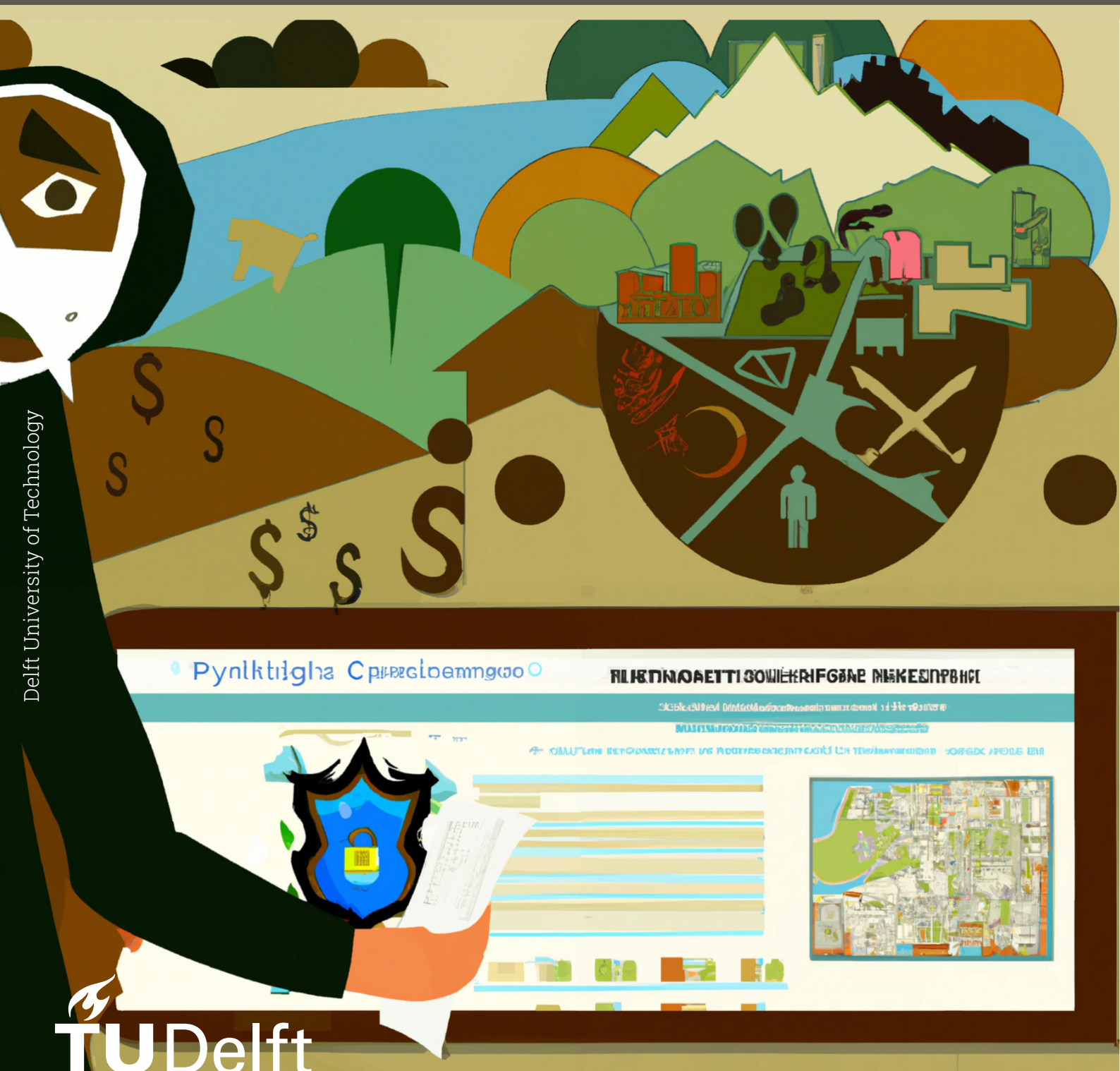
Once upon a Tuesday

Longitudinal analysis of the vulnerability management of Dutch municipalities

Yana Angelova



Delft University of Technology

TU Delft

Once upon a Tuesday

Longitudinal analysis of the vulnerability management of Dutch municipalities

by

Yana Angelova

to obtain the degree of Master of Science
Software Technology Track
with a 4TU specialization in Cyber Security

at the Delft University of Technology,
to be defended publicly on Tuesday October 17, 2023 at 15:45.

Student number: 4649370

Thesis committee:	Prof. dr. M. J. G. van Eeten,	TU Delft, chair
	Dr. R. S. van Wegberg,	TU Delft, first supervisor
	Prof. dr. G. Smaragdakis,	TU Delft, second supervisor
	A. Ethembabaoglu,	TU Delft, daily supervisor

An electronic version of this thesis can be found: `http://repository.tudelft.nl/`
Cover: Image created with the assistance of DALL·E 2.

TU Delft

Preface

When I first came to TU Delft, I did not know what to expect from my university life. Back then the weather was nice, I was fresh out of high school, and I was filled with ambition for new knowledge. Only one of those things stayed the same throughout my journey at this university. This work is the culmination of 6 years filled with joy, stress, knowledge, sweat, blood, and tears and it would be unfair if I did not acknowledge the people who made it possible.

First and foremost I would like to extend my gratitude to my supervisors in this journey called “master thesis". Thank you Rolf for allowing me to explore a side of cybersecurity, closely related to our society and always being optimistic even when I did not feel that way. Aksel, thank you for all the chats during my journey of discovering what I was doing and apologies for all the suffering you had to go through with my complicated visualisations. A special thank you goes to Michel and Georgios for always bringing valuable insights into our meetings and encouraging me to do my best with the tricky aspects of my research.

A special thank you also goes to everyone in the Cybersecurity group at TPM. You all made me feel so welcome and as part of the group. These 9 months would not have been as enjoyable for me if it were not for every single one of you. I will see you next Thursday!

To my friends, I owe both a thank you and an apology. Apologies for all the time I have been complaining to you and using you as rubber ducks, and for going MIA when the stress was getting the best of me. But most importantly thank you for always being there for me, whether it was for a cup of coffee or a spontaneous gossip session.

