Network intrusion prediction model using external data

Irène Förstel

TU Delft

Network intrusion prediction model using external data

by

Irène Förstel

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Tuesday August 26, 2021 at 10:00 AM.

Student number:	5149398	
Project duration:	January 8, 2021 – August 26, 2021	
Thesis committee:	Prof. Erkin,	TU Delft, Chair of committee
	dr. Gañan,	TU Delft, supervisor
	dr. Cockx,	TU Delft
	mr. Vermeer	TU Delft, daily supervisor

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft

Abstract

Nowadays does the internet presence of companies increase, and with it, their attack surface and the probability of breaches: every information system in the company's network may be an entry point for an outsider. Therefore, companies need to secure their information systems. However, current risk assessment frameworks fail to connect the security measures with the impact of future breaches, making it difficult for a company to prioritize their security investments: what security indicators should they look at to limit the impact of future intrusions?

In this report, we study how to collect external security indicators from a company's network, and how to process this information to build an intrusion prediction model. First, we build a scanning tool to retrieve relevant security indicators from the company's network (such as services misconfigurations or vulnerabilities) and public datasets. Then, we associate the collected indicators with incidents data from a Managed Security Service Provider in order to model using a Random Forest algorithm the probability of intrusions. Finally, we analyze the most significant indicators according to the model in an effort to find which indicators are the most relevant to evaluate the company's security posture. When we assess our model on real company's data, it achieves 92% accuracy on intrusions prediction.

Preface

Working on a master thesis project is a challenging task, especially during a pandemic. I learned a lot during these two years at TUDelft, and thanks to all the different professors I now have a much precise overview of cybersecurity (and computer sciences in general) than I had before. This research project also helped me a lot to understand what I wanted to work on in my future career.

I would like to thank Professor Van Eeten for steering me towards this project, as well as Dr. Gañan for supervising it. Their advice and suggestions throughout the project were extraordinarily helpful. I also would like to thank Mr. Vermeer, who advised me daily through all the topics I was unfamiliar with.

I also want to thank my friends for supporting me the whole year, especially the friends I made in the Manege de Prinsenstad, for welcoming me for two years and sharing with me the beautiful dutch culture. Many thanks also to my boyfriend for his help and support!

Finally, I would like to thank my family for their encouragement and support during my whole studies.

Irène Förstel
Delft, August 2021

Contents

List of Figures

1 Introduction

1.1. Motivation

Nowadays does the internet presence of most enterprises increase [42]: websites, cloud storage, or VPN networks are now common. An external observer can abuse it to collect data about a given company and, if he is malicious, exploit vulnerabilities and cause breaches. Such intrusions may be disastrous: if precious assets are destroyed, or the client's data is disclosed, the enterprise will suffer harmful consequences. Legal consequences can include financial penalties if sensitive data becomes public, and often require the company to notify the concerned customer. This leads to a decrease of trust in the company and a bad reputation in general.

This growing internet presence leads to an increase in the attack surface [42] (in the US, the number of data breaches increased by more than 100% between 2012 and 2020 [23]): internet-facing hosts may be vulnerable, either because of outdated software, security misconfigurations (open ports, poorly configured applications), or other attack vectors.

These misconfigurations are one way for attackers to infiltrate the private network of a company or temper with its information systems and cause breaches. Therefore, they can be considered as security indicators: they give a first indication of the security posture of a company. However, the large number of possible misconfigurations makes the exhaustive scanning of those security indicators difficult. The data exposed to an external observer can then be linked to the probability of breaches, since it is related to the security posture of a company [29]: more vulnerable hosts openly available from the outside of a network will give an attacker more entry points to abuse the network. Modeling future breaches depending on security indicators could help companies to find the most urgent vulnerabilities to patch or security controls to implement to minimize the impact of an attack.

From an attacker's point of view, the openly available hosts already tell a lot about a company's security posture and will be a target for information collection in the attack preparation. Anyone can do an external scan of the common misconfigurations and vulnerabilities, and the results can be useful to build a first approximation of the company's security indicators. External measurements are often imprecise since an outsider can only collect information about the surface of the company's network, maybe missing some important vulnerabilities on the internal network or the human factor. Also, the publicly available data about breaches is scarce [14] (companies don't want their private security information to be shared online), which makes the analysis of the security indicators (and their influence on breaches) difficult. However, this collected data can help to build a security assessment model based on the companies own security indicators rather than theoretical frameworks.

From the company's point of view, it invests in security measures to defend its network against outsiders. These security measures aim at patching the misconfigurations and vulnerabilities to prevent attacks. However, the efficacy of these security measures has historically always been difficult to measure [47], leading to a lack of incentives to invest in those security measures. Since the investment is hard to quantify, companies prefer investing in features that lead to direct outcomes. Having a more precise way of measuring the impact of security measures (as well as the impact of their absence) would help companies to calculate the best security investments depending on their budget and the threat landscape, ultimately leading to better overall internet security.

1.2. Problem statement

The goal of the project is then to predict future security incidents, based on external data, and to evaluate the accuracy of such a solution.

To what extent can we predict the occurrence and severity of future breaches in a company based on external data? To answer this question, we aim to build a precise model, using only data gathered by external scans of the enterprise's network.

Our research aims at building a model that could predict future breaches in a company based on external measurements of its network. The scope of our research is limited to a company's network, both in terms of incidents (for example, physical intrusions are not covered in our study) and security indicators.

1.2.1. Features and data collection

First, **what are the security indicators an external observer can deduce from scanning methods?** The observer can effortlessly discover from the outside several features, such as open ports and their configuration. He can only guess however the values of some other features: the port value may give indications of the applications running, or the packets' values may give indices on the OS running on the hardware since each OS tends to have its combination of default values. However, these default values can be hidden behind firewalls or network configuration, meaning the result is not always 100% accurate. This uncertainty in the feature measurement is an additional parameter to take into account while selecting the relevant indicators.

What are the most useful security indicators for that purpose? An external observer can measure a lot of different parameters, and gather additional data through external databases such as blacklists (public blacklists are a list of IPs from where malicious behavior such as spam or scanning originated). Given that there can be 65 535 ports on a given machine, and a company has more than one machine facing the internet, it is crucial to limit the number of features to measure. Otherwise, the model will be needlessly complicated to compute (even impossible for companies with many machines). To answer this question, we first need to look at the existing literature. Researchers have studied the topic: many papers provide a list of useful features to measure (and how to process them to obtain even more interesting features, such as dynamic ones). After creating a first model, this choice of features will be again evaluated (by comparing the influence of every feature alone or in combination with others) and, if necessary, modified.

How can those features be collected? Plenty of tools already exist to scan networks, such as nmap or ZMap. However, network scanning may raise alerts on the internal firewalls (scanning is not a malicious activity per se, but since hackers often use it to find vulnerabilities, it may lead to alert and blacklisting if it is not done carefully). Additionally, external datasets such as security reports and APIs exist, which can be useful to gather different data as well as to compare the temporal evolution of a feature, which may be helpful as an additional feature.

How can the data be labeled and preprocessed? After having been collected, the data will have to be preprocessed to fit the machine learning model. The labels must be coherent through the data: some labels may be grouped to avoid unbalanced data (for example, grouping together two security levels or incident types to keep enough data on each group to train and test a model).

1.2.2. Machine Learning model and parameters

Finally, **what machine-learning model can best predict future breaches based on the data we collected?** Using machine learning techniques will help to classify the data and predict future breaches based on past events. However, many different models exist: each one has different parameters that need to be carefully tuned to avoid overfitting or other prediction problems. The literature provides examples of models chosen for such a goal (for example, *Liu et al.* [29] uses Random Forest), but the parameters need to be tested and evaluated to give the best possible prediction. The model can also be a white or a black box, depending on the company's need for explainable results and precision in the prediction.

The prediction of breaches would be more interesting if the severity would be part of the prediction. **To what extent is it possible to predict such data?** Machine-learning classifiers give interesting results when given sufficient data. However, when limiting to a specific incident type or severity level, the collected data may not be enough to provide a good classifier. Some trade-offs may then work

better overall, for example by grouping several security levels when there is not enough data on one of them.

1.3. Proposed solution

1.3.1. Research steps

The goal of the project is to deliver an assessment of the most useful external security indicators that can be used to build a network intrusion prediction model (as well as an automated tool that measure these indicators, based on existing toolchains) and their influence on the prediction result and the probability of a breach; and an assessment of the quality of the model: is it useful for companies and does it give them insights about the probable future breaches and the indicators to look at to reduce the breaches' impact?

We divided the research into the following steps:

- The first step is to gather information about the companies we base our study on. What are their IP space, what are the indicators we can collect, and what are the possible useful features that we should collect to have a broad overview of the security posture of all those companies? Scanning is not the only option here, since many online reports and APIs also provide useful information about a given host. We chose features depending on what was useful and what we had access to (since some companies may not give their permission to proceed to a scan of their device, and some APIs may not be available without paying a fee).
- After choosing the features, the implementation part leads to a script that can collect those features and save them to use for later. The script itself is only a tool but necessary for the project.
- The next step is to implement a machine learning algorithm that will take as input the collected features (as well as the incident and alert data) and returns a prediction of future breaches with a given accuracy score. To improve the results and make the prediction more accurate, we tune the parameters until satisfied.
- Once we finalized the algorithm, it returns a set of results to analyze: first, we compute the influence of different features, to answer the questions about the most influential security indicators, and what to prioritize when looking to improve the security posture of a company; second to study the overall accuracy of the prediction: can the prediction be useful even after making approximations (i.e. collecting only external data and not internal data)?

1.3.2. Contribution

The solution adds the following contribution to the scientific research in the scanning and breach prediction domain: first, it provides an assessment of the most indicators features to include in a security posture assessment, based on precise incident data that not everyone has access to, as well as an automated scanning tool to retrieve these features as an external observer. Second, it provides a model to evaluate those features in a concrete context, an evaluation of the overall performance of a prediction system based on external data (for the features, but not the incidents, which allows more precision in the prediction since public incident data is scarce).

Later, a company could use the model by testing it on its private incident data, to find the most interesting security features to take care of. Since the company knows the cost of implementing each feature, and the model helps to estimate the cost of a future breach, the company can then estimate the best investments to make.

2

Background and literature

2.1. Security Investment

This master thesis is part of the TUDelft project ”Towards Outcome-based Cybersecurity Risk Management” in collaboration with a Managed Security Service Provider (MSSP). The project's goal is to study on a large scale the factors that cause a breach and the relationship between controls and security levels, both at companies' and end-users' levels. Most risk assessment methods in companies are indeed based on security frameworks (such as NIST [1], or ISO 27001 [21]) that mainly consist of standards and best practices to reduce the security risk. However, these frameworks only provide a list of controls and very little indication of how these controls are reducing the security risk and the impact of future breaches, especially for a single company; and fail to connect the security measures with the impact of future breaches, making it difficult for a company to prioritize their security investments. Many companies don't have the incentives to put in place all the recommended good practices described in such frameworks, either because they can't invest in all the necessary security equipment at once or either because they think it is not a priority. *Kipp et al.* [26] explain indeed that the market alone fails to give companies incentives to properly secure their networks due to the lack of information and coordination between the different actors.

It is then helpful for the companies to consider a more precise risk assessment method: what are the probable breaches, and how are they impacting the company? What are the most efficient controls against such intrusions, to prevent or limit the impact of the incident? Instead of using a theoretical description of an ideal secure network as a comparison basis, it would be interesting to understand how the security level of a company is related to the controls put in place.

Researchers have studied the topic, using data gathered as an external observer, and considering that the controls inside the company's network heavily influence what a third party can see from the company's network (open ports, servers, etc.), and therefore that the security level can be estimated using the data. However, predicting breaches using only external data may return poor results since the public information about reported incidents is scarce: companies are reluctant about sharing security information online, even anonymized; and the absence of any public framework to report incidents [25] does not add incentives to disclose breaches. Such databases cover only a small fraction of the incidents and not all the threat landscape targetting the companies. Using such incomplete data, it is difficult to build an accurate prediction model.

That's why it is interesting to continue the research on the influence of externally measurable factors on breaches, but using more precise data, especially incident data. To that extent, the MSSP provides us with a set of incidents (manually investigated by security analysts, gathered while monitoring client companies), with multiple levels of security, that can be used in combination with external data to predict more accurately future breaches and test whether the external data can provide a precise measurement of the company's security posture. After studying the features used in the model, we link predicted breaches to specific features, depending on the incident type or the breach's severity. These results will benefit the companies, as they will have access to tools to assess more accurately their risk. They can then decide with more precision what controls will be the best investment to reduce the probability of breach and limit the impact of an incident.

2.2. Scanning and data retrieval

To build such a model, accurate data is essential. To collect the data that is appropriate to measure the company's security indicators (or an approximation of), various scanning tools and APIs may be used.

The internet is a vast space that has been studied for years. Many projects had for objective to measure all the machines connected to the internet and that led to the development of scanning tools, such as Zmap, and APIs such a Shodan to access that data.

2.2.1. Scanning Definition

Since the internet was created, the number of machines belonging to the global network increased dramatically: the development of IoT devices alone increased a lot in the past few years to reach 12 billions connection [17, 22] in 2020. Many different people are then interested in studying the remote hosts in the network, which led to various scanning techniques. From security researchers to automated services, to attackers, being able to access some information about a remote host is useful for a lot of people.

The purpose of a scan is to collect data about a remote host. The scanning machine sends a probe packet (often using the ICMP protocol, but various methods exist, see chapter 3.3) to the targeted host and waits for a response. The results may indicate whether a given host is up or not, what ports are open on that host, or whether it is protected by a firewall or not. Depending on specific values inside the probe packets, the remote host may react differently: depending on the protocols used, the applications it runs, and the operating system behind it, the answer will be different. By analyzing the answer (or the lack thereof), an external observer can guess this information even without being granted access to the remote host.

Such information may be useful for many different usages: some use it to scan the entire internet on a specific feature and study trends across the globe; others target a specific network range to obtain the most information about it. Nowadays, scanning is a necessary step in the open-source intelligence collection, especially to understand the threat landscape and the trends in internet security.

2.2.2. Methods, tools and datasets

The scanning method that allows finding whether an IP address is a live host or not is the following: the scanning machine sends a packet requiring a response, using the **ICMP** (Internet Control Message Protocol), which is used to send error and control messages through the internet. If no firewall in between the two hosts prevents the receiving host to answer the packet, the scanning machine will receive an ICMP answer and deduce that the host is up, as well as other information. The scanning machine can then ask for more specific information, such as the open ports (port scan), as well as the services running behind it. Other types of ping exist (using different options that can help determine whether the host is down, or a firewall is preventing the host to receive or answer the ping request), and can be used in addition to having a more detailed overview of the machine and its access condition.

Most of the services run under officially assigned ports: the IANA has indeed made a list of the official port for the most popular services or the ones that have official RFC [2], with common ports and protocols described in table 2.1. It is often assumed (but not always true) that a service running under port X is the officially assigned service.

By querying specific services, the scanning machine can learn more about them: for example, an HTTP query will lead to an HTTP response if the host is an HTTP server. Each service has its own security configuration, some of which can be abused if they are not configured properly (or not up-to-date with security patches). For example, old versions of SSL/TLS certificates (used widely to protect websites but also other forms of encrypted communication) are vulnerable to a security vulnerability called ”Heartbleed” [40]. Some certificates may not be valid (either because they are too old, or not trusted by the browsers, which may indicate a security issue).

Various tools can be used to collect such data: first, one can use scripts such as nmap, Zmap [9] or masscan [16], and change the parameters to target the specific hosts or ports that one want to investigate.

Nowadays can the fastest scanning tools such as Zmap [9] or masscan [16] retrieve the entire internet in less than an hour (given that the scanning machine has enough bandwidth to send so many packets per second without flooding his network) [16]. This led to new ways of seeing the internet, and the emergence of datasets such as Shodan [31] that gather data about every public host in the world

port number	officially assigned protocol
20	File Transfer Protocol (FTP) Data Transfer
21	File Transfer Protocol (FTP) Command Control
22	Secure Shell (SSH)
23	Telnet - Unencrypted remote login service
25	Simple Mail Transfer Protocol (SMTP) E-mail Routing
53	Domain Name System (DNS) service
80	Hypertext Transfer Protocol (HTTP)
110	Post Office Protocol (POP3) for email clients
119	Network News Transfer Protocol (NNTP)
123	Network Time Protocol (NTP)
143	Internet Message Access Protocol (IMAP) for email management
161	Simple Network Management Protocol (SNMP)
194	Internet Relay Chat (IRC)
443	HTTP Secure (HTTPS) HTTP over TLS/SSL

Table 2.1: Common ports and the officially assignated protocol

(even hosts that should not be made public, for security reasons: for example, the internal Industrial Control Systems that operate the machines). If these services are useful for security engineers and researchers (since it allows these people to collect open-source intelligence about the networks to protect), they also permit attackers to find information about the network they want to attack, and therefore their existence and accessibility may be considered as an ethic problem. John Matherly, the founder of the Shodan search engine, however, affirms in an interview [8] that the real attackers who wish to prepare illegal attacks on other networks don't use the engine, because it requires to give personal information to be used -and attackers don't want to be tracked.

2.2.3. State of the art

All these tools have security applications: in their article, *Durumeric et al.* [10] describe the possible security applications for their scanner. Since the main advantage of Zmap over Nmap is its speed, researchers mainly use the scanner for trend analysis. Indeed, Zmap can scan the entire internet in a limited time, which makes it possible to periodically run the same scan and analyze the differences. For example, the article analyses the proportion of HTTP servers that use weak keys for their certificates: by doing such a scan repeatedly over several months, the authors observe general trends. However, this research is quite large: it targets the entire internet. In their research [10], the authors analyze different scans. They find that tools such as Zmap or masscan were mostly used to target an important portion of the IP space (whereas the smaller scans used other tools). These tools are built for general purposes because they are meant to be used in various different projects.

In our research, we are more interested in small-scale scans. We need to build a tool that combines the existing scanning tools to our exact purpose and retrieves the specific security indicators we are interested in; since we only target a few companies, precision is essential. Such a tool doesn't exist and will be part of our contribution.

In the end, internet scanning is often investigated. But it is not often linked to the controls and security indicators of a given company, and rather studied as a whole or as part of attackers' behavior. Indeed, researchers often target a single security indicator and study it in depth over a large number of hosts or all the internet. Our approach is different since we only target a small set of hosts from specific companies and study in depth multiple security indicators.

2.3. Related work

2.3.1. Intrusion Prediction Models

The importance of prediction

Many researchers have conducted studies on the topic of intrusion detection in cybersecurity. Indeed, with the increasing number and sophistication of information systems networks, a manual network inspection to check if there is no intrusion is not possible anymore. The automated detection systems

allow an administrator to filter the many alerts received to consider only the relevant events.

However, even if the detection systems are efficient, they only give the security administrator an extremely short time window to act and avoid any harm to a company's assets. Indeed, once the attacker gained access to the network, he has a wide range of actions, and it may be too late for the defenders to prevent the attack. This conclusion emphasizes the importance of a more proactive approach to network intrusions: by predicting a breach rather than detecting it, the defenders have a larger time window to reflect on security investments and how they could protect their network, depending on their priorities and assets.

State of the art

Since the intrusion detection research topic is older, it is also more advanced, and research about intrusion prediction is not as prominent [18]. It is, however, gaining popularity because of the advantages gained by using a proactive approach.

Defenders have adopted various approaches to building beneficial intrusion prediction models, as described in figure 2.1.

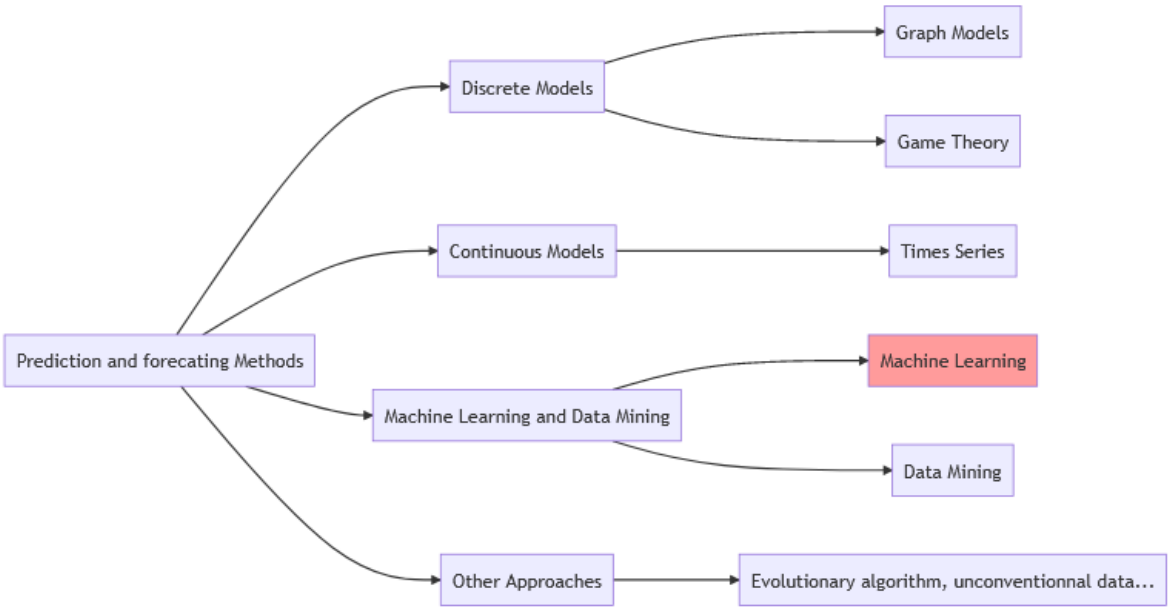


Figure 2.1: Attack prediction and forecasting methods [18]

Various researchers have studied these methods:

- **Discrete models**: models such as attack graphs or Markov models are discrete models (using defined steps, unlike continuous models). In their study, *Zhengdao et al.* use a Markov Model to predict the future steps and intentions of an attacker based on the systems calls sequences [46]. This model allows a defender to react before the end of the attack; however, the time window remains short since the model predicts the future step of an ongoing attack rather than future intrusions. Predicting the attacker's next steps is interesting for security administrators to take immediate actions to prevent the attack, but not for security investments. Since the attack is already happening, there are no possibilities to implement new security measures to prevent the intrusion.

  Similarly, the attack graphs model how an attacker achieves his end goals (for instance privilege escalation, or data exfiltration) step by step [24]. Some attack graphs studies [24] consider the security indicators of the network to build the attack graph, by adding values to each network

nodes depending on its security and impact if compromised. However, the primary usage of attack graphs is to achieve attack prediction (i.e. predicting the next steps of the attacker) rather than intrusion prediction (predicting the occurrence or not of intrusion) and its usefulness for preparing security investment is therefore limited.

- **Continuous models**: Many works have used time series because of the temporal nature of the network data. *Shi et al.* [39] combine time series and an artificial immunity model to build a prediction model of the network security situation. However, their work is not based on security indicators. If the time series gives an accurate prediction of the occurrence of incidents, its usefulness for prioritizing the security indicators that lead to an incident is limited.

