamic environment has been a hot scientific field for decades due to its wide applicability ranging from Economics and Operational Research to modern applications of Artificial Intelligence. For this reason, solid mathematical frameworks have been developed to accurately describe the decision-making process in such a setting and to provide guarantees that a *strategy* (i.e., a plan of actions) is *optimal*. The basic forms of uncertainty considered are due to the outcome of the agent's actions (i.e., in a stochastic system the same action might not result in the same outcome) and the uncertainty due to faulty observations (i.e., an underlying system state component is observed with possible inaccuracy).

The basic framework for studying sequential decision problems for stochastic systems but with perfect observability (i.e., there is uncertainty about the outcome of the actions but not about the accuracy of the observation of the system state) is the Markov Decision Process (MDP) framework [75], [8]. An MDP is defined as a tuple $< S, A, R, T >$ where S is the *state space*, A is the *action space*, $R: S \times A \rightarrow \mathbb{R}$ is the (instantaneous) *reward function* and $T: S \times A \rightarrow S$ is the *transition matrix*. In the standard MDP model the state and action spaces are finite and the time is discretized into distinct time instances. In an MDP, the decision-maker wants to maximize a *long-term reward* criterion (not just the immediate reward R). If the time duration (or *time horizon*) is known a priori, then this is the *finite horizon case*, where the agent aims at maximizing the *expected future (discounted) sum of rewards*

$$E\left\{\sum_{t=0}^{T}\rho^t R(s_t,a_t)\right\}, \quad (1)$$

where the expectation is with respect to future states and actions, $s_t \in S, a_t \in A$ are the state and action at time t, respectively and $\rho \in [0,1]$ is a discount factor. Agent's goal is to find the *optimal policy* $\pi = (\pi_0, \dots, \pi_{T-1})$ which maximizes (1). $\pi_t: S \rightarrow A$ is a *decision rule* that maps the set of states to the set of actions. In case the time horizon is not known a priori, or the process never terminates (*infinite horizon case*) the usual maximization criterion is the following

$$E\left\{\sum_{t=0}^{\infty}\rho^t R(s_t,a_t)\right\}, \quad (2)$$

where now it is $\rho \in [0,1)$ to ensure that (2) is bounded. For MDPs, it has been shown that the only information that is needed for a strategy to be *optimal* is the current system state (*Markov policies*), instead of the complete *history* of past states and actions (i.e., the whole *information* that the agent has at its disposal at a time instant). This is an attractive feature of MDPs that is not shared with its partially observable counterpart (i.e., POMDP), as we will see later on. Moreover, for the infinite horizon case (*see* (2)), it is shown that there always exists an optimal policy which is Markov and additionally it is time-independent (*Markov stationary policy*), meaning that the optimal policy consists of the same decision rule $\pi_t: S \rightarrow A$ for every different time $t \in [0, \infty)$. This is not the case for the finite horizon case optimal policies. Finally, for the aforementioned MDP models there always exist optimal policies that are *deterministic* (i.e., policies where each decision rule completely determines – with probability one – which action to be taken at every state and time).

For a given policy π, (1) can be computed with the following recursive equation (due to the Markovian property of the model)

$$V_t(\pi, s) = R(s, \pi_t(s)) + \rho \sum_{s' \in S} T(s, \pi_t(s), s') V_{t+1}(\pi, s'), \quad (3)$$

by setting $V_T(\pi, s) = 0$ for all $s \in S$ and by starting from time $T-1$ and working backwards to time 0 (*dynamic programming - principle of optimality* [276]). Using this decomposition, the optimal value function can be computed by using the dynamic programming equation

$$V_n^*(s) = max_{a \in A}\{R(s,a) + \rho \sum_{s' \in S} T(s,a,s') V_{n-1}^*(s')\}, \quad (4)$$

where V_n^* is the value function of the optimal policy π^* and n are the remaining time steps. This method of finding the optimal policy is called *Value Iteration* (VI).

The corresponding value function for the infinite horizon case and given a stationary policy π is

$$V(\pi, s) = R(s, \pi(s)) + \rho \sum_{s' \in S} T(s, \pi(s), s') V(\pi, s'), \quad (5)$$

Note that (5) defines a system with $|S|$ equations and $|S|$ unknowns. Applying the VI algorithm in (5) gives the optimal value function and the optimal stationary policy. For solving infinite horizon MDPs, the *Policy Iteration* (PI) algorithm can be applied as well [75].

In many problems the assumption of full observability of the state is not valid. For such cases, a generalization of MDP, the Partially Observable Markov Decision Process (POMDP) framework was developed. A POMDP is defined as a tuple $< \mathrm{S}, \mathrm{A}, \mathrm{T}, \mathrm{R}, \mathrm{O}, \mathrm{Z} >$ where $\mathrm{S}, \mathrm{A}, \mathrm{T}$ are the same as in MDP model. Z is a set of observations

which act as *signals* on the trust state of the system. Associated with the observations there is an *observation model/function* $\mathrm{O}: \mathrm{S} \times \mathrm{A} \rightarrow \Pi(Z)$, where $\Pi(Z)$ denotes a *probability distribution* over Z. Finally, the reward function can take a more general form as $R: S \times A \times S \times Z \rightarrow \mathbb{R}$. The agent at every time epoch has not access to the previous or current states, but only to the set of the observations he has received up to that time (as well as to the previous actions selected).

In order to act optimally in such a setting the agent has to devise policies that map the entire information it possesses (i.e., the history of observations and actions) at every time to the action set. This is computationally expensive, as this history grows with time. An alternative to that option is to keep a *sufficient statistic* that encapsulates all the available information that the history of the process. In the POMDP model described above this sufficient statistic exists and it is called *belief state*. A belief state is denoted as b and it is a probability distribution over the system states. Given a belief vector b and the new action and observation received, the new belief vector b' can be computed via the Bayes' rule and hence the past history is not needed, preserving in this way the Markovian property of the model. Exploiting this fact, the original POMDP over states S can be re-cast as an observable MDP over the belief states $B = \Pi(S)$, which is the space of all probability distributions over S. However, the new *belief-state MDP* is a continuous state MDP (infinite number of states) and although the dynamic programming equations hold, as well as the properties of the optimal policy, the computation of the optimal policy is a much harder task in terms of complexity. The state space of the belief-state MDP is B and the optimal policy is a mapping from B to the action set.

For a finite horizon POMDP the optimal value function is *piecewise linear and convex* (PWLC) [281], [87]. By exploiting this property, the first exact algorithm for solving a POMDP was developed. The value function in an infinite horizon POMDP remains convex, but its piecewise linearity is lost (in general). The optimal policy in a POMDP has the same properties as in the MDP model, meaning that there is always a deterministic optimal policy in finite horizon that depends only on the belief state and in infinite horizon there is always an optimal policy that is additionally stationary.

Due to the intractability of the exact algorithms for realistic problem sizes (solving a finite horizon POMDP is PSPACE-complete [45] and for an infinite horizon POMDP the problem is undecidable [251]), *approximate methods* are used to solve a POMDP. These approximate methods can be categorized into *offline* and *online* algorithms and they can be combined in a hybrid fashion. Offline algorithms specify, prior to execution, the best available action for every situation, while online algorithms compute a policy by planning online for the current belief state encountered. For an excellent survey on approximate algorithms on POMDPs, the interested reader can refer to [307].

4.2.2 Game theory

Decision-making in a multi-agent environment where multiple *rational* agents, or *players*, (i.e., individual utility-maximizers) interact and the actions of one agent affect the rewards realized by the others is more challenging than the single-agent decision-making models described in the previous subsection, as there is extra uncertainty on the behavior of the other agents and the environment can now be affected by all agents' actions. A *game* is a description of the strategic interaction between the players. A *strategy* for a player is a complete plan of actions in all possible situations that may be encountered throughout the game. If the strategy specifies to take a unique action in a situation then it is called a *pure strategy*. On the other hand, if the stategy specifies a probability distribution for all possible actions in a situation then the strategy is referred to as a *mixed strategy*. The most widely used solution concept for a game is *Nash Equilibrium* (NE). A NE is a set of strategies of the players, each one of which constitutes *best-response* to the other strategies simultaneously. A NE describes a steady state condition of the game; no player would prefer to change his strategy as that would lower his payoffs given that all other players are adhering to the prescribed strategies. Formally, as set of strategies $s_1^*, \dots, s_N^*$ for players $1, \dots, N$ with utilities $U_i(s_i^*, s_{-i}^*)$ for player $i \in \{1, \dots, N\}$ and $-i \in \{1, \dots, N\} \backslash \{i\}$ if

$$U_i(s_i^*, s_{-i}^*) \geq U_i(s_i, s_{-i}^*) \tag{6}$$

for every strategy $s_i \neq s_i^*$, for every player $i \in \{1, \dots, N\}$.

Games can be categorized into *static*, which are played for one time only, and *dynamic*, where the players interact repeatedly for multiple times [68]. Next, we present (Figure 4.2) a simple game, called *Prisoner's dilemma*, in its static form where the players interact once.

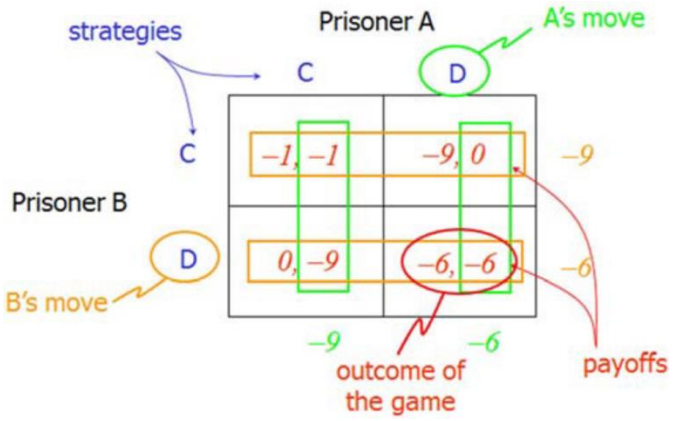


Figure 4.2. Description of the prisoner's dilemma game

In Prisoner's dilemma it is assumed that two individuals have been arrested. The police suspects that they have committed some form of crime; if nobody confesses to the police (i.e., *defects*), they will be jailed for 6 years. If only one confesses and the other defects, then the latter will go free and the first one will be jailed for 9 years. If they both confess, they get 1 year each. In this game, the only NE is both prisoners to defect and not confess the crime since they don't have a previous knowledge of what the other player will do.

In the aforementioned game both players move *simultaneously*. Players can take actions sequentially, as well. A game in which each player is aware of the moves of all other players that have already taken place is called *perfect information game*. Examples of perfect information games are: chess, tic-tac-toe, and go. A game where at least one player is not aware of the moves of at least one other player that have taken place is called an imperfect information game.

A game in which information about the strategies and payoff for other players is incomplete and a player assigns a 'type' to other players. Such games are labelled *Bayesian games* due to the use of Bayesian analysis in predicting the outcome.

In a *dynamic game*, players interact for more than one stages in each of which the players can consider their action. It can be considered as a sequential structure of the decision-making problems encountered by the players in static games. The sequences of the game can be either finite, or infinite.

A game that involves probabilistic transitions through several states of the system is called *Stochastic Game* (SG). The game progresses as a sequence of states. The game begins with an *initial state*; the players choose actions and receive a payoff that depends on the current state of the game and the players' actions, and then the game transits into a new state with a probability that depends upon players' actions and the current state.

Next, we will present some characteristics for dynamic games only, since we are interested in exploiting such games in the development of the iIRS in order to derive optimal defense strategies against far-sighted attackers that are capable of launching elaborate multi-stage attack plans in order to achieve their objectives. Various kinds of games have been proposed in literature and their solutions is highly dependent on their structure. For a comprehensive treatment of GT, the interested reader can refer to [252], [68], [159], [357].

The multi-agent extension of MDPs and POMDPS are *Stochastic Games* (SGs), also called *Markov Games*, and *Partially Observable Stochastic Games* (POSGs), respectively. SGs were introduced by Shapley [203] and they are defined as $< N, S, A, P, R >$, where N is a finite set of players, S is a finite set of states, $A = A_1 \times \ldots \times A_n$ with A_i, $i \in N$ denoting a set of actions available to player i (the set of available actions can depend on the state as well), $P: S \times A \times S \rightarrow [0,1]$ is the transition probability function and $R = R_1, \ldots, R_n$

where $R_i: S \times A \rightarrow \mathbb{R}$ is the reward function for player $i \in N$. Note that the state transitions, as well as the reward realized by every player now depend on the actions of all players. Regarding the overall (long-term) rewards that each agent aims at maximizing, the less problematic case and perhaps the most common in literature is the *future discounted rewards* (*see* (1), (2)) and we will focus on this one here.

Every n-player (general-sum) discounted-reward SG admits a NE. Actually, a stronger property has been proved for this class of SGs which states that a *Markov Perfect Equilibrium* (MPE) always exists. A strategy profile is an MPE if all agents' strategies are Markov strategies and it is a NE regardless of the game's starting state.

Computing equilibria in (discounted-reward) SGs can be accomplished by using a modified version of Newton's method to a nonlinear program formulation of the problem. If the game is *zero-sum*, an algorithm, which is based on VI, proposed by Shapley can be used. For details on solving SGs the interested reader can refer to [22], where multiple sub-classes of SGs, along with the respective algorithms to solve them are presented.

In SGs it is assumed that the players have *complete* on the state of the game. Extending SGs to include the case when the players observe incompletely the system state is a non-trivial task and an area of active research. As SGs extend MDPs to the multi-agent setting, POSGs extend POMDPs in the same fashion. In this kind of games, each agent has its own observation model and as a result, each agent has access to different information. For this reason, such games can be also characterized as *dynamic games of asymmetric information*. Hence, POSGs combine characteristics of SGs and *games of incomplete information* (*Bayesian Games*).

This class of games is quite expressive and models strategic interactions that describe accurately the system dynamics in a wide range of applications. For this reason, it has attracted interest both by AI community [99], [84] as well as from decentralized control community [21], [81] with the researchers in both communities studying problems that fall within this broad category. Different assumptions on the observation model and on the utility functions of each agent give rise to different game models which need different treatment. One case of great interest and wide applicability is the one where the agents make their own private observations and take their own actions independently but they try to maximize a common objective (*team problem*), which is known as Decentralized POMDP (Dec-POMDP) [99] (since the agents do not have individual reward function and do not antagonize it is not a game but it is an extension of single-agent POMDPs to the multi-agent (cooperative) setting with great interest in a variety of applications).