Finally, I would like to express my gratitude to my family, both back home in Bulgaria and here in the Netherlands. Without you, none of this would have been possible. Your support is what keeps me pushing forward. A special thanks to Philippos, who should probably receive a master’s degree of his own just for putting up with me in these last few months.

This thesis is in memory of Mirjam.

Yana Angelova
Delft, October 2023

P.S.: If you feel like I have forgotten to thank you, consider this one just for you.

Abstract

In recent years, more and more emphasis has been put on the importance of good preventative cyber security and vulnerability management techniques such as “Patch Tuesday". Despite the increased importance, not all organisations have the same resources and knowledge when it comes to securing their networks against cyber adversaries.

This research tries to examine the vulnerability posture of Dutch municipal ICT networks. To accomplish this a network ranges dataset was curated using open source intelligence techniques. These networks, related to current and previous Dutch municipalities, have been used to collect network data scans and observe the changes in software products and versions. Based on the data collected we can observe the software update moments for different organisations and analyse how often software products are kept up to date. Using this network scan data and a subset of open-source products, we were able to construct a case study analysis about the general trends of vulnerability management and the influencing factors thereof. This was done through timeline analysis, involving also software update releases, security advisories, and publicly disclosed vulnerability exploits. Our findings show uncoordinated strategies within the different organisations and rare proactive security behaviour.

Another contribution of this study is in the sphere of reconnaissance and open source intelligence gathering, showing that publicly available information alone is a time-consuming procedure that renders very few useful data points. These later findings have implications for both adversaries as well as security organisations, as reliable data could only be obtained through direct contact with the underlying municipality.

Contents

List of Figures

List of Tables

1

Introduction

With the ever-increasing reliance on technology in modern society, cybersecurity has become a crucial aspect of information systems management. Vulnerability management is a key component of cybersecurity that focuses on identifying, prioritising, and mitigating security vulnerabilities within an organisation's infrastructure. In recent years, vulnerability management has gained significant attention from the academic community and the industry due to the growing number of cyber attacks and the potential damage they can cause.

In recent years, the growth of the internet and digital technology has revolutionised the way organisations carry out their daily activities. Municipalities are no exception to this trend, and they have extensively adopted digital solutions to manage their affairs. While this has brought about numerous benefits, it has also increased the vulnerability of municipal IT systems to cyber-attacks. This vulnerability arises from the ever-expanding and complex nature of technological systems, making it increasingly challenging to identify and address security threats.

Software security patches are an essential mitigation strategy that organisations can use to address vulnerabilities [16]. However, patching can be a challenging task, and different factors can influence an organisation's vulnerability management strategies [26]. As a result, organisations exhibit different security behaviours, which can affect the efficiency and effectiveness of their vulnerability management efforts. Therefore, it is essential to investigate these behaviours over time and understand the decision-making processes involved.

The main focus of this research is on the vulnerability management of IT systems in Dutch municipalities. Municipalities are an excellent target for this analysis because of the societal importance of their IT infrastructure and the sensitivity of the data they collect and process. This work aims to observe the vulnerability management practices of Dutch municipalities over time and derive any factors that influence the process.

The study will adopt an observational research design, which involves observing vulnerability management practices in the wild, without direct access to the municipalities' IT systems. This approach ensures that municipalities will not alter their behaviour under the knowledge that they are being observed, thus providing an accurate representation of their vulnerability management practices. Furthermore, this approach could give useful insights into what information could be derived using open source intelligence techniques (OSINT), a practice commonly employed by cyber adversaries.

2 Background

The term vulnerability has multiple definitions used in literature, all-encompassing the notion of “Weakness in an information system, system security procedures, internal controls, or implementation that could be exploited or triggered by a threat source" as explained by the National Institute of Standards and Technology [19]. NIST further defines vulnerability management as: “An Information Security Continuous Monitoring (ISCM) capability that identifies vulnerabilities [Common Vulnerabilities and Exposures (CVEs)] on devices that are likely to be used by attackers to compromise a device and use it as a platform from which to extend compromise to the network." For this study, we have adopted these two definitions as well.

In this chapter, we will further look into the current research approaches to the challenges of vulnerability management. In section 2.1 we will focus on the research studies proposing vulnerability management solutions. Following, section 2.2 will explain more about the approach of interviewing security practitioners to assess an organisation’s vulnerability management practices. A deeper look into the few papers performing outside analysis will be presented in section 2.3. Finally, the research gap and focus of this research will be explained in section 2.4.

2.1. Solution focused research

The field of vulnerability management and research thereof is not new, but it has gained more popularity in recent years. A literature review performed by Dissanayake et al. [8] lays out the different aspects of vulnerability patch management, the challenges, and possible solutions found in the literature. From this work, we have identified that a big part of the research focus is on the “Vulnerability Scanning, Assessment and Prioritisation" part of the vulnerability management process. Although a lot of work has been published analysing this part, very few case studies have been conducted, with the majority of papers using artificially created data to evaluate their approach or compare their solutions to other already existing ones.

A lot of effort is put into proposing solutions for the management of vulnerable systems, such as prioritisation strategies and mitigation techniques. Some of these studies try to tackle the problem of uniform identification of vulnerabilities and the absence of unified resources for vulnerability management. An example study is that of Varela-Vaca et al [27], in which they propose a feature-model-based tool that could help identify and prioritise vulnerabilities present in a given system. Their tool presents the possibility to query and analyse multiple vulnerabilities and exploit repositories, assisting security officers in their management tasks.

Finally, OHare, Macfarlane, and Lo [20] propose a tool that could do vulnerability discovery using internet-wide scan data from open-access scanning services. This research is more in line with the goals of the current work, however, its main focus is on identifying vulnerabilities and not analysing the management techniques taken by the affected organisations.

2.2. Practice evaluation research

The field of evaluating vulnerability management techniques is a bit less studied in comparison. Nevertheless, the majority of this research is focused on approaches based on collaboration with the researched entities, primarily through user studies and interviews.