- **Machine Learning and data mining models**: various machine learning models have been tested, with different methodologies. These models are described in section 2.3.4.

  Data mining has also been used to predict intrusions: *Husàk et al.* used such a methodology to forecast attacks and the next steps of an attacker. Instead of relying on a human-curated list of attacks steps, the authors used a sequential mining algorithm to correlate security alerts with incidents and later predict future steps in an intrusion. If the data mining algorithms perform well in such a situation, the authors did not test them in intrusion prediction in the long term.

- **Other models**: researchers use other more specific models to target specific attacks. In their study *Kwon et al.* [28] used the network traffic indicators collected from a honeypot and a regression model to model the probability of Distributed Denial of Service (DDoS) attack happening. However, their study concerns only a specific type of attack and aims at predicting the type and intensity of this attack rather than the target or the security indicators that are related to such an attack. Because of this, it is in our study more interesting to consider other models which can predict multiple types of intrusions.

  Unconventional data sources may also be used to discover specific attacks or new trends: instead of focusing on traditional data sources for intrusion prediction or detection (such as incidents or alerts databases, or security indicators inputs), researchers try to gather information about future attacks on online forums, or on the dark web. In their study [37], the authors target the attackers and defenders discussion boards (such as their blog or social media) and extract relevant information using text mining. This allows the researchers to detect in advance some cyberattacks such as *wannacry*. However, these unconventional sources are often unrelated to a specific company and, as a result, are not the most efficient tools for building an intrusion prediction system based on the company's security indicators.

2.3.2. Conclusion

Various studies have been conducted on the topic of attack and intrusion prediction. However, many of the techniques discussed focus on short-term predictions, such as predicting the next steps of an attacker. In the long-term perspective, our research aims at predicting intrusion and breaches before they take place, to make the company able to invest in the necessary defenses to counterattack such an attacker, which is not possible with short-term predictions. The introduction of security indicators in our research also makes discrete models or attack graphs complex to apply since they would not be scalable. By comparing existing works to the goal of our research, it appears that the machine learning models are the most relevant: indeed, discrete models are not scalable, time series are not always possible when collecting security indicators, and unconventional data only targets specific attacks.

2.3.3. The importance of a collaborative environment

Many research has been carried out using public incidents datasets. However, the threat landscape for incident prediction is evolving quickly. Since the open data is scarce, companies may rely exclusively on their own alerts and incidents data. But often a single company's alerts are not enough to understand the threat landscape and the risks encountered by the companies network.

In their research, *Hùsak et al.* emphasized the importance of data sharing between companies [20]: they used a collaborative approach to alerts correlation, by using the SABU database that allows users

to share alerts. Similar to other works ([19, 18]), the authors predict future intrusions by correlating the incidents with the alerts generated on multiple different networks.

Indeed, considering a single network as the only source for the alerts or the security indicators significantly limits the data, especially for machine learning or data mining solutions. A specific company may have never yet encountered a specific incident and alert group, but they still may be relevant for prediction purposes.

Limitations of publicly shared databases

Companies still may be reluctant to use shared databases of alerts or security indicators, as well as intrusions. First, sharing such data may induce concerns about privacy and reputation loss. Sharing details about intrusions may give insights into how the companies' networks are structured, and potential clients may be reluctant to join a network prone to intrusions. Second, there are legal requirements to satisfy to share data online, and profit from the shared data: laws such as the European GDPR require specific steps and maintenance to be taken for such a shared database. This means that building a publicly shared database requires much maintenance, and many companies may not have incentives to participate. Additionally, attackers may gain access to the public database and use it to counter the predictions, making them useless.

To counter such limitations, our research uses a private incidents database collected by an MSSP, which contains the incidents that occurred on several companies' networks, since the MSSP has several clients. This allows us, contrary to previous works, to benefit from detailed incident data from various sources.

2.3.4. Predictions using security indicators

Predicting breaches is an interesting topic since it allows companies to take action before the incident, depending on its predicted impact. That's why researchers have conducted many studies on this topic. Using security indicators allows researchers to better understand the link between controls and intrusions.

Using malicious behavior symptoms

First, work has been written on how to predict breaches for a given company in a given time window [30, 29, 12]. External indicators vary from paper to paper, but one that researchers regularly use is the index of malicious activities. It uses public blacklists as an indicator of these malicious behaviors (spam, phishing, scanning, etc.) originating from a company's network. That is what *Liu et al.* [30] did. In this article, the authors predicted future breaches based on the data found on public blacklists: indeed, if an IP is blacklisted, it means that malicious behavior such as sending spam or scanning other networks originated from it; therefore, the company's network is not safe, because an intruder was able to gain access to it and use it for non-intended behavior. It is not only a waste of resources for the company but a display of poor network security since a non-authorized third party was able to bypass the security features in place. It is not a direct measurement of the security posture of the company (it does not measure any controls implemented), but the two seem strongly related since a company with strong security policies would prevent some attackers to gain access to the network and conduct malicious activities. First, the authors measured the company's IP space by querying the Internet Registries to find what portion of the IP space was allocated to the company; then they counted how many IPs (per day) in that space were associated with malicious behavior. The result (and the dynamic features deduced from it, such as the frequency of change in the number of malicious IPs) were given to Support Vector Machines to predict breaches. However, *Kotzias et al.* [27] studied the threat landscape through blacklists and more specialized datasets (such as the reputation logs of downloaded files, obtained from internal measurements) and conclude that blacklists alone do not match the accuracy of the other datasets, and thus should be exploited with precaution. In our research, we consider blacklists along with several other external security indicators for a better precision.

Using additional indicators

To refine the model, researchers have proposed collecting additional features in addition to blacklists. Some works, such as *Liu et al.* [29] consider the mismanagement symptoms of a network as a set of indicators that measure the security posture of a company. After finding the allocated IP space of the company, they scan it and search for common misconfigurations such as open DNS resolvers (that can

lead to amplification attacks), BGP misconfigurations, or open SMTP mail relays. The new features are then combined to the VERIS community database [13], Hackmaggedon [34] and the Web Hacking Incidents Database [4] (three public databases of cyber incidents) to build a machine learning model to forecast breaches. Based on the results, the paper then discusses the relative influence of every security indicator, to determine which are the most important. *Edwards et al.* [12] also use internal data (information about file-sharing in the company as provided by the company BitSight) as well as botnet data from the Anubis network to complete the external features, and predict incidents. However, they consider only one type of incident, the botnet infection. Our goal is broader: to build a useful model for risk assessment, all types of attacks need to be considered.

These models make use of several external datasets and use them to predict incidents. However, they consider only the presence (or absence) of incidents in the prediction and do not try to examine the influence and the correlation of the diverse features on the incident type or incident severity. To estimate the impact of a cyber incident, it would be interesting to have precise results on the correlation between selected indicators and the incident type, severity, or probability. Public breaches datasets may not be precise enough (since the incident data is often not reported) to build such a model. Our research further analyze the results of the model by studying the important features: the goal is not only to build an intrusion prediction model but also to make it useful for risk assessment.

More precise predictions

Only predicting the occurrence (or not) of a breach in the next months isn't the most useful feature for risk assessment, since it does not indicate anything about the type or the severity of the breach. Research has been conducted on the study of incident type prediction: using statistical models rather than machine-learning ones, some papers [11] [38] [44] modeled the distribution of breach to predict future incidents and the induced costs. *Edwards et al.* [11] examined a dataset of data breaches (both malicious and accidental) to model the distribution of such incidents and find the probability of large breaches happening in the next years. If this work helps to estimate the general costs of breaches in a country as a whole (here, the United States), it does not rely on the security posture of a given company to estimate the costs for a single actor. In their article, *Xu et al.* [44] use a different approach for the modelization of the cyber incidents since it describes the breaches by a stochastic process, but the purpose of analyzing trends in the data remains the same (rather than providing results and tools that could be used by a company as a concrete model to reflect on their own security investments).

Sarabi et al. [38] go further in the construction of a statistical model to analyze the probability of each incident type based on the business profile of a given company. Indeed, by computing the conditional risk prediction depending on the industry type (business, computer, health... as indicated by the company's webpage category on Alexa), the paper gives an accurate prediction of the incident type. This allows a company to reduce the field of security investments and to focus more on selected incident types, depending on the industry type.

However, the results are quite broad and are mainly influenced by the industry type rather than the security posture of the company. It does not study the link between the security posture and the (future) incident type. Our research adopts a different methodology: rather than studying the trends among a group of companies, we focus on concrete security data and try to link directly those to the probability of breach, regardless of the industry group the company belongs to.

Conclusion

To conclude, the public data on breaches is scarce since many data breaches go unreported [14], therefore the model built on such data may be not precise enough to allow companies to allocate at best their security resources or estimate the impact of future breaches. That is why it is interesting to combine external data (to estimate the company's security posture based on information that anyone can access) and internal incidents reports, to have a more fine-grained analysis of the situation. It would also be interesting to link the estimated impact to the security posture (as it has been measured externally), to have an overview of the relationship between the external features and the impact of a breach: this study has been done in [29], but only for the prediction of breaches, and without linking the features to the incident type or the impact of the breach; in [12] (but only for botnet infection) without linking the features to the incident type or severity. To evaluate the result of security investments, it is useful to study the relationship between the measured features and the outcome (i.e. the impact of future breaches), and not only predict the occurrence of a breach.

Finally, these studies are mostly focused on studying trends among a large set of companies rather than directly linking the security posture of a company to the probability of incidents. A more concrete analysis would help to understand the link between security investments and the impact of future breaches if the available data about security allows such an analysis. For a company, it would provide an additional source of information to decide the security investments; for researchers, it would provide a base list of interesting security features to look at (or how to compute it depending on openly available data) for security posture measurements. Our research aims at solving this gap by providing an intrusion prediction model based on precise incident data, to understand the influence of external security indicators and the probability of breach inside a company's network.

3

Scanner

3.1. Introduction

An outsider (being either an attacker or a security researcher) can access some information about the network of a company by querying the public IP addresses inside it. However, such analysis requires some research to define a specific goal since the number of features that can be collected from a live host is huge: with 65 535 possible ports and a large number of possible protocols for each host, an exhaustive analysis of a large IP range is not possible for time reasons. *Zhang et al.* [45] select a limited number of representative security features (such as DNS security features, or whether the SSL certificates are trusted by the browser or not). *Liu et al.* [29] draw on this research and select five of these eight features to conduct their research (the Open recursive resolver, source port randomization, BGP misconfigurations, the untrusted certificates, and the SMTP server relaying).

We based our scanning script on these features as well, but couldn't analyze the results without the approval of the companies themselves. Since companies can see a scan of their network as a nuisance (because it consumes resources) as a nuisance, due to ethical concerns and consent issues with the MSSP, the scanning didn't occur.

Such a scanner is an effective tool to measure the security posture because it captures precise data associated with a given company, and homogeneous data among multiple companies if multiple companies are tested. Many scanning tools already exist, but either focusing on security features we are not interested in or as toolboxes with many possibilities and no guide to distinguish the useful features from the less useful. Our tool then acts as a stand-alone scanning tool that can gather accurate data for security analysis.

3.2. Methodology

The goal of the scanning tool is to measure the most interesting security indicators, i.e. some external measurements on the public hosts of a company related to security: for example, whether a protocol implements the recommended best practices (or not) is a security indicator.

The external features to measure can be collected using the following methodology:

- Performing a first literature review of the work done on scanning a company's network and scanning tools as well as useful features to collect for breach prediction. The literature review gives the first list of indicators to use in the model.
- Then considering the already existing tools and how to use them: many scanning tools (Zmap, etc.) already perform part of the work. However, they need to be tuned (and sometimes require additional scripts) to gather exactly the features we want.
- Contacting the companies to avoid any ethical issues: some may be reluctant on network scanning, or may blacklist any IP that conducts scanning on their network.
- The data couldn't be collected because the companies didn't give their consent. However, this can be done in further works.

3.3. Scanning methods

A scan is part of the network reconnaissance strategy: its goal is to determine what hosts are up and what ports are open on the available hosts.

A network scan is based on the following architecture: the sending host sends a probe (it can be a UDP or TCP probe, depending on the type of scan) to the receiving host. In a default configuration, the receiving host, if it is up, sends a response back to the sending host after receiving the probe. If the sending host receives the response, it means that the receiving host was up; otherwise not.

Different scanning methods exist:

- The **TCP scan**: the client sends a SYN (or another specific packet for other methods but the SYN is the most common method) packets to the host on the port he wants to test. If the port is open, it responds with an SYN/ACK packet, and if it is closed, it responds with an RST packet (to reset the TCP channel);
- The **UDP scan**: the client sends a packet to the host, and if the port is closed it responds with an ICMP packet (port unreachable) which indicates that the port is closed and otherwise, it is open.

Different scanning strategies also exist:

- The **vertical port scan** targets a single host and for every (or a subset of every) port, sends a packet to see whether this port is open or not. This strategy is often blocked by firewalls since it is easy to detect by automated rules and often used by attackers that target a specific machine as part of the reconnaissance strategy;
- The **horizontal port scan** targets a single port on a (possibly huge) list of hosts; it can be used by an attacker to exploit a specific vulnerability in any machine, for example. However, it does not give a precise overview of the other services running on a given machine.
- The **block scan** targets a list of IP addresses as well as a list of ports, and every port is tested for every machine.

This scan returns a list of ports and available hosts, which are not in themselves a result of the security posture and need to be further analyzed in order to return interesting data.

3.4. Tools used

A large number of tools already exist to perform scanning, depending on the goal of the user.

Nmap is a famous scanner that is often used in research since it allows every user to write and use custom scripts for more specific scanning purposes. For example, we can launch the Nmap script against DNS servers to check whether they are open resolvers or not. It makes it easy to design a first scanning tool to gather whatever information we need. Nmap is connection-oriented: it sends a connection request to the target and waits for the response before reaching the next target.

Because of that behavior, Nmap is not fast and therefore is not indicated for scanning large networks since it takes too much time to retrieve the results. That is why faster tools exist, such as **masscan** or **Zmap**. Both scanners are asynchronously running, which means they do not wait to receive the answer to their connection request before asking the next target. It makes them less accurate (unable to recognize dropped packets for example) but also much faster.

Zmap is a fast scanner: with a 1Gb Ethernet connection, it can scan the entire IPv4 space (2^{32} different IPs) in around 45 minutes. It solves the problem of the scanning speed. However, Zmap does not yet support the multi-port scan, which is an extensive part of our research: it only targets a single port for each running instance. To perform a multiple port scan, we should run multiple instances of Zmap simultaneously, which reduces the overall efficiency of the software. That's why we rather use masscan because it is better oriented towards our research objectives.

However, masscan is only useful for finding the hosts and open ports on a given network, whereas my analysis goes further. That is why we set up masscan to find the list of open ports, and on this port perform further analysis to obtain details. For this analysis, we used different tools, as described in figure 3.1:

- **Zgrab** to grab the banners for a predetermined list of common services;

- **dig** to obtain more information about the DNS configuration if the host is a DNS server (i.e. has its port 53 open);
- The **OpenSSL** library to check the certificates of a host;
- **Nmap** to check options about the SMTP hosts (Nmap alone is quite slow, but it is not problematic when checking a specific configuration on only few hosts)

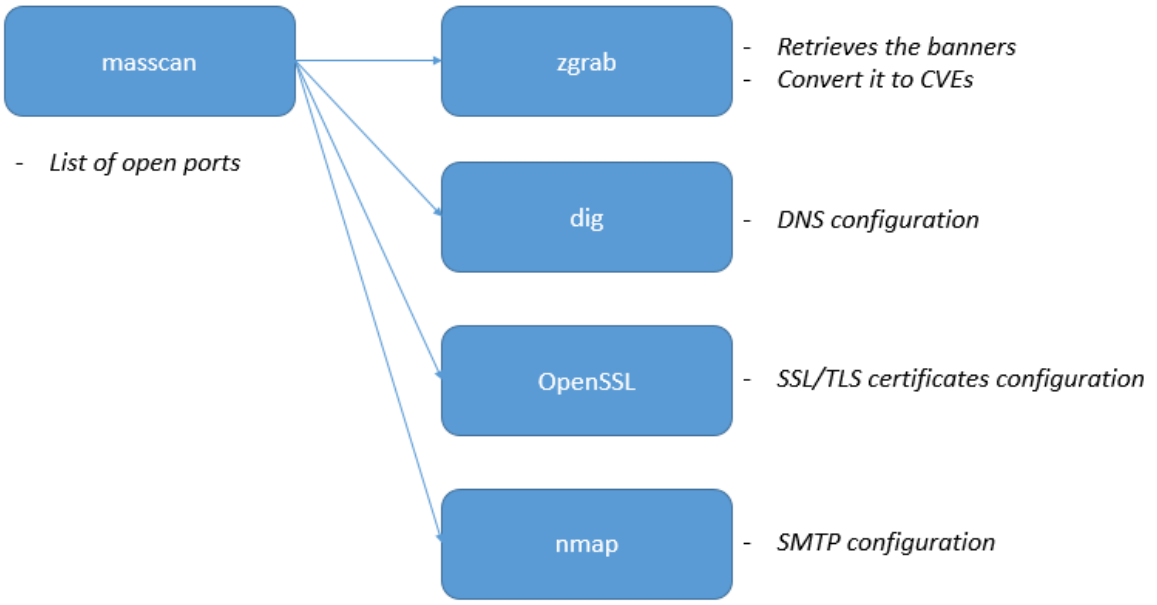


Figure 3.1: Schema of the scanning tool flow

3.5. Scanning for services

The first goal of the scanning tool is to detect what services run on a given host. Many services may run on a host, and multiple services may run on different ports. Vertical scanning is often impossible or impractical, either because it takes too much time (the goal is to scan many hosts and not only a few); or because it is automatically blocked by a firewall after a few steps. To tackle that issue, we chose the following trade-offs:

- Only scanning 1000 ports amongst the Nmap top ports (which is enough to catch around 90% of the TCP open ports and 49% of the UDP ports according to the Nmap documentation [6]): this limits the time resources it takes to scan a single host and is more likely to be allowed by a firewall;
- Considering that each service runs in the officially assigned IANA port (so instead of testing a probe of each possible service on each possible port, we assumed that the port number gave the right indication about the service), instead of testing every possible protocol on every open port.

It allows the script to scan in a reasonable amount of time.

The first step of the scanning process is to retrieve the available hosts and their open ports on the given network. This is done using the **masscan** tool in the command line:

```
masscan <network blocks> -p <ports> --rate 10000 -oJ results.json
```

3.6. Misconfigurations

Once the tool has detected which service runs on a given host on a given port, it can go more in-depth to detect the presence or absence of any known misconfigurations. We checked for the following misconfigurations, based on the ones detected by *Liu et al.* [29] (where the prediction model achieves 90% accuracy):

- Whether a DNS server is an open resolver or not. An attacker can abuse this type of DNS server to provoke amplification attacks, and therefore should be avoided in the company's private network or filtered to allow only the employees of the company to access it;
- Whether a DNS server implements source port randomization (to protect against cache poisoning attack);
- Whether an SMTP server is an open relay (which can be abused to send spam messages);
- Whether an HTTP host has a valid certificate that the browser trusts (i.e. whether the chain of trust is validated by a root certificate authority among the trusted CA database from Firefox [5]);

Rather than an indication of vulnerabilities, these misconfigurations are an indication of the security posture of a company: they go against the recommended best practices of Internet protocols and therefore, are representative of a lack of security interest in the company.

3.6.1. DNS misconfigurations

The *Domain Name System* (DNS) service stores the information about the domain names. Each DNS server keeps data about multiple domain names, such as the IP address of the machine that runs the process associated with the domain name.

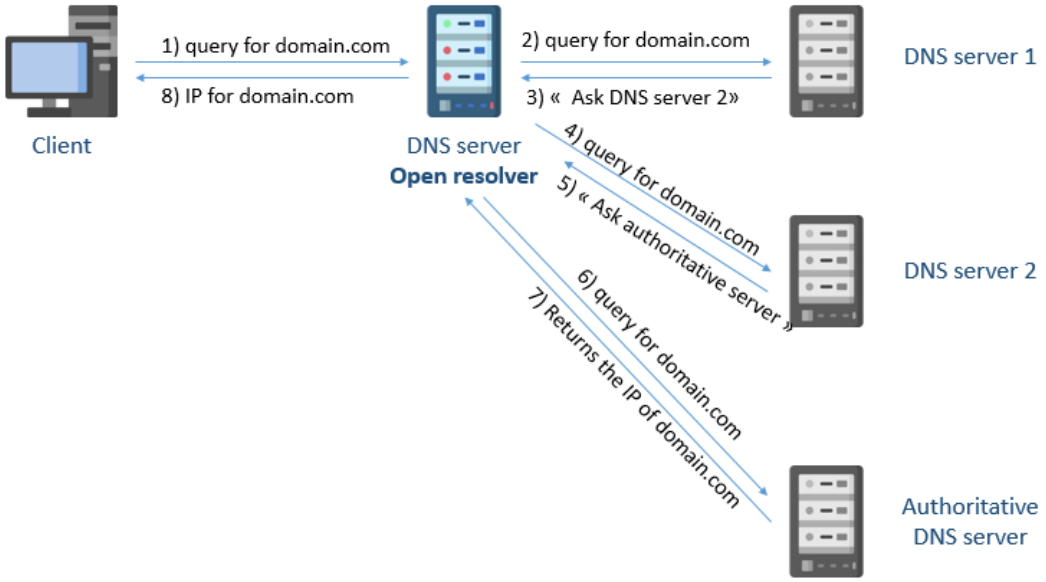
An open resolver DNS server answers recursive queries from anyone: when the server itself doesn't know the answer to the request, it will ask another DNS server about it and return the full answer to the client who originated the request (see figure 3.2). However, an attacker can easily make such a request in the name of another victim machine, which will then receive the large packets, as described in figure 3.3. Since the query packet is much smaller than the answer packet, a single attacker with limited power can generate a huge volume of packets and flood the victim machine in what is called an amplification attack. The resulting Denial of Service can make entire websites and services unavailable, such as the attack that targeted Spamhaus in 2013, caused by open resolvers [35].

Another attack that specifically targets the DNS service is the cache poisoning attack (also called DNS spoofing): to redirect users to a malicious page (instead of the real benign page), an attacker can attempt to answer the DNS queries received by the server **before** the DNS server itself: the client will cache the first answer it receives and will be redirected to the wrong machine, as described in figure 3.4. DNS packets contain a query ID that the attacker needs to guess to perform a successful cache poisoning (if the query ID of the answer is not the same as the query ID from the request, the client will ignore it). However, the attacker can perform the query as many times as he wants using the DNS extra records: even if the likelihood of guessing the right 16-bits query ID is very low, it becomes much larger in hundreds of attempts. To counteract such an attack, one solution is to increase the entropy of the numbers to guess and **randomize the source port** of the query: the attacker also has to guess the source port, which makes the likelihood of a successful attack much lower.