The difficulty that arises in these games lies in the fact that each agent has access to different information, meaning that they have different histories of past observations of the system state (and possibly about agents' past actions) and as a result the agents form different beliefs about the game that is played. Thus, an important aspect in this literature is the *information structure* of each agent and the assumptions on how this information is shared among the agents. One approach to deal with this asymmetry in beliefs was proposed in [21], where the authors define a so-called *Common Information Based Markov Perfect Equilibrium* (CIB-MPE) where the agents form a belief on the part of the history that is known to all agents (i.e., *common history*) and provide a Backward Induction Dynamic Programming algorithm to find these equilibria. An important aspect of this work is that the authors study different cases of how the agents share information among them to form the common history where this Dynamic Programming procedure can be performed. Another Dynamic Programming algorithm was proposed in [84] where a different belief was defined, called *multi-agent belief* which is a distribution over states and policies of other agents. More recently, in [81] the authors extend [21] to study the case where the common information-based belief depends on the agents' strategies and define *structured Bayesian perfect equilibria* (SPBEa), which is subset of Perfect Bayesian Equilibria (PBE), along with a Dynamic Programming procedure to compute them. Another important work in this domain is [16], where the authors provide results on the structure of the value function for zero-sum POSGs. All the aforementioned works refer to the finite-horizon case. In this literature, only [47] studies the infinite horizon case where SPBEa are computed by solving a single-shot fixed-point equation and a corresponding forward recursive algorithm.

4.2.3 Learning methods and online algorithms

An important aspect of decision-making in dynamic environments is the one of *learning*. Learning algorithms try to devise (learn) an effective policy (ideally the optimal policy) when some component of the model is unknown. For example, in the MDP setting the agent could be unaware of the transition matrix and/or of the reward function. So, the question arises whether an agent in such a setting can come up with the best policy through repeated interactions and received feedback of its actions by the environment. The learning literature is vast [288], [357], so here we will present a brief overview of *Reinforcement Learning* (RL) [288], while keeping focus on the characteristics that seem appealing to the needs of Cyber-Trust project.

One of the most well-known and widely used RL algorithms is the Q-Learning (QL) algorithm [47]. Its importance lies in the fact that it learns the optimal policy in an MDP (infinite horizon), under the assumptions that each state-action pair is visited infinitely often and the learning parameter is decreased appropriately. This is done without requiring any knowledge about the state transition function or the reward function, but the agent interacts repeatedly with the environment by only having knowledge of the state it resides in and a received reward signal at every time instant.

Extending RL from MDPs to their multi-agent counterpart SGs poses difficulties due to the *non-stationarity* of the environment as there are other agents interacting with the environment and performing their own learning process. In multi-agent learning the notion of "optimality" of the agents' learning process needs to be revisited and researchers have proposed some criteria that a learning algorithm has to fulfill, such as safety, *Hannan consistency* and *rationality* [357].

Towards this direction, a QL-based algorithm, called *minimax-Q*, was proposed in [216] for two-player zero-sum SGs. In minimax-Q each player assumes that the other player will select the action that minimizes the former player's payoff. Under the same conditions that assure convergence of QL to the optimal policy in MDPs, Minimax-Q converges to the value of the game in *self play* (i.e., play against itself) in zero-sum games. The same author in [215] extended this algorithm to present *friend or foe Q-Learning* (FFQ) for general-sum SGs. In FFQ the learner assumes that the other agents will act either as foes (i.e., they will act to minimize its reward), or as friends (i.e., they will act to maximize its reward). The assumption that the other will follow the behavior dictated by a NE of the game was utilized in [163] for the Nash-Q Learning algorithm for general-sum SGs. The algorithm requires a set of very strict requirements to guarantee convergence to a NE in self play. For a recent discussion on QL for games, the interested reader can refer to [112].

In partially observable domains the techniques applied in MDPs are no longer applied. In a POMDP as discussed earlier, approximate solutions have received increased attention due to the complexity of exact techniques. These techniques are divided in *model*-based and *model-free*. In model-based techniques lie the *point-based value iteration methods*, which instead of planning over the entire belief space, they plan only for a part of the belief space that is *reachable* from. This part of the belief space is sampled through agent's interactions with the environment. Other model-based approaches include *grid-based approximations*, in which a (fixed or variable) grid is used to describe the belief simplex, *policy search*, in which a search for a good policy is performed within a restricted class of controllers and *heuristic search*, in which after defining an initial belief as the root node a tree is built that branches over action-observation pairs, each of which recursively induces a new belief node [233]. When the model of the POMDP is not available (e.g., the state transition probabilities), the previous methods cannot be applied. Model-free methods are categorized in *direct* and *indirect* RL methods. Indirect methods reconstruct the POMDP model through repeated interactions with it and then this POMDP can be solved by one model-based method. On the other hand, direct methods utilize true model-free techniques without reconstructing the POMDP. In these methods the policy usually maps a subset of the previous acquired observations (history *window*) to actions [233].

4.3 Cyber-defense and decision-making

In this subsection we review the basic cyber-defense models based on STC and GT. We focus on state-based models. One feature that distinguishes the various models is the assumption of the level of *observability* of the system underlying state. This characteristic affects both the modelling, as well as the solution algorithms

for the derivation of the optimal strategies. We start by presenting the single-agent models and game-theoretic models for IRSs in observable domains and then we present the respective models for partially observable domains.

4.3.1 Decision-making in fully observable domains

In [299] an MDP-based IRS is proposed. The *state* is comprised of an *attack vector*, which contains as many variables as the number of attacks detectable by the IDSs, and a set of *system variables*. The authors consider a set of *response actions* as countermeasures and take into account system security and system operation to assign the costs for the various response actions. To deal with the large number of states, the authors employ the sub-optimal rollout-based Monte-Carlo algorithm, named UCT [202], and compare its performance with the classic Value Iteration (VI) algorithm [276]. Through extensive simulations, they show that when a small reward degradation is acceptable, the planning time can be improved significantly. A high-level architecture of the system proposed in [299] is presented in Figure 4.3.

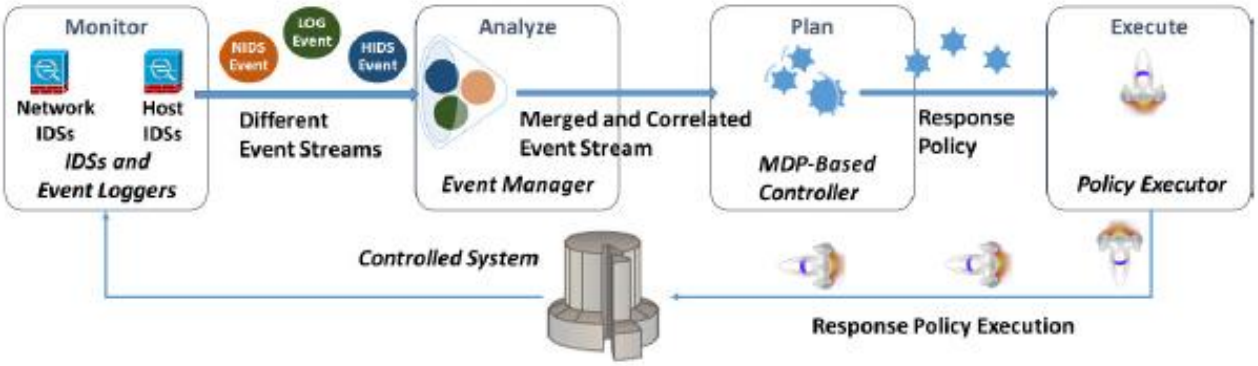


Figure 4.3. High-level architecture [299]

The multi-agent equivalent (i.e., there are multiple *rational* decision-makers interacting with each other) of an MDP is a general-sum S*tochastic Game* (SG). This framework was utilized in [193] to model the interactions between the attacker and the network administrator. They use a non-linear program to compute the *Nash Equilibria* (NEa) of the SG [159], which are multiple. They illustrate by experimental results that the NE strategies are meaningful and that they can be utilized by a network administrator as a useful tool to provide insight and discover potential attack strategies that can compromise network security.

4.3.2 Decision-making in partially observable domains

To account for the partial observability of the system state by the defender, caused by IDS anomalies, and to provide a more realistic model, a host-based IRS, called ALPHATECH Lightweight Autonomic Defense System (aLADS), was proposed in [253]. The authors modelled the defender's problem as a POMDP. In their modelling the trade-off between the security achieved by the countermeasures and the network availability is captured and extensive simulations are performed to illustrate the effectiveness of the proposed IRS in protecting its host, a Linux-based web-server, against an automated Internet worm attack. The proposed host-based IRS is presented schematically in Figure 4.4.

In [91] a cyber-defense model is built upon a Bayesian attack graph (BAG) [354], where the nodes represent *system attributes* - attributes can be seen as *attacker capabilities* - (e.g., attacker permission levels on a given machine, vulnerabilities of a service or system, or information leakage) and the edges represent exploits (i.e. events that allow the attacker to use their current set of capabilities (attributes) to obtain further capabilities). They assume a probabilistic behavior for the attacker and study the defender problem, meaning the problem of selecting the optimal defense actions in order to prevent the attacker from reaching its goals. They assume *partial* observability, in the sense that the defender receives noisy alerts from an Intrusion Detection System (IDS) about the system security state. The problem is formulated as a POMDP and it is

solved using the Cassandra's C-software package, called `pomdp-solve` [316], to obtain the defense policy for a small sample network.

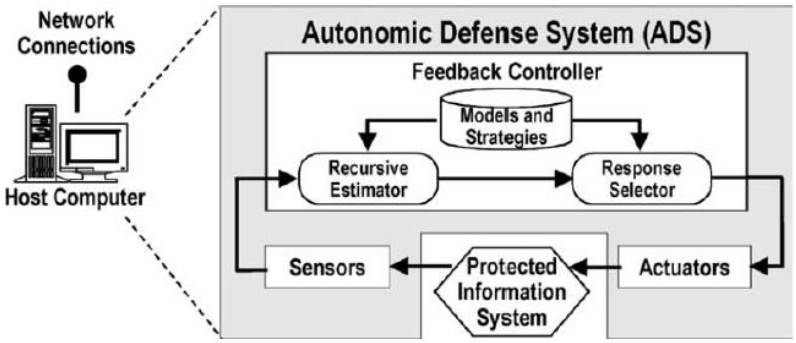


Figure 4.4. Autonomic host-based IDS [253]

The authors extended this work in [90] to present a more expressive model to allow for more complex dependencies among exploits, a more realistic observation model (i.e., alerts are triggered by exploit activity and are subject to false alarms) and they assume different attacker possible strategies. The proposed IRS's architecture is presented in Figure 4.5. Finally, they follow a *Monte-Carlo sampling* approach to develop a scalable online defense algorithm, based on the POMCP algorithm [80], to deal with the scalability issues raised in [91] due to large state spaces.

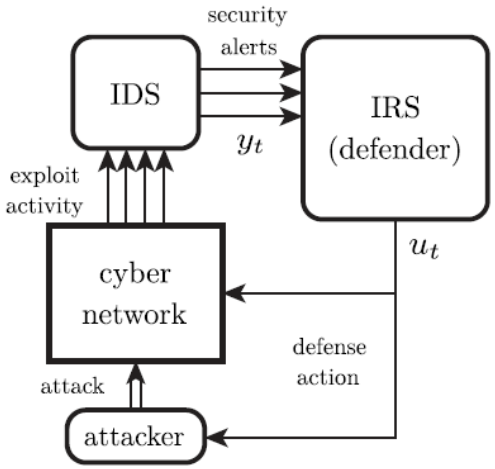


Figure 4.5. Intrusion response system architecture

One limitation of the previous works is that the attacker is not *rational* (i.e., it does not take actions that maximize its utility, but it is assumed to follow a set of pre-specified attack strategies). In fact, extending POMDPs to the game setting where multiple rational agents interact and possess different information (i.e., *asymmetric information*) is a rather challenging task and procedures for computing optimal strategies for this kind of games, which are called *asymmetric information dynamic games*, is an area of active research [21], [81] (*see* section 4.2 for more information).

In the area of cyber-security there are some research efforts that model the problem using variations of the aforementioned kind of games. In [319] the authors propose a dynamic game between the defender and the attacker interacting on a BAG, following the modelling proposed in [91]. Both players move simultaneously. The system state is imperfectly observed by the defender, while the attacker observes it without errors. The authors utilize a simulation-based methodology, called *empirical game-theoretic analysis* [276], to construct and analyse game models over some heuristic strategies. As the formulated game falls into the category of POSGs which is far too complex for analytic solution, the authors employ this simulation-based methodology

to evaluate these heuristic strategies. They show that the defence heuristics proposed outperform many baselines and they are robust to the defender's uncertainty of the true system state.

In [292], Zonouz et al. use a sequential Stackelberg stochastic game formulation to propose an intrusion response and recovery engine, called RRE. RRE is a two-layer architecture, with a local and a global layer, to deal with the scalability issues for large-scale networks. More specifically, RRE's local engines are located in host computers and aim at protecting their corresponding host computers. They receive IDS alerts, which are stored subsequently in the *alert database*. RRE's global engine gets high-level information from all host computers in the network, decides on optimal global response actions to take, and coordinates RRE agents to accomplish the actions by sending them relevant response commands. In addition to local security estimates from host computers, network topology is also fed into the global engine in the form of an *attack-response tree* (which is defined in [292]). The high-level architecture of RRE is depicted in Figure 4.6.