An evaluation study on different update strategies was performed by Di Tizio, Armellini, and Massacci [7], where the authors compared different update approaches to known Advanced Persistent Threats (APTs). This research presents a more realistic comparison of different mitigation techniques by using reported APTs that have been used in real-life attack campaigns. Nevertheless, the focus of the study is on evaluating the strategies and not on investigating a real-life example of vulnerability management.

When it comes to real-life case studies, researchers focus on user study-based research with different organisations. In their work Gerace and Cavusoglu [13] have conducted a series of surveys with IT professionals in order to identify which pre-defined factors are perceived as crucial for patch management and how organisations view patching. They have also studied the manner in which patches are applied and the perceived effectiveness of this process.

Another interview-based work is that of Tiefenau and Häring [25]. In their paper, they have explored not only what factors influence patching, but what are the general behaviour and attitude of system administrators in a corporate environment. They have also identified certain challenges present when it comes to patching outdated systems. These types of research provide a better understanding and explanation of the different vulnerability management techniques.

2.3. Outside analysis research

There have been, nevertheless, a few papers investigating the “outside" view of vulnerability management and the posture of different systems. When we talk about the “outside" view we mean research conducted without direct contact with the organisations and subjects of the study. In most cases, this is achieved through the use of network scans.

One of the first works employing this approach is by Durumeric et al [9]. They have evaluated the vulnerability state of the Alexa Top 1 Million domains and the IPv4 address space concerning the Heartbleed exploit. This study is entirely focused on one specific vulnerability present in OpenSSL, which Heartbleed exploits. They have used network scan data to determine the speed of vulnerability mitigation following the disclosure of the Heartbleed vulnerability in 2014. Another key aspect of this study is the investigation of the effectiveness of vulnerability notification on system patching. The main characteristic of this research is the fact that it was conducted for the Heartbleed vulnerability, one of the most widely covered vulnerabilities in recent years. This narrow scope could not be used to draw any conclusions on the overall vulnerability management of different organisations.

A similar, internet-wide research was done by Demir et al [6]. They have focused their study on vulnerabilities in website technologies. In their work, they show the current vulnerability state of the Web and examine the possible security implications related to not updated web software products. Because of their web focus, this study has different goals than the current work, nevertheless, the approach to achieving these goals is similar.

Another focused research is that of West and Moore [28]. In their paper, they investigate the update level of OpenSSH on an internet-wide basis using the backport information. This paper puts forward the notion that version numbers alone, do not always tell the whole story about the update status of a system. The authors have opted for an analysis of OpenSSH patching practices, because of its widely available patch-level information derived from the backports. Although restrictive in scope, this paper has been used as model research during the execution of this study.

2.4. Research gap and contributions

From the previous work, we can establish that the majority of efforts in vulnerability management research go towards analysing solutions and mitigation techniques or making use of user studies to investigate vulnerability management in different organisations. Only a few studies have looked at publicly available data and used it to do vulnerability analysis. Furthermore, most often these studies focus on a single service or vulnerability and do not relate what factors might be influencing the observed patching trends.

As such this research's contributions to the scene of vulnerability management analysis are found in its longitudinal character and outside-view approach. These contributions will be achieved by answering the following research questions:

RQ: What are the security practices of Dutch municipalities concerning vulnerability management?

SQ1: What can be derived about the update behaviour of municipalities from open source data?

SQ2: Are there factors influencing these security practices?

We want to establish what information can be derived without direct access or knowledge about the organisation's networks and whether they follow proactive patching practices. The analysis will focus on the most popular services present, in order to broaden the scope of the previous works and look into data for a time period of two years, in order to derive the underlying vulnerability management. Furthermore, the focus on Dutch municipalities illustrates a more concrete picture of the patching practices of (a subset of) governmental institutions, which do not always have the same cybersecurity capabilities as corporate organisations.

As such, the contribution this research attempts to make are:

- Develop a framework methodology for open source data collection of Dutch municipal networks and network data
- Perform exploratory analysis of the current network data related to Dutch municipalities
- Present case study analysis of longitudinal vulnerability exposure and management

3

Data & Methodology

This research employs a step-wise methodology where each step relies on the data and results collected in the previous steps. In this chapter, we explore the different data needed for the study in section 3.1, the steps taken in our approach in more detail in section 3.2, and briefly discuss the ethical implications of the research in section 3.3.

3.1. Data

We have three main sources of data for this study: municipality names and changes thereof, IP ranges associated with these municipalities, and network scan data retrieved for these ranges. It is in this order that the data is considered and explained next.

Further supplementary data is collected concerning the software product's release dates, security advisories as published by the National Cyber Security Center, and publicly disclosed vulnerability exploits.

3.1.1. Municipality names

As of 1st January 2023 in the Netherlands, there are 342 municipalities [11]. This has not always been the case, with a number of municipality changes introduced every year. Because of the longitudinal nature of this research, this dataset includes information on the municipality names for each year for the time period between 2001 and 2023 [12]. This dataset has been created using the publicly available list of municipalities from 2001 and applying the announced official changes in structure or name between 2001 and 2023 for each year. As such for each year the data shows how many municipalities there have been in the Netherlands and their names. The full list of municipality names and the announced changes can be found in Appendix A.

This approach of manually reproducing the records for each year was based on the lack of uniformity in the data archive of CBS before 2010.

3.1.2. Municipal network ranges

The municipal network ranges have been aggregated using three distinct data sources. For all sources, we have only considered municipality-owned networks, meaning that ICT assets hosted on cloud infrastructure were not searched for and consecutively used in this research. This choice to exclude assets hosted on the cloud was based on the fact that acquiring the municipality specific IP addresses was infeasible using the proposed methodology. Furthermore, we have seen in previous works [28] that cloud hosted infrastructure does not suffer from the same flaws in vulnerability management as networks administrated by the organisations themselves.