3.6.2. SMTP misconfiguration

The Simple Mail Transfer Protocol (DNS) is a protocol dedicated to sending and receiving electronic mail messages on a mail server. It frequently runs on port 25 over a TCP channel, and the client initiates a connection by sending string commands. Contrary to other mail protocols such as IMAP or POP, it is used only to deliver messages (and not managing email inboxes on the user side).

Some SMTP servers are configured as **open relayers**: they pass on every message they receive, without looking at where it comes from or where it is supposed to go. Because of this lack of security, an attacker may abuse the open mail relays such as spammers who send messages in bulk from these open relays. Because of this behavior, the open relayers are often blacklisted; and a given company shouldn't configure one of its public mail servers in that way.



Icons made by Freepikwww.flaticon.com

Figure 3.2: Schema of a query to an open resolver server

3.6.3. SSL/TLS certificates

With the rising adoption of encryption over the internet, many hosts nowadays use an SSL or TLS certificate to authenticate themselves and encrypt the connection with another machine using a public/private key pair. A hierarchy of authorities is needed to validate the end-entity certificate: at the top of the chain of trust, a Certificate Authority (CA) validates an intermediary certificate (which can then validate another intermediate certificate, and so on), which will validate itself the end-entity certificate. If any certificate in the chain of trust is not valid, the end-entity certificate is not valid as well, which compromises the security of the host.

Multiple reasons may cause a certificate to be invalid: certificates have specific dates of validity, after which they are not valid anymore. If a company cannot renew its certificates in time, they become invalid.

Some certificates may be valid but not properly configured: for example, some cryptographic schemes options for configuring a certificate are not up-to-date with today standards in cryptography and should not be used (for example, when using the RSA algorithm, keys should at least have 1024 bits otherwise it is feasible to break it with brute-force computation). Some others may be validated by an intermediary certificate or a certificate authority that is not trusted by most of the browsers, either because they are self-signing, or because they are not part of the certificate authority trusted by Mozilla of Chrome.

3.7. Banners and CVEs

Another potential security attribute to analyze is the banner collection: the banner request (using a tool such as ZGrab) may return as a result a string containing information about the application used by the service as well as the version (e.g. 'OpenSSH_8.0'). We can use this information to find the associated CPE (Common Platform Enumeration), which is a formalized naming scheme for applications and services, using the CPE dictionary [32]. The CPEs obtained can then be linked to the CVE (Common Vulnerabilities and Exposures) that are related to specific software and version, using NIST API [3] and the CVE database, as described in figure 3.5. Each host may therefore be linked with multiple CVE depending on the security misconfigurations that appear in the banner (if the software is not up-to-date with security patches for example). However, many banners don't contain any information about the software used and its version, which prevents us to draw a conclusion about the security status (i.e. safe or unsafe) of the given host.

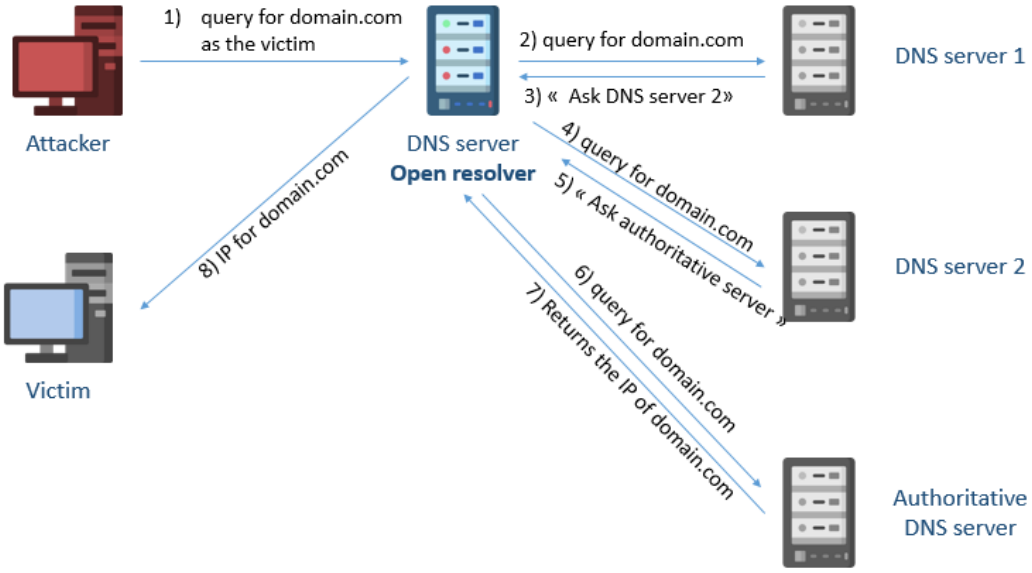


Figure 3.3: Schema of an amplification attack

To perform the banner grabbing, we used the **Zgrab** tool, which takes as input the host to scan and the service to ask. Several services are supported by Zgrab (HTTP, TLS, MySQL, MSSQL...). If either one of these services is running on the host, we launch a Zgrab banner grabbing process.

The severity of the security flaws found in the companies' machines can be another feature. Similar to the other scanned features, it requires however the company's approval before investigation.

3.8. Scanner's performance

3.8.1. Performance on a local network

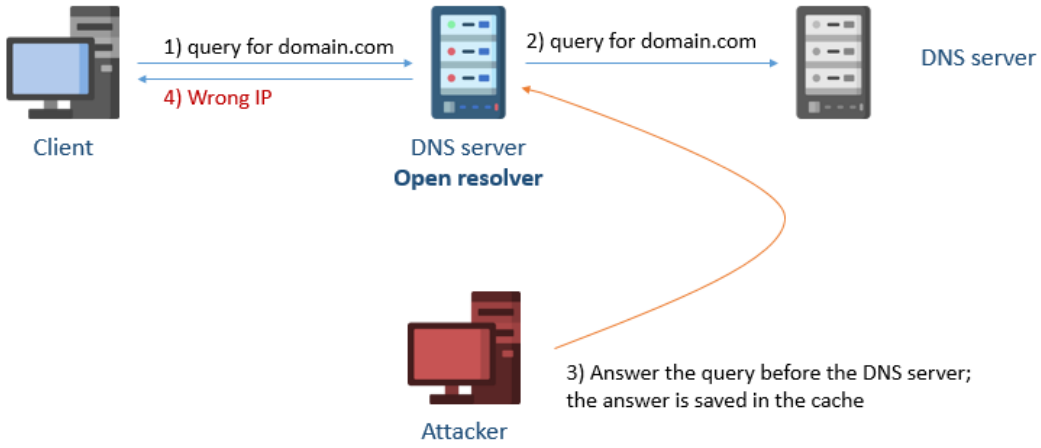
The scanner was first tested on a TUDelft local network using an available virtual machine, with the following CPU: `Intel(R) Xeon(R) Gold 5218N CPU @ 2.30GHz`. To scan the local network of 256 hosts on 1000 ports, the performance was the following:

- **Time**: 56,2461s
- **CPU usage**: up to 15%.

In total, 10 hosts were up, including 4 DNS servers and 3 HTTP servers (open on ports 80, 8080, and 443). We can decrease the time it takes for the scanner to go over the whole network by increasing the number of packets sent by masscan. However, this may overload the network depending on its bandwidth; some parts of the scan (such as API requests or using specific tools such as dig) still take a fixed amount of time that can't be reduced.

3.8.2. Performance on real hosts

Testing on a local network only does not capture well the scanning time, since real hosts outside a local network may cause additional network delays. The local TUDelft network also has a reduced set of hosts which may not represent the diversity of hosts found in other companies' networks (such as additional services that may take longer to analyze). That's why it would be interesting to measure the scanner's performance on a set of real networks; however, this requires prior approval from the network's owner for ethical reasons, and couldn't be performed on our research.



Icons made by Freepikwww.flaticon.com

Figure 3.4: Schema of a cache poisonning attack

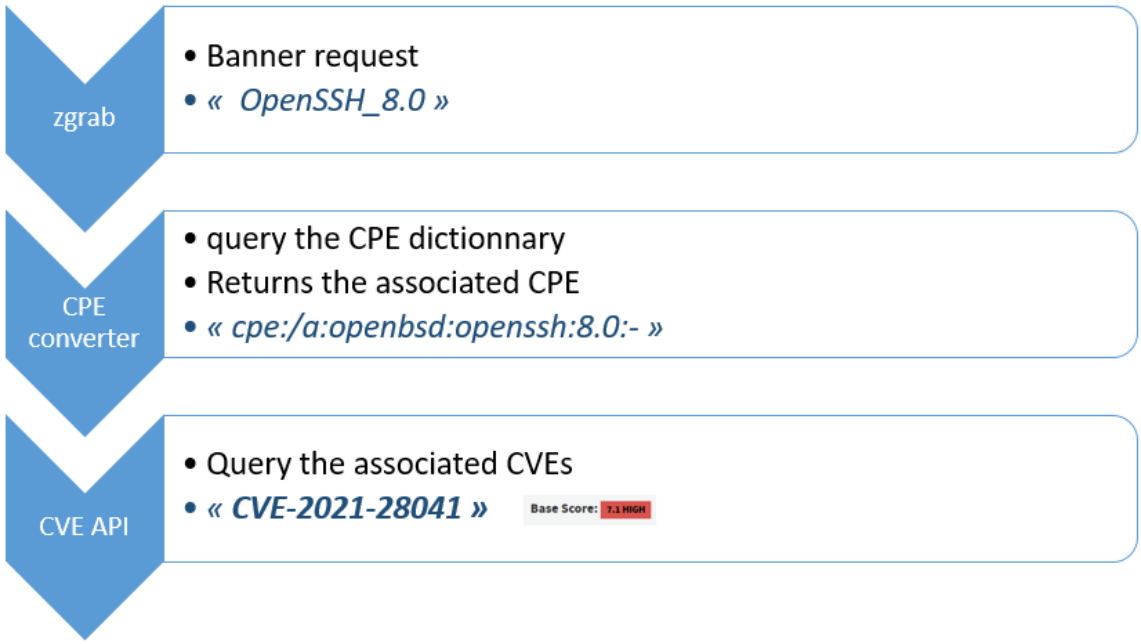


Figure 3.5: From banners to CVEs

4

Datasets

4.1. Introduction

To achieve our research, we need to gather data. The scanning algorithm gathers part of it. However, it is necessary to add other sources of data (such as the incidents, and features that can't be gathered by usual scanning methods). The goal of this part is to find security indicators (from an outsider point of view) that will later feed the machine learning algorithm. Therefore the collected data needs to be consistent and have a significant volume (i.e. too little data may lead to incorrect results in the data analysis part).

In addition to the scanning tool, therefore, we studied several datasets which collected various security features.

We separate the gathered data into four different sections: first, the companies' data, containing their IP range as well as the industry they are working in; then the data gathered using various API and datasets features, as well as the Shadow server reports (to which we dedicated a whole section since the reports are complex); finally, the incident dataset, containing a set of incidents that occurred on the companies' network, provided by the MSSP.

4.2. Methodology

After creating the scanner, the internal data (the alerts and incidents dataset) can be collected, alongside additional external data about companies' networks, using the following methodology:

- Collecting internal data and requesting the incident data from the MSSP.
- To replace the security features that couldn't be collected by the scanner, gathering data from Shadowserver, an external security dataset; analyzing the available reports to find the most useful ones.
- Making every data consistent (labeling on severity levels and incident type), ready to be used in the context of a machine learning algorithm.

4.3. Companies dataset

To perform the data analysis, we need the IP networks used by the companies. Indeed, we target a specific list of companies (from which we have detailed incident data), but our scanning tool targets hosts.

For anonymity reasons, the list of companies is based both on the MSSP clients and non-clients, with no way to distinguish between them. The companies' names were also anonimized in this report.

The client data is comprised of **60 companies** associated with their area of interest, from e-commerce platforms to banks and telecommunications companies. Since some other datasets have historical data (dating back to 2014), we also made a temporal analysis using historical WHOIS data. Indeed, the IP space evolves constantly, and the companies may have drastically changed their IP space. Therefore, considering only the actual network of a given company may lead to errors in further analysis. That's why we adopted the following methodology:

- For each date of the historical WHOIS data, search through the list of dutch companies (i.e. companies that have IP addresses listed as Dutch in the ”country” field of the database) for the exact name of the company;
- If no IP is found for a given company, search if it is not because of wrong name formatting (i.e. BV instead of B.V.). This step was enough to solve the majority of companies not found.

This methodology results in a list of networks associated with a company. There may be errors among the results:

- the IP blocks may contain sub-networks that are allocated to another company;
- there may be some misclassifications of IPs for a few days after a change in a company's IP space since the WHOIS data is studied at intervals of a few days;
- the company may have additional IPs that are not part of the blocks found. For example, the company may have additional web services hosted on a third-party server (e.g. cloud storage) or may use other third-party services to increase their online presence (Content Management Systems...). Even if those IP addresses are not technically part of the network owned by the company, they are still part of the company's security posture and any vulnerability on those hosts will influence the company's overall security.

These errors may lead to inaccurate security indicators: for example, if a company rent a network block to a client with a poor security posture, they might be several intrusions happening on that network block. In the end, the prediction model may return features that are relevant for the client rather than the company we are really interested in. However, the question of finding the exact IPs belonging to a given entity is a complicated study. The number of features collected on the IP blocks should be enough to counterbalance the limited number of erroneous IPs.

The temporal data as seen in figure 4.1 shows that a few companies changed their network a lot. Whereas there is companies who remained quite stable over time (with few changes in the size of their network), some others encountered drastic changes (see the blue line, which represents a company that multiplies the size of its network by a factor of 10 after 2018). This proves that taking into account the evolution of the companies' IP space is a necessary step.

This dataset allows us to map companies with their network. This is a necessary step because the hosts we consider are part of a company, and therefore are heavily influenced by the security posture of the company (rather than being independent machines); it makes little sense then to consider the security posture of a single host, especially since the incidents are often linked to a company rather than a single host or IP address.

4.3.1. The Spamhaus dataset

In addition to scanning features, researchers often consider the blacklist data [29]: the addresses added to blacklists are IP addresses that take part in malicious behavior (sending spam messages for example). When such malicious behavior originates from a company's network, it is considered as an indication for its security posture (even if this feature alone is not enough to estimate the security posture [27]). To find the blacklist information, we used the Spamhaus dataset [43]. This dataset consists of a daily list of IP addresses or domain names separated in the following files:

- **Spamhaus Block List** (SBL): a list of addresses that have been involved in email related malicious behavior (sending bulk spam messages for example);
- **Exploits Block List** (XBL): a list of addresses that are infected by third party exploits or malware;
- **Policy Block List** (PBL): a list of end-user addresses that should not send unauthenticated SMTP emails. Since this list does not show any malicious behavior, it was not used in my analysis.
- **Domain Block List** (DBL): a list of domains that have a poor reputation. Since I am working on IP addresses and not domains, this list was not used in my analysis.

This dataset allows us to consider additional information coming from external sources that a scanning tool alone could not have detected.

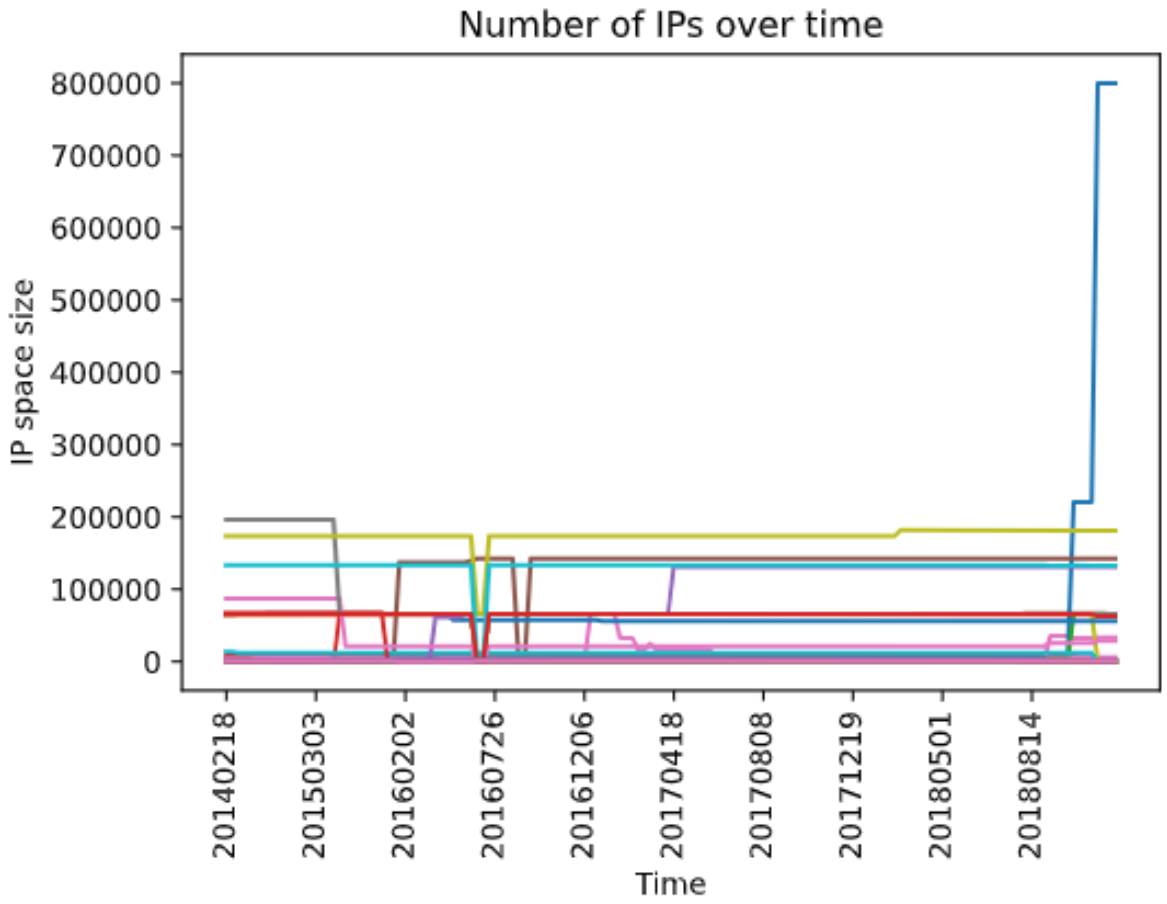


Figure 4.1: Evolution of the number of IPs for the considered companies over time

4.4. Shadowserver dataset

4.4.1. Presentation

In addition to the other APIs and datasets, the Shadowserver dataset [41] consists of a large number of reports, covering the time window between 2014 and today (depending on the report, some may cover a different time window). Each report is divided into several columns, containing a timestamp (time at which the scan was conducted), the IP address (IPv4 for most of the reports) of the host that falls under the description of the report, and several additional information that help to categorize the host.

The data are grouped by geographical regions: we only used the *netherlands-geo* part of the dataset since the companies we are interested in are Dutch.

The Shadowserver dataset covers a large number of security features such as the one we were looking for by creating the scanner (even if not all the scanner's features are represented in the Shadowserver reports: for example, the SMTP open relay misconfiguration is not present), with the addition of the historical data. Since the scanning couldn't take place without the approval of the companies, no data could be collected. The Shadowserver features may replace this data for our current research. In further works, the scanning tool can be used instead of third-party datasets such as the Shadowserver reports.

4.4.2. List of the reports

Some reports target a specific service for security reasons; for example, a service that provides no encryption or authentication and may leak data; or a service that may lead to an amplification attack, and list every IP address that publicly offers this service.

The following services may be abused and lead to amplification attacks: QOTD (Quote of the day), Netbios (to help application communicate on a local network), CharGen (used for debugging on TCP

networks), NTP (Network Time Protocol), SNMP...

Some other reports may be subject to misconfiguration and lead to abuse of the network, or concerns protocols that should not be accessible to the general public (even without configurations problems or vulnerabilities). It concerns protocols such as:

- **FTP**: because it is not encrypted and may leak information;
- **vulnerable SMTP**: contains a list of SMTP servers that are vulnerable to specific vulnerabilities;
- **vulnerable HTTP**: HTTP servers that only use vulnerable basic authentication;

Another type of report already talks about the breach since the **http vulnerable** report describes only the websites that researchers determined were breached. This is a result of the security posture so it must not be used in the same condition as the other reports.

4.4.3. Overview of reports

Some reports may not contain information that we deemed useful for our research. For example, some reports only use IPv6: since our research is based on IPv4 networks, the IPv6 information is useless to us. These reports were discarded and not used further in the research.

A first analysis of the reports shows, for each dataset, the number of IP addresses of a given company that is mentioned in each report depending on the day; it shows that some features are evolving in time (for example, more IP addresses are accessing the Microsoft sinkhole in 2014-2015 than in 2020-2021).

It is noteworthy to notice that the Shadowserver foundation allows the owners of network blocks to opt-out from the scans they do. This may increase the number of false negatives in the dataset since some companies may not appear in some reports because they chose not to (rather than this being the result of their security posture).

4.5. The MSSP dataset

The alert and incident dataset is divided into two files: the first is the alert dataset, i.e. all the alerts triggered by the rules implemented in the firewalls; the second is the dataset of incidents, i.e. a set of grouped alerts that an analyst inside the SOC deemed to be related and causing a breach.

In the literature, the incident data used for breach prediction is often public: for example, *Liu et al.* use [29] public incidents databases. The incidents are distinct, since public databases only contain successful attacks (unsuccessful ones are not reported), whereas the MSSP dataset contains more incidents, with successful hacks but also attempted incidents. It makes the MSSP dataset more detailed; the classification will then be more precise and the model may be able to predict different types of incidents and not only the successful attacks that are reported in public databases.