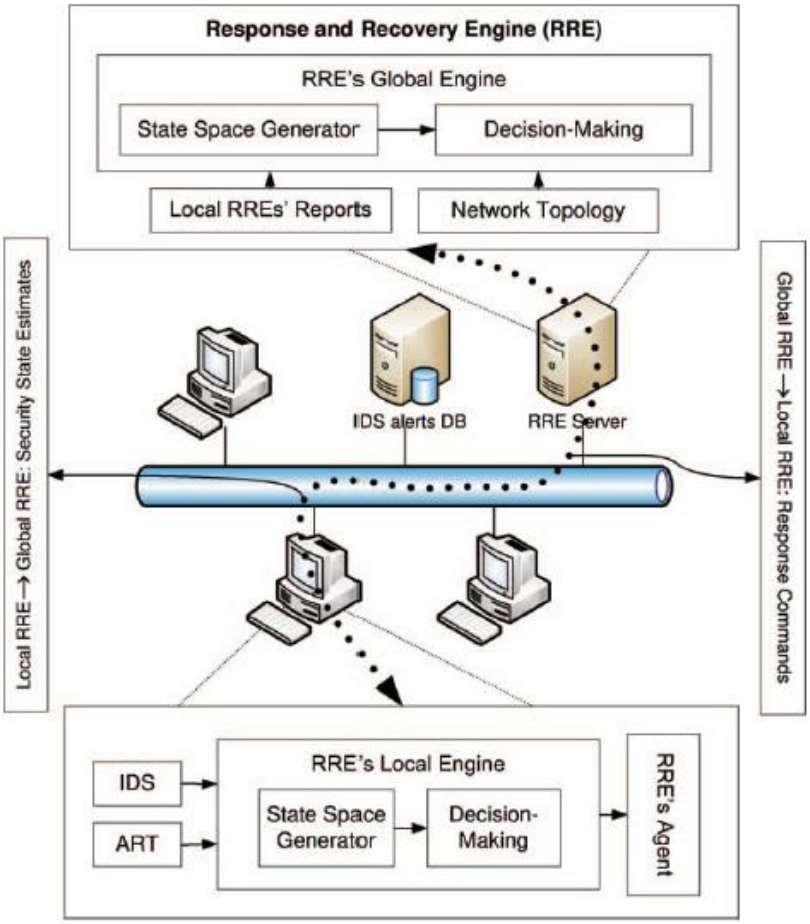


Figure 4.6. RRE architecture [292]

In the Stackelberg game formulation proposed, RRE acts as the *leader*, while the attacker acts as the *follower*. They assume a finite set of states, which the security condition of the system. After RRE selecting a defense action, the system transits probabilistically to a new state and then the attacker (after observing RRE's action) selects an attack action, resulting in a new system transition (probabilistically). The model proposed considers partial observability of the system state by the defender (i.e., the defender receives noisy observations by the IDS about the system state subject to false alarms and miss detections). Due to the partial observability of the model, the defender solves a POMDP problem to find the best-response defense action by employing *value-iteration* technique [276].

Table 4.1. State-of-the-art intrusion response system models

Paper	Problem formulation	Observability (defender)	Observability (attacker)
[299]	MDP	Full	Full
[253]	POMDP	Partial	Partial
[91]	POMDP	Partial	Partial
[90]	POMDP	Partial	Partial
[193]	SG (general sum)	Full	Full
[292]	Sequential Stackelberg stochastic game	Partial	Full
[319]	One-sided incomplete information dynamic game (finite horizon)	Partial	Full

The state-of-the-art intrusion response system models, based on SCT and GT, that have been proposed are summarized in Table 4.1.

4.3.3 Observation models based on intrusion detection systems

An important aspect of the research efforts on attacker-defender interactions for cyber-security is how the controller (defender) observes the system security state and how it is informed about any attacks performed in the system. In a cyber-security system this information is provided by the IDS, which is prone to false alarms and miss-detections. Hence, it is important to see how the state-of-the-art works build such observation models.

In [90] the information arrives at the defender in the form of a sequence of security alerts generated by the IDS as the attacker attempts exploits and progresses through the network. Each exploit if attempted, has an associated set of alerts that can generate and more than one exploit can generate the same alert. The authors consider the case when some exploits may not generate any alerts, which corresponds to the case of *stealthy* exploits. The probabilities of (correct) detection for each exploit and the probabilities of false alarms for each alert are predefined and assumed to be known by the defender. At every time instant the defender receives an *observation* vector of security alerts which consists of all security alerts triggered. This observation vector is utilized by the defender to update its belief about the system state. The same authors in their previous work [91] assume a simpler observation model without considering false positives occurrence.

The observation model in [292] accounts for both false positives and false negatives events. The IDS alerts taken as input by RRE's local engines are sent to and stored in the alert database to which each local engine subscribes to be notified when any of the alerts related to its host computer is received.

In [319] each node in the BAG is associated with a binary signal indicating whether this security condition is active or not. The signals are assumed to be independently distributed, over time and nodes. The defender receives an observation vector at every time epoch which is comprised of these signals.

4.4 Recommendations

From the state-of-the-art review, we observe that the main challenges for deploying fully automated dynamic IRSs that effectively protect cyber-systems from intelligent attackers, able to employ elaborate strategies in order to gain access in a cyber-system over the course of time, are the following two factors.

- *Complexity*: Optimal control for dynamic processes is a well-investigated subject and it is known that there are complexity issues as the state space in an MDP (with finite state space) gets larger (*curse of dimensionality* [75]). The situation gets even worse when the state is partially observable, which is the case in the POMDP model. However, in the cyber-security problem the POMDP framework is more suited, due to the fact that in reality IDSs are subject to false alarms and miss-detections.

- *Rationality*: Most works in cyber-security assume a *non-strategic* attacker. This is due to the fact that solving dynamic games of asymmetric information (i.e., the attacker and the defender have access to different information at every time instant) is a challenging task and an area of active research [81]. However, this direction needs to be pursued in order to provide a complete and realistic cyber-security framework as well as to deliver useful information to security administrators.

In Cyber-Trust project an autonomic IRS system is envisaged that will alleviate the aforementioned main challenges through novel theoretical results driving the development of efficient cyber-defense algorithms. More specifically, the *structure* of the cyber-defense problem will be explored to tackle the complexity concerns, so that under certain conditions the optimal defense policy is characterized by a special structure that is efficiently determined. *Monotone optimal policies* are such examples [329].

Regarding the rationality of the attacker, novel advances in games of asymmetric information are currently investigated [81] to be employed to the cyber-defense problem and provide efficient algorithms that better model an intelligent attacker's behavior and result in more efficient defense strategies. Moreover, the derivation of the structural results (*see* above) for the multi-agent setting will further be pursued.

Finally, an interesting research avenue is studying the (more realistic) case where some components of the model, e.g. the state transition matrix, the utility functions, etc., are unknown to the agents. In this case, *learning schemes*, could be employed. A recent research effort towards this direction is presented in [373], where a Q-Learning – based algorithm is developed for adaptive cyber-defense on BAGs when the defender does not have a priori knowledge of the utility functions.

5. Conclusions

This deliverable performed a detailed review of the current state-of-the-art in Cyber-Trust's key proactive technologies, namely cyber-threat intelligence techniques, trust establishment and risk assessment, as well as game-theoretic security for intelligent cyber-defense. A number of prominent research directions in each area has been identified, along with key challenges that have to be tackled in the design of Cyber-Trust's solutions, and a number of tools/technologies that are related to these areas and could be employed for the provisioning of the platform's functionalities. The main findings of this report are summarized below:

- Regarding crawling, the exploitation of centralized or hybrid architectures of topic-specific crawlers coupled with best-first strategies is suggested for the identification of high-quality links. Moreover, technical issues regarding access to, duplicate elimination of, and freshness of crawled content needs to be further elaborated particularly for the deep/dark web.
- Data management issues can be tackled by utilizing user-level data for the ranking of sources, the use of machine-learning algorithms for text processing and concept identification/leveraging (e.g. to identify different types of actionable intelligence), and the exploration of solutions based on NoSQL datastores to tackle storage efficiency issues.
- Cyber-threat intelligence sharing, further validated the findings of previous deliverables, where MISP (resp. STIX/TAXII) was identified as the dominant sharing platform (resp. the de facto CTI exchange standards that are also supported by MISP).
- Regarding behavioral aspects in trust management systems, dynamic models and signature-based scanning are far more mature technologies than MUDs whose adoption is substantially limited. This is also the case with the usage of TPMs in IoT devices, which necessitates the implementation of software-based remote attestation techniques to perform integrity checking.
- From the examined trust models, the ones in [149], [222], [369] have been shown to be resilient to many attacks and could fit Cyber-Trust smart home or mobile domain; however, their use in a large-scale IoT context should be tested.
- None of the identified trust management systems' implementations is directly usable in Cyber-Trust, since many of them lack important features or have performance issues; however, concepts from Keynote, SAFE, and ReTrust can be exploited.
- Regarding trust- and risk-aware defense, incorporation of trust metrics in computing the probability attack existence can leverage the precision of attack detection; however, further experimentation is needed to identify efficient methods and formulate representative optimization problems in order to optimally select the security controls minimizing the residual risk.
- The primary challenges that have been identified for developing an automated and dynamic IRS for Cyber-Trust are related to the complexity issues arising due to the large state space needed for an accurate modelling of the cyber-defense problem and the attackers' rationality, i.e. the extent to which they are assumed to be intelligent/strategic.
- Tackling both challenges by means of enforcing some special, yet realistic, structure to the defender's policies (e.g. monotone policies) seems to be possible and are currently explored along with recent efficient algorithms that have been proposed for games where the players (defenders and attackers) have different degree of knowledge about the game's parameters.

6. References

[1] A. A. Garje, Prof. Bhavesh Patel, Dr. B. B. Meshram. Realizing Peer-to-Peer and Distributed Web Crawler. International Journal of Advanced Research in Computer Engineering & Technology Volume 1, Issue 4, June 2012.

[2] A. Abbasi, W. Li, V. Benjamin, S. Hu, and H. Chen, “Descriptive Analytics: Examining Expert Hackers in Web Forums,” in 2014 IEEE Joint Intelligence and Security Informatics Conference, 2014, pp. 56–63.

[3] A. Arabsorkhi, M. Sayad Haghighi and R. Ghorbanloo, "A conceptual trust model for the Internet of Things interactions," 2016 8th International Symposium on Telecommunications (IST), Tehran, 2016, pp. 89-93.

[4] A. Attiah, M. Chatterjee, and C. C. Zou, “A Game Theoretic Approach to Model Cyber-Attack and Defense Strategies,” in IEEE International Conference on Communications, 2018, vol. 2018–May.

[5] A. Barua, Thomas, S.W., Hassan, A.E., 2014. What are developers talking about? An analysis of topics and trends in Stack Overflow. Empirical Software Engineering 19, 619–654. doi:10.1007/s10664-012-9231-y

[6] A. Benjamin, Victor & Chen, Hsiu-chin. (2013). Machine Learning for Attack Vector Identification in Malicious Source Code. IEEE Intelligence and Security Informatics. 21-23. 10.1109/ISI.2013.6578779.

[7] A. Calder and S. Watkins. IT governance: A manager's guide to data security and ISO 27001/ISO 27002. Kogan Page Ltd, 2008.

[8] A. Cassandra, “Exact and Approximate Algorithms for Partially Observable Markov Decision Processes,” PhD thesis: Brown University, 1998.

[9] A. Emmanuuel Chukwudi, “Game Theory Basics and Its Application in Cyber-Security,” Adv. Wirel. Commun. Networks, vol. 3, no. 4, p. 45, 2017.

[10] A. Fielder, E. Panaousis, P. Malacaria, C. Hankin, and F. Smeraldi, “Decision support approaches for cyber-security investment,” Decis. Support Syst., vol. 86, pp. 13–23, 2016.

[11] A. Gkoulalas-Divanis and G. Loukides. PCTA: privacy-constrained clustering-based transaction data anonymization. In 2011 International Workshop on Privacy and Anonymity in Information Society (PAIS), pages 1–10, 2011.

[12] A. H. F. Laender, B. A. Ribeiro-Neto, A. S. da Silva, and J. S. Teixeira, “A brief survey of web data extraction tools,” ACM SIGMOD Rec., vol. 31, no. 2, p. 84, Jun. 2002.

[13] A. Hamza, D. Ranathunga, H. Habibi Gharakheili, M. Roughan and V. Sivaraman, "Clear as MUD: Generating, Validating, and Applying IoT Behavioural Profiles", ACM Sigcomm Workshop on IoT Security and Privacy (IoT S&P), Budapest, Hungary, Aug 2018.

[14] A. Harth, Jürgen Umbrich, Stefan Decker: MultiCrawler: A Pipelined Architecture for Crawling and Indexing Semantic Web Data. International Semantic Web Conference 2006: 258-271

[15] A. Heydon, & Najork, M. (1999). Performance Limitations of the Java Core Libraries. Java Grande.

[16] A. J. Wiggers, F. A. Oliehoek, and D. M. Roijers, “Structure in the value function of zero-sum games of incomplete information,” In proc. of the AAMAS Workshop on Multi-Agent Sequential Decision Making in Uncertain Domains (MSDM), May 2015.

[17] A. Lazarevic, Kumar V., Srivastava J. (2005) Intrusion Detection: A Survey. In: Kumar V., Srivastava J., Lazarevic A. (eds) Managing Cyber-Threats. Massive Computing, vol 5. Springer, Boston, MA, DOI: https://doi.org/10.1007/0-387-24230-9_2

[18] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. In the IEEE International Conference on Data Engineering (ICDE), page 24, 2006.

[19] A. Monreale, G.L. Andrienko, N.V. Andrienko, F. Giannotti, D. Pedreschi, S. Rinzivillo, and S. Wrobel. Movement data anonymity through generalization. Transactions on Data Privacy, 3(2):91– 121, 2010.

[20] A. Mouton, Pierre-François Marteau. Exploiting Routing Information Encoded into Backlinks to Improve Topical Crawling. SoCPaR 2009: 659-664

[21] A. Nayyar et al., “Common information based Markov perfect equilibria for stochastic games with asymmetric information: Finite games," IEEE Trans. Automatic Control, vol. 59, no. 3, pp. 555-570, 2014.

[22] A. Neyman, and S. Sorin, “Stochastic games and applications,” vol. 570. Springer Science & Business Media, 2003.

[23] A. Ntoulas, Junghoo Cho, Christopher Olston: What's new on the web?: the evolution of the web from a search engine perspective. WWW 2004: 1-12

[24] A. Patcha and Park J-M., “An overview of anomaly detection techniques: Existing solutions and latest technological trends”, Computer Networks 51, 2007, pp. 3448–3470.

[25] A. Patrascu and E. Simion, “Game theory in cyber-security defence,” 2013 Int. Conf. Electron. Comput. Artif. Intell. ECAI 2013, no. June 2013, 2013.

[26] A. S. Mujumdar, Gayatri Masiwal and Dr. B. B. Meshram. “Analysis of Signature-Based and Behavior-Based Anti-Malware Approaches.”.

[27] A. Sachan, Wee-Yong Lim, Vrizlynn L. L. Thing: A Generalized Links and Text Properties Based Forum Crawler. Web Intelligence 2012: 113-120

[28] A. Sapienza, S. K. Ernala, A. Bessi, K. Lerman, and E. Ferrara, “DISCOVER: Mining Online Chatter for Emerging Cyber-Threats,” in Companion of the The Web Conference 2018 on The Web Conference 2018 - WWW ’18, 2018, pp. 983–990.