MaxMind The first source of network ranges is MaxMind. MaxMind is a service that provides different types of network-related data including mapping from IP to geographical locations (GeoIP) [17]. Using their GeoIP database we could query for networks, which use the word “gemeente" (Dutch for municipality) in their name. This renders a list of network names, the date on which these networks have been observed, and the CIDR ranges associated with them. An example MaxMind result for the query “gemeente" looks as follows:

Name	Networks	Date
“Network of Gemeente X"	11.222.333.0/25	20180130

Table 3.1: Example MaxMind result for the query “gemeente"

As we can see the network’s name is “Network of Gemeente X", it contains the CIDR range `11.222.333.0/25` and it was valid as of `2018/01/30`.

RIPE The second source used is the database of RIPE NCC records. RIPE NCC is the regional internet registry for Europe, the Middle East, and parts of Asia [22]. Because of their overseeing capabilities of IP records, the data from their databases could be considered the closest to the “ground truth". Here as well the query was for InetNums (their equivalent of IP ranges consisting of starting and ending IP addresses) containing the word “gemeente" in their name or description. Once again the resulting data consists of an IP range, the name of the range, and the date of the last modification to the retrieved record. The following is an example received as a result of our query:

inetnum:	11.222.333.0 - 11.222.333.444
netname:	NL-GEMEENTEX-NET4
descr:	Network of Gemeente
country:	NL
admin-c:	DUMY-RIPE
tech-c:	DUMY-RIPE
status:	ASSIGNED PA
remarks:	Service: MPN
remarks:	Please send SPAM reports to postmaster@ibm.net
remarks:	Please send ABUSE reports to abuse@ibm.net
mnt-by:	EU-IBM-NIC-MNT
created:	1970-01-01T00:00:00Z
last-modified:	2001-09-21T22:38:06Z
source:	RIPE
remarks:	**********
remarks:	* THIS OBJECT IS MODIFIED
remarks:	* Please note that all data that is generally regarded as personal
remarks:	* data has been removed from this object.

Table 3.2: Example RIPE result for the query “gemeente"

In this case, the fields of interest for us are the inetnum `11.222.333.0 - 11.222.333.444`, corresponding to the network with name “NL-GEMEENTEX-NET4", which was last modified om `2001-09-21T22:38:06Z`.

Hurricane Electric The final data source is Border Gateway Protocol (BGP) records retrieved from Hurricane Electric. BGP is a standard protocol used by Internet Service Providers (ISPs) and Autonomous Systems (AS) to exchange routing and reachability information on the Internet [4]. The Hurricane Electric BGP Toolkit provides an easy lookup for networks using parts of their name [15]. Once again the word “gemeente" was used to retrieve the associated CIDR ranges. For this data only the network name and CIDR range were available. These data points have been considered current in February 2023. Here the example results are as follows:

Name	Network
Gemeente X	11.222.333.0/25

Table 3.3: Example Hurricane Electric result for the query “gemeente"

Here we see the network with the name “Gemeente X" and CIDR range `11.222.333.0/25`.

The decision to use three distinct datasets has been in part based on the premise of multi-source validation, where multiple sources agree that the IPs belong to a municipality, and partially on the scarcity of the available data. It is important to note, that networks which do not contain municipality-identifying words such as “gemeente" would not be discovered using this data collection approach. As such the currently collected dataset could be seen as a lower bound on the municipality network landscape, but it should not be seen as the full picture. This is an intrinsic limitation of the open-source nature of this study. Further analysis of this dataset will be provided in chapter 4.

3.1.3. Network scan data

The final data set of this research is network scan data associated with the IPs related to municipalities. Numerous services provide such data the most popular being Censys and Shodan [5] [23]. Bennett, Abdou, and van Oorschot [3] have shown that when comparing the two services the results are largely similar. For the purpose of this work, the scanning service of choice is Shodan. The network scan data retrieved from it consists of numerous banner tags such as “IP string", “port", “timestamp", etc. The banner tags of interest for this study can bee seen in Table 3.4. A further feature is the possibility to request the history scans for an IP address, allowing you to see any potential changes to the underlying services over time.

As such this study has collected current network scan data from April 2023 as well as historic data on all discovered IPs. These results contain entries for currently and historically active IP addresses. In this study, the terms entries and records are used interchangeably to refer to the scanned results retrieved from Shodan. Further analysis of the collected data will be provided in chapter 4.

3.1.4. Software release dates, vulnerability identifiers, and vulnerability exploits

To facilitate the longitudinal aspect of the analysis a few extra supplementary data points have been gathered. Each software product is assigned a unique Common Platform Enumeration (CPE) that encodes the vendor, product, and version of the given product. Using the `cpe` one can distinguish between different versions of the same product. Another useful identifier is the Common Vulnerabilities and Exposures (CVEs), these uniquely enumerate software vulnerabilities as discovered by Mitre [2]. Finally, some vulnerabilities have publicly disclosed exploits, that abuse the found bugs or misconfigurations in the software to gain unauthorised access or

[1] For the full list and description of tags visit: `https://datapedia.shodan.io/`

[2] `https://cve.mitre.org/`

Property Name	Type	Description	Required
cpe	array of strings	CPE information in the old, deprecated format	
cpe23	array of strings	CPE information in the 2.3 format	
hash	integer	Numeric hash of the "data" property which is helpful for finding other IPs with the exact same information	Yes
ip	integer	Numeric IP address which can be more efficient for storing or indexing	
ip_str	string	String representation of the IP address	
org	string	Name of the organisation that manages the IP	
os	string	Operating system	
port	integer	The port being scanned	Yes
product	string	Name of the software that powers the service	
timestamp	string	Date and time that the banner was collected in UTC time	Yes
version	string	The version of the product that powers the service	

Table 3.4: Description of relevant Shodan banner tags [1]

hinder one or more of the CIA triad (Confidentiality, Integrity, Availability). All these data points will be used in this research and have been collected as follows.

In order to observe individual update patterns, we first need to collect the relevant product information and dates. The software release dates for the most popular open-source products in the dataset have been collected. For each product the public repositories have been searched for announcements of new version releases and these dates have been saved. Whenever needed, the mailing list archives for the products have also been searched if the repositories do not contain enough information. A full list of the used public repositories and mailing list archives can be found in Appendix B.