The incident dataset is the one we mostly used on our machine learning model: it is related to real incidents, and not automated alerts, therefore is a better indication of a breach. It contains the following columns:

- **incident_id**
- **submit_stamp**, in this format: ”2009-10-06 08:58:50+00”
- **alert_level**, which can take the following values: -2 (not interesting), -1 (false positive), 0 (undetermined), 1 (interesting), 2 (low-risk), 3 (high-risk), 4 (successful hack attack);
- **category**: describes the alert level in a string rather than an integer;
- **subject**: string that identifies the company the incident is related to;
- **open_stamp**
- **response_stamp**
- **done_stamp**
- **is_escalated**: whether the MSSP manually escalated the incident to the company;

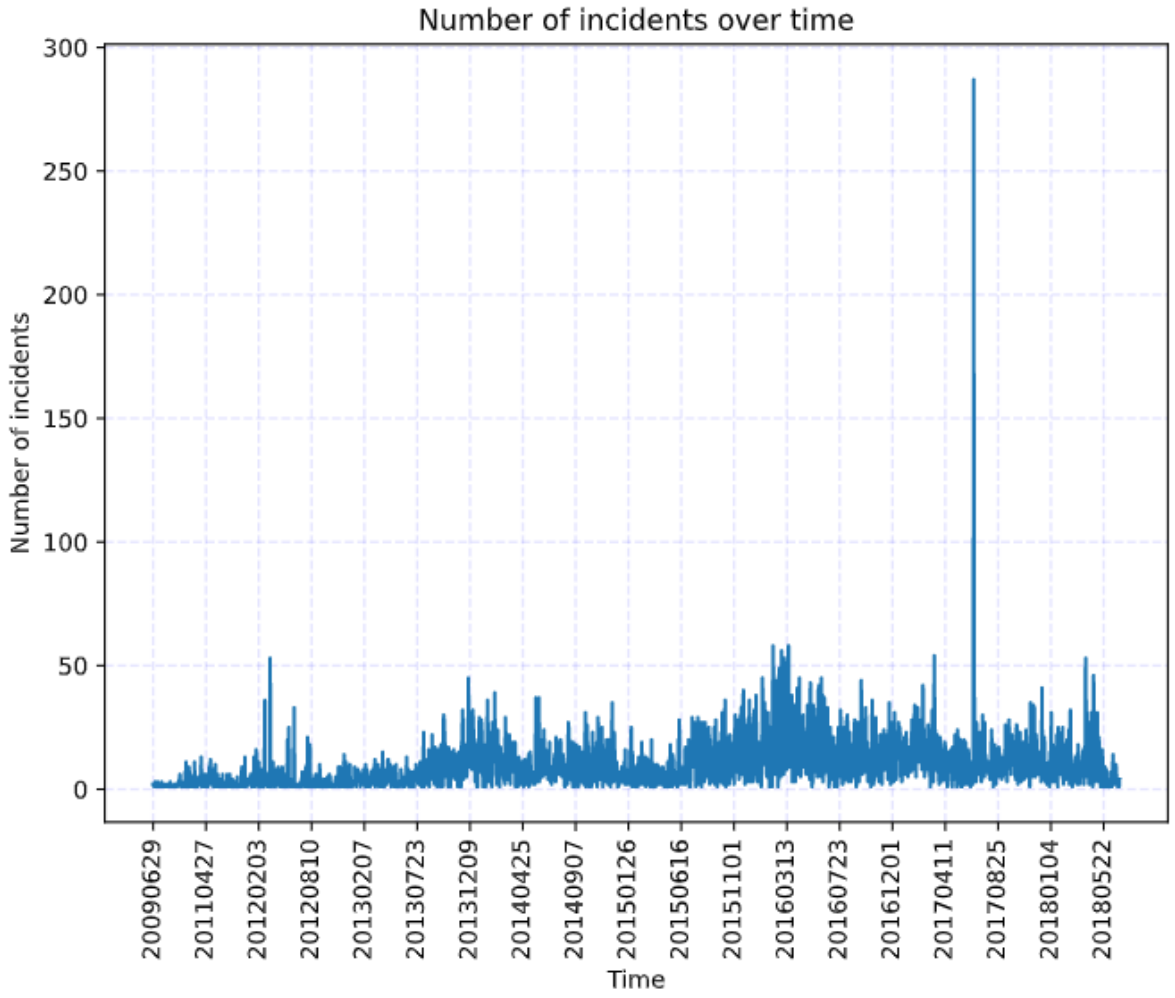


Figure 4.2: Number of incidents per day over time

- **description**: based on the description of the first alert that belongs to the incident;

As we can observe in figure 4.2, the MSSP dataset contains incidents from 2009 to 2018. Most of the time, the number of incidents stays below 50. However, this concerns all incidents (i.e. all samples with a severity score strictly superior to 0).

We can indeed notice in figure 4.3 that the proportion of incidents on a given day highly depends on the severity score: as expected, the largest numbers of incidents are always the incidents with a severity score of 1 (interesting but not risky). Indeed, on the 24 190 incidents considered, 19 697 have an alert level of 1. The incidents of other severity appear a lot less in the dataset: there are only 1938 incidents of severity level 2 (low-risk), and 2541 incidents of severity level 3 (high risk). The severity level 4 (successful attack) is the rarest, and only occurs 14 times in the dataset.

In figure 4.4, we notice that the companies considered have a different repartition of incidents over time. Most of them register less than 50 incidents per day, except for the peak (already seen in the other graphs) in 2017 (which is, as seen in figure 4.3, only composed of ”interesting” incidents i.e. incidents without risks) for a single company.

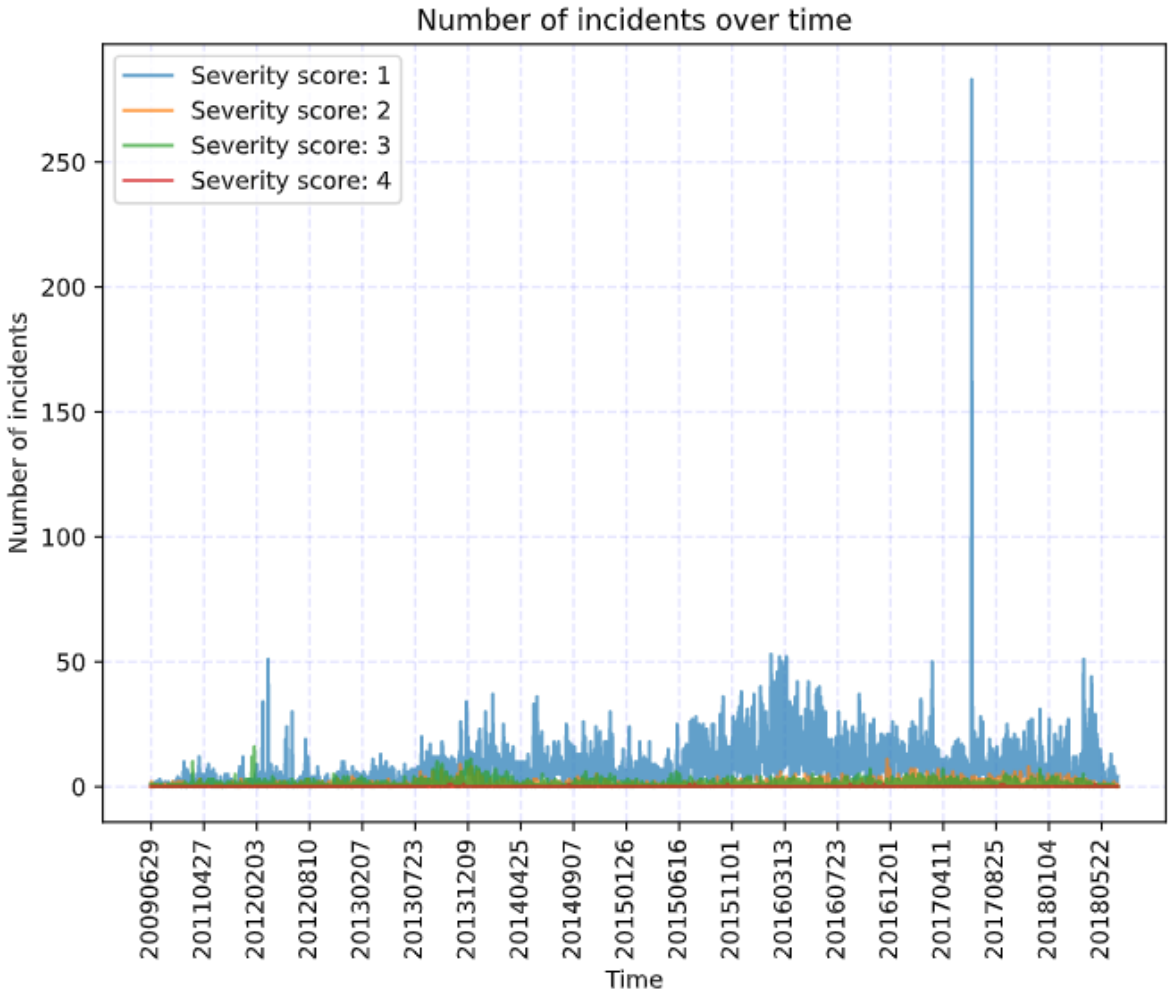


Figure 4.3: Number of incidents per day over time depending on the severity

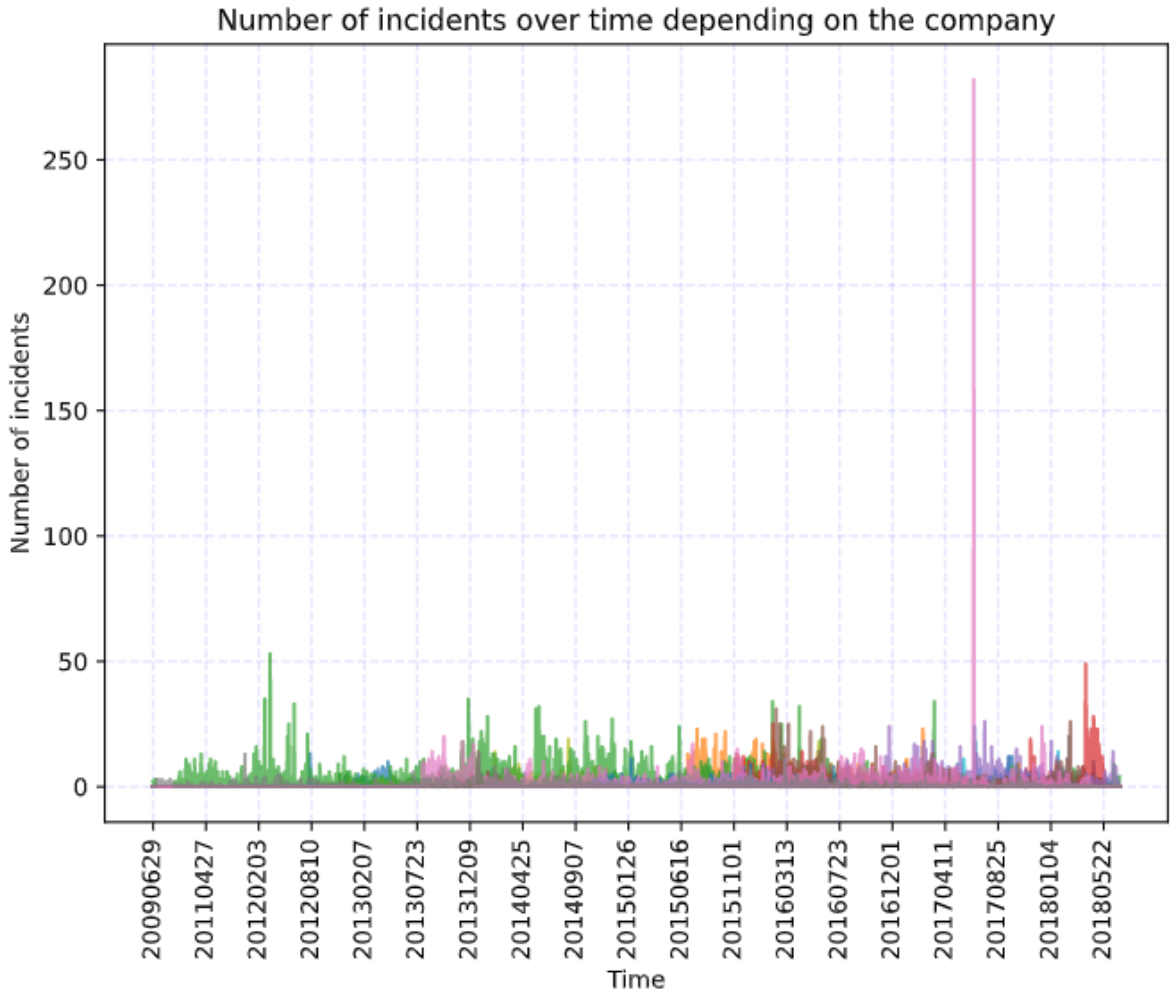


Figure 4.4: Number of incidents per day over time depending on the company

5

Machine learning model

5.1. Introduction

Once the data from the scanning or the datasets are collected, a first statistical observation shows that the features are not strongly correlated with the label (i.e. whether at least an incident occurred on that day or not), as seen in figure 5.1: when studying the correlation between the features and the labels, we observe that all the features have a correlation score with the labels below 0.22. This shows that a simple statistical analysis of the features pattern is not efficient enough to predict breaches and that a machine learning model will help us to explore the data.

Following the forecasting methodology described in the paper by *Liu et al.* [29] but using a different set of features, we train a Random Forest Classifier model to classify whether an incident happened on a given day.

5.2. Methodology

After having collected and set up the data, the next step is to build the prediction model based on the collected data. The methodology is the following:

- From the collected data, building the features that seem interesting as determined by the literature review. Some features are directly given by the external data or need a bit of processing (for example, the number of blacklisted IPs per day), and some others need to be build from the external data (for example, the average number of blacklisted IPs or the frequency in the variation of the number of blacklisted IPs).
- First performing a literature review of the possible models, with their advantages and drawbacks, as well as a review of the processing steps; choosing a model based on this review and the prediction needs.
- Preprocessing the data (to avoid unbalanced categories for instance).
- Building the classifier as we defined it in the previous steps to run it on the training data.
- Testing the classifier and studying the results: what is the accuracy score? What accuracy measurements are the best to represent the classifier?
- Tuning the parameters to obtain the best possible results and conclude: what are the important parameters and why? What are the possible trade-offs (e.g. between the false positive and false negative rates)? What do they mean in our study?
- In the final model, studying the relative influence of each feature (what are the features that produce relevant results and the features that could be dismissed to simplify the model without performance drop?) and draw conclusions on the relationship between the external features and the security posture.

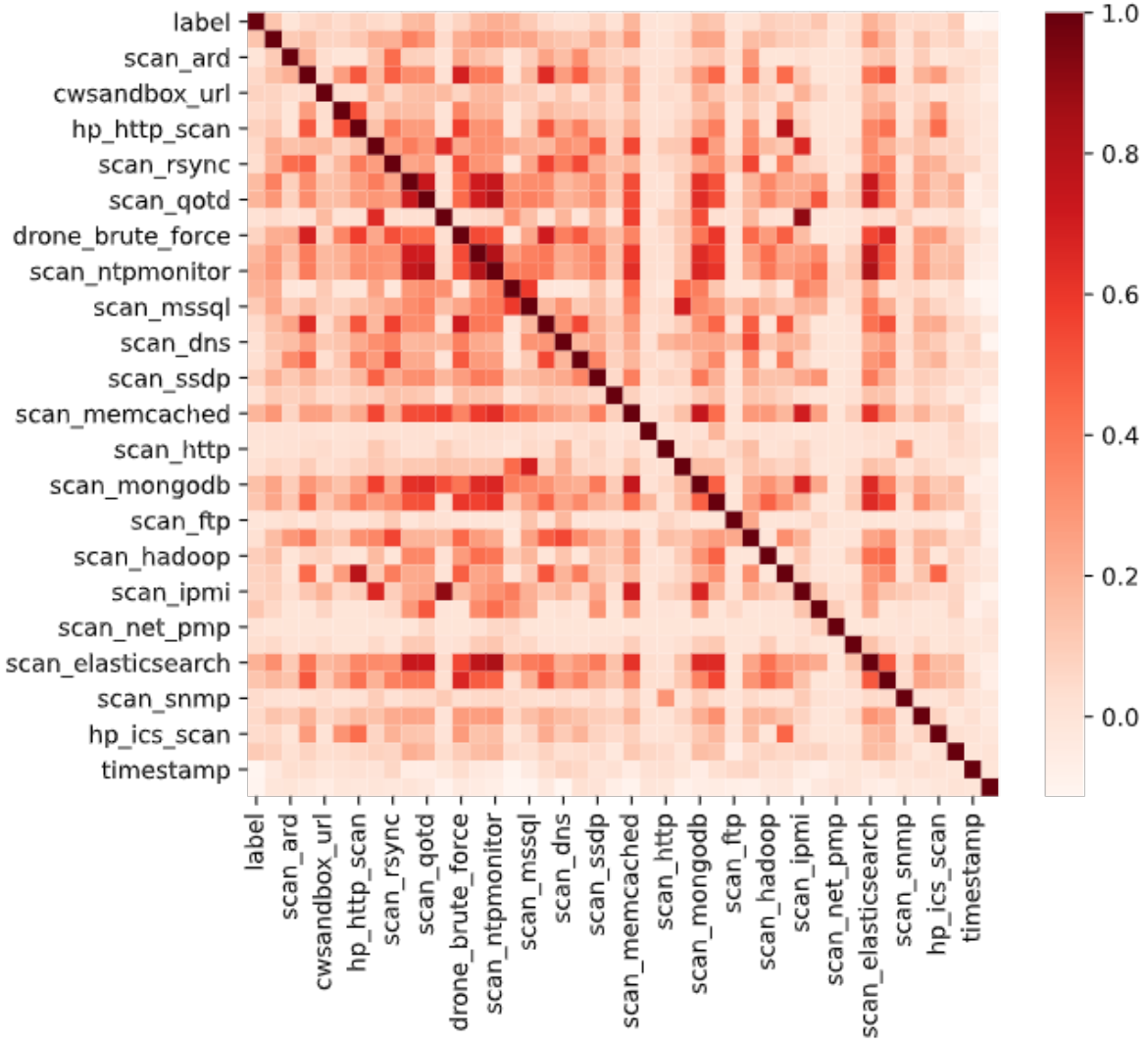


Figure 5.1: Correlation of each feature

5.3. Features set

To build a prediction model, we used the results from the scanning along with the external datasets. The scanning tool gives us the following features:

- The *scanned features (4)*: using the scanning tool, we can list the hosts that implement specific misconfigurations, and group them by companies to see how many hosts are misconfigured in the network of a specific company;

- The *blacklist data (1)*: the scanning tool also retrieves specific blacklist data using an external dataset. For a given company, we can then count the number of hosts that are mentioned in the blacklists on a given day.

- The *CVE data (1)*: using the scanning tool, we can also retrieve the CVE associated with a given host and group them by companies to keep an overall severity score over each company.

As soon as we obtain the authorization to scan the companies' networks, these features can be used in the machine learning model.

As we are still waiting for the scanning authorization, we used another set of features instead, using the additional datasets:

- The *Shadowserver features* (64): the security reports provide us data separated into 64 different categories (each category focused on a specific security misconfiguration or a specific protocol as described in section 4.4). For each category, we count the number of hosts from a given company inside the report for each day. It gives us 64 features to feed the machine learning model.

However, these features need a bit of preprocessing before being relevant for the machine learning model.

5.4. Preprocessing

Before feeding a machine learning model with the data, however, preprocessing steps are necessary: we need to convert every data into numbers, and make the features represent as much as possible the security indicators (and not any other indicator, such as the network size of the company, which is not an information related to security).

5.4.1. Encoding

Building the dataframe

To analyze the data and feed it to a machine learning model, we build a list of data samples based on the collected security features. Since temporal data is available, we grouped the data by day (depending on the company): for a given day, for a given company, we associated all together the different security features concerning that company for that day (i.e. the number of hosts belonging to the company associated with the security feature). The dataframe has the following organization, as described in table 5.1.

day	company ID	security feature 1	security feature 2
04/06/2018	1	0	0
04/06/2018	2	5	2
05/06/2018	1	0	3

Table 5.1: Organization of the dataframe with different security features

Taking size into account

One important thing to consider is the influence of the size of a company: the number of available hosts is not a security feature, but will directly affect the number of incidents on the network. To better represent the security features, we then choose to encode the size in the features themselves, by dividing the number of hosts mentioned in the Shadowserver dataset by the size of the company's network on the given day.

The industry type

Another feature that can be deemed relevant is the company's industry: each company is associated with an industry name (whether it is a bank, a hospital, a telecommunications company, etc.). Since the machine learning algorithm works only with numbers, we converted the strings to integers.

Various methods exist to convert categorical data to a format usable by a machine learning algorithm:

- **One-hot encoding**: for each sample, a new column is created for every possible category. For a sample belonging to a given category, we put 1 in the column of the given category and 0 for every other column. For example, if there were three possible categories: e-commerce, web-hosting provider, and university, a data sample from a company would have the following values, as described in table 5.2. However, this encoding method leads to a lot of additional columns, which may lead to overfitting and increase the time complexity of the model, as the number of features increases with the number of categories.
- **Ordinal encoding**: each possible category is associated with an integer, and this integer is used instead of the string. For example, if e-commerce is associated with the value 1, web-hosting provider is associated with the value 2, university with the value 3, the data sample belonging to an e-commerce company will have the following values (see table 5.3). This does not increase the

industry	e-commerce	web hosting provider	university
e-commerce	1	0	0
web hosting provider	0	1	0
university	0	0	1

Table 5.2: One hot encoding for a e-commerce company

number of features for the machine learning model, which is good to avoid overfitting; however, it creates a create a partial order between the industries, which is something that does not exist in reality and may cause bias in the machine learning model. This is the solution we chose in the model because the data samples already have a lot of features, and overfitting would be problematic using one-hot encoding; since the *industry feature* is not selected in the final model (because it is not an indication of the security posture), the addition of a partial order is not important.

industry	category
e-commerce	1
web hosting provider	2
university	3

Table 5.3: Ordinal encoding for a e-commerce company

Similar to the size, this feature will give indications on the number of incidents, especially for industry types such as web-hosting providers or internet service providers (since the IP blocks are still considered as part of the company, even if it is rented to clients, and the large size of the network will produce more incidents) and can help to predict future incidents with better accuracy. However, the industry is not a security indicator in itself and should not be used in the prediction, since the algorithm's goal is to model the security posture. Taking the industry into account would lead the model to consider the global incidents trends in the specific industry group considered rather than the security posture and would penalize outliers (e.g. the companies in a given industry group that have a far better -or worse- security posture than average).

Labeling the incidents

The goal of the model is to predict whether there was at least an incident on a given day depending on the features. It was encoded as follows:

- if an incident (of severity score strictly superior to 0, since the incident with a severity score equal to or below 0 are not incidents but false positives, see section 4.5) or more occurred on a given sample (i.e. a specific day for a specific company), this sample is labeled as 1 (meaning at least an incident happened).
- Otherwise, the sample is labeled as 0.

This binary labeling allows us to compute the accuracy as well as the precision and recall of the classifier, which is important to measure the effectiveness of the prediction.

5.4.2. Sliding windows

To take into account the previous states of the company to detect whether it is more prone to an attack or not, one can shift the previous values of a security feature to encode the state of the previous day as well as the day before. This helps the machine learning algorithm to detect patterns and not only single points in space. In their article, *Liu et al.* [29] used similar sliding windows, but only for the blacklist data since the authors didn't have access to the temporal data for the other security features.

On the contrary to time series (which are not designed for Random Forest models), sliding windows only describe the past state of the security indicators used as features (and not the occurrence or not of incidents in these past states).

For a given security feature, for a size of the sliding window of 2, the data would be transformed as described in table 5.4.

Sample ID	feature	label
1	1	0
2	0	0
3	3	1

(a) Initial data

Sample ID	feature	feature - 1	feature - 2	label
1	1	/	/	0
2	0	1	/	0
3	3	0	1	1

(b) Transformed data

Table 5.4: Data transformation when using a temporal sliding windows of size 2

The first n samples for a temporal sliding window of size n don't have values in some of their columns, and therefore must be discarded, because we don't know the values before the first sample.