[29] A. Shameli-Sendi, H. Louafi, W. He and M. Cheriet, "A Defense-Centric Model for Multi-step Attack Damage Cost Evaluation," 2015 3rd International Conference on Future Internet of Things and Cloud, Rome, 2015, pp. 145-149, doi: 10.1109/FiCloud.2015.39

[30] A. Singh, Mudhakar Srivatsa, Ling Liu, Todd Miller: Apoidea: A Decentralized Peer-to-Peer Architecture for Crawling the World Wide Web. Distributed Multimedia Information Retrieval 2003: 126-142

[31] A. Srinivasan, J. Teitelbaum and J. Wu, "DRBTS: Distributed Reputation-based Beacon Trust System," 2006 2nd IEEE International Symposium on Dependable, Autonomic and Secure Computing, Indianapolis, IN, 2006, pp. 277-283. doi: 10.1109/DASC.2006.28

[32] A. K. Singh, Goyal N. (2017) MalCrawler: A Crawler for Seeking and Crawling Malicious Websites. In: Krishnan P., Radha Krishna P., Parida L. (eds) Distributed Computing and Internet Technology. ICDCIT 2017. Lecture Notes in Computer Science, vol 10109. Springer.

[33] A. Khan, Dilip Kumar Sharma: Self-Adaptive Ontology based Focused Crawler for Social Bookmarking Sites. IJIRR 7(2): 51-67 (2017)

[34] Apache Group, “Apache Spot”, http://spot.incubator.apache.org/ (accessed February 14, 2019)

[35] Apache SOLR, available at http://lucene.apache.org/solr/, last accessed 2019.

[36] B. Bamba, Ling Liu, James Caverlee, Vaibhav Padliya, Mudhakar Srivatsa, Tushar Bansal, Mahesh Palekar, Joseph Patrao, Suiyang Li and Aameek Singh. DSphere: A Source-Centric Approach to Crawling, Indexing and Searching the World Wide Web. In Proceedings of International Conference on Data Engineering, 2007

[37] B. Ladd, “The Race Between Security Professionals and Adversaries,” 2017. [Online]. Available: https://www.recordedfuture.com/vulnerability-disclosure-delay/. [Accessed: 01-Feb-2019].

[38] B. Škorić, de Hoogh, S.J.A. & Zannone, N., “Flow-based reputation with uncertainty: evidence-based subjective logic”, International Journal of Information Security 15(4), 2016, pp. 381-402, https://doi.org/10.1007/s10207-015-0298-5

[39] B. Yates, R.A. (2003). Information retrieval in the Web: beyond current search engines. Int. J. Approx. Reasoning, 34, 97-104.

[40] B.C.M. Fung, K. Wang, and P.S. Yu. Anonymizing classification data for privacy preservation. IEEE Transactions on Knowledge and Data Engineering (KDE), 19(5):711–725, 2007.

[41] B.C.M. Fung, K. Wang, and P.S. Yu. Top-down specialization for information and privacy preservation. In the International Conference on Data Engineering (ICDE), pages 205–216, 2005.

[42] C. C. Yang, X. Tang, and B. M. Thuraisingham, "An analysis of user influence ranking algorithms on Dark Web forums," in ACM SIGKDD Workshop on Intelligence and Security Informatics - ISI-KDD '10, 2010, pp. 1–7.

[43] C. Corritore, B. Kracher and S. Wiedenbeck, "On-line trust: concepts, evolving themes, a model", Int. J. Human-Computer Studies, vol. 58, no. 6, pp. 737-758, 2003.

[44] C. Dwork. Differential privacy. In Automata, languages and programming, pages 1–12, 2006.

[45] C. H. Papadimitriou and J. N. Tsitsiklis, "The complexity of Markov decision processes," Mathematics of Operations Research, vol. 12, no. 3, pp. 441-450, 1987.

[46] C. Iliou, George Kalpakis, Theodora Tsikrika, Stefanos Vrochidis, Ioannis Kompatsiaris: Hybrid Focused Crawling for Homemade Explosives Discovery on Surface and Dark Web. ARES 2016: 229-234

[47] C. J. C. H. Watkins, "Learning from Delayed Rewards," PhD thesis, Cambridge University, Cambridge, England, 1989.

[48] C. Johnson, NIST, NIST SP 800-150: Guide to Cyber-Threat Information Sharing https://csrc.nist.gov/CSRC/media//Projects/Forum/documents/aug-2016/tues300_sp800-150_cjohnson.pdf

[49] C. Labovitz, A. Ahuja, and M. Bailey, "Shining Light on Dark Address Space," 2001.

[50] C. S. D. Brown, "Investigating and prosecuting cyber-crime: Forensic dependencies and barriers to justice," Int. J. Cyber-Criminol., vol. 9, no. 1, pp. 55–119, 2015.

[51] C. T. Do et al., "Game Theory for Cyber-Security and Privacy," ACM Comput. Surv., vol. 50, no. 2, pp. 1–37, 2017.

[52] C. Vassilakis, N. Kolokotronis, K. Limniotis, C.M. Mathas, K.P. Grammatikakis, D. Kavallieros, G, Bilali, S. Shiaeles and J. Ludlow, "D2.1 Threat landscape: trends and methods", Cyber-Trust project, 2018

[53] C. Zhang, Jingwei Zhang: InForCE: Forum data crawling with information extraction. IUCS 2010: 367-373

[54] C. Zimmer, Christos Tryfonopoulos, Gerhard Weikum: MinervaDL: An Architecture for Information Retrieval and Filtering in Distributed Digital Libraries. ECDL 2007: 148-160

[55] C. A. Ardagna, E. Damiani, S.D.C. di Vimercati, S. Foresti and P. Samarati, "Trust Management", in: "Security, Privacy, and Trust in Modern Data Management", Milan Petkovic and ·Willem Jonker (eds), ISBN 978-3-540-69860-9, Springer Berlin Heidelberg New York, 2007.

[56] C. C. Aggarwal, Al-Garawi, F., & Yu, P.S. (2001). Intelligent crawling on the World Wide Web with arbitrary predicates. In Proceedings of the Tenth World Wide Web Conference (pp. 96–105). New York: ACM Press.

[57] Cloud Security Alliance, "Cloud Trust Protocol Daemon Prototype", 2016, available at https://github.com/CloudSecurityAlliancePublic/ctpd (accessed February 18, 2019).

[58] Cloud Security Alliance, "CTP Data Model and API", rev. 2.13, 2016. Available at https://downloads.cloudsecurityalliance.org/assets/research/cloudtrust-protocol/CTP-Data-Model-And-API.pdf (accessed February 18, 2019).

[59] Cloud Security Alliance, "The CTP prototype back office API", rev. 0.2, October 2015. Available at https://github.com/CloudSecurityAlliancePublic/ctpd/blob/master/client/ CTP-Admin-API.pdf (accessed February 18, 2019).

[60] K. Limniotis, et al., "Threat actors' attack strategies", Cyber-Trust, Deliverable D2.5, 2018.

[61] S. Skiadopoulos, et al., "Threat sharing methods: comparative analysis", Cyber-Trust, Deliverable D2.2, 2018

[62] R. Binnendijk, et al., "Architecture and design specifications", Cyber-Trust, Deliverable D4.1, 2019

[63] D. Ashutosh Dixit: URL ordering policies for distributed crawlers: a review. CoRR abs/1611.01228 (2016)

[64] D. Bra, P.M.E. & Post, R.D.J. (1994). Information retrieval in the World Wide Web: Making client-based searching feasible. In Proceedings of the First World-Wide Web Conference (pp. 183–192). New York: ACM Press.

[65] D. Buttler, Ling Liu, and C. Pu, “A fully automated object extraction system for the World Wide Web,” in Proceedings 21st International Conference on Distributed Computing Systems, pp. 361–370.

[66] D. Canali, Vigna, G., Kruegel, C.: Prophiler : a fast filter for the large-scale detection of malicious web pages. In: Proceeding of 20th International Conference on World Wide Web, pp. 197–206 (2011)

[67] D. Chen, Guiran Chang, Dawei Sun, Jiajia Li, Jie Jia and Xingwei Wang. “TRM-IoT: A trust management model based on fuzzy reputation for internet of things.” Comput. Sci. Inf. Syst. 8 (2011): 1207-1228.

[68] D. Fudenberg and J. Tirole, “Game Theory,” Cambridge, MA, USA: MIT Press, 1991.

[69] D. Güemes-Peña, & Nozal, Carlos & Sánchez, Raúl & Maudes-Raedo, Jesús. (2018). Emerging topics in mining software repositories: Machine learning in software repositories and datasets. Progress in Artificial Intelligence. 7. 10.1007/s13748-018-0147-7.

[70] D. H. Chau, Shashank Pandit, Samuel Wang, Christos Faloutsos: Parallel crawling for online social networks. WWW 2007: 1283-1284

[71] D. Kumar Sharma, A. K. Sharma: A Novel Architecture for Deep Web Crawler. IJITWE 6(1): 25-48 (2011)

[72] D. L. Quoc, Christof Fetzer, Pascal Felber, Etienne Rivière, Valerio Schiavoni, Pierre Sutra: UniCrawl: A Practical Geographically Distributed Web Crawler. CLOUD 2015: 389-396

[73] D. Lin, S. Gurung, W. Jiang, and A. Hurson. Privacy-preserving location publishing under road-network constraints. In 15th International Conference on Database Systems for Advanced Applications, DASFAA ’10, pages 17–31, 2010,

[74] D. Lopresti and Andrew Tomkins. 1997. Block edit models for approximate string matching. Theor. Comput. Sci. 181, 1 (July 1997), 159-179. DOI=http://dx.doi.org/10.1016/S0304-3975(96)00268-X

[75] D. P. Bertsekas, “Dynamic programming and optimal control,” Belmont, MA: Athena scientific, 2005.

[76] D. Sepandar. Kamvar, Mario T. Schlosser, and Hector Garcia-Molina. 2003. The Eigentrust algorithm for reputation management in P2P networks. In Proceedings of the 12th international conference on World Wide Web (WWW '03). ACM, New York, NY, USA, 640-651. DOI: https://doi.org/10.1145/775152.775242

[77] D. Shen, C. Drive, E. Blasch, A. Snaa, and M. Kruger, “Game Theoretic Solutions to Cyber-Attack and Network Defense Problems,” in Security, no. Track 2, 2015.

[78] D. Shen, G. Chen, L. Haynes, and E. Blasch, “Strategies comparison for game theoretic cyber-situational awareness and impact assessment,” FUSION 2007 - 2007 10th Int. Conf. Inf. Fusion, no. August, 2007.

[79] D. Shen, Genshe Chen, Jose B. Cruz, Jr., Leonard Haynes, Martin Kruger, and Erik Blasch. 2007, “A markov game theoretic data fusion approach for cyber-situational awareness,” in Proc. SPIE Defense+ Security, vol. 3, pp. 65710F–65710F.

[80] D. Silver and J. Veness, “Monte-Carlo planning in large POMDPs,” in Proc. Adv. Neural Inf. Process. Syst., 2010, pp. 2164–2172.

[81] D. Vasal, A. Sinha, and A. Anastasopoulos, “A systematic process for evaluating structured perfect Bayesian equilibria in dynamic games with asymmetric information,” IEEE Trans. Automatic Control, 2018.

[82] D. Yadav, A. K. Sharma, J. P. Gupta, N. Garg, A. Mahajan: Architecture for Parallel Crawling and Algorithm for Change Detection in Web Pages. ICIT 2007: 258-264

[83] Deloitte, 2015, Building an informed community New cyber-threat landscape makes sharing intelligence imperative, https://www2.deloitte.com/content/dam/Deloitte/ca/Documents/risk/cyber%20security-CTI-sharing.PDF

[84] E. A. Hansen, D. S. Bernstein, and S. Zilberstein, “Dynamic programming for partially

observable stochastic games," in proc. of the National Conference on Artificial Intelligence, pp. 709–715, 2004.

[85] E. C. Amadi, G. E. Eheduru, F. U. Eze, and C. Ikerionwu, "Game Theory Application in Cyber-Security ; A Review," in IEEE 3rd International Conference on Electro-Technology for National Development (NIGERCON) Game, 2018, no. January.

[86] E. Damiani, De Capitani di Vimercati, Stefano Paraboschi, Pierangela Samarati, and Fabio Violante, "A reputation-based approach for choosing reliable resources in peer-to-peer networks". In Proceedings of the 9th ACM conference on Computer and communications security (CCS '02), Vijay Atluri (Ed.). ACM, New York, NY, USA, 2002, 207-216. DOI=http://dx.doi.org/10.1145/586110.586138

[87] E. J. Sondik, "The optimal control of partially observable Markov processes," Ph.D. thesis, Stanford University, 1971.

[88] E. Lear, R. Droms, D. Romascanu, "Manufacturer Usage Description Specification", draft-ietf-opsawg-mud-25, available at https://tools.ietf.org/html/draft-ietf-opsawg-mud-25 (accessed February 12, 2019)

[89] E. Linstead, & Lopes, Cristina & Baldi, Pierre. (2009). An Application of Latent Dirichlet Allocation to Analyzing Software Evolution. 813 - 818. 10.1109/ICMLA.2008.47.

[90] E. Miehling, M. Rasouli, and D. Teneketzis, "A POMDP Approach to the Dynamic Defense of Large-Scale Cyber-Networks." IEEE Transactions on Information Forensics and Security, vol. 13, no. 10, pp. 2490-2505, 2018.

[91] E. Miehling, M. Rasouli, and D. Teneketzis, "Optimal defense policies for partially observable spreading processes on Bayesian attack graphs," in Proc. 2nd ACM Workshop Moving Target Defense, 2015, pp. 67–76.

[92] E. Tapiador JM, Garcia-Teodoro P and Diaz-Verdejo JE., "Anomaly detection methods in wired networks: a survey and taxonomy.", Computer Networks 27(16), 2004, pp. 1569–1584.

[93] ENISA, "The cost of incidents affecting CIIs," Aug. 2016.

[94] ENISA, 2015, Cyber-Security Information Sharing: An Overview of Regulatory and Non-regulatory Approaches, https://www.enisa.europa.eu/publications/cybersecurity-information-sharing

[95] ENISA, CTI Information Sharing, https://www.enisa.europa.eu/events/cti-eu-event/cti-eu-event-presentations/cti-information-sharing/

[96] ENISA, ENISA Threat Landscape 2017, ENISA, 2018.