As explained earlier other relevant data points are the software vulnerabilities associated with each version of the considered products as retrieved from the National Vulnerability Database (NVD) [3], the security advisories as published by the National Cyber Security Center (NCSC) [4] and the publicly available vulnerability exploits from ExploitDB [5] and Cybersecurity & Infrastructure Security Agency's (CISA) Known Exploited Vulnerabilities Catalog [6]. To collect these the following steps have been executed:

1. For each product the corresponding `cpe2.3` identifier was retrieved from the Shodan network scan records.
2. Using the collected `cpe2.3` identifiers the NVD was queried and the associated CVE identifiers and publication dates for each CPE have been saved. For the `OpenSSH` CVEs the approach as explained by West and Moore [28] was used, whenever a backport version was available. Using their approach, the CVEs for a main OpenSSH version were retrieved

[3] `https://nvd.nist.gov/vuln/search`
[4] `https://advisories.ncsc.nl/advisories`
[5] `https://www.exploit-db.com/`
[6] `https://www.cisa.gov/known-exploited-vulnerabilities-catalog`

and employing manual inspection of the change logs the CVE identifiers corresponding to backport fixes have been removed up to the current backport version.

3. Using the found CVEs the advisory database of the NCSC was queried for published security advisories. The ones related to the product at hand have been saved with their publication date and the severity attached to them.
4. Using the found CVEs in step 2, the ExploitDB and CISA's Known Exploited Vulnerabilities Catalog were searched for publicly disclosed vulnerability exploits. These have been saved, together with their publication date and where applicable whether they have been verified or not.

This data will be further used in the longitudinal part of this research as presented in chapter 6.

3.2. Approach

The research approach follows accordingly from the data above. We have summarised it in the following 5 general steps:

1. A List of municipality names per year has been created for the period 2001 - 2023.
2. The different network range datasets have been searched for municipality-related IP ranges. Those ranges have been verified against the previous point's data to see the coverage of the dataset.
3. The found IPs have been used to retrieve current scan data from April 2023 and the data has been analysed for popular services and trends.
4. The historical data for the IPs has been retrieved and analysed. Data points of vulnerability management have been extracted.
5. The final analysis aggregates the previous findings and draws conclusions on the overall vulnerability management of the selected municipalities and services.

Using this approach the research has been carried out in two main phases.

First, an exploratory study has been conducted to analyse the available services and their information on the current scan data. As well as to establish the presence of data points associated with vulnerability management. This process will be further explained in greater detail in chapter 4.

Second, a longitudinal analysis was conducted in order to retrieve the vulnerability management trends for a selected subset of municipalities and services. This will further be discussed in chapter 6.

An overarching, simplified view of the whole methodology is presented in Figure 3.1.

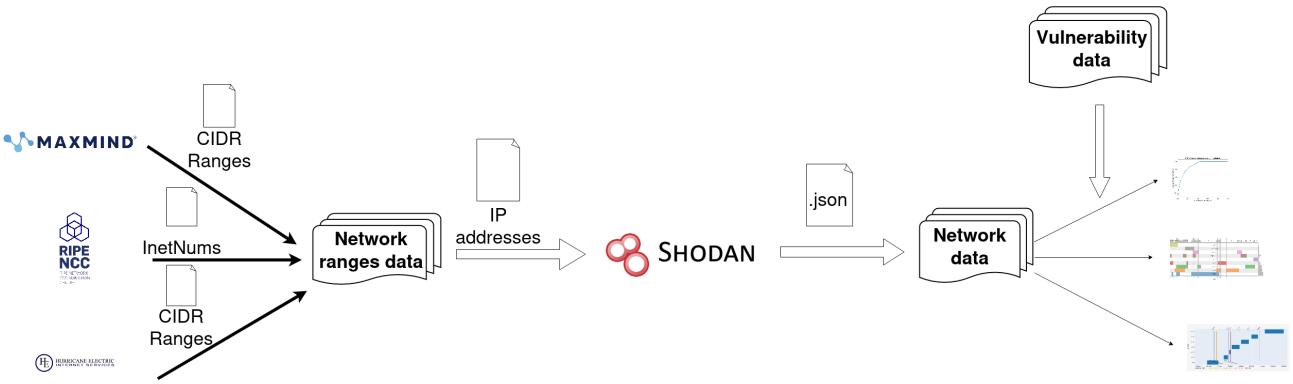


Figure 3.1: Methodology framework as executed during the research

As explained earlier, the vulnerability data collection has its own methodological flow. For ease of readability, this process is focused visualised in Figure 3.2.

In the chapters to follow, the use of this data will be explained in more detail and the performed analysis will be presented.

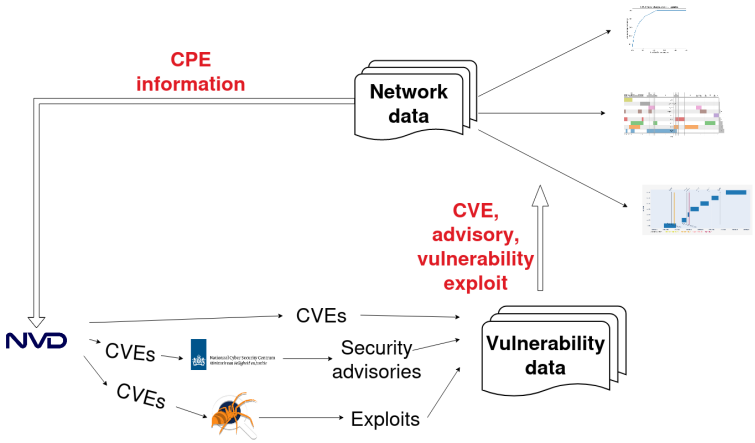


Figure 3.2: Methodology used for vulnerability data collection

3.3. Ethics

One of the main ethical considerations for this research is the decision to not perform the network scans ourselves. Both passive and active scans could put an unnecessary load on the IT systems of the municipalities and we opted for the already available data provided by scanning services. The services provide guidance on how anyone can ban their networks from the scanning procedures, thus allowing for an opt-out option from further research.