That way, the previous values of each feature are encoded into a new column, meaning that the classification algorithm can interpret temporal patterns. However, for a multivariate time series analysis such as the one performed here, adding several columns per feature can rapidly lead to overfitting. That is why it is essential to limit the number of features to apply the sliding windows to. In our research, we selected the 4 most important features (as determined by a previous run of the classification model without the sliding windows) and applied sliding windows of various sizes to it to analyze how they influenced the results.

The results are described in section 6.1.2.

5.4.3. Unbalanced data

Since the data is unbalanced (the occurrences of incidents are far below the size of the data without incident), additional algorithms may help to have better accuracy of the prediction. We used the SMOTE algorithm [7] using the `imblearn` Python library.

The goal of the SMOTE algorithm is to create new training data in the minority class based on the already existing samples. The algorithm processes the data as follows, for every sample of the minority class:

- It finds the *k* nearest neighbors of the considered sample (with a default value of $k = 5$);
- Depending on the sampling strategy (i.e. the number of new samples we want to create) the algorithm then selects randomly some of those neighbors, and for each sample in the selection:
- It creates a synthetic sample between the original sample and its neighbor by taking the distance between the features, multiplying it by a random number between 0 and 1, and add it to the originated sample (therefore, it creates a new sample on the line between the original sample and one of its nearest neighbor).

According to the algorithm's authors, it performs best when combined with undersampling of the majority class: using both oversampling on the minority class and undersampling on the majority class, the training set becomes more balanced, which is helpful for the classification.

The results obtained using SMOTE on the incident classifiers are described in section 6.1.2.

5.5. Choosing a model

Many papers [29, 12] about breach prediction chose the Random Forest learning model: *Liu et al.* [29] focus on the Random Forest because it works well with large and diverse datasets, and was successfully used in previous internet-related studies.

A Random Forest classifier is a set of Decision trees: instead of using a single decision tree classifier, it uses an ensemble of it and combines the results of every tree by a majority vote. It helps to reduce overfitting.

The Random Forest classifier also makes it easy to determines the feature importance after training the model. This is essential for our research since the main goal is not to blindly predict the occurrence or not of an incident, but rather to link it to the security features, to do an analysis of the security posture depending on the incident predicted risks.

Other types of classifiers exist, such as the Support Vector Machine or the Multi-Layer Perceptron, and we tested their performance on the incident classification to compare them, using the same methodology for each classifier, as described in section 5.6. The results are visible in figure 5.2.

Overall, the Random Forest Classifier performs better than other classifiers used for the same purpose: as we can see in figure 5.2, for a given False Positive Rate (i.e. the proportion of genuine samples that are classified as incidents), the False Negative Rate of the Random Forest is far lower than the other models. Less False Negative means a better detection accuracy and fewer attacks missed. This is important: attacks are far less frequent and a detection method that returns everything as a non-attack will have a good accuracy score but would be disastrous for a company since any missed attack may have a huge impact. Overall, the Random Forest has an area under the ROC curve (AUC) equal to 1, which is better than the Multi-Layer Perceptron ($AUC = 0.96$) and SMV ($AUC = 0.81$) because it is closer to the perfect model. The Decision-Error trade-off curve confirms this: for a false positive rate of 1%, the random forest has a false negative rate of 1%, whereas the MLP is over 5% and the SVM is far over 50%.

That's why we chose the Random Forest model to do our classification.

5.6. Training and testing methodology

The first step is to train the model on a subset of the data. To do so, we split the data into two sets, with a training-testing ratio of 67%-33% (33% of the initial data is used to test the model after training).

We first test the model using one Shadowserver report, the compromised websites report, to label the data; then using the MSSP incident data. Using the MSSP incident data, we tested different parameters to find whether they are useful or not, using the following methodology:

- First, training the model for every company and testing it and evaluating it on the testing set;
- Then, evaluating the influence of the SMOTE algorithm by training the model using different SMOTE parameters and evaluating them;
- Considering and analyzing the relevant features by training the model and ranking the features by their influence on the classification;
- Then, training and evaluating the model on every company separately to analyze the differences;
- Then, considering larger time windows (i.e. instead on grouping the features per day, grouping it per month or year) and evaluating the differences;
- Adding features using the sliding windows algorithm to understand how the temporal patterns affect the classification;
- Finally, training separately the model on the different incident categories to take the incident severity into account in the classification.

5.7. Testing with the compromised website data

5.7.1. Introduction

Since the incident data given by the MSSP is confidential, we worked first on building the model using a public incident dataset: the report about compromised websites from the Shadowserver archives. The data is scarce so the model is not perfect but allows us to find interesting results.

5.7.2. The incident data

The incident data is composed of a list of websites that have been identified as compromised, through an insecure version of applications running on it or weak credentials. The compromised websites are then used for malicious activities such as sending spam messages or participating in DDoS attacks; we are however not interested in the resulting behavior of the hosts but rather the fact that they are compromised.

In the figure 5.3, we have an overview of the proportion of compromised websites over time. The total number figure makes clear that the overall number of compromised websites declined over time, going from more than 40 in 2014 to less than 10 in 2019. It could be because security is taken more seriously nowadays (with both a more active security community and retribution if the security of a website is problematic: for example, legal actions can be taken). However, the data from the report is not safe from false negatives, i.e. the researchers may have missed multiple compromised websites

that do not appear in the dataset even if an attacker was in control of the host. The decline over time may then results from the attackers being more stealthy, or the security breaches being more challenging to verify, rather than the websites being more secure.

We also notice that the two most prominent companies that appear in the graph are a web hosting provider and an internet service provider. It means that both of these companies provide clients with a lot of websites controlled by the client. The fact that a website is compromised is then more an indication of the client's security posture than the company's security posture, and it is out of our scope.

The figure 5.3 also shows the limit of the dataset: of all the 60 companies that are part of our research, only 10 have had, at some point in time, a host that appears in the compromised website dataset. It means that the data gathered for all the other 50 companies will be useless to classify future breaches. This shows the necessity to also train the classifier on a more precise incident dataset. Overall, the only type of incidents covered in this dataset is whether the website was compromised or not, with no indication of severity. The dataset is then missing all the other types of incidents, and the results may be biased (depending on the industry, a web hosting provider is far more likely to have a compromised website incident on the network blocks it owns than a company that only has a single website hosted on a third-party provider).

5.7.3. Prediction's Accuracy

Considering all the companies at once

After training the model, we tested it on a separate training set, which size is 1/3 of the total data. When all the companies are considered at once, the classifier obtains accuracy scores to measure its performance.

First, the measure of the number of false positives and false negatives helps to see if the model is working properly. In the table 5.5, we see that, first, the data is unbalanced because the number of true positives is almost 50 times less than the number of true negatives. It emphasizes the importance of algorithms such as SMOTE and undersampling to correct the classification training, otherwise, the results may be really poor in terms of detection. However, the classification obtains a high number of false positives detection (i.e. samples classified as incidents when they were benign), almost 1/2 of the number of true positives which may cause a lot of false alerts. Some incidents are also missed, but the number of false negatives is quite low. Depending on the companies needs, some parameters may be modified to change the proportion of false positives and false negatives: to have a more accurate detection, one may use a classifier with more false positives; on the contrary, if the false alerts are damaging (because the workforce is mobilized in vain), one may choose a higher detection threshold, but would be at risk of missing more incidents.

	Predicted Negatives	Predicted Positives
Real Negatives	44040	479
Real Positives	155	956

Table 5.5: Confusion matrix

More precise score measurements give a better overview of the performance of the model. In the table 5.6, we observe that the accuracy is high (around 0.98). This means that overall, the classifier has a good prediction. However, the accuracy does not detail the percentage of missed incidents or false alerts and may be a poor indicator of the prediction quality, especially in the context of an unbalanced dataset such as the one we are using. That's why we considered the precision and recall scores, which are a better indicator of the usefulness of the prediction in the context of cyber-security incidents. First, the **precision** is low (around 0.66), which means that a lot of incidents are investigated in vain, when they are not, in fact, incidents. However, the **recall** (the proportion of detected incidents over the total number of incidents) is quite high (around 0.86), meaning that the number of incidents missed is low. This is important in the case of security incidents, where one successful incident may cause huge damages to the company, both to the finance and the image of the company. Finally, the F1 score is a combination of precision and recall: it is the harmonic mean of the two metrics. It is here high (around 0.75) which means that the higher recall compensated for a lower precision. Depending on the ultimate goal of the company, some parameters of the model may be modified to find a better trade-off between

the recall and the precision of the classifier.

Finally, the Out-of-bag score provides an additional way of validating the model, but faster than the cross-validation technique since it doesn't require retraining the model each time. Here, the OOB score, similar to the accuracy, is high (around 0.95).

Accuracy	0.9861
Precision	0.6662
Recall	0.8605
F1 score	0.7510
OOB score	0.9475

Table 5.6: Scores of the model

An additional way of evaluating the model would be to examine the results' curves. As we can see in figure 5.4, the following curves can be analysed:

- First, the **ROC curve** has quite a large area under the curve (0.95), near the score of a perfect classifier (1) and far from the one of a random decision (0.5). However, this curve is not the most interesting to look at since the interesting part of the curve is only located in the top left corner;

- The **precision-recall curve** is, according to Takaya Saito and Marc Rehmsmeier [36] more interesting to study than the ROC curve, especially in the case of unbalanced datasets, because a ROC curve is heavily influenced by the huge number of data without incidents. Since the curve is closer to the top right corner than to the straight line at 0.5, it means that the results returned by the classifier are quite good, which is summarized by the Averaged Precision (AP) equal to 0.78. It is not a perfect classifier (AP = 1) but it is closer to 1 than it is to 0.5 (which would be the AP of a classifier that predicts that all samples belong to the same class).

- Finally, the **Detection-Error Trade-off curve** represents the number of false alarms depending on the number of errors (incidents missed). This curve clearly shows that in order to have the number of missed incidents below a given acceptable threshold, one must be ready to accept a given proportion of false alarms; the fewer incidents are missed, the more some genuine samples are classified as incidents. It is up to a given company, depending on its own IT resources and security strategy to choose the classification threshold.

Considering the companies separately

Considering all the companies at once gives a first idea of the classification score of the model. However, the results returned are generic, while the goal of the project is to link the security posture of a specific company to the breach probability. By training the model on a single company only, the results returned are quite different (the important features are not the same, and not in the same order) and more specific to the company itself. Because of the lack of incident data, only 6 companies can be trained over the incident data. Here are the result after testing the model, as we can see in figure 5.5.

We notice first that the accuracy and OOB scores are quite high among the companies. However, a closer look at the precision and recall scores shows discrepancies among the companies: whereas some perform well (e.g. number 5), some others have either a poor recall (company 1 with a recall of 0.4727) or poor precision (e.g. company 4 with a precision of 0.3157). This can be explained by the discrepancies in the incident data: indeed, most of the incidents are found in company 5 (which then performs well overall) whereas companies such as 4 have very little incident data, which makes the model more difficult to train accurately. With a more precise incident dataset, the performance may be higher.

On average, the accuracy and OOB score of the model trained over single companies are a little below the accuracy of the model trained over all the data at once. Overall, the precision is a bit better: false positives are less likely to occur when the training set is restricted to a given company's data, but the recall is lower: the data from other companies help to find future incidents.

Mean accuracy	0.9292
Mean OOB	0.9326
Mean precision	0.7446
Mean recall	0.7826
Mean F1 score	0.7280

Table 5.7: Average scores over the companies

5.7.4. The relevant features

The Shadowserver incident dataset is not a private dataset and its data will be different from the incidents gathered by the MSSP. It is still interesting to see how the model performs on such a dataset. By choosing a random forest classifier, we have access to the influence of each feature, which gives us the following result (see figure 5.6).

This classification is not a simple measure of correlation. As we can see in figure 5.1, the most important features are not the ones that have the best correlation with the label data (since the features strongly correlated with the labels were the *Elastic Search scan*, or *NTP monitor scan*, which do not appear in the top 5 most important features).

When we first compute the features' importance considering all features and not only security ones, we notice the following:

- The timestamp is the most important feature, which is coherent with the incident dataset since the number of compromised websites evolved over time (and in a similar way for every company); this means that the security posture (or the apparent security posture given that the incident data may be incomplete) changed over time.
- The size is a relevant feature as well: logically, a bigger company will have more breaches. However, since the size is not a security feature in itself, we later did the classification without this feature.
- The industry type is also important for the classification; however, this may be caused by the unbalanced dataset and the fact that most incidents are found in companies such as a web hosting provider or an internet service provider. Since these companies rent a large part of their network as their main activity, this means that they also have more incidents.

These features however are not part of the security posture of a company; they are relevant for a breach prediction algorithm but not for a security posture analysis, and from now on we discarded them before doing training and testing the model. The goal of our research is indeed to relate the future breaches to the security posture and not other external parameters.

However, the default implementation of the features importance computation in scikit-learn is biased: researchers showed this in [33], by adding to a dataset a randomly generated feature (which therefore should not have any influence on the classification). The sklearn feature importance described that the random feature was more important than other features, which logically seems wrong, so the author came with their own feature classification implementation. Different ways of doing so may be used: first, one can evaluate the model, drop a single feature and evaluate the model again; the score of the feature is then the score difference between the classification with and without the feature. However, this computation takes a lot of computation time (since the model needs to be trained again for each dataset with a single feature missing), so another version may be implemented: the Random Forest Permutation Importance. The model is trained and tested against the dataset; for each feature, the values are randomly permuted, and the new dataset is again tested in the random forest. The difference in the score makes the feature score.

Ordered by importance, the most interesting features are the following: the SNMP scan, the DNS open resolver scan, the Memcached scan, the IPMI scan, and the open portmapper scan (see table 5.8). As we can see in the table, the two implementation presents significant differences in the feature ordering: the most important feature (scan SNMP) remains the same, but the permutation importance implementation consider the DNS open resolver as the second most important feature (while it is only the 5th in the sklearn default implementation). However, the top 6 features in both cases are the same (even if their relative importance and their rank differ). Let's consider these features.

Feature	Importance
scan_snmp	0.2473
scan_memcached	0.1149
scan_ipmi	0.1137
scan_portmapper	0.09408
dns_openresolver	0.0808
scan_mssql	0.0646
scan_nat_pmp	0.0558
scan_ntp	0.0555
scan_ntpmonitor	0.0498
scan_redis	0.0204

(a) Using sklearn default implementation

Feature	Importance
scan_snmp	0.0349
dns_openresolver	0.0189
scan_memcached	0.0182
scan_ipmi	0.0178
scan_mssql	0.0106
scan_portmapper	0.0078
cisco_smart_install	0.0074
scan_ntpmonitor	0.0034
scan_ntp	0.0023
scan_netbios	0.0009

(b) Using permutation importance

Table 5.8: Top 10 most important features

- The **Open SNMP** report: the report collected the information about the hosts running the Simple Network Management Protocol on port 161 (UDP) with an outdated software version as well as default credentials that could be abused in an amplification attack. We see in figure A.1 that the evolution of this protocol is not the same as the compromised website report (there is no overall downward trend). However, the vast majority of the hosts are found in the same company, which may reduce the accuracy of the result on a dataset with more companies.
- The **DNS open-resolver** report: this report collects the DNS server that can act as an open resolver, as I did in section 3.6.1. We observe in figure A.2 that a huge majority of the open resolvers are found on the same company network, which may be the result of its clients' activity rather than its security posture since the considered company is an Internet Service Provider.
- The **open memcached** report: the Memcached store [15] is a distributed memory caching system, which allows to speed up web applications by better allocating the system's memory. However, this application provides no authentication, therefore any open Memcached store is prone to attacks. Again, we notice in figure A.3 that few companies are represented in this dataset.
- The **open IPMI report**: the Intelligent Platform Management Interface (IPMI) is an interface used to monitor networking information systems. This protocol however suffers from security issues (for example, the passwords must be stored unencrypted, which is a bad security practice since anyone having physical access to the server may find the passwords), which means it shouldn't be openly available to everyone. As noticed in figure A.4, the proportion of open IPMI servers is not significantly reduced in time but only a few companies appear in it.
- The **Open MS-SQL Server Resolution Service** report: the server resolution service is a protocol to ask a Microsoft-SQL server (a database management system) information about itself. It should not be openly available since an attacker can abuse it for amplification attacks. As we can see in figure A.5, the number of affected hosts reduced over time on average, but the ISP remains the company with the largest proportion of IP addresses found.
- The **Open Portmapper** report: the portmapper service was discovered to be used in denial-of-service attacks. Since the service can be part of an amplification attack, it, therefore, should not be openly available. Figure A.6 visualizes the evolution of the number of portmapper services found: in some companies, there is a downward trend; however other companies seem to keep a stable number.

As we can see, some of those features are similar to the one collected by the scanner (the DNS open resolver). Some others are not, such as the Open SNMP report. It would be interesting to aggregate the features found by the scanner (e.g. the Open mail relay) to the Shadowserver features and see which performs best; however, it would be complicated since the reports provide historical data that the scanner can't obtain.

However, most of the results from the Shadowserver data are found in the same group of companies, and some companies are represented only in a few reports. This means that the Shadowserver data

will be sufficient to predict incidents in the few companies that are represented; for the other companies, the data may not be representative enough to make a precise enough classification.