[97] ENISA, Standards and tools for exchange and processing of actionable information/ENISA, November 2014

[98] EUROPOL, IOCTA 2016 Internet organized crime threat assessment, EUROPOL, 2016.

[99] F. A. Oliehoek, and C. Amato, "A concise introduction to decentralized POMDPs," vol. 1, Springer International Publishing, 2016.

[100] F. Ahmadi-Abkenari, Ali Selamat: An architecture for a focused trend parallel Web crawler with the application of clickstream analysis. Inf. Sci. 184(1): 266-281 (2012)

[101] F. Bao and Ing-Ray Chen. 2012. Dynamic trust management for internet of things applications. In Proceedings of the 2012 international workshop on Self-aware internet of things (Self-IoT '12). ACM, New York, NY, USA, 1-6. DOI=http://dx.doi.org/10.1145/2378023.2378025

[102] F. Bao, I. Chen and J. Guo, "Scalable, adaptive and survivable trust management for community of interest based Internet of Things systems," 2013 IEEE Eleventh International Symposium on Autonomous Decentralized Systems (ISADS), Mexico City, Mexico, 2013, pp. 1-7. doi: 10.1109/ISADS.2013.6513398

[103] F. Bonchi, L.V.S. Lakshmanan, and W.H. Wang. Trajectory anonymity in publishing personal mobility data. ACM SIGKDD Explorations Newsletter, 13(1):30–42, 2011.

[104] F. Buccafurri, Gianluca Lax, Antonino Nocera, Domenico Ursino: Crawling Social Internetworking Systems. ASONAM 2012: 506-510

[105] F. Erlandsson, Bródka, P., Boldt, M., & Johnson, H. (2017). Do We Really Need to Catch Them All? A New User-Guided Social Media Crawling Method. Entropy, 19, 686.

[106] F. Erlandsson, Roozbeh Nia, Martin Boldt, Henric Johnson, Shyhtsun Felix Wu: Crawling Online Social Networks. ENIC 2015: 9-16

[107] F. Medjek, D. Tandjaoui, I. Romdhani and N. Djedjig, "A Trust-Based Intrusion Detection System for Mobile RPL Based Networks," 2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Exeter, 2017, pp. 735-742. doi: 10.1109/iThings-GreenCom-CPSCom-SmartData.2017.113

[108] F. Menczer, Pant, G., & Srinivasan, P. (2004). Topical Web crawlers: Evaluating adaptive algorithms. ACM Transactions on Internet Technology, 4(4), 378–419.

[109] F. Stumpf, Tafreschi, O., Röder, P., & Eckert, C. (2006, November). A robust integrity reporting protocol for remote attestation. In Second Workshop on Advances in Trusted Computing (WATC'06 Fall) (pp. 25-36).

[110] F. Zhao, Jingyu Zhou, Chang Nie, Heqing Huang, Hai Jin: SmartCrawler: A Two-Stage Crawler for Efficiently Harvesting Deep-Web Interfaces. IEEE Trans. Services Computing 9(4): 608-620 (2016)

[111] F., R., Brito, P.H., Melo, J., Costa, E., Lima, R., & Freitas, F.L. (2012). An architecture-centered framework for developing blog crawlers. SAC.

[112] G. Arslan, and S. Yüksel, "Decentralized Q-learning for stochastic teams and games," IEEE Transactions on Automatic Control, vol. 62, no. 4, pp. 1545-1558, 2017.

[113] G. Danezis et al., Privacy and Data Protection by Design – from policy to engineering. ENISA. 2014.

[114] G. Drakopoulos, A. Kanavos, P. Mylonas, and S. Sioutas, "Defining and evaluating Twitter influence metrics: a higher-order approach in Neo4j," Soc. Netw. Anal. Min., vol. 7, no. 1, p. 52, Dec. 2017.

[115] G. Ghinita, P. Kalnis, and Yufei Tao. Anonymous publication of sensitive transactional data. Knowledge and Data Engineering, 23(2):161–174, 2011.

[116] G. Ghinita, Y. Tao, and P. Kalnis. On the anonymization of sparse high-dimensional data. In the International Conference on Data Engineering (ICDE), pages 715–724, 2008.

[117] G. Hattori, Matsumoto, K., Ono, C., & Takishima, Y. (2010). Identification of malicious web pages for crawling based on network-related attributes of web server. 2010 4th International Universal Communication Symposium, 355-361.

[118] G. Hurlburt. 2017. Shining Light on the Dark Web. Computer 50, 4 (April 2017), 100-105. DOI: https://doi.org/10.1109/MC.2017.110

[119] G. Husari, Ehab Al-Shaer, Mohiuddin Ahmed, Bill Chu, and Xi Niu. 2017. TTPDrill: Automatic and Accurate Extraction of Threat Actions from Unstructured Text of CTI Sources. In Proceedings of the 33rd Annual Computer Security Applications Conference (ACSAC 2017). ACM, New York, NY, USA, 103-115. DOI: https://doi.org/10.1145/3134600.3134646

[120] G. Karame, I. T. Christou and T. Dimitriou, "A Secure Hybrid Reputation Management System for Super-Peer Networks," 2008 5th IEEE Consumer Communications and Networking Conference, Las Vegas, NV, 2008, pp. 495-499. doi: 10.1109/ccnc08.2007.116

[121] G. Loukides, A. Gkoulalas-Divanis, and B. Malin. COAT: Constraint-based anonymization of transactions. Knowledge and Information Systems, 28(2):251–282, 2011.

[122] G. Marmol, Perez, M.: Providing trust in wireless sensor networks using a bioinspired technique. Telecommunication Systems, Vol. 46, Number 2, pp 163-180. (2010)

[123] G. O. Arocena and A. O. Mendelzon, "WebOQL: restructuring documents, databases and Webs," in Proceedings 14th International Conference on Data Engineering, pp. 24–33.

[124] G. Pant, & Srinivasan, P. (2005). Learning to crawl: Comparing classification schemes. ACM Transations on Information Systems, 23(4), 430–462.

[125] G. Pant, & Srinivasan, P. (2006). Link contexts in classifier-guided topical crawlers. IEEE Transactionson Knowledge and Data Engineering, 18(1), 107–122.

[126] G. Pant, Srinivasan, P., & Menczer, F. (2002). Exploration versus exploitation in topic driven crawlers. Paper presented at the Second World Wide Web Workshop on Web Dynamics, Honolulu, Hawaii.

[127] G. Pavai, T. V. Geetha: Improving the freshness of the search engines by a probabilistic approach based incremental crawler. Information Systems Frontiers 19(5): 1013-1028 (2017)

[128] G. Suryanarayana, J. R. Erenkrantz and R. N. Taylor, "An architectural approach for decentralized trust management," in IEEE Internet Computing, vol. 9, no. 6, pp. 16-23, Nov.-Dec. 2005.

[129] G. Suryanarayana, Justin R. Erenkrantz, Scott A. Hendrickson, Richard N. Taylor, “PACE: An Architectural Style for Trust Management in Decentralized Applications”, Proceedings of the fourth IEEE/IFIP working conference on software architecture, 2004, pp. 221-230.

[130] G. Teodoro P., Diaz-Verdejo J., Macia-Fernandez G. and Vazquez E., “Anomaly-based network intrusion detection: Techniques, systems and challenges”, Computers and Security, vol. 28, 2009, pp. 18-28.

[131] G. Valkanas, Alexandros Ntoulas, Dimitrios Gunopulos: Rank-Aware Crawling of Hidden Web sites. WebDB 2011

[132] G. Yücel, Çağatay; Koltuksuz, Ahmet; Ödemiş, Murat; Kademi, Anas Mu’azu; Özbilgin, “A Programmable Threat Intelligence Framework for Containerized Clouds,” in International Conference on Cyber-Warfare and Security, 2018.

[133] H. Bullot, Shyam K. Gupta, Mukesh K. Mohania: A Data-Mining Approach for Optimizing Performance of an Incremental Crawler. Web Intelligence 2003: 610-615

[134] H. Cenk Özmutlu, Seda Özmutlu: An architecture for SCS: A specialized web crawler on the topic of security. ASIST 2004: 317-326

[135] H. Chen, Chung, Y., Ramsey, M., & Yang, C. (1998a). A smart bitsy spider for the Web. Journal of the American Society for Information Science, 49(7), 604–619.

[136] H. Chen, Chung, Y., Ramsey, M., & Yang, C. (1998b). An intelligent personal spider (agent) for dynamic internet/intranet searching. Decision Support Systems, 23(1), 41–58.

[137] H. J. Oh, Dong-Hyun Won, Chonghyuck Kim, Sung-Hee Park, Yong Kim: Design and implementation of crawling algorithm to collect Deep Web information for web archiving. Data Techn. and Applic. 52(2): 266-277 (2018)

[138] H. Salah and M. Eltoweissy, "PETRA: Personalized Trust Management Architecture (Application Paper)," 2016 IEEE 17th International Conference on Information Reuse and Integration (IRI), Pittsburgh, PA, 2016, pp. 287-296, doi: 10.1109/IRI.2016.46

[139] H.Y. Limanto, Giang, N.N., Trung, V.T., Zhang, J., He, Q., & Nguyen, Q.H. (2005). An information extraction engine for web discussion forums. WWW.

[140] Haskell Team, Haskell and Android, https://wiki.haskell.org/Android (accessed February 18, 2019)

[141] http://basex.org, last accessed 2019.

[142] http://www.dtsearch.com

[143] https://ec.europa.eu/newsroom/document.cfm?doc_id=45631

[144] https://github.com/TeamHG-Memex/Formasaurus

[145] https://www.elastic.co/products/elasticsearch

[146] https://www.sap.com/products/hana.html

[147] I. Avraam & Anagnostopoulos, I. (2011). A Comparison over Focused Web Crawling Strategies. Ion Proceedings of 15th Conference on Informatics (PCI).

[148] I. C. Entre, A. Niversity, A. K. Koko, O. Tate, and D. Omputer, “A Review of Game Theory Approach To Cyber-Security Risk Management,” Niger. J. Technol., vol. 36, no. 4, pp. 1271–1285, 2017.

[149] I. Chen, F. Bao and J. Guo, "Trust-Based Service Management for Social Internet of Things Systems," in IEEE Transactions on Dependable and Secure Computing, vol. 13, no. 6, pp. 684-696, 1 Nov.-Dec. 2016.
doi: 10.1109/TDSC.2015.2420552

[150] I. Chen, J. Guo and F. Bao, "Trust Management for SOA-Based IoT and Its Application to Service Composition," in IEEE Transactions on Services Computing, vol. 9, no. 3, pp. 482-495, 1 May-June 2016.
doi: 10.1109/TSC.2014.2365797

[151] I. Dionysiou, K. Harald Gjermundrød, David E. Bakken, "GUTS: A Framework for Adaptive and Configureable Grid User Trust Service", Proceedings of the 6th International Workshop on Security and Trust Management, 2010, pp. 84-99

[152] I. Stoica, Robert Tappan Morris, David Liben-Nowell, David R. Karger, M. Frans Kaashoek, Frank Dabek, Hari Balakrishnan: Chord: a scalable peer-to-peer lookup protocol for internet applications. IEEE/ACM Trans. Netw. 11(1): 17-32 (2003)

[153] IBM Data warehouse solutions, available at https://www.ibm.com/analytics/data-warehouse, last accessed 2019.

[154] J. Audun & Ismail, Roslan. (2002). The Beta Reputation System. In: Proceedings of the 15th Bled Conference on Electronic Commerce.

[155] J. Cho, Garcia-Molina, H., & Page, L. (1998). Efficient crawling through URL ordering. Computer Networks and ISDN Systems, 30(1–7), 161–172.

[156] J. Cho, Hector Garcia-Molina: The Evolution of the Web and Implications for an Incremental Crawler. VLDB 2000: 200-209

[157] J. Domingo-Ferrer and R. Trujillo-Rasua. Microaggregation- and permutation-based anonymization of movement data. Journal of Information Sciences, 208:55–80, November 2012.

[158] J. Edwards, Kevin S. McCurley, John A. Tomlin: An adaptive model for optimizing performance of an incremental web crawler. WWW 2001: 106-113

[159] J. Filar, K. Vrieze, "Competitive Markov decision processes," Springer, Berlin Heidelberg New York, 1996.

[160] J. Golbeck, (2019). Personalizing applications through integration of inferred trust values in semantic web-based social networks.

[161] J. Guo, Ing-Ray Chen, Jeffrey J.P. Tsai, A survey of trust computation models for service management in internet of things systems, Computer Communications, Volume 97, 2017, Pages 1-14, ISSN 0140-3664, https://doi.org/10.1016/j.comcom.2016.10.012

[162] J. Harman, T. R. Nides, and S. R. Gerber, "THE DEEP WEB AND THE DARKNET: A LOOK INSIDE THE INTERNET'S MASSIVE BLACK BOX," 2015.

[163] J. Hu, M. Wellman, "Nash Q-learning for General-Sum Stochastic Games," The Journal of Machine Learning Research vol. 4, pp. 1039–1069, 2003.

[164] J. Jiang, Nenghai Yu, Chin-Yew Lin: FoCUS: learning to crawl web forums. WWW (Companion Volume) 2012: 33-42

[165] J. Liu, Zhaohui Wu, Lu Jiang, Qinghua Zheng, Xiao Liu: Crawling Deep Web Content through Query Forms. WEBIST 2009: 634-642

[166] J. Lu, Yan Wang, Jie Liang, Jessica Chen, Jiming Liu: An Approach to Deep Web Crawling by Sampling. Web Intelligence 2008: 718-724

[167] J. M. Hsieh, Steven D. Gribble, Henry M. Levy: The Architecture and Implementation of an Extensible Web Crawler. NSDI 2010: 329-344

[168] J. Madhavan, David Ko, Lucja Kot, Vignesh Ganapathy, Alex Rasmussen, Alon Y. Halevy: Google's Deep Web crawl. PVLDB 1(2): 1241-1252 (2008)

[169] J. Magidson, (1996). Maximum likelihood assessment of clinical trials based on an ordered categorical response. Drug Information Journal, 30(1), 143-170.

[170] J. Ni, N. Li and W.H. Winsborough, “Automated trust negotiation using cryptographic credentials”, Proceedings of the 12th ACM Conference on Computer and Communications Security, Alexandria, VA, USA, 2005

[171] J. R. Nurse, Creese, S., & De Roure, D. (2017). Security risk assessment in Internet of Things systems. IT Professional, 19(5), 20-26.

[172] J. Raftery, (2003). Risk analysis in project management. Routledge.