Another risk of such research is that the discovered trends could put the underlying organisation at greater risk of attacks by malicious actors. To overcome this issue, any discussed results would be anonymised and no real organisation's names or identifying information will be disclosed.

4

Exploratory Analysis

In order to perform longitudinal analysis, we need insight into the information available to us. To get a better understanding of the collected data, a step that we should refer to as “Exploratory Analysis" was conducted. In this chapter we explain the methodology followed in order to validate the correctness of the collected network ranges in section 4.2, followed by general visualisations and descriptive statistics based on these ranges in section 4.3. The steps to follow, which are performed on the network scan data are described in section 4.4 with their corresponding validations and visualisations.

4.1. Network ranges data collection

As explained in chapter 3, the three sources of IP range information have been RIPE NCC (RIPE), MaxMind, and Hurricane Electric. Each data source has provided a different number of network ranges associated with Dutch municipality networks. The biggest source has been RIPE with 4,455 `inetnum` ranges of IP addresses, followed by MaxMind with 4,253 CIDR network ranges, and finally Hurricane Electric with only 53 CIDR ranges. It is important to mark that the different source have differences in their time period coverage. The Hurricane Electric ranges are only a snapshot of early 2023, the MaxMind ones cover a time period between 2009 and 2023, and finally RIPE spans over 2001 and 2023. Another important difference is that depending on the network mask or the range in the InetNum, these ranges differ in the amount of IP addresses they posses. As such it is misleading to directly compare the number of ranges retrieved from the different sources, and instead we should consider the absolute number of IPs that are part of these ranges as shown in section 4.3.

4.2. Network ranges validation

The different datasets used, obtain their data points in different manners, thus requiring further checks to establish their validity.

In the case of RIPE, we believe that the data is truthful, because of RIPE NCC’s function as an IP registry. We do acknowledge the possible limitation of these ranges that lies in the fact that old ranges could remain registered past their lifetime, due to factors such as missed de-registration or no new allocation of the used IP addresses.

For the data collected from MaxMind and Hurricane Electric, a series of manual checks have been conducted to establish their validity. The setup of these checks is as follows:

1. The CIDR ranges and their network names have been collected.
2. A query has been performed using the CIDR range on the RIPE NCC historic whois database.

3. All entries in the historic whois database have been checked to establish the correctness of the network's name.

The RIPE historic whois database provides information on changes to the data for registered IP addresses. For our use case, the main information necessary is the network name and description associated with the records. If either of these entities provides information similar to the network name presented by MaxMind or Hurricane Electric, the corresponding range was deemed accurate. If the RIPE records present different information or no information that can be used for the verification this has been marked. This analysis step was performed on all CIDR ranges found by Hurricane Electric (53 ranges) and a subset of the MaxMind ones, all ranges dating from 2022 (22 ranges).

After the analysis of 22 ranges, the data retrieved from MaxMind contains 16 ranges that match with the RIPE records, 4 ranges for which RIPE fails to provide (enough) information for a concrete conclusion, and 2 ranges where the organisation to which the range belongs according to the RIPE records is different to what is stated in the name of the network as provided by MaxMind. The results for the network ranges found by Hurricane Electric show similar results. Out of the 53 ranges discovered, 46 ranges match with the records' information available to RIPE, 4 ranges have not (enough) information on RIPE in order to validate their accuracy, and 3 ranges have a mismatch between the information provided by Hurricane Electric and what is available on RIPE.

As it can be seen on average below 20% of ranges are mistakenly labelled by MaxMind and Hurricane Electric. For the purpose of this study, all found ranges are used in the consecutive steps of the analysis, since further filtering of municipality IP data is also performed in the steps that follow.

4.3. Network ranges visualisation

Once we have established the validity of the network ranges retrieved, we proceed to visualise the different characteristics of the network data from all the data sources.

4.3.1. IP address overlap

Firstly, we establish the overlap between the different data sources we use. That is done by comparing the IP addresses contained within the CIDR ranges for each data source and visualising it in a Venn diagram. In total we have discovered 383,831 IP addresses which have at one point in time been allocated to a Dutch municipality network. As it can be seen from Figure 4.1 the IPs found by Hurricane Electric are entirely present in either the RIPE or MaxMind found addresses. This can be explained by the fact that the Hurricane Electric dataset is the only one that encapsulates a single time-frame.

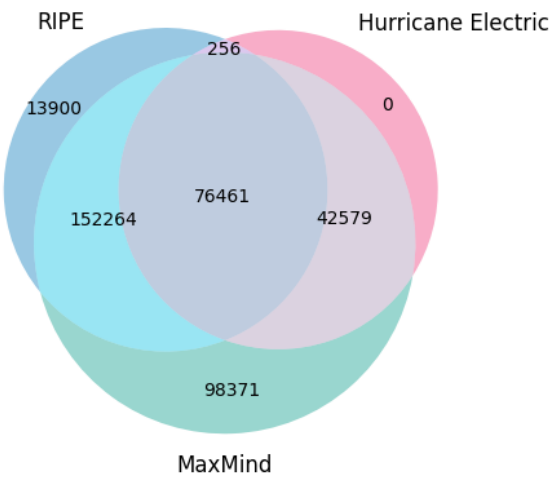


Figure 4.1: Venn diagram showing IP overlap between the 3 sources [1]

For a better representation of how alike the MaxMind and RIPE datasets are we computed the Jaccard index. This similarity coefficient is a numeric value between 0 and 1, which represents how similar two sets are, with values closer to 1 meaning bigger similarity. In this case, the Jaccard similarity coefficient is 0.596, revealing that although they contain some differences in ranges, RIPE and MaxMind are rather similar.

4.3.2. Overtime IP usage

Another insightful trend is the number of IPs evolving over the available time period. For this visualisation only the dataset of MaxMind is considered due to the flaws in the other two datasets. The data retrieved from Hurricane Electric presents a snapshot of the network ranges that we can only guarantee is correct for 2023. Because of that, this dataset will be excluded. In turn, the RIPE dataset could also not be considered a good representation due to the fact that it only captures the date of the final modification to the network ranges. We have no information about when the range was first used nor when it was terminated or replaced.