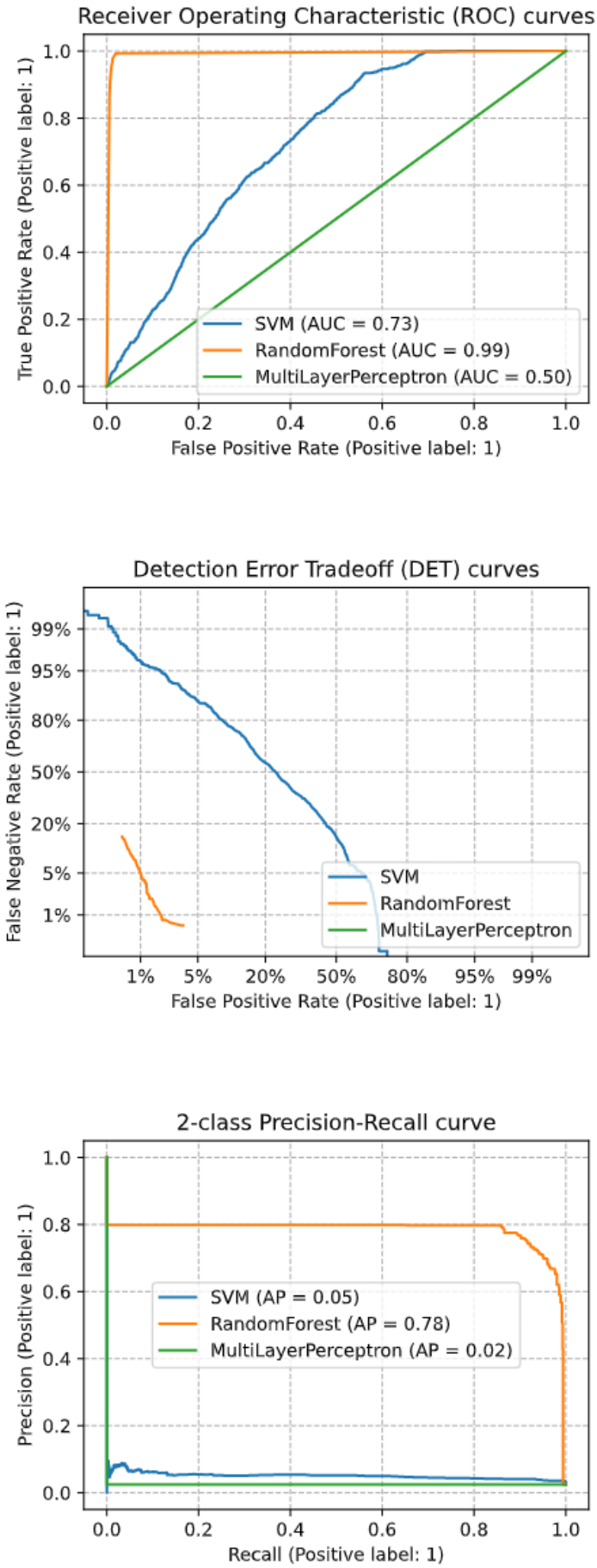


Figure 5.2: DET and ROC curves for multiple models

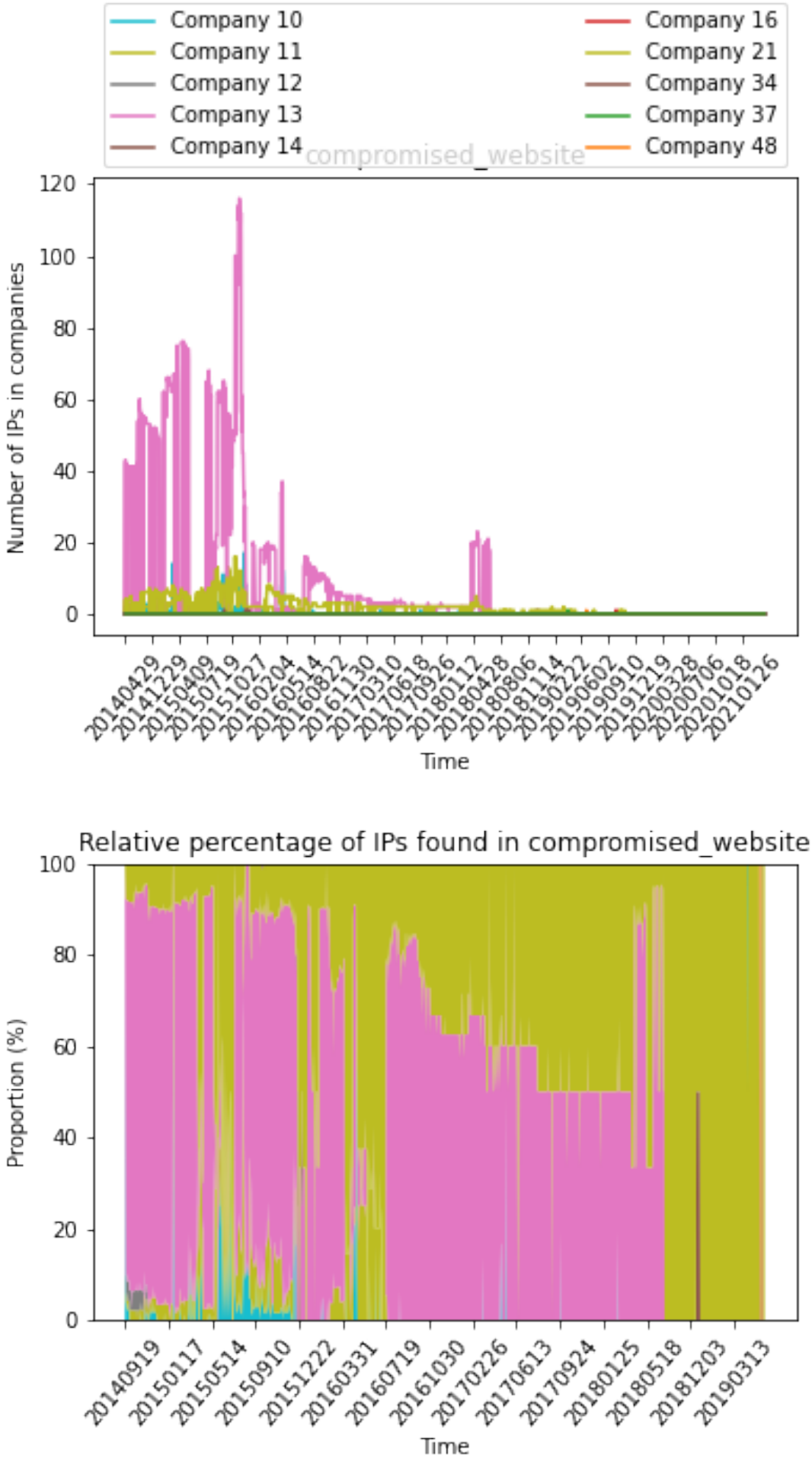


Figure 5.3: Numbers and proportions of compromised websites in the report

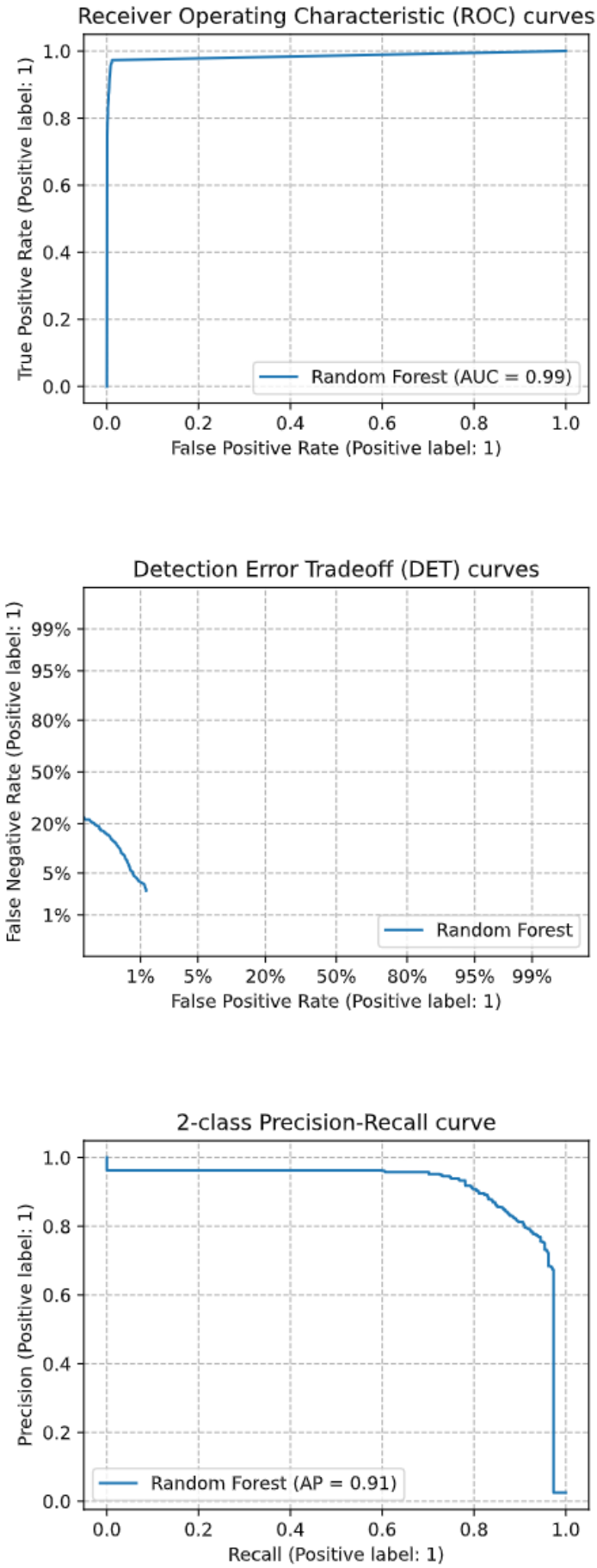
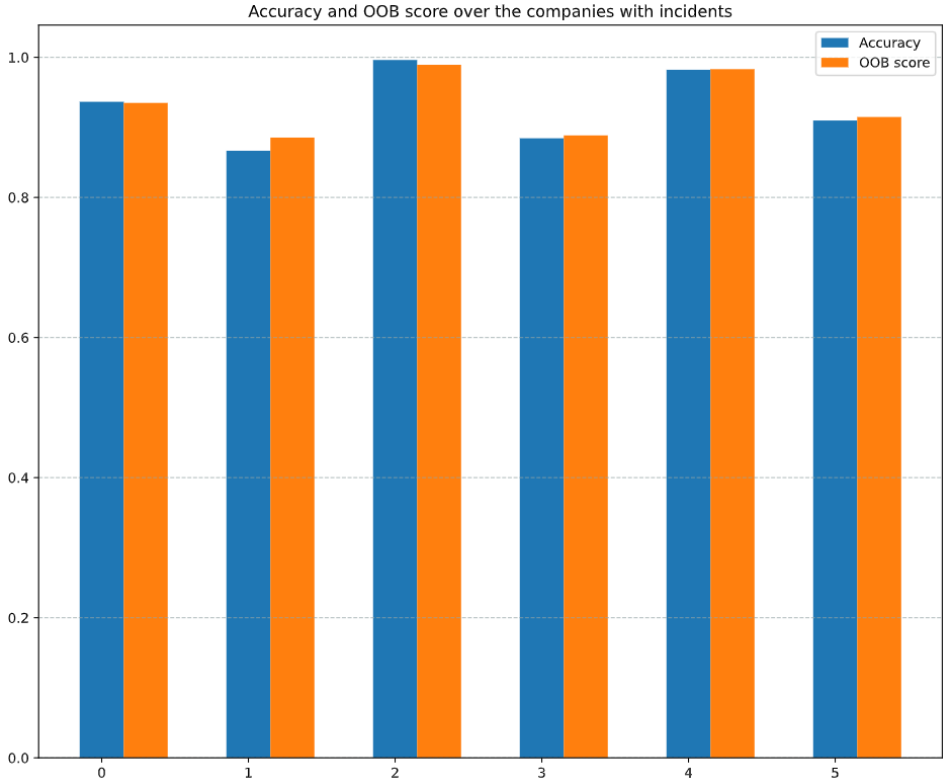
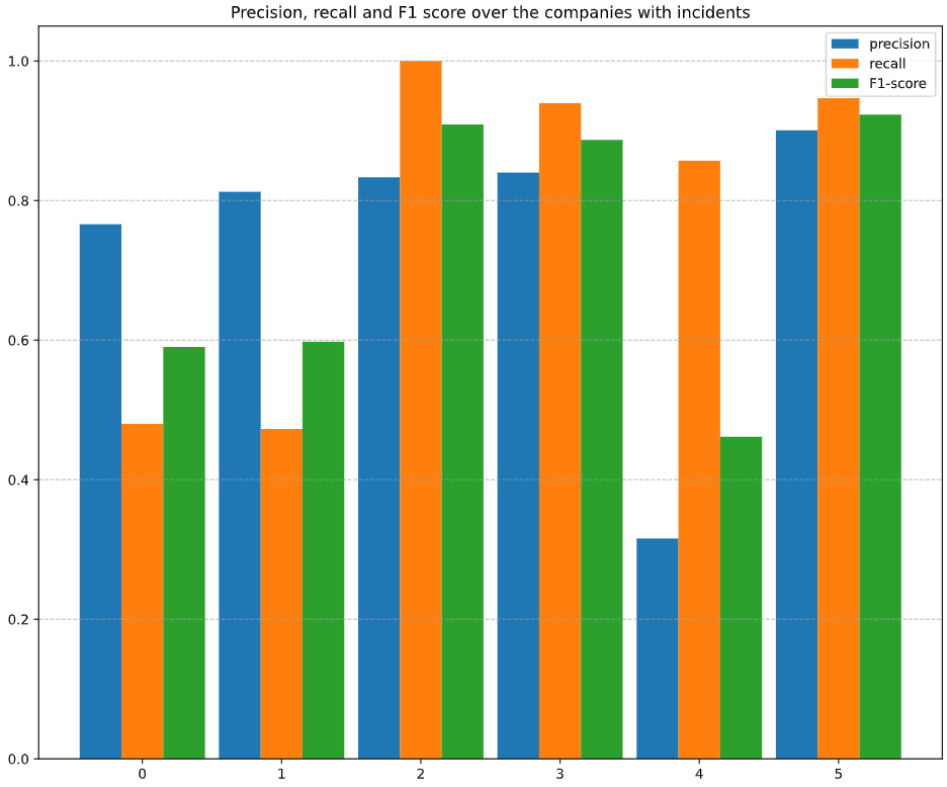


Figure 5.4: Evaluation curves of the Random Forest classifier



(a) Accuracy and OOB score



(b) Precision, recall and F1 score

Figure 5.5: Scores over the single companies data

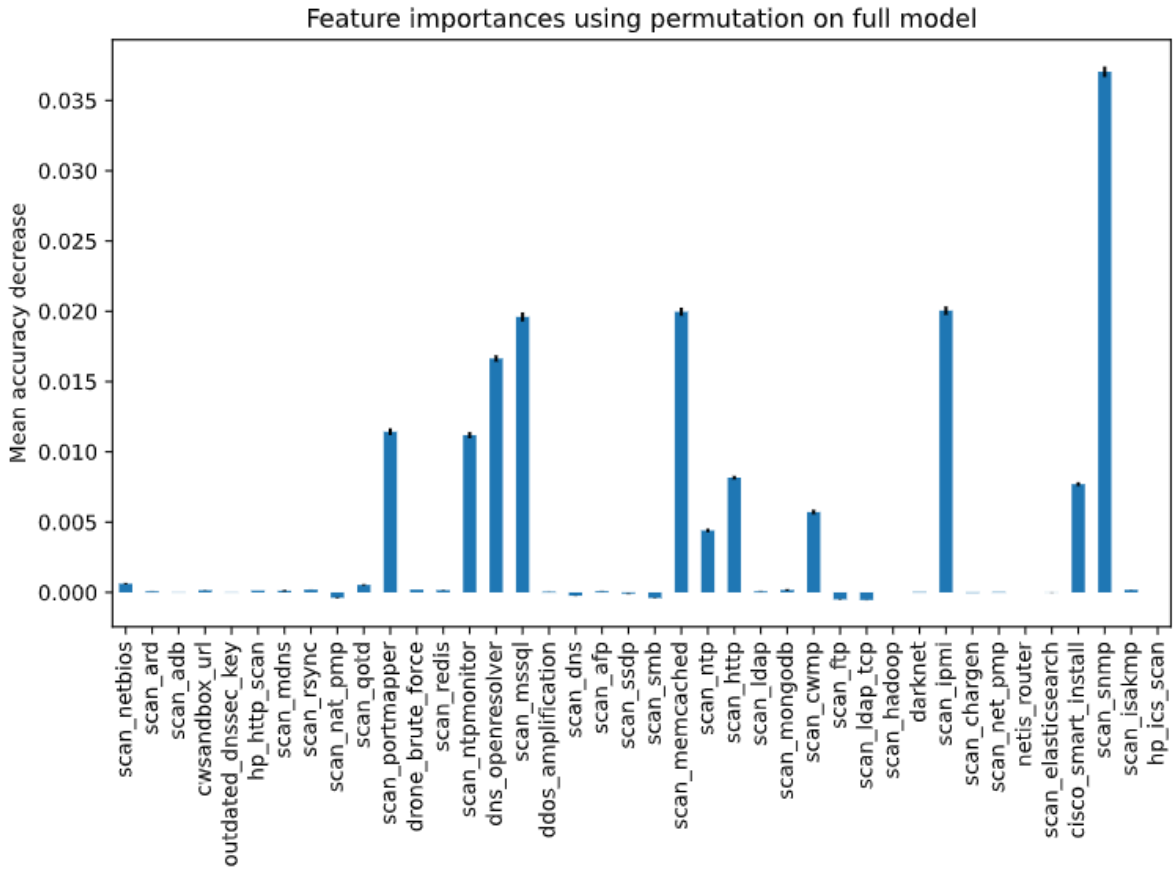


Figure 5.6: Importance of each feature in the dataset

6

Results and discussions

6.1. Results

The Random Forest algorithm is a classification algorithm that allows us to compute the importance of every feature in the model. Each feature is given a rank depending on its influence on future incidents. A highly ranked feature has a stronger link to a future incident. The company may consider it as more important to investigate than other features and invest in the related security controls in priority.

In this chapter, the results are computed using the incident data given by the MSSP: instead of using Shadowserver's incident data (that were only covering compromised websites, and no other types of incidents), the incidents are coming from an external dataset as described in the section 4.5. The incidents are linked to the other features by comparing the incidents' timestamp with the dates of the security features, as well as the company they belong to.

The use of private incident data differs from most of the papers we studied on intrusion prediction, where authors relied on public databases. The results will then differ since the private incident data is closer to reality than public data.

6.1.1. Scores

To evaluate the model, we can use different scores:

- The **accuracy**: the accuracy is simply a measure of the number of correct predictions over the size of the testing set. The accuracy is high in our model. However, the testing set is unbalanced (the occurrence of incidents is often far less likely than the non-occurrence of an incident), which means that a model that classifies everything as non-incident will have a high accuracy score. Since missing an incident may be disastrous for the company, accuracy is not the most important score we can look at.
- The **precision**: the precision score measures the number of detected and real incidents (true positives elements) over the number of detected incidents. A high precision score means that there are very few false positives in the prediction and that few incidents were investigated in vain. The precision is essential because predicting an incident that does not exist will waste the workforce on a non-existing threat.
- The **recall**: combined with the precision score, the recall score gives the number of detected and real incidents (true positive elements) over the number of real incidents. A high recall means that very few incidents were missed by the detection tool: it is then critical to consider the recall in a cyber-security setting where an approximate prediction gives very little information.
- The **F1 score**: the F1 score is the harmonic mean of precision and recall, to combine both in a single measurement.
- The **Out-of-Bag (OOB) score**: the OOB score is specific to the Random Forest classifier and is effective to validate it. Each tree of the forest is trained with a subset of the training data; the remaining data can then go through the tree, and the valid prediction from these "out of bag" samples is added to the OOB score.

6.1.2. Prediction of breach occurrence

First, we tested the machine learning algorithm using the incident data given by the MSSP as described in section 4.5. Overall, this data is more precise than the Shadowserver incident data (it contains more incidents, of various severity attribute and type); but it is not linked to the shadowserver data and the two are more difficult to combine than it was to work with only the shadowserver data.

Scores on all the companies

The machine learning algorithm is first tested without using SMOTE against all the companies at once. The following scores are returned, as described in table 6.1.

Accuracy	0.9150
OOB	0.9164
Precision	0.7258
Recall	0.0237
F1 score	0.0459

Table 6.1: Average scores over the companies

First, the accuracy and the OOB score remain high (0.92), but they are inferior to the one obtained with the MSSP incidents dataset (0.99). The precision also becomes higher (0.73) than the previous values (0.67) found in table 5.6; however, the recall score has dropped to less than 0.1, which is an extremely low value. Indeed, the classification has a huge number of false negatives (incidents missed) which makes it useless; even if the few incidents found are really precise (the precision value is high), there are too many incidents missed to be useful as an incident prediction model. The F1 score is low as well because it is computed as the harmonic means of precision and recall and is extremely sensitive to low values.

Such a discrepancy with the values obtained with the Shadowserver incidents may be explained by the scarcity of the data: the Shadowserver data is concentrated on a few companies (such as web-hosting providers with large network blocks rented to clients that are responsible for the security posture of their server), which are not necessarily the same as the MSSP client companies. This leads to companies never being mentioned in most of the Shadowserver features: either because the initial scan was not precise enough, or because the companies security posture has already corrected the necessary misconfigurations. In the first case, the scanner described in chapter 3 will help solve the problem by doing a real-time scanning over the interesting companies; in the second case, additional data needs to be gathered; for example by considering the data from the internal network as well as the external data, to have a more precise representation of the security posture and therefore a more precise prediction.

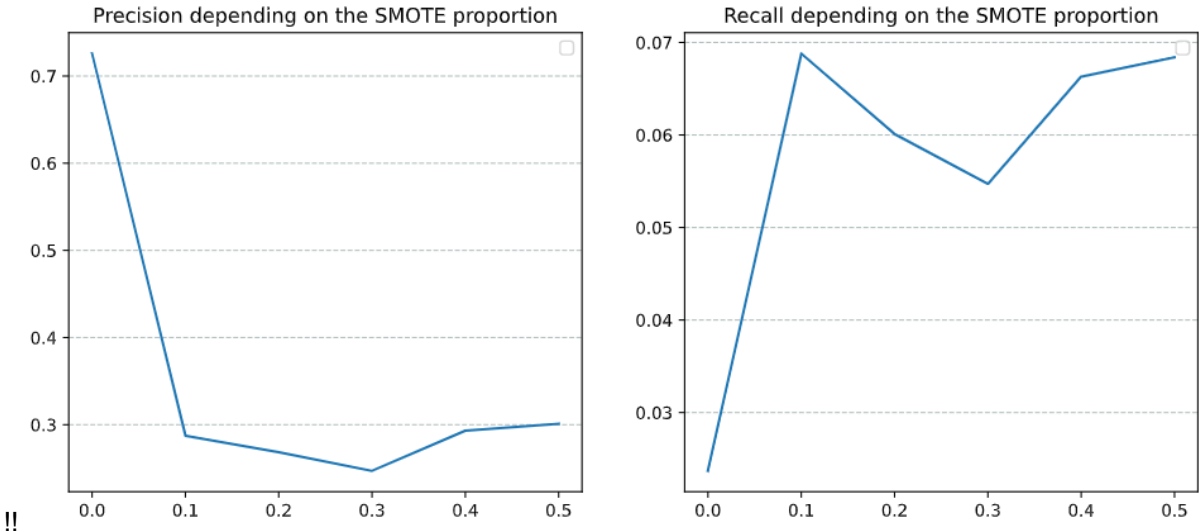
Influence of SMOTE

To increase the recall of the algorithm, we can test the influence of the preprocessing steps, especially the SMOTE of the minority class (the class with incidents) and the undersampling of the majority class.

As we notice in figure 6.1, increasing the SMOTE oversampling (adding samples until the proportion of the minority class samples reaches the defined sampling strategy proportion) produces an increase in the recall score and a decrease in the precision score. However, the increase in the recall score is poor (the proportion of false negatives samples in the prediction remains high), while the precision decreases a lot. Depending on the objectives of the prediction, the SMOTE proportion may be increased, but this has drawbacks on the precision of the model.

Relevant features

As we did in section 5.7.4, we compute the relevant features for this model as well. We notice that the features are not similar to the previously found features; however, this result may be biased by the lack of data: since there are very few Shadowserver reports containing data for the MSSP client companies, the pool of features to choose from is reduced a lot.



!!

Figure 6.1: Evolution of the precision and recall with the SMOTE sampling strategy

feature	importance
scan_ntp	0.013376
dns_openresolver	0.005852
scan_http	0.001839
cwsandbox_url	0.000334
scan_qotd	0.000334
scan_isakmp	0.000167
scan_chargen	0.000167

Table 6.2: Influential features using permutation importance

Mean accuracy	0.8356
Mean OOB	0.8087
Mean precision	0.3997
Mean recall	0.5734
Mean F1 score	0.3625

Table 6.3: Average scores over the single companies

Results on individual companies

Because of the limited data, there are few companies in which the model obtains a classification (can label incidents data from non-incident data).

As we notice in table 6.3, the classifier performs slightly better on selected single companies, when the data is sufficient. The mean recall is greatly improved. However, we notice discrepancies between the companies: with the same parameters, some obtain a high recall but a low accuracy, and other the contrary (high accuracy but lower recall). It shows that every company is quite different, and to obtain a precise result for a concrete company, considering only the data from the company is better. However, this requires a lot of data that is not also available, and the results will heavily depend on the company's past data. It may be problematic for a company that recently changed its network's configuration because past data won't be accurate anymore.

Using larger time windows

We obtained all the previous results by considering the security features over a single day (and predicting whether at least an incident happened on a given day or not). However, the incidents are not the direct results of a misconfiguration's appearance. There might be a large time window between the

moment the misconfiguration first appears, and the moment a related incident happens. That's why it is also interesting to consider larger time windows, such as the number of hosts that have a given misconfiguration over a month or a year, rather than a single day.

First, we consider the average number of hosts that have a given misconfiguration over a month for a given company (divided by the company's network size), as well as the variance, and classify them into two categories, depending on whether at least one incident happened on that company on that month. The results are described in table 6.4. We notice that, even if the accuracy (0.7419) is slightly lower than the previous scores in table 6.1 (0.9150), the precision and the recall are higher (with a precision of 0.8333 > 0.7258 and a recall of 0.0966 > 0.0237). However, the recall remains low, which leads to a lot of missed incidents.

Accuracy	0.7419
OOB	0.7547
Precision	0.8333
Recall	0.0966
F1 score	0.1732

Table 6.4: Average scores over the companies with a monthly time window

By considering the same features, but over a time window of a year (instead of a month), the results are better, as described in table 6.5. Indeed, if the accuracy (0.6769) is lower, the precision (0.875) and especially the recall (0.2593) scores perform better on such a time window. It is indeed simpler for the classifier to decide whether an incident happens on a large time window since every data sample is associated with security data from a whole year (instead of just a month or a single day). However, the model is overall less precise since it only predicts whether at least an incident will happen over a year for a given company.

Accuracy	0.6769
OOB	0.6183
Precision	0.875
Recall	0.2593
F1 score	0.4

Table 6.5: Average scores over the companies with a yearly time window

Sliding windows

In order to identifies temporal patterns as described in section 5.4.2, the algorithm was tested using the top features computed in section 6.1.2, and various sizes of sliding windows, as described in table 6.6.

	0 (no sliding window)	10	20	30	40	50
Accuracy	0.9014	0.898	0.8993	0.8984	0.8971	0.8976
OOB	0.6777	0.6748	0.6731	0.672	0.6728	0.6691
Precision	0.256	0.1941	0.198	0.1859	0.1831	0.181
Recall	0.06	0.0592	0.0654	0.0545	0.0603	0.0615
F1 score	0.0972	0.0907	0.0984	0.0843	0.0907	0.0918

Table 6.6: Scores depending on the sliding window size

The results in table 6.6 show that using a sliding window preprocessing does not improve the performance of the algorithm. The incidents are the results of the security posture, but the delay may be longer than the size of the sliding window (around 50 days in the largest sliding window size).

Predicting the incident severity

To do a successful impact assessment, the occurrence or not of an incident is not the only parameter to consider. Indeed, many intrusions with a low impact may happen without causing damages. However, incidents with high severity will be less frequent but more damaging. The goal of a company is then to mitigate as much as possible the incidents with a high severity level rather than spending too much budget on preventing the small incidents without impact.