[173] J. Shin, Joo, G., & Kim, C. (2016). XPath based crawling method with crowdsourcing for targeted online market places. 2016 International Conference on Big Data and Smart Computing (BigComp), 395-397.

[174] J. Wang and F. H. Lochovsky, “Data extraction and label assignment for web databases,” in Proceedings of the twelfth international conference on World Wide Web - WWW ’03, 2003, p. 187.

[175] J. Yuan and X. Li, "A Reliable and Lightweight Trust Computing Mechanism for IoT Edge Devices Based on Multi-Source Feedback Information Fusion," in IEEE Access, vol. 6, pp. 23626-23638, 2018. doi: 10.1109/ACCESS.2018.2831898

[176] J. Zhang, R. Shankaran, A. O. Mehmet, V. Varadharajan and A. Sattar, “A trust management architecture for hierarchical wireless sensor networks”, IEEE Local Computer Network Conference, Denver, CO, 2010, pp. 264-267, doi: 10.1109/LCN.2010.5735718

[177] J. Zhao, Peng Wang: Nautilus: A Generic Framework for Crawling Deep Web. ICDKE 2012: 141-151

[178] J. J. Gardner, L. Xiong, Y. Xiao, J. Gao, A.R. Post, X. Jiang, and L. Ohno-Machado. Share: system design and case studies for statistical health information release. Journal of the American Medical Informatics Association, 20(1):109–116, 2013.

[179] J. M. Yang, Rui Cai, Chunsong Wang, Hua Huang, Lei Zhang, Wei-Ying Ma: Incorporating site-level knowledge for incremental crawling of web forums: a list-wise strategy. KDD 2009: 1375-1384

[180] Joint Technical Committee ISO/IEC JTC 1, “International standard NEN-ISO/IEC 27001: Information technology - Security techniques - Information security management systems - Requirements”, NEN-ISO/IEC, 2013.

[181] K. Boukadi, Mouna Rekik, Molka Rekik, Hanêne Ben-Abdallah:
FC4CD: a new SOA-based Focused Crawler for Cloud service Discovery. Computing 100(10): 1081-1107 (2018)

[182] K. Eldefrawy, Rattanavipanon, N., & Tsudik, G. (2017, July). HYDRA: hybrid design for remote attestation (using a formally verified microkernel). In Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks (pp. 99-110). ACM.

[183] K. Govindan, and Prasant Mohapatra. “Trust Computations and Trust Dynamics in Mobile Adhoc Networks: A Survey.” IEEE Communications Surveys & Tutorials 14 (2012): 279-298.

[184] K. Gupta, Vishal Mittal, Bazir Bishnoi, Siddharth Maheshwari, Dhaval Patel: AcT: Accuracy-aware crawling techniques for cloud-crawler. World Wide Web 19(1): 69-88 (2016)

[185] K. Irwin and T. Yu, “Preventing attribute information leakage in automated trust negotiation”, Proceedings of the 12th ACM Conference on Computer and Communications Security, Alexandria, VA, USA, 2005.

[186] K. LeFevre, D.J. DeWitt, and R. Ramakrishnan. Incognito: Efficient full-domain k-anonymity. In ACM SIGMOD international conference on Management of data, pages 49–60, 2005.

[187] K. LeFevre, D.J. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In 22nd International Conference on Data Engineering (ICDE), page 25, 2006.

[188] K. LeFevre, D.J. DeWitt, and R. Ramakrishnan. Workload-aware anonymization. In 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 277–286. ACM, 2006.

[189] K. Ntemos, J. Plata-Chaves, N. Kolokotronis, N. Kalouptsidis, and M. Moonen, “Secure Information Sharing in Adversarial Adaptive Diffusion Networks,” IEEE Trans. Signal and Information Processing over Networks, vol. 4, no. 1, pp. 111-124, 2018.

[190] K. Ntemos, N. Kolokotronis, and N. Kalouptsidis, “Trust-based strategies for wireless networks under partial monitoring,” in Proc. European Signal Processing Conference (EUSIPCO), pp. 2591-2595, 2017

[191] K. Pham, Aécio S. R. Santos, Juliana Freire: Learning to Discover Domain-Specific Web Content. WSDM 2018: 432-440

[192] K. S. McCurley: Incremental Crawling. Encyclopedia of Database Systems 2009: 1417-1421

[193] K. W. Lye, and J. M. Wing, "Game strategies in network security," International Journal of Information Security vol. 4, no. 1-2, pp. 71-86, 2005.

[194] K. Xu, Gao, K.Y., & Callan, J.P. (2018). A Structure-Oriented Unsupervised Crawling Strategy for Social Media Sites. CoRR, abs/1804.02734.

[195] K. Klinke & Renn, O. (2002). A New Approach to Risk Evaluation and Management: Risk-Based, Precaution-Based, and Discourse-Based Strategies 1. Risk Analysis: An International Journal, 22(6), 1071-1094.

[196] L. Bonomi and L. Xiong. A two-phase algorithm for mining sequential patterns with differential privacy. In 22nd ACM international conference on Conference on information and knowledge management, pages 269–278, 2013.

[197] L. Chen, Landfermann, R., Löhr, H., Rohe, M., Sadeghi, A. R., & Stüble, C. (2006, November). A protocol for property-based attestation. In Proceedings of the first ACM workshop on Scalable trusted computing (pp. 7-16). ACM.

[198] L. Creager, “MUD: The Solution to Our Messy Enterprise IoT Security Problems?”, available at https://www.darkreading.com/endpoint/mud-the-solution-to-our-messy-enterprise-iot-security-problems/a/d-id/1332384 (accessed February 13, 2019)

[199] L. Invernizzi, Benvenuti, S., Cova, M., Kruegel, C., Vigna, G.: EVILSEED : a guided approach to finding malicious web pages. In: IEEE Symposium on Security and Privacy (SP), pp. 428–442 (2012)

[200] L. Jiang, Zhaohui Wu, Qian Feng, Jun Liu, Qinghua Zheng: Efficient Deep Web Crawling Using Reinforcement Learning. PAKDD (1) 2010: 428-439

[201] L. Karttunen. Applications of Finite-State Transducers in Natural-Language Processing. Lecture Notes in Computer Science. 2000.

[202] L. Kocsis and C. Szepesvari, “Bandit based monte-carlo planning,” In Machine Learning: ECML 2006, pp. 282–293, Springer, 2006.

[203] L. S. Shapley, “Stochastic games,” in proc. of the National Academy of Sciences, vol. 39, pp. 1095–1100, 1953.

[204] L. Sweeney. k-anonymity: A model for protecting privacy. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 10(05):557– 570, 2002.

[205] L. Tchankova, (2002). Risk identification–basic stage in risk management. Environmental management and health, 13(3), 290-297.

[206] L. Xiong and Ling Liu, "PeerTrust: supporting reputation-based trust for peer-to-peer electronic communities," in IEEE Transactions on Knowledge and Data Engineering, vol. 16, no. 7, pp. 843-857, July 2004. doi: 10.1109/TKDE.2004.1318566

[207] M. Bahrami, Mukesh Singhal, Zixuan Zhuang: A cloud-based web crawler architecture. ICIN 2015: 216-223

[208] M. Blaze, J. Ioannidis and A.D. Keromytis, “Experience with the KeyNote Trust Management System: Applications and Future Directions”, Proceedings of the First International Conference on Trust Management (iTrust 2003), Springer-Verlag LNCS 2692, pp. 284-300, 2003

[209] M. Chau, & Chen, H. (2003). Comparison of three vertical search spiders. IEEE Computer, 36(5), 56–62.

[210] M. Diligenti, Coetzee, F.M., Lawrence, S., Giles, C.L., & Gori, M. (2000). Focused crawling using context graphs. In Proceedings of the 26th Conference on Very Large Databases (pp. 527–534). New York: ACM Press.

[211] M. Ester, Grob, M., & Kriegel, H. (2001). Focused Web crawling: A generic framework for specifying the user interest and for adaptive crawling strategies. In proccedings of VLDB.

[212] M. Hay, V. Rastogi, G. Miklau, and D. Suciu. Boosting the accuracy of differentially private histograms through consistency. VLDB Endowment, 3(1-2):1021–1032, 2010.

[213] M. Hurst, Alexey Maykov: Social Streams Blog Crawler. ICDE 2009: 1615-1618

[214] M. K. Muchahari and S. K. Sinha, "A New Trust Management Architecture for Cloud Computing Environment", 2012 International Symposium on Cloud and Services Computing, Mangalore, 2012, pp. 136-140, doi: 10.1109/ISCOS.2012.30

[215] M. Littman, "Friend-or-Foe Q-learning in General-Sum Games," in proceedings of the Eighteenth International Conference on Machine Learning, pp. 322–328, 2001.

[216] M. Littman, "Markov Games as a Framework for Multi-Agent Reinforcement Learning," in proc. of the Eleventh International Conference on Machine Learning, pp. 157–163, 1994.

[217] M. Mahmud, Mohammed Shamim Kaiser, M. Mostafizur Rahman, Md Arifur Rahman, Antesar M. Shabut, Shamim Al Mamun and Amir Hussain. "A Brain-Inspired Trust Management Model to Assure Security in a Cloud Based IoT Framework for Neuroscience Applications." Cognitive Computation 10 (2018): 864-873.

[218] M. Menzel, Markus Klems, Hoàng Anh Lê, Stefan Tai: A Configuration Crawler for Virtual Appliances in Compute Clouds. IC2E 2013: 201-209

[219] M. Naghavi, Mohsen Sharifi: A Proposed Architecture for Continuous Web Monitoring Through Online Crawling of Blogs. CoRR abs/1202.1837 (2012)

[220] M. Najork, & Wiener, J.L. (2001). Breadth-first search crawling yields high-quality pages. In Proceedings of the World Wide Web Conference (pp. 114–118). New York: ACM Press.

[221] M. Najork: Web Crawler Architecture. Encyclopedia of Database Systems 2009: 3462-3465

[222] M. Nitti, R. Girau and L. Atzori, "Trustworthiness Management in the Social Internet of Things," in IEEE Transactions on Knowledge and Data Engineering, vol. 26, no. 5, pp. 1253-1266, May 2014. doi: 10.1109/TKDE.2013.105

[223] M. Ozcelik, Mert & Irmak, Erdal & Ozdemir, Suat. (2017). A hybrid trust based intrusion detection system for wireless sensor networks. 1-6. 10.1109/ISNCC.2017.8071998.

[224] M. P. Wellman, "Putting the agent in agent-based modeling. Autonomous Agents and Multi-Agent Systems", vol. 30, no. 6, pp. 1175–1189, 2016.

[225] M. S. Faisal, A. Daud, A. U. Akram, R. A. Abbasi, N. R. Aljohani, and I. Mehmood, "Expert ranking techniques for online rated forums," Comput. Human Behav., Jul. 2018.

[226] M. Smith, (2002). Tools for navigating large social cyberspaces. Commun. ACM, 45, 51-55.

[227] M. Srivatsa, Balfe S., Paterson K.G. and Rohatgi P., "Trust management for secure information flows". In Proceedings of the 15th ACM conference on Computer and communications security (CCS '08). ACM, New York, NY, USA, 2008, 175-188. DOI: https://doi.org/10.1145/1455770.1455794

[228] M. Srivatsa, Li Xiong, and Ling Liu, "TrustGuard: countering vulnerabilities in reputation management for decentralized overlay networks". In Proceedings of the 14th international conference on World Wide Web (WWW '05), 2005, ACM, New York, NY, USA, 422-431. DOI: https://doi.org/10.1145/1060745.1060808

[229] M. Terrovitis and N. Mamoulis. Privacy preservation in the publication of trajectories. In 9th International Conference on Mobile Data Management (MDM), pages 65–72, 2008.

[230] M. Terrovitis, J. Liagouris, N. Mamoulis, and S. Skiadopoulos. Privacy preservation by disassociation. Very Large Data Bases Endowment (PVLDB), 5(10):944–955, 2012.

[231] M. Terrovitis, N. Mamoulis, and P. Kalnis. Local and global recoding methods for anonymizing set-valued data. International Journal on Very Large DataBases (VLDBJ), 20(1):83–106, 2011.

[232] M. Theoharidou, Mylonas, A., & Gritzalis, D. (2012, June). A risk assessment method for smartphones. In IFIP International Information Security Conference (pp. 443-456). Springer, Berlin, Heidelberg.

[233] M. Wiering, and M. Van Otterlo, "Reinforcement learning. Adaptation, learning, and optimization," vol. 12, page 51, 2012.

[234] M. C. Marneffe and Christopher D. Manning. 2008. The Stanford typed dependencies representation. In Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation (CrossParser '08). Association for Computational Linguistics, Stroudsburg, PA, USA, 1-8.

[235] Marlabs, "DARK WEB AND THREAT INTELLIGENCE," 2018.

[236] Microsoft Azure, available at https://azure.microsoft.com/en-us/services/search/, last accessed 2019.

[237] Microsoft Data transformation services, available at https://azure.microsoft.com/en-us/services/sql-data-warehouse/, last accessed 2019.

[238] MongoDB, available at https://www.mongodb.com, last accessed 2019.

[239] N. Djedjig, D. Tandjaoui, F. Medjek and I. Romdhani, "New trust metric for the RPL routing protocol," 2017 8th International Conference on Information and Communication Systems (ICICS), Irbid, 2017, pp. 328-335. doi: 10.1109/IACS.2017.7921993

[240] N. Glance, Hurst, M., Nigam, K., Siegler, M., Stockton, R., & Tomokiyo, T. (2005a). Analyzing online discussion for marketing intelligence. In Proceedings of the 14th International World Wide Web Conference (pp. 1172–1173). New York: ACM Press.

[241] N. Kushmerick, "Wrapper induction: Efficiency and expressiveness," Artif. Intell., vol. 118, no. 1–2, pp. 15–68, Apr. 2000.

[242] N. Li, J.C. Mitchell and W.H. Winsborough, "Beyond proof-of-compliance: Security analysis in trust management", Journal of the ACM, 52(3):474–514, 2005.

[243] N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In the IEEE International Conference on Data Engineering (ICDE), volume 7, pages 106–115, 2007.

[244] N. Mohammed, B.C.M. Fung and M. Debbabi. Walking in the crowd: anonymizing trajectory data for pattern analysis. In 18th ACM Conference on Information and knowledge Management (CIKM), pages 1441– 1444, 2009.

[245] N. Mohammed, X. Jiang, R. Chen, B.C.M. Fung, and L.σ Ohno-Machado. Privacy-preserving heterogeneous health data sharing. Journal of the American Medical Informatics Association, 20(3):462–469, 2013.