The distribution of IP addresses, overtime can be seen in Figure 4.2. From the histogram, it can be seen that the number of IP addresses allocated over the years has a steady trend with some fluctuations. The relatively lower number of IPs allocated in the years 2021 and 2022, could be explained by the fact that for those years the number of found municipality networks in general is much lower. This could be due to the renaming of networks to exclude the word “gemeente" in them, which would result in the inability of our query to find such results. It is also important to note, that not all allocated IP addresses end up being used. This will be further shown in the next section.

[1] NB: Diagram not to scale due to library restrictions

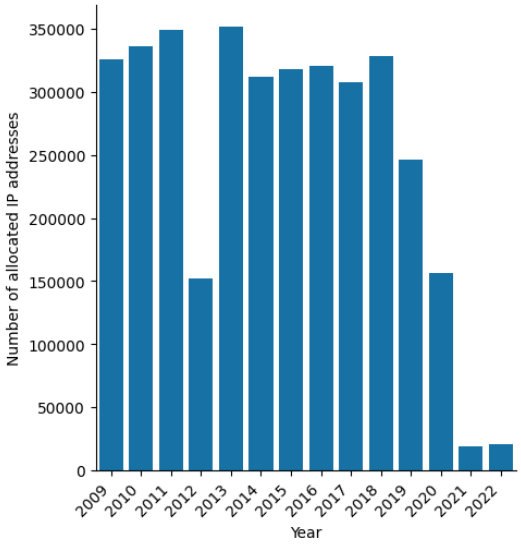


Figure 4.2: An yearly overview of the allocated IP addresses from MaxMind

Another interesting trend over time is the use of old names for municipalities that have been part of the yearly municipal restructuring. In these restructuring there are three main types of changes:

1. The name of the municipality changes but its territory and composition do not.
2. The municipality is acquired or merged with another (bigger) one, which results in a change of name and the ceasing of the existence of the old (smaller) municipality.
3. The municipality acquires another smaller municipality, which results in a change in territory and composition but no change of name is needed.

For the time period between 2001 and 2023, there have been 104 such restructurings [24]. It is expected that after such a restructuring, the old names of merged or re-named municipalities will no longer be in use. However, the data shows that this is not the case. Of the observed restructuring (31 fully present and 69 where at least one of the affected municipalities is present) there are 12 cases where the old name has been used after the year the change has taken place. The period in which the old name is still used for the IP networks varies between 2 and 10 years with an average of 5.5 years.

4.3.3. Network growth

A further point of exploration is the growth of the municipalities' networks over time. For the majority of municipalities, this growth is not always observable due to the small size of their networks and the scarcity of historical data available in order to make this observation. However, a good example could be found in the networks of one of the bigger municipalities such as Den Haag. In Figure 4.3 we can see the evolution of their IP space. Each colour in the endpoint represents a different year between 2001 and 2023 when a final network change has been done. These changes are directly related to the last modified date, retrieved from the RIPE records. This way of representing the network space uses a prefix graph and is a nice way to observe the connectivity and evolution of a given network.

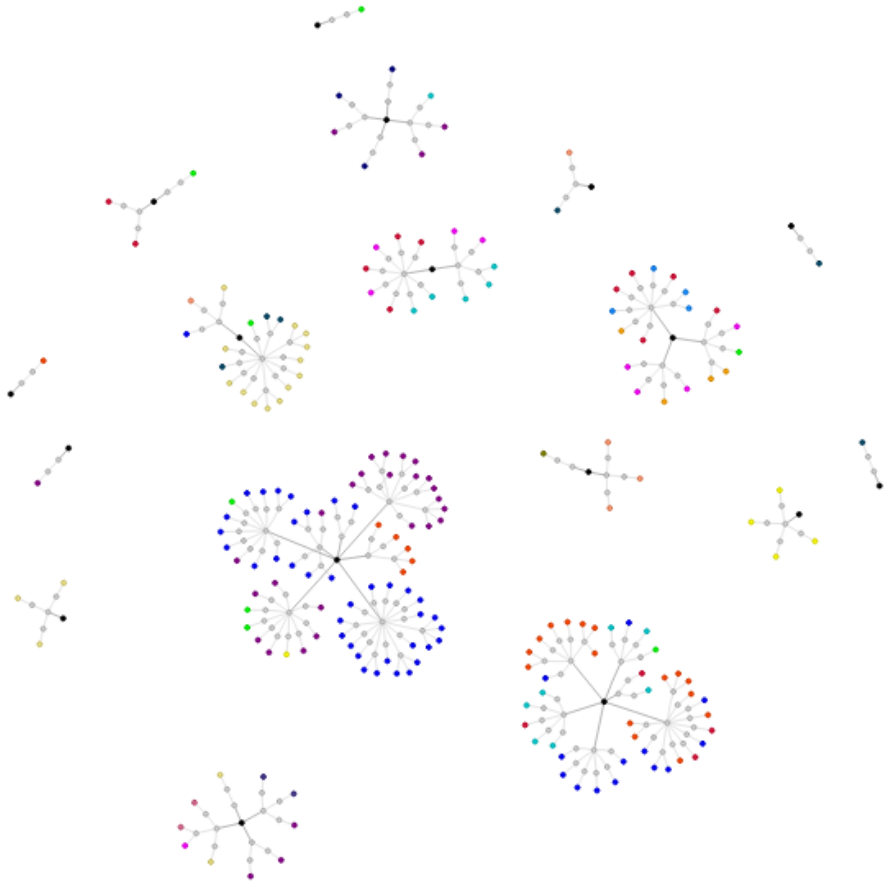


Figure 4.3: Network evolution graph municipality Den Haag

As we can see there are two main big graphs in this figure, which suggests that those are the two main networks of the municipality. Furthermore, we can derive that in 2015 (visualised by the blue end nodes) the municipality had a large expansion of their IT networks.