To do so, we separated the incident data into severity categories, from 1 (low severity) to 4 (successful attack) based on the score given by the MSSP security engineers. Then, the prediction was made on each category separately. For each severity category, we trained a separate classifier where the labels are changing. For a given severity category, and a given data sample, the label is set to 1 only if there is at least an incident of the given severity level corresponding to the sample (i.e. same company and same date).

The results can be found in table 6.7.

	Interesting (1)	Low-risk (2)	High risk (3)	Successful attack (4)
Accuracy	0.9155	0.9669	0.9598	0.9998
OOB	0.6784	0.6736	0.6923	0.6667
Precision	0.2733	0.0489	0.0621	/
Recall	0.0722	0.0556	0.1016	0
F1 score	0.1142	0.052	0.0771	/

Table 6.7: Scores depending on the incident severity level

As we notice in the table 6.7, the results for the interesting incidents (incidents with a severity label of 1) are similar to the overall results (see table 6.1), with a low precision score and an even lower recall. It is quite logical because the majority of incidents are often low-level incidents (the successful attacks are less frequent). In the second category (low-risk), we notice that the precision score dropped: the proportion of true positives was indeed especially low for this category, and a lot of events were misclassified as either false positives or false negatives. As the severity increases, the number of false positives increases, and the precision decreases. It may be the result of the data provided to the classifier because there are too few security features concerning those companies to build an efficient classifier. For the last category (Successful attacks), the number of incidents belonging to that category in the dataset was too low to conduct a successful classification, and every incident was missed. We need to group this class needs with a lower class otherwise the model won't be able to predict any incident.

We also notice that the severity category doesn't influence too much the important features, which remain the same overall. It is probably a result of the lack of interesting features: since there are too few features to classify the incidents, the few same features are used for every severity category.

6.2. Limitations

Scanning

The scan as we intended to do it was not possible, because the companies didn't give us the authorization for scanning their network. Therefore, the scanning tool cannot be evaluated on the companies we worked on. When looking at the model results using Shadowserver incident data, we notice that some important features are collected by the scanner (e.g. the DNS open-resolver feature); some others however are not present in the Shadowserver reports (e.g. the SMTP open relay) and there is, for now, no way to estimate the importance of the other collected features.

It also means that the feature data is entirely dependant on a third party (here, Shadowserver). The same methodology may be applied to the data obtained after scanning. However, the lack of temporal data may lead to incorrect results and a more difficult computation of the features per company.

Lack of data

As seen in section 5.7.4, most of the Shadowserver's security features considered do not apply to every company: several companies do not appear in most of the features. For example, in the incident data, only 10 different companies among the company we worked on appear. On the IPMI scan feature (see

figure A.4), only seven distinct companies among the 60 companies we worked on appear. Over the top 6 most relevant features in this section, only 23 different companies are represented, which is less than half of the total number of companies we worked on.

Since most of the companies are not even represented in the security features data, it is impossible to classify their security posture (even with a more precise incident dataset such as the one provided by the MSSP). To solve this limitation, more precise datasets are required, i.e. datasets that provide a faithful representation of every concerned company rather than a global overview that omit several companies.

Evaluation on a single company

The incident classification on a single company is relying on past data (the security posture of the previous days and on which days previous incidents occurred). It means that the classification would not be accurate as soon as the company changes its network configuration (for example, by acquiring new machines or changing its business model).

In order to tackle this issue, the company would need to also use the classification made by considering all the companies at once, to take into account different security configurations and how they are linked to the probability of incidents. However, this may lead to less precise results.

7

Conclusion

7.1. Answer to the research questions

The goal of our project was to answer the following research questions:

- **To what extent can we predict the occurrence and severity of future intrusions in a company, based on external data?**: the main goal of the research was to build a model to predict breaches. As we saw in the results section 6.1, the prediction is often not accurate enough because of the lack of data in the security features dataset, and the model might need more specific data. But in a few cases (especially on companies often mentioned on the Shadowserver dataset), the model is accurate enough to provide a relevant list of features.
- **What are the most relevant indicators an external observer can retrieve from the network of a company?**: the model provides a list of the most relevant features, as described in the table 6.2.
- **How can those indicators be collected?**: using our scanning tool (which combines various existing techniques to obtain the precise results we are interested in), or third-party datasets such as Shadowserver, these indicators can be retrieved.
- **How can the data be labeled and preprocessed?**: as described in section 5.4, multiple steps of encoding and preprocessing were required to tune the data for the machine learning model.
- **What machine-learning model can best predict future breaches?**: using a Random Forest Classifier, we were able to predict incidents and rank the security features to answer our research questions.

7.2. Further work

7.2.1. Scanning and data retrieval

A significant contribution to extend our research would be to evaluate the scanner and see how it performs compared to the Shadowserver data when used for intrusion prediction. Since the companies didn't authorize the scanning of their network, there was no test in real-world conditions and therefore no indication of performance.

If the evaluation performs poorly, the scanner can be further extended: for example by adding features such as the most important indicators determined on the Shadowserver dataset that are not implemented yet. When targeting a specific company, the scanning part is crucial, to obtain precise indicators (that may not be available when using a dataset that performs scanning over a large number of different networks, where there is a trade-off between precision and scanning time).

Adding other security features, especially internal security features may help the classification. Indeed, the internal network of a company, even if it is less open to an attacker, is part of the security posture of a company and any vulnerability or misconfiguration on it may lead to severe incidents. These features also need to be considered when doing a security assessment, and they may provide

additional precision in the data collection. Therefore it would be interesting to collect those features and see how they perform in the incident prediction compared to a model using external data.

With a more accurate dataset of security indicators, the prediction might be more accurate and therefore more valuable for companies to use in actual investments. Once the scanner is authorized to run in real-world conditions, we can gather more data and combine it with the existing classifier to complete the analysis.

7.2.2. Predicting the incident type

Our classifier can predict the incident severity (with more or less accuracy depending on the number of data on a given severity category), which is crucial for calculating the impact of an incident. However, another category that would help a company to estimate the impact of a future breach is the **incident type**. Depending on its type, the incident may not lead to the same response or have the same impact. For example, a distributed Denial of Service attack may cause more damage to an e-commerce company (where the availability is crucial) than to a municipality company. Classifying the incidents depending on their type would then add a layer of precision to the prediction and allow for more fine-grained decisions depending on the goal and assets of the considered company.

7.3. Contribution

To summarize, we divided our research into two parts: first, we created a scanner based on existing tools that retrieves a set of security features from a company as an external observer. We selected the set of security features based on existing literature [29] as well as additional data that seemed relevant for our analysis (such as the CVE analysis). We then combined the existing techniques to build a new scanning tool that retrieves precisely the security features we are interested in.

Once the indicators are retrieved (or using an additional dataset if the scanning itself is not possible), we use them after preprocessing as input for a machine learning classifier, using the Random Forest model, since it performs better than other models. In addition, we use the incident data provided by an MSSP to label the security features and evaluate the model, which performs with an accuracy of 0.92.

After the classifier's training, it returns a list of the features that influenced the classification the most. This feature list depends on several parameters and might be distinct for every company since not every company has the same security posture and indicators. If the classification performs well on Shadowserver's incident dataset, the precision and especially the recall of the model decrease when using the MSSP incident dataset for labeling. Indeed, the external dataset is not precise enough on the security features for most companies to provide a valuable classification: when selecting the companies with enough data, the model returns a far more precise classification of the incidents.

The classifier returns a list of the most important features, which may be interesting for companies looking at investing in their security posture, and for this research in the breach prediction and security posture analysis domains. By adding more detailed security features (such as the internal security features) to the security posture, which is the goal of the project our research belongs to, the classification will be more accurate and therefore more useful for real-case usage. The scanner and classifier we designed are then building blocks on which a detailed modelization of the security posture can be built.

A

Additional figures

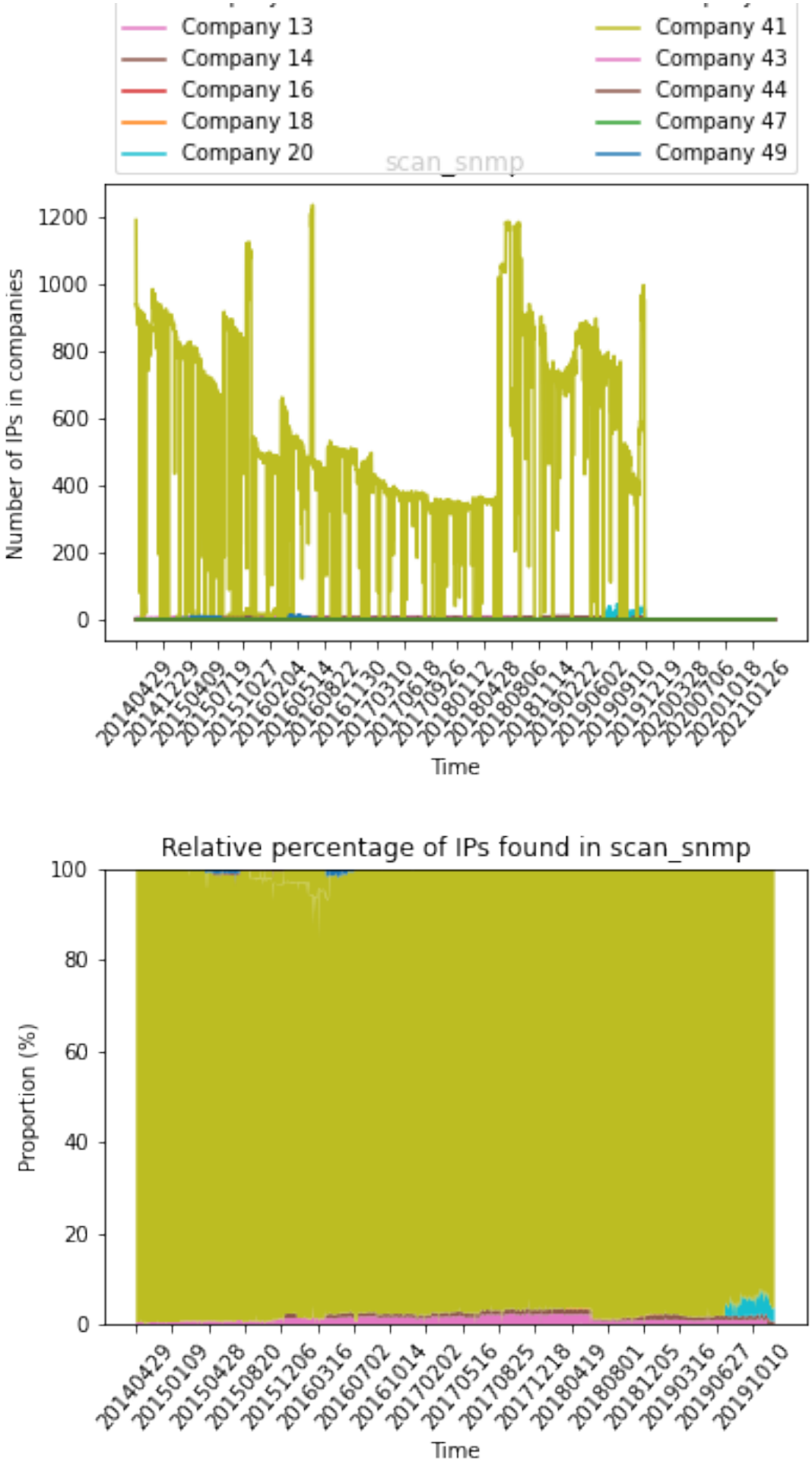
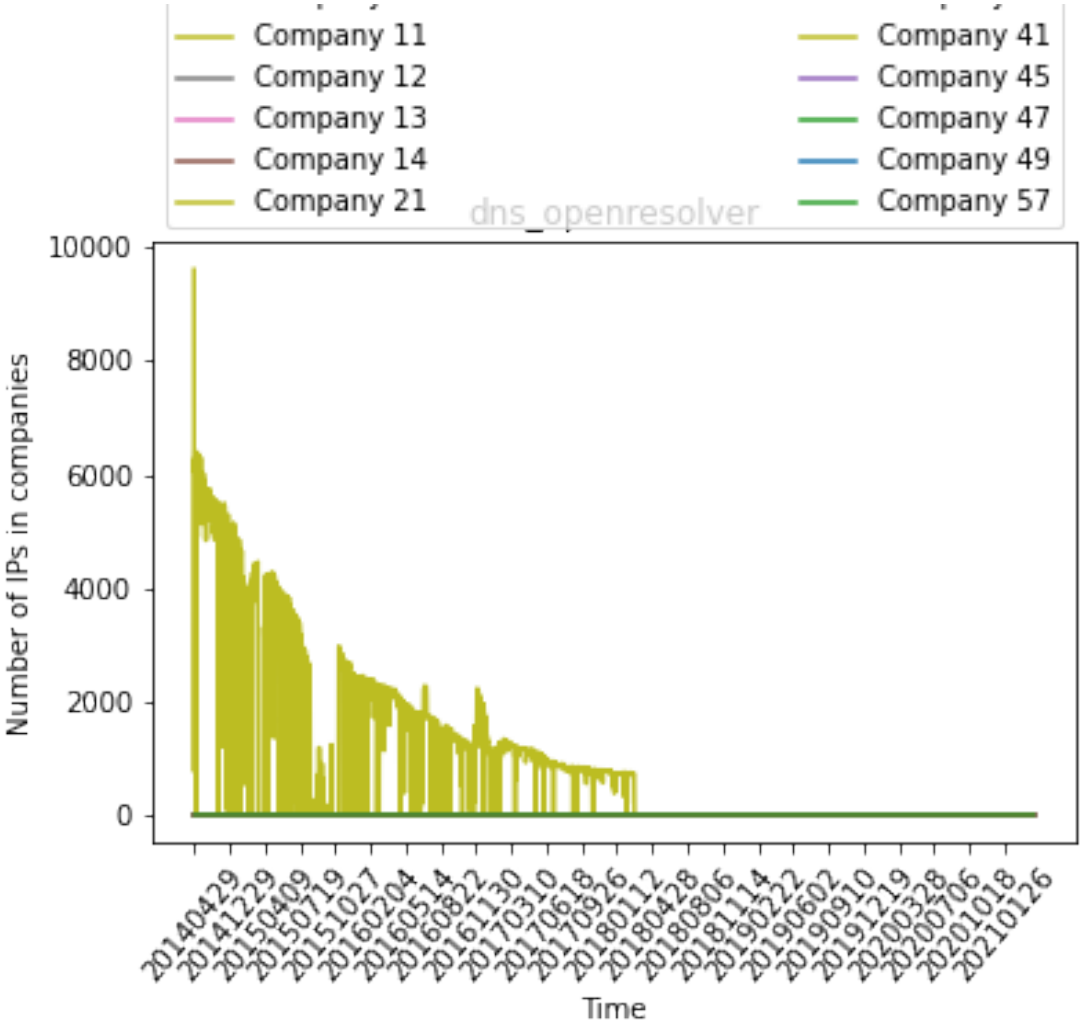