[246] N. Poolsappasit and I. Ray, "Dynamic Security Risk Management Using Bayesian Attack Graphs", IEEE Transactions on Dependable and Secure Computing, 9(1), January/February 2012.

[247] NIST, "Guide to Cyber-Threat Information Sharing," 2016.

[248] O. Abul, F. Bonchi, and M. Nanni. Anonymization of moving objects databases by clustering and perturbation. Information Systems, 35(8):884–910, 2010.

[249] O. Abul, F. Bonchi, and M. Nanni. Never walk alone: Uncertainty for anonymity in moving objects databases. In 24th International Conference on Data Engineering (ICDE), pages 376–385, 2008.

[250] O. Blanvillain, Kasioumis, N., & Banos, V. (2014). BlogForever Crawler: Techniques and Algorithms to Harvest Modern Weblogs. WIMS.

[251] O. Madani, S. Hanks, and A. Condon, "On the undecidability of probabilistic planning and infinite-horizon partially observable Markov decision problems," In Proceedings of
the Sixteenth National Conference on Artificial Intelligence. (AAAI-99), pp. 541–548, 1999.

[252] O. Morgenstern, and J. Von Neumann, "Theory of games and economic behaviour," Princeton university press, 1953.

[253] O. P. Kreidl, and T. M. Frazier, “Feedback control applied to survivability: A host-based autonomic defense system,” IEEE Trans. Rel., vol. 53, no. 1, pp. 148–166, Mar. 2004.

[254] O. Vikas, Nitin J. Chiluka, Purushottam K. Ray, Girraj Meena, Akhil K. Meshram, Amit Gupta, Abhishek Sisodia: WebMiner--Anatomy of Super Peer Based Incremental Topic-Specific Web Crawler. ICN 2007: 32

[255] Oracle data warehouse, available at https://www.oracle.com/database/data-warehouse.html, last accessed 2019.

[256] P. Bedi, Anjali Thukral, Hema Banati: Focused crawling of tagged web resources using ontology. Computers & Electrical Engineering 39(2): 613-628 (2013)

[257] P. Berger, Patrick Hennig, Justus Bross, Christoph Meinel: Mapping the Blogosphere-Towards a Universal and Scalable Blog-Crawler. SocialCom/PASSAT 2011: 672-677

[258] P. Bonatti and P. Samarati, “A unified framework for regulating access and information release on the web”, Journal of Computer Security, 10(3):241–272, 2002.

[259] P. Didier, Cyber-Security - IT Security Meets OT Security, ODVA 2018 Industry Conference & 18th Annual Meeting, October 10, 2018, https://www.odva.org/Portals/0/Library/Conference/Paper%206_2018-ODVA-Conference_Didier_Cyber%20Security%20-%20IT%20Security%20meets%20OT%20Security_FINAL.pdf (accessed February 13, 2019)

[260] P. F. Baldi, Cristina V. Lopes, Erik J. Linstead, and Sushil K. Bajracharya. 2008. A theory of aspects as latent topics. SIGPLAN Not. 43, 10 (October 2008), 543-562. DOI: https://doi.org/10.1145/1449955.1449807

[261] P. Jaganathan, T. Karthikeyan: Highly Efficient Architecture for Scalable Focused Crawling Using Incremental Parallel Web Crawler. JCS 11(1): 120-126 (2015)

[262] P. Likarish, & Jung, E. (2009). A targeted web crawling for building malicious javascript collection. CIKM-DSMM.

[263] P. Liu, “A Game Theoretic Approach to Cyber-Attack Prediction,” Training, pp. 1–26, 2005.

[264] P. N. Mahalle, P. A. Thakre, N. R. Prasad and R. Prasad, "A fuzzy approach to trust based access control in internet of things," Wireless VITAE 2013, Atlantic City, NJ, 2013, pp. 1-5. doi: 10.1109/VITAE.2013.6617083

[265] P. Nespoli, D. Papamartzivanos, F. G. Mármol, and G. Kambourakis, “Optimal countermeasures selection against cyber-attacks: A comprehensive survey on reaction frameworks,” IEEE Communications Surveys & Tutorials, 2017.

[266] P. Samarati. Protecting respondents’ identities in microdata release. IEEE Transactions of Knowledge and Data Engineering (TKDE), 13(6):1010– 1027, 2001.

[267] P. Sarbanes. Sarbanes-oxley act of 2002. The Public Company Accounting Reform and Investor Protection Act. Washington DC: US Congress, 2002.

[268] P. Srinivasan, Mitchell, J., Bodenreider, O., Pant, G., & Menczer, F. (2002, July). Web crawling agents for retrieving biomedical information. Paper presented at International Workshop on Agents in Bioinformatics (NETTAB), Bologna, Italy.

[269] P. Vassiliadis, A. Simitsis, P. Georgantas, M. Terrovitis, S. Skiadopoulos. A generic and customizable framework for the design of ETL scenarios. Inf. Syst. 30(7): 492-525, 2005.

[270] P. Vassiliadis, Z. Vagena, S. Skiadopoulos, N. Karayannidis, T.K. Sellis. ARKTOS: towards the modeling, design, control and execution of ETL processes. Inf. Syst. 26(8): 537-561, 2001.

[271] P. Vassiliadis, Z. Vagena, S. Skiadopoulos, Nikos Karayannidis. ARKTOS: A Tool For Data Cleaning and Transformation in Data Warehouse Environments. IEEE Data Eng. Bull. 23(4): 42-47, 2000.

[272] PostgreSQL Full Text Search, available at https://www.postgresql.org/docs/current/textsearch.html, last accessed 2019.

[273] Q. Cao, Vamsi Thummala, Jeffrey S. Chase, Yuanjun Yao, Bing Xie: Certificate Linking and Caching for Logical Trust. CoRR abs/1701.06562 (2017)

[274] Q. Tan, Prasenjit Mitra: Clustering-based incremental web crawling. ACM Trans. Inf. Syst. 28(4): 17:1-17:27 (2010)

[275] Q. Zheng, Zhaohui Wu, Xiaocheng Cheng, Lu Jiang, Jun Liu: Learning to crawl Deep Web. Inf. Syst. 38(6): 801-819 (2013)

[276] R. Bellman, "Dynamic Programming," Princeton University Press, 1957; republished 2003.

[277] R. Cai, Jiang-Ming Yang, Wei Lai, Yida Wang, Lei Zhang: iRobot: an intelligent crawler for web forums. WWW 2008: 447-456

[278] R. Chen, B.C. Fung, N. Mohammed, B.C. Desai, and K. Wang. Privacy-preserving trajectory data publishing by local suppression. Information Science, 231:83–97, 2013.

[279] R. Chen, B.C.M. Fung, and B.C. Desai. Differentially private trajectory data publication. Computing Research Repository, abs/1112.2020, 2011.

[280] R. Chen, N. Mohammed, B.C.M. Fung, B.C. Desai, and L. Xiong. Publishing set-valued data via differential privacy. Very Large Data Bases Endowment (PVLDB), 4(11):1087–1098, 2011.

[281] R. D. Smallwood, and E. J. Sondik, "The optimal control of partially observable Markov processes over a finite horizon," Operations Research, vol. 21, no. 5, pp. 1071–1088, 1973.

[282] R. Ferreira, Fred Freitas, Patrick H. S. Brito, Jean Melo, Rinaldo Lima, Evandro Costa: RetriBlog: An architecture-centered framework for developing blog crawlers. Expert Syst. Appl. 40(4): 1177-1195 (2013)

[283] R. Ferreira, Rinaldo Lima, Jean Melo, Evandro Costa, Frederico Luiz Gonçalves de Freitas, Henrique Pacca Loureiro Luna: RetriBlog: a framework for creating blog crawlers. SAC 2012: 696-701

[284] R. Friedman and Amit Portnoy. 2015. A generic decentralized trust management framework. Softw. Pract. Exper. 45, 4 (April 2015), 435-454. DOI=http://dx.doi.org/10.1002/spe.2226

[285] R. Gaur, Dilip Kumar Sharma: Focused crawling with ontology using semi-automatic tagging for relevancy. IC3 2014: 501-506

[286] R. Madaan, Ashutosh Dixit, A. K. Sharma, Komal Kumar Bhatia: A Framework for Incremental Domain-Specific Hidden Web Crawler. IC3 (1) 2010: 412-422

[287] R. Mayer, J. Davis and F. Schoorman, "An integrative model of organizational trust", The Academy of Management Review, vol. 20, no. 3, pp. 709-734, 1995.

[288] R. S. Sutton and A. G. Barto, "Reinforcement Learning: An Introduction," The MIT Press, Mar. 1998.

[289] R. Trujillo-Rasua and J.p Domingo-Ferrer. On the privacy offered by (k, δ)-anonymity. Information Systems, 38(4):491–494, June 2013.

[290] R. Yarovoy, F. Bonchi, L.V.S. Lakshmanan, and W.H. Wang. Anonymizing moving objects: how to hide a mob in a crowd? In 12th International Conference on Extending DataBase Technology (EDBT), pages 72–83, 2009.

[291] R. J. Bayardo and R. Agrawal. Data privacy through optimal k-anonymization. In 21st International Conference on Data Engineering (ICDE), pages 217–228, 2005.

[292] S. A. Zonouz, H. Khurana, W. H. Sanders, and T. M. Yardley, "RRE: A game-theoretic intrusion response and recovery engine," IEEE Trans. Parallel Distrib. Syst., vol. 25, no. 2, pp. 395–406, Feb. 2014.

[293] S. Agarwal, & Sureka, A. (2015). A Topical Crawler for Uncovering Hidden Communities of Extremist Micro-Bloggers on Tumblr. #MSM.

[294] S. Agarwal, Ashish Sureka: A Topical Crawler for Uncovering Hidden Communities of Extremist Micro-Bloggers on Tumblr. #MSM 2015: 26-27

[295] S. Catanese, Pasquale De Meo, Emilio Ferrara, Giacomo Fiumara, Alessandro Provetti: Crawling Facebook for social network analysis purposes. WIMS 2011: 52

[296] S. Chakrabarti, Punera, K., & Subramanyam, M. (2002). Accelerated focused crawling through online relevance feedback. In Proceedings of the 11th International World Wide Web Conference (pp. 148–159). New York: ACM Press.

[297] S. Chakrabarti, Van Den Berg, M., & Dom, B. (1999). Focused crawling: A new approach to topic-specific resource discovery. In Proceedings of the Eighth World Wide Web Conference (pp. 1623–1640). New York: ACM Press.

[298] S. G. Shaila, A. Vadivel: Architecture specification of rule-based Deep Web crawler with indexer. I. J. Knowledge and Web Intelligence 4(2/3): 166-186 (2013)

[299] S. Iannucci, and S. Abdelwahed, “A probabilistic approach to autonomic security management,” in Proc. 13th IEEE Int. Conf. Autonomic Comput., Jul. 2016, pp. 157–166.

[300] S. Kabou et al., A Survey on Privacy Preserving Dynamic Data Publishing, International Journal of Organizational and Collective Intelligence, Volume 8, Issue 4, 2018.

[301] S. Michel, Peter Triantafillou, Gerhard Weikum: MINERVAinfinity: A Scalable Efficient Peer-to-Peer Search Engine. Middleware 2005: 60-81

[302] S. Namal, H. Gamaarachchi, G. MyoungLee and T. Um, "Autonomic trust management in cloud-based and highly dynamic IoT applications," 2015 ITU Kaleidoscope: Trust in the Information Society (K-2015), Barcelona, 2015, pp. 1-8. doi: 10.1109/Kaleidoscope.2015.7383635

[303] S. Prajapati, Kumar, Suvamoy Changder and Anirban Sarkar. “Trust Management Model for Cloud Computing Environment.” CoRR abs/1304.5313 (2013): n.

[304] S. R. Etesami, and T. Başar, “Dynamic Games in Cyber-Physical Security: An Overview,” Dynamic Games and Applications, pp. 1-30, 2019.

[305] S. Raghavan and Hector Garcia-Molina. 2001. Crawling the Hidden Web. In Proceedings of the 27th International Conference on Very Large Data Bases (VLDB '01)

[306] S. Ratnasamy, Paul Francis, Mark Handley, Richard M. Karp, Scott Shenker: A scalable content-addressable network. SIGCOMM 2001: 161-172

[307] S. Ross, J. Pineau, S. Paquet, and B. Chaib-Draa, “Online planning algorithms for POMDPs,” Journal of Artificial Intelligence Research, vol. 32, pp. 663-704, 2008.

[308] S. S. Jodha Khalsa, Chris A. Mattmann, Ruth E. Duerr: Deep Web crawling for insights from polar data. IGARSS 2017: 376-379

[309] S. Sarwade, Patil, P.D.D.: Document-based and URL-based features for automatic classification of cross-site scripting in web pages. IOSR J. Eng. 3, 1–10 (2013)

[310] S. Sizov, & Biwer, Michael & Graupmann, Jens & Siersdorfer, Stefan & Theobald, Martin & Weikum, Gerhard & Zimmer, Patrick. (2002). The BINGO! System for Information Portal Generation And Expert Web Search.

[311] S. Sizov, Graupmann, J., & Theobald, M. (2003). From focused crawling to expert information: An application framework for Web exploration and portal generation. In Proceedings of the 29th International Conference on Very Large Databases (pp. 1105–1108). New York: ACM Press.

[312] S. Ugurel, & Krovetz, Robert & Lee Giles, C. (2002). What’s the code? automatic classification of source code archives. 639-644. 10.1145/775047.775141.

[313] S. Ye, Juan Lang, Shyhtsun Felix Wu: Crawling Online Social Graphs. APWeb 2010: 236-242

[314] Searchdaimon, available at http://www.searchdaimon.com, last accessed 2019.

[315] T. Binz: Crawling von Enterprise Topologien zur automatisierten Migration von Anwendungen: eine Cloud-Perspektive. University of Stuttgart 2015

[316] T. Cassandra. pomdp-solve: POMDP solver software, v5.4, 2003.

[317] T. Fu, Ahmed Abbasi, and Hsinchun Chen. 2010. A focused crawler for Dark Web forums. J. Am. Soc. Inf. Sci. Technol. 61, 6 (June 2010), 1213-1231. DOI=http://dx.doi.org/10.1002/asi.v61:6

[318] T. Furche, Georg Gottlob, Giovanni Grasso, Christian Schallhart, Andrew Jon Sellers: OXPath: A language for scalable data extraction, automation, and crawling on the Deep Web. VLDB J. 22(1): 47-72 (2013)

[319] T. H. Nguyen, M. Wright, M. P. Wellman, and S. Baveja, “Multi-stage attack graph security games: Heuristic strategies, with empirical game theoretic analysis,” in Proc. ACM Workshop Moving Target Defense, 2017, pp. 87–97.

[320] T. H. Noor, Quan Z. Sheng, Abdullah Alfazi, Anne H. H. Ngu, Jeriel Law: CSCE: A Crawler Engine for Cloud Services Discovery on the World Wide Web. ICWS 2013: 443-450

[321] T. Peng, Lu Liu. Clustering-Based Topical Web Crawling for Topic-Specific Information Retrieval Guided by Incremental Classifier. International Journal of Software Engineering and Knowledge Engineering 25(1): 147-168 (2015)

[322] T. S. Dybedokken, “Trust Management in Fog Computing”, NTNU, 2017

[323] T. Sommestad, M. Ekstedt and L. Nordstrom, "Modeling Security of Power Communication Systems Using Defense Graphs and Influence Diagrams," in IEEE Transactions on Power Delivery, vol. 24, no. 4, pp. 1801-1808, Oct. 2009., doi: 10.1109/TPWRD.2009.2028796

[324] T. Spyridopoulos, G. Karanikas, T. Tryfonas, and G. Oikonomou, “A game theoretic defence framework against DoS/DDoS cyber-attacks,”. Computers & Security, vol. 38, pp. 39-50, 2013.

[325] T. Yu, M. Winslett and K.E. Seamons, “Supporting structured credentials and sensitive policies trough interoperable strategies for automated trust”, ACM Transactions on Information and System Security (TISSEC), 6(1):1–42, 2003

[326] The OpenIOC Framework. http://www.openioc.org.

[327] U. Jensen, (2002). Probabilistic risk analysis: foundations and methods.

[328] V. Crescenzi and G. Mecca, “Grammars have exceptions,” Inf. Syst., vol. 23, no. 8, pp. 539–565, Dec. 1998.

[329] V. Krishnamurthy, “Partially Observed Markov Decision Processes," Cambridge University Press, 2016.

[330] V. L. Carolina, Mendoza and João H. Kleinschmidt. 2016. Mitigating On-Off attacks in the internet of things using a distributed trust management scheme. Int. J. Distrib. Sen. Netw. 2015, Article 233 (January 2016), 1 pages. DOI: https://doi.org/10.1155/2015/859731

[331] V. Merekoulias et al., "A trust management architecture for autonomic Future Internet," 2010 IEEE Globecom Workshops, Miami, FL, 2010, pp. 616-620, doi: 10.1109/GLOCOMW.2010.5700394

[332] V. Plachouras, Florent Carpentier, Muhammad Faheem, Julien Masanès, Thomas Risse, Pierre Senellart, Patrick Siehndel, Yannis Stavrakas: ARCOMEM Crawling Architecture. Future Internet 6(3): 518-541 (2014)

[333] V. Shkapenyuk and Suel T. Design and Implementation of a high-performance distributed web crawler. In Proc. 18th Int. Conf. on Data Engineering, 2002, pp. 357–368.

[334] V. Thummala, Jeffrey S. Chase: SAFE: A Declarative Trust Management System with Linked Credentials. CoRR abs/1510.04629 (2015)

[335] W. Liu, X. Meng, and W. Meng, “ViDE: A Vision-Based Approach for Deep Web Data Extraction,” IEEE Trans. Knowl. Data Eng., vol. 22, no. 3, pp. 447–460, Mar. 2010.

[336] W. T. Yih, Po-hao Chang, and Wooyoung Kim. 2004. Mining Online Deal Forums for Hot Deals. In Proceedings of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence (WI '04). IEEE Computer Society, Washington, DC, USA, 384-390.

[337] X. Huang, R. Yu, J. Kang and Y. Zhang, "Distributed Reputation Management for Secure and Efficient Vehicular Edge Computing and Networks," in IEEE Access, vol. 5, pp. 25408-25420, 2017. doi: 10.1109/ACCESS.2017.2769878

[338] X. Liao, Kan Yuan, XiaoFeng Wang, Zhou Li, Luyi Xing, and Raheem Beyah. 2016. Acing the IOC Game: Toward Automatic Discovery and Analysis of Open-Source Cyber-Threat Intelligence. In Proceedings

of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS '16). ACM, New York, NY, USA, 755-766. DOI: https://doi.org/10.1145/2976749.2978315

[339] X. Ou, W.F. Boyer and M.A. McQueen, "A scalable approach to attack graph generation". In Proceedings of the 13th ACM conference on Computer and communications security (CCS '06). ACM, New York, NY, USA, 336-345, 2006. DOI: https://doi.org/10.1145/1180405.1180446

[340] X. Wu, Li F (2017) A multi-domain trust management model for supporting RFID applications of IoT. PLoS ONE 12(7): e0181124. https://doi.org/10.1371/journal.pone.0181124

[341] X. Xiao and Y. Tao. Anatomy: Simple and effective privacy preservation. In 32nd International Conference on Very Large Data Bases, VLDB '06, pages 139–150. VLDB Endowment, 2006.

[342] X. Xiao, G. Wang, and J. Gehrke. Differential privacy via wavelet transforms. Knowledge and Data Engineering, 23(8):1200–1214, 2011.

[343] X. Zhang, A. Tsang, W. T. Yue, and M. Chau, "The classification of hackers by knowledge exchange behaviors," Inf. Syst. Front., vol. 17, no. 6, pp. 1239–1251, Dec. 2015.

[344] Y. B. Saied, Alexis Olivereau, Djamal Zeghlache, and Maryline Laurent. 2013. Trust management system design for the Internet of Things: A context-aware and multi-service approach. Comput. Secur. 39 (November 2013), 351-365. DOI=http://dx.doi.org/10.1016/j.cose.2013.09.001

[345] Y. Bakhdlaghi, "Snort and SSL/TLS Inspection", SANS Institute, 2017, https://www.sans.org/reading-room/whitepapers/vpns/snort-ssl-tls-inspection-37735

[346] Y. Guo, Kui Li, Kai Zhang, Gang Zhang: Board Forum Crawling: A Web Crawling Method for Web Forum. Web Intelligence 2006: 745-748

[347] Y. Guo, Liu, Z., Zhang, K., & Zhang, G. (2006). Board Forum Crawling: A Web Crawling Method for Web Forum. 2006 IEEE/WIC/ACM International Conference on Web Intelligence (WI 2006 Main Conference Proceedings)(WI'06), 745-748.

[348] Y. He, Dong Xin, Venkatesh Ganti, Sriram Rajaraman, Nirav Shah: Crawling Deep Web entity pages. WSDM 2013: 355-364

[349] Y. Hwei-Ming, Vrizlynn L. L. Thing: An enhanced intelligent forum crawler. CISDA 2012: 1-8

[350] Y. Li, Li Zhao, Xinran Liu, Peng Zhang: A Security Framework for Cloud-Based Web Crawling System. IEEE WISA 2014: 101-104

[351] Y. Li, Meng, X., Wang, L., & Li, Q. (2006). RecipeCrawler: Collecting recipe data from WWW incrementally. In Proceedings of the 7th International Conference on Web-Age Information Management (pp. 263–274). Washington, DC: IEEE.

[352] Y. Li, Yuping Wang, Erfeng Tian: A New Architecture of an Intelligent Agent-Based Crawler for Domain-Specific Deep Web Databases. Web Intelligence 2012: 656-663

[353] Y. Li, Yuping Wang, Jintao Du: E-FFC: an enhanced form-focused crawler for domain-specific Deep Web databases. J. Intell. Inf. Syst. 40(1): 159-184 (2013)

[354] Y. Liu, and H. Man, "Network vulnerability assessment using Bayesian networks," Proc. SPIE, vol. 5812, pp. 61–71, Mar. 2005.

[355] Y. Lu, H. He, H. Zhao, W. Meng, and C. Yu, "Annotating Structured Data of the Deep Web," in 2007 IEEE 23rd International Conference on Data Engineering, 2007, pp. 376–385.

[356] Y. Ruan, Arjan Durresi, A survey of trust management systems for online social communities – Trust modeling, trust inference and attacks, Knowledge-Based Systems, Volume 106, 2016, Pages 150-163, ISSN 0950-7051, https://doi.org/10.1016/j.knosys.2016.05.042

[357] Y. Shoham, and K. Leyton-Brown, "Multiagent systems: Algorithmic, game-theoretic, and logical foundations," Cambridge University Press, 2008.

[358] Y. Wang, Jiang-Ming Yang, Wei Lai, Rui Cai, Lei Zhang, Wei-Ying Ma: Exploring traversal strategy for web forum crawling. SIGIR 2008: 459-466

[359] Y. Wang, Jianguo Lu, Jessica Chen, Yaxin Li: Crawling ranked Deep Web data sources. World Wide Web 20(1): 89-110 (2017)

[360] Y. Wang, Jianguo Lu, Jessica Chen: Crawling Deep Web Using a New Set Covering Algorithm. ADMA 2009: 326-337

[361] Y. Wang, Jianguo Lu, Jessica Chen: TS-IDS Algorithm for Query Selection in the Deep Web Crawling. APWeb 2014: 189-200

[362] Y. Wang, Jianguo Lu, Jie Liang, Jessica Chen, Jiming Liu: Selecting queries from sample to crawl Deep Web data sources. Web Intelligence and Agent Systems 10(1): 75-88 (2012)

[363] Y. Xu, K. Wang, A. Fu, and R.C. Wong. Publishing skewed sensitive microdata. In SIAM International Conference on Data Mining, pages 84–93, 2010.

[364] Y. Xu, K. Wang, A.W.-C. Fu, and P.S. Yu. Anonymizing transaction databases for publication. In Knowledge Discovery and Data Mining, pages 767–775, 2008.

[365] Y. Zhang, Shuo Zeng, Li Fan, Yan Dang, Catherine A. Larson, and Hsinchun Chen. 2009. Dark Web forums portal: searching and analyzing Jihadist forums. In Proceedings of the 2009 IEEE international conference on Intelligence and security informatics (ISI'09). IEEE Press, Piscataway, NJ, USA, 71-76.

[366] Y. Zhou, & Reid, Edna & Qin, Jialun & Chen, Hsiu-chin & Lai, Guanpi. (2005). US Domestic Extremist Groups on the Web: Link and Content Analysis. IEEE Intelligent Systems. 20. 44-51. 10.1109/MIS.2005.96.

[367] Y. Zhou, Qin, J., Lai, G., Reid, E., & Chen, H. (2006). Exploring the Dark Side of the Web: Collection and Analysis of U.S. Extremist Online Forums. ISI.

[368] Z. A. Khan, Johanna Ullrich, Artemios G. Voyiatzis, and Peter Herrmann. 2017. A Trust-based Resilient
[369] Routing Mechanism for the Internet of Things. In Proceedings of the 12th International Conference on Availability, Reliability and Security (ARES '17). ACM, New York, NY, USA, Article 27, 6 pages. DOI: https://doi.org/10.1145/3098954.3098963

[370] Z. Bazrafshan, H. Hashemi, S. M. H. Fard and A. Hamzeh, "A survey on heuristic malware detection techniques," The 5th Conference on Information and Knowledge Technology, Shiraz, 2013, pp. 113-120. doi: 10.1109/IKT.2013.6620049

[371] Z. Fang, X. Zhao, Q. Wei, G. Chen, Y. Zhang, C. Xing, W. Li, and H. Chen, “Exploring key hackers and cybersecurity threats in Chinese hacker communities,” in 2016 IEEE Conference on Intelligence and Security Informatics (ISI), 2016, pp. 13–18.

[372] Z. Han, N. Marina, M. Debbah, and A. Hjørungnes, “Physical layer security game: interaction between source, eavesdropper, and friendly jammer,” EURASIP J. Wirel. Commun. Netw., 2009.

[373] Z. Hu, M. Zhu, and P. Liu, “Online algorithms for adaptive cyber-defense on bayesian attack graphs,” in Proc. of the 2017 Workshop on Moving Target Defense, pp. 99-109, ACM, 2017, October.

[374] Z. Inayat, A. Gani, N. B. Anuar, M. K. Khan, and S. Anwar, “Intrusion response systems: Foundations, design, and challenges,” J. Netw. Comput. Appl., vol. 62, pp. 53–74, Feb. 2016.

[375] Z. Zhang, Guoqing Dong, Zhaohui Peng, Zhongmin Yan: A Framework for Incremental Deep Web Crawler Based on URL Classification. WISM (2) 2011: 302-310

[376] Z. Zhang, Olfa Nasraoui, Roelof van Zwol: Exploiting Tags and Social Profiles to Improve Focused Crawling. Web Intelligence 2009: 136-139

[377] Z. Zhang, Olfa Nasraoui: Profile-Based Focused Crawler for Social Media-Sharing Websites. ICTAI (1) 2008: 317-324

[378] S. Noel, D. Wijesekera, C. Youman: Modern Intrusion Detection, Data Mining, and Degrees of Attack Guilt, In: Barbará D., Jajodia S. (eds) Applications of Data Mining in Computer Security. Advances in Information Security, vol 6. 2002:1-31, Springer, Boston, MA

[379] F. Cuppens, N. Cuppens-Boulahia, W. Kanoun, Y. Bouzida, A. Croissant: Expression and deployment of reaction policies. SITIS : 4th IEEE Conference on Signal Image Technology and Internet Based Systems (SITIS’08), Nov 2008:118 - 127

[380] J.E. López de Vergara, E. Vázquez, A. Martin, S. Dubus, M.N. Lepareux: Use of ontologies for the definition of alerts and policies in a network security platform. Journal of Networks 4(8), 2009:720–733

[381] D. Newman, K. M. Manalo, E. Tittel: Intrusion Detection Overview. Chapter in “CSIDS Exam Cram 2”, Que, 2004, ISBN-10: 0789730227.

[382] I. J. Martinez-Moyano, E. H. Rich, and S. H. Conrad: Exploring the detection process: integrating judgment and outcome decomposition. Intelligence and Security Informatics. ISI 2006. Lecture Notes in Computer Science, vol 3975. 2006, Springer, Berlin, Heidelberg.

[383] Fortinet: False Positive Mitigation for SQL Injection signatures. 2018. http://help.fortinet.com/fweb/580/Content/FortiWeb/fortiweb-admin/sqli_signature_fpm.htm (accessed February 24, 2019)

[384] S. Shiaeles, et al., “Use case scenarios”, Cyber-Trust, Deliverable D2.3, 2018