Figure A.1: SNMP scan



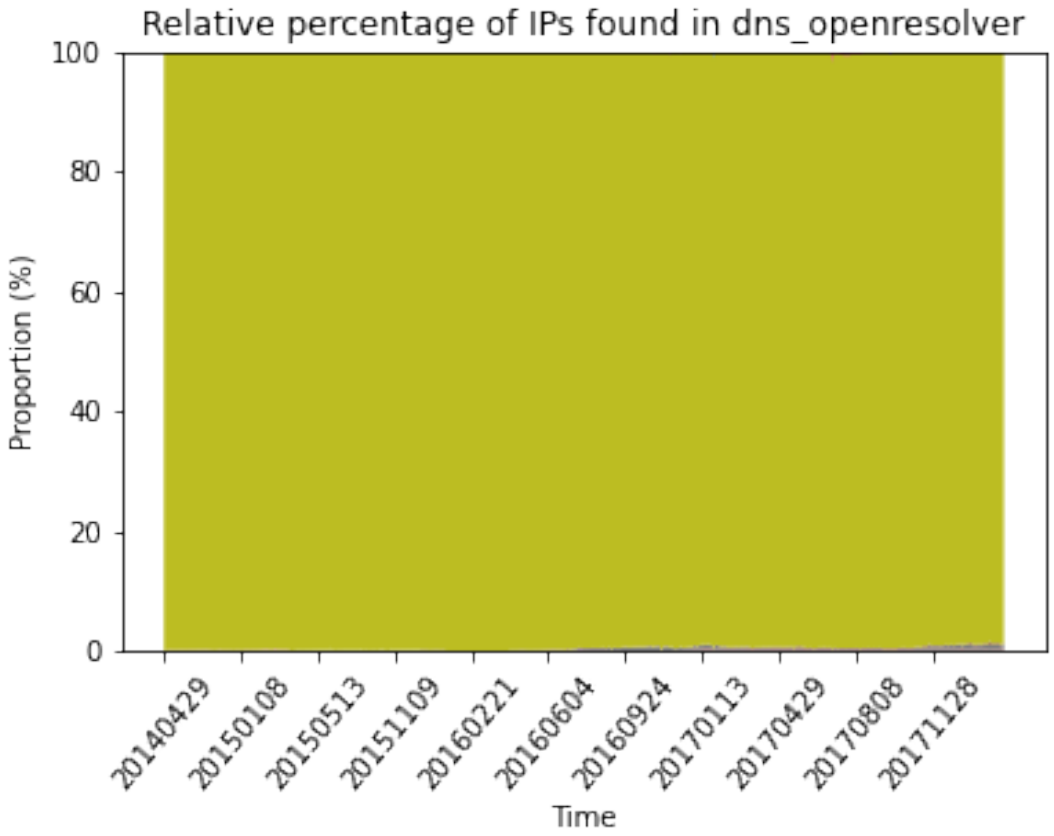


Figure A.2: DNS Open-resolvers scan

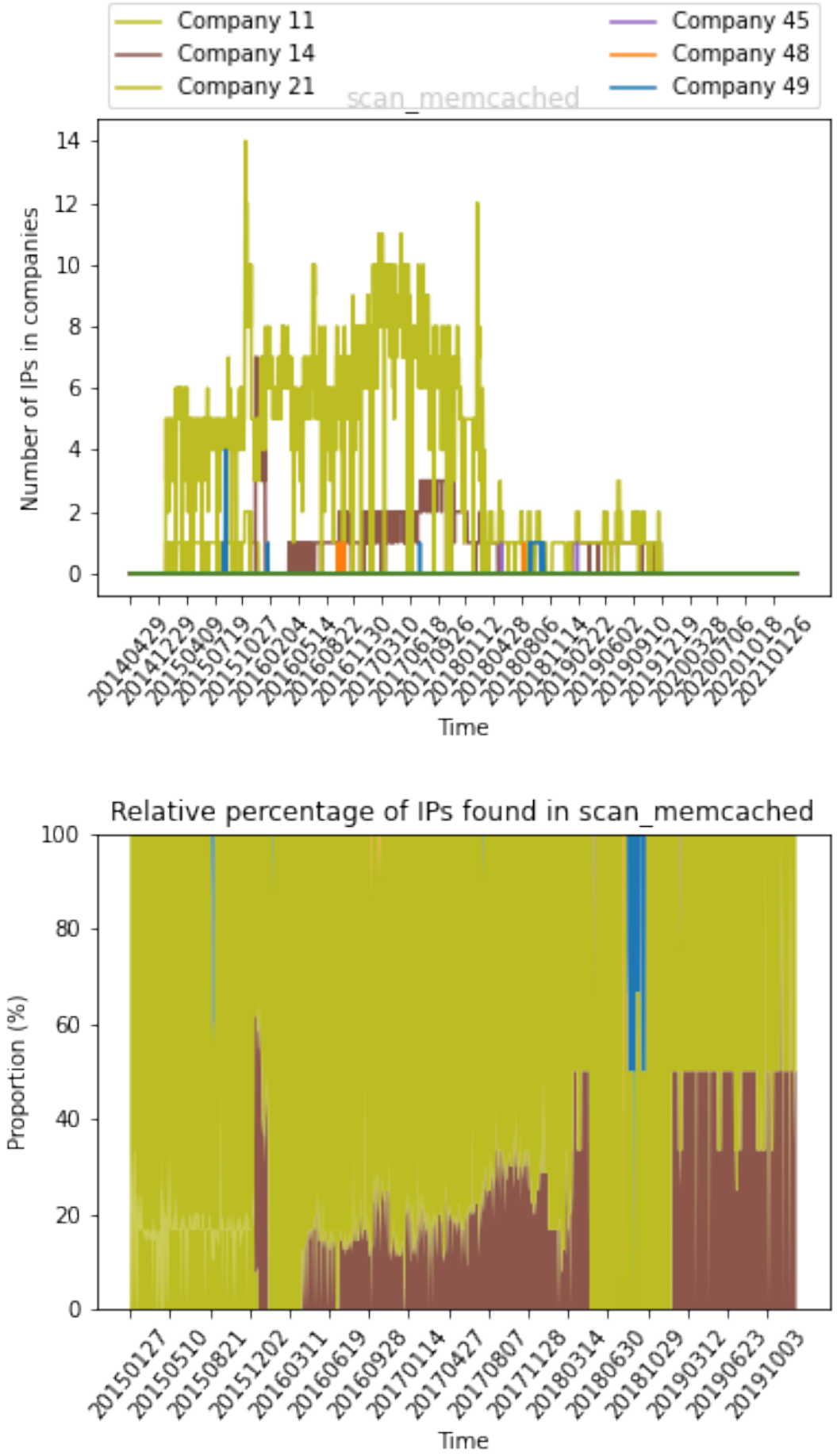


Figure A.3: Open memcached scan

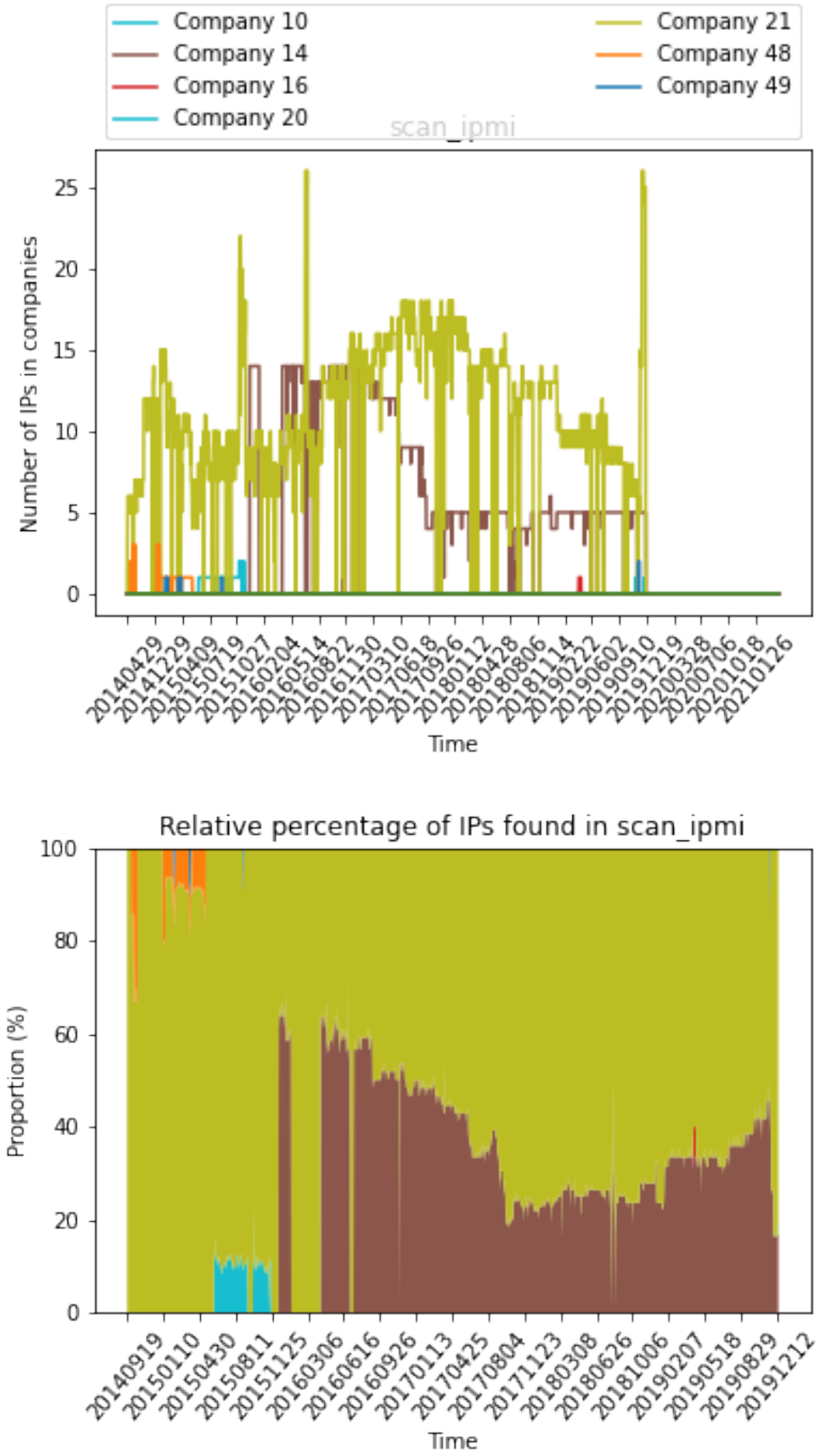


Figure A.4: IPMI scan

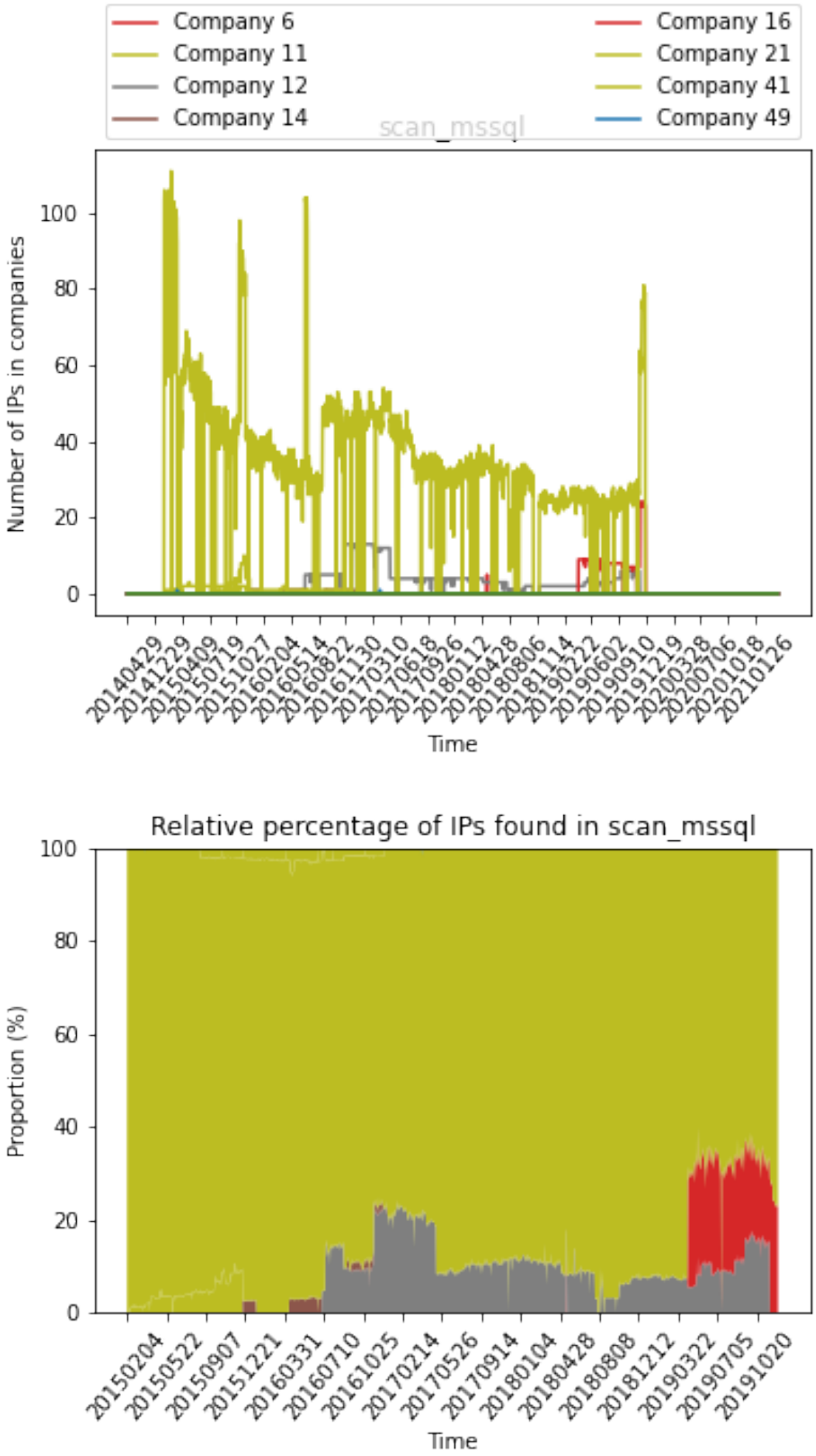
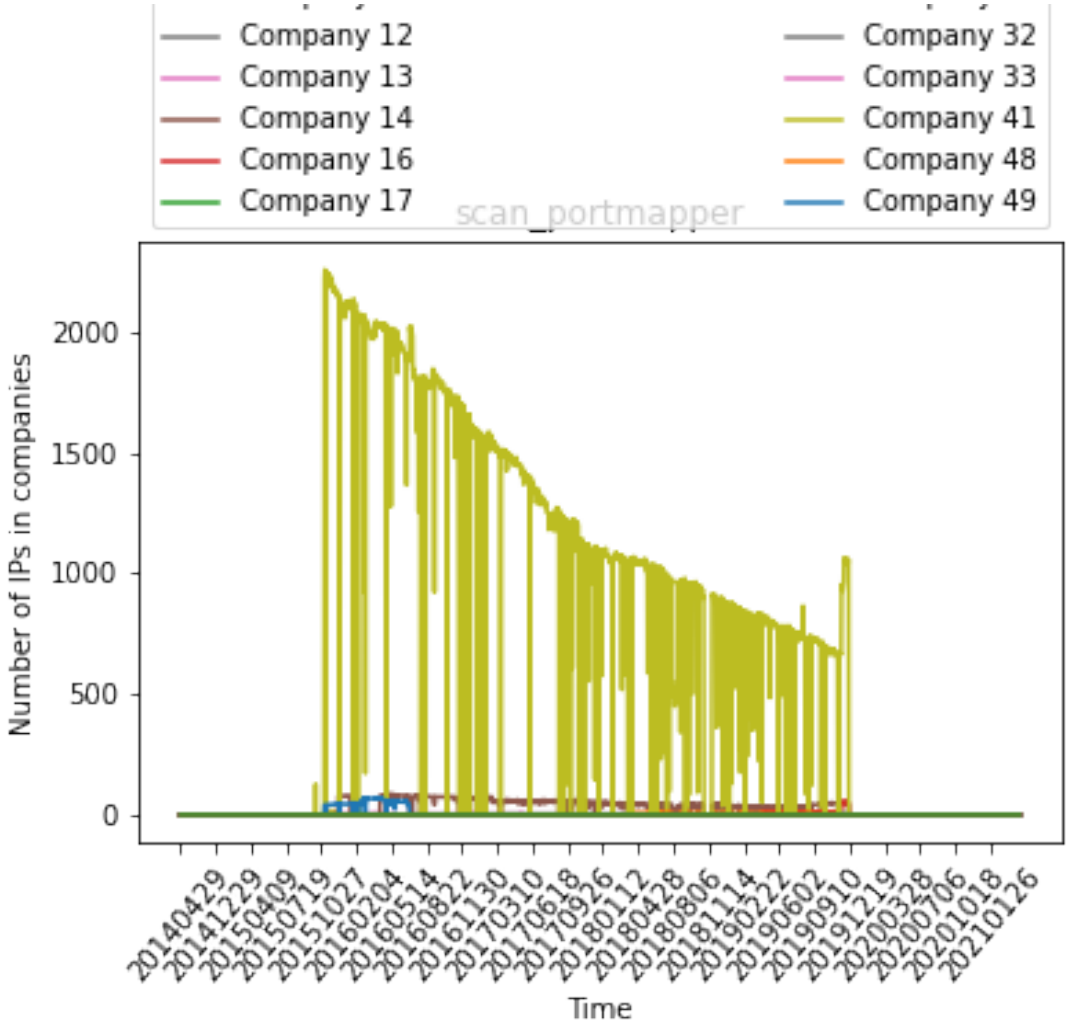


Figure A.5: MSSQL scan



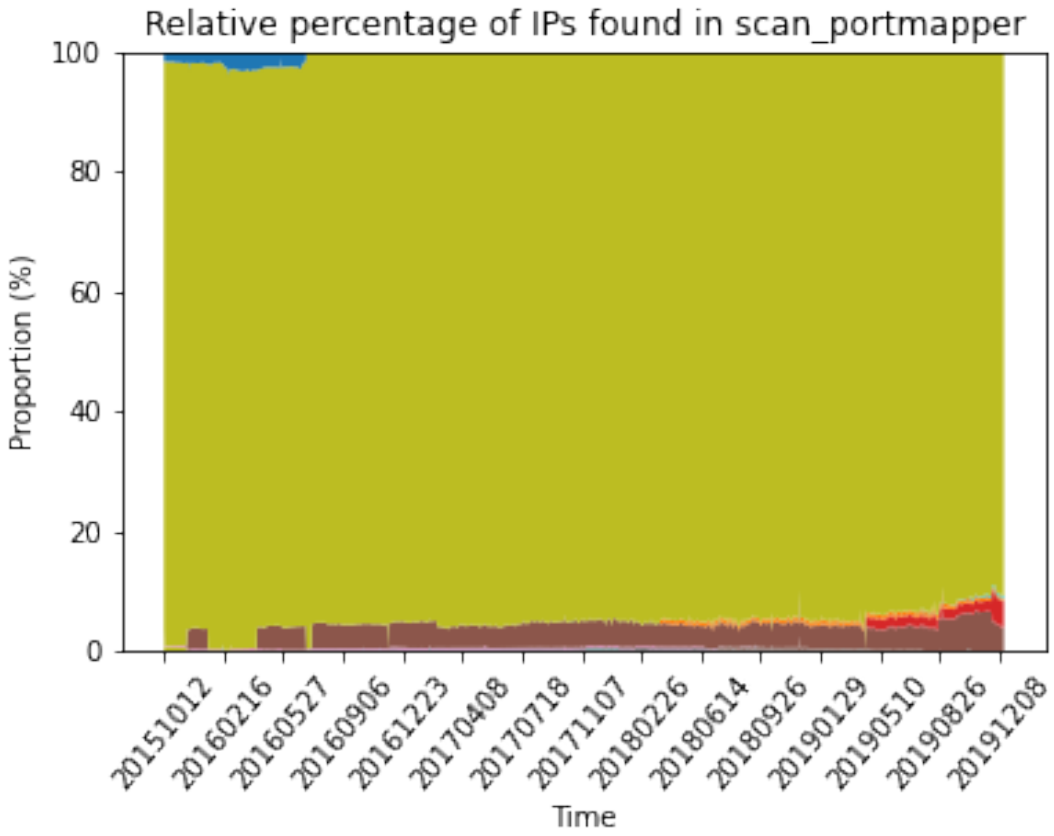


Figure A.6: Portmapper scan

Bibliography

[1] Nov. 2020. URL: https://www.nist.gov/cyberframework.

[2] June 2021. URL: https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml.

[3] *Automation Support for CVE Retrieval*. URL: https://csrc.nist.gov/CSRC/media/Projects/National-Vulnerability-Database/documents/web%5C%20service%5C%20documentation/Automation%5C%20Support%5C%20for%5C%20CVE%5C%20Retrieval.pdf.

[4] Ryan C. Barnett. *Web-Hacking-Incident-Database*. 2009. URL: http://projects.webappsec.org/w/page/13246995/Web-Hacking-Incident-Database.

[5] *CA/Included Certificates*. URL: https://wiki.mozilla.org/CA/Included_Certificates.

[6] *Chapter 4: Port Scanning Overview*. URL: https://nmap.org/book/port-scanning.html#most-popular-ports.

[7] N. V. Chawla et al. “SMOTE: Synthetic Minority Over-sampling Technique”. In: *Journal of Artificial Intelligence Research* 16 (June 2002), pp. 321–357. ISSN: 1076-9757. DOI: 10.1613/jair.953. URL: http://dx.doi.org/10.1613/jair.953.

[8] Sam Clements. *Is Shodan Really the World’s Most Dangerous Search Engine? [Interview with John Matherly]*. Apr. 2013. URL: https://www.vice.com/en/article/9bvxmd/shodan-exposes-the-dark-side-of-the-net.

[9] Zakir Durumeric, Eric Wustrow, and Alex Halderman. 2013. URL: https://zmap.io/.

[10] Zakir Durumeric, Eric Wustrow, and J. Alex Halderman. “ZMap: Fast Internet-wide Scanning and Its Security Applications”. In: *22nd USENIX Security Symposium (USENIX Security 13)*. Washington, D.C.: USENIX Association, Aug. 2013, pp. 605–620. ISBN: 978-1-931971-03-4. URL: https://www.usenix.org/conference/usenixsecurity13/technical-sessions/paper/durumeric.

[11] Benjamin Edwards, Steven Hofmeyr, and Stephanie Forrest. “Hype and heavy tails: A closer look at data breaches”. In: *Journal of Cybersecurity* 2.1 (Dec. 2016), pp. 3–14. ISSN: 2057-2085. DOI: 10.1093/cybsec/tyw003. eprint: https://academic.oup.com/cybersecurity/article-pdf/2/1/3/26672851/tyw003.pdf. URL: https://doi.org/10.1093/cybsec/tyw003.

[12] Benjamin Edwards, Jay Jacobs, and Stephanie Forrest. *Risky Business: Assessing Security with External Measurements*. 2019. arXiv: 1904.11052 [cs.CR].

[13] Elemind.com. *The VERIS Community Database (VCDB)*. URL: http://veriscommunity.net/vcdb.html.

[14] Michel J.G. Van Eeten Fabio Bisogni Hadi Asghari. “Estimating the size of the iceberg from its tip. An investigation into unreported data breach notifications”. In: June 2017.

[15] Brad Fitzpatrick. *a distributed memory object caching system*. Nov. 2020. URL: https://memcached.org/.

[16] Robert David Graham. *Masscan*. Jan. 2021. URL: https://github.com/robertdavidgraham/masscan.

[17] *How many iot devices Are there in 2021? More than ever!* Aug. 2021. URL: https://techjury.net/blog/how-many-iot-devices-are-there/.

[18] Martin Husák and Pavel Čeleda. “Predictions of Network Attacks in Collaborative Environment”. In: *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*. 2020, pp. 1–6. DOI: 10.1109/NOMS47738.2020.9110472.

[19] Martin Husák and Jaroslav Kašpar. “AIDA Framework: Real-Time Correlation and Prediction of Intrusion Detection Alerts”. In: *Proceedings of the 14th International Conference on Availability, Reliability and Security*. ARES ’19. Canterbury, CA, United Kingdom: Association for Computing Machinery, 2019. ISBN: 9781450371643. DOI: 10.1145/3339252.3340513. URL: https://doi.org/10.1145/3339252.3340513.

[20] Martin Husák and Jaroslav Kašpar. “Towards Predicting Cyber Attacks Using Information Exchange and Data Mining”. In: *2018 14th International Wireless Communications Mobile Computing Conference (IWCMC)*. 2018, pp. 536–541. DOI: 10.1109/IWCMC.2018.8450512.

[21] *ISO/IEC 27001 - Information security management*. Apr. 2020. URL: https://www.iso.org/isoiec-27001-information-security.html.

[22] Allan Jay. *Number of Internet of Things (iot) connected devices worldwide 2021/2022: Breakdowns, growth and predictions*. Mar. 2021. URL: https://financesonline.com/number-of-internet-of-things-connected-devices/.

[23] Joseph Johnson. *U.S. data breaches and exposed records 2020*. Mar. 2021. URL: https://www.statista.com/statistics/273550/data-breaches-recorded-in-the-united-states-by-number-of-breaches-and-records-exposed/.

[24] Kerem Kaynar. “A taxonomy for attack graph generation and usage in network security”. In: *Journal of Information Security and Applications* 29 (2016), pp. 27–56. ISSN: 2214-2126. DOI: https://doi.org/10.1016/j.jisa.2016.02.001. URL: https://www.sciencedirect.com/science/article/pii/S2214212616300011.

[25] Katarina Kertysova et al. *Ensuring awareness and resilience of the private sector across Europe in face of mounting cyber risks*. 2018, pp. 70–117.

[26] Emanuel Kopp and Lincoln Kaffenberger. *Cyber risk, market failures, and financial stability*. Aug. 2017. URL: https://www.imf.org/en/Publications/WP/Issues/2017/08/07/Cyber-Risk-Market-Failures-and-Financial-Stability-45104.

[27] Platon Kotzias et al. “Mind your Own Business: A Longitudinal Study of Threats and Vulnerabilities in Enterprises”. In: *Network and Distributed Systems Security Symposium*. San Diego, CA, USA, Feb. 2019. DOI: https://dx.doi.org/10.14722/ndss.2019.23522. URL: https://www.ndss-symposium.org/wp-content/uploads/2019/02/ndss2019_03B-1-2_Kotzias_paper.pdf.

[28] Dongwoo Kwon et al. “DDoS attack volume forecasting using a statistical approach”. In: *2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. 2017, pp. 1083–1086. DOI: 10.23919/INM.2017.7987432.

[29] Yang Liu et al. “Cloudy with a Chance of Breach: Forecasting Cyber Security Incidents”. In: *Proceedings of the 24th USENIX Conference on Security Symposium*. SEC’15. Washington, D.C.: USENIX Association, 2015, pp. 1009–1024. ISBN: 9781931971232.

[30] Yang Liu et al. “Predicting Cyber Security Incidents Using Feature-Based Characterization of Network-Level Malicious Activities”. In: Mar. 2015, pp. 3–9. DOI: 10.1145/2713579.2713582.

[31] John Matherly. *Search Engine for the Internet of Everything*. 2009. URL: https://www.shodan.io/.

[32] *Official Common Platform Enumeration (CPE) Dictionary*. URL: https://nvd.nist.gov/products/cpe.

[33] Terrence Parr et al. *Beware Default Random Forest Importances*. Mar. 2018. URL: https://explained.ai/rf-importance/index.html.

[34] Paulo Passeri. *Hackmageddon, Information Security Timelines and Statistics*. June 2021. URL: https://www.hackmageddon.com/.

[35] Matthew Prince. *The DDoS That Knocked Spamhaus Offline (And How We Mitigated It)*. Aug. 2018. URL: https://blog.cloudflare.com/the-ddos-that-knocked-spamhaus-offline-and-ho/.

[36] Takaya Saito and Marc Rehmsmeier. “The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets”. In: *PLOS ONE* 10.3 (Mar. 2015), pp. 1–21. DOI: 10.1371/journal.pone.0118432. URL: https://doi.org/10.1371/journal.pone.0118432.

[37] Anna Sapienza et al. “DISCOVER: Mining Online Chatter for Emerging Cyber Threats”. In: *Companion Proceedings of the The Web Conference 2018*. WWW ’18. Lyon, France: International World Wide Web Conferences Steering Committee, 2018, pp. 983–990. ISBN: 9781450356404. DOI: 10.1145/3184558.3191528. URL: https://doi.org/10.1145/3184558.3191528.

[38] Armin Sarabi et al. “Risky business: Fine-grained data breach prediction using business profiles”. In: *Journal of Cybersecurity* 2.1 (Dec. 2016), pp. 15–28. ISSN: 2057-2085. DOI: 10.1093/cybsec/tyw004. eprint: https://academic.oup.com/cybersecurity/article-pdf/2/1/15/10833165/tyw004.pdf. URL: https://doi.org/10.1093/cybsec/tyw004.

[39] Yuanquan Shi et al. “An immunity-based time series prediction approach and its application for network security situation”. In: *Intelligent Service Robotics* 8.1 (2014), pp. 1–22. DOI: 10.1007/s11370-014-0160-z.

[40] Synopsys. *The Heartbleed Bug*. 2014. URL: https://heartbleed.com/.

[41] *The Shadowserver Foundation*. URL: https://www.shadowserver.org/.

[42] Emmanouil Vasilomanolakis et al. “Taxonomy and Survey of Collaborative Intrusion Detection”. In: *ACM Computing Surveys* 47 (May 2015). DOI: 10.1145/2716260.

[43] The Spamhaus Project Webteam. URL: https://www.spamhaus.org/.

[44] M. Xu et al. “Modeling and Predicting Cyber Hacking Breaches”. In: *IEEE Transactions on Information Forensics and Security* 13.11 (2018), pp. 2856–2871. DOI: 10.1109/TIFS.2018.2834227.

[45] Jing Zhang et al. *On the Mismanagement and Maliciousness of Networks*. 2014.

[46] Zhang Zhengdao, Peng Zhumiao, and Zhou Zhiping. “The Study of Intrusion Prediction Based on HsMM”. In: *2008 IEEE Asia-Pacific Services Computing Conference*. 2008, pp. 1358–1363. DOI: 10.1109/APSCC.2008.107.

[47] Yueran Zhuo and Senay Solak. “Measuring and optimizing cybersecurity investments: A quantitative portfolio approach”. In: *IIE Annual Conference. Proceedings*. Citeseer. 2014, p. 1620.