This however is a rather big municipality, hence the opportunity to observe the network changes and growth. For the majority of the dataset, these network graphs have rendered only small singular graphs with very few nodes. This approach suffers greatly from the lack of historical data and the incompleteness of some municipality network ranges. Because of that, it is not reliable to use the available network ranges data to estimate the network growth of the municipalities in this dataset.

4.4. Scan data analysis

As mentioned before, not all allocated IP addresses end up being used. In order to get a better picture of the accurate network topology, we should focus more on the hosts that have services online. In this study, the active IP addresses are approximated by the IP addresses observed by the scanning service Shodan. This approximation reveals a lower bound on the number of used IPs since there is a possibility that certain IPs have not been discovered by the network scrapers of Shodan.

In our data and the analysis to follow we distinguish between 3 distinct entities: Municipality name, IP address, and Shodan record. The associations between these 3 distinct entities are presented in UML form in Figure 4.4.

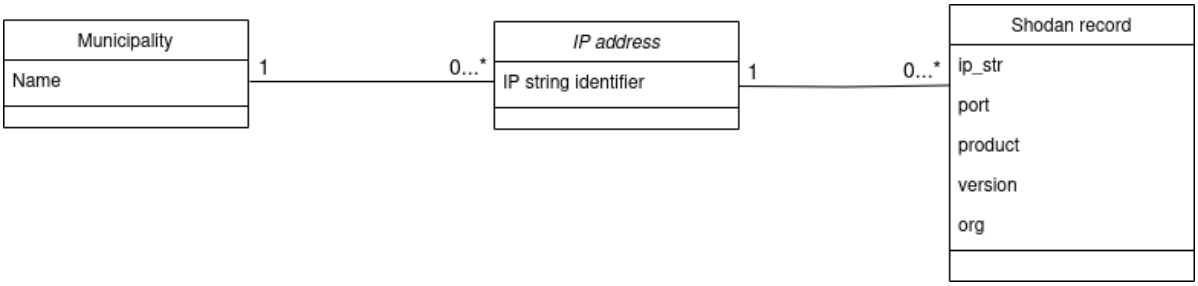


Figure 4.4: UML diagram of relations between the 3 entities

The relations between these three entities as represented in the diagram is as follows:

- A Shodan record is associated to 1 distinct IP address, which is further linked to 1 distinct municipality name.
- An IP address can have multiple Shodan records for its different open ports and dates on which the scan was performed.
- A municipality name can be connected to multiple IP addresses as part of their network assets.

An explanation of the data discovery and filtering process can be seen in Figure 4.5. The first two data points of “Found ranges" and “Found IP addresses" have already been described in the previous section.

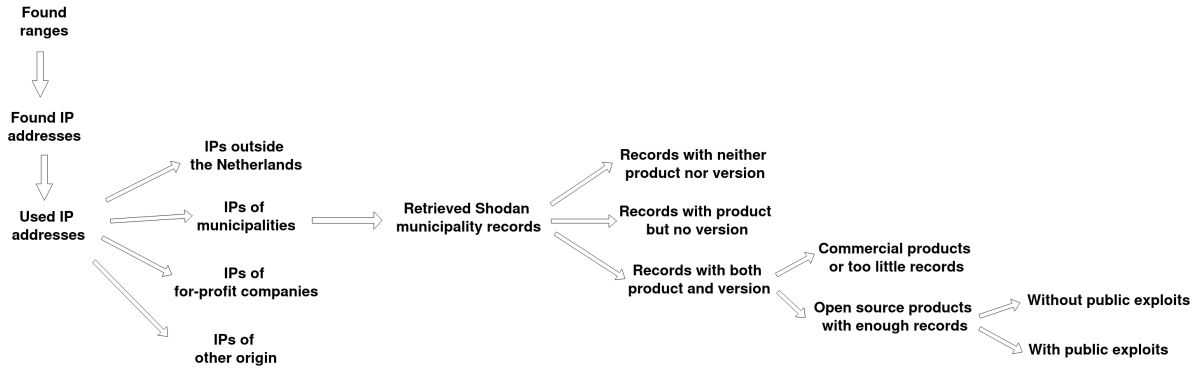


Figure 4.5: Path of data discovery and filtering

For this section, the Shodan results correspond to scanned network banner information retrieved from all the IPs found in the previous steps. These scans have been conducted around the month of April 2023. Using all collected IPs that have at one point in time been related to a municipality network, we receive currently active results for 6,354 unique IPs of 285 unique municipalities from Shodan. This shows that only a small subset of IP addresses are actively used. It is also visible that not all municipalities in the Netherlands are are represented in our dataset as the current number of municipalities is 342. In general municipalities have multiple IP addresses in their networks which we can analyse, explaining the bigger number of IPs compared to municipalities.

4.4.1. Data filtering

As network addresses could often change ownership, it is important to only continue our analysis on records that are still related to a municipality. To distinguish that a number of filters have been used.

Firstly, the records have been checked for their geographical location. The Shodan records contain tags with this information such as `country`$_{c}$*odeand*

As can be seen, from the records located in the Netherlands, roughly a bit more than half of the IPs are still associated with municipality networks. From the rest, more than 1 thousand IPs, the majority have transferred ownership to for-profit companies. For the IPs located outside of the Netherlands, the majority of them stay located within Europe and in close geographical proximity to the Netherlands. However, there are some addresses present that end up reallocated to entities in Brazil and the Islamic Republic of Iran.

4.4.2. Filtering validation

As it can be seen in ?? a considerable amount of the IP addresses, which have once been associated with a municipality range are currently owned by other entities. In order to validate these findings we have performed manual checks similar to those used for the network range validation. Here the following steps were taken:

1. A sample of 100 random IPs have been uniformly selected from all categories that no longer relate to a Dutch municipality (outside the Netherlands, for-profit company, and others).
2. A query has been performed using each IP address on the RIPE NCC historic whois database.
3. All entries in the historic whois database have been checked to establish the validity of the networks ownership.

Of all 100 IPs, 30 IPs have RIPE whois data available and the other 70 have no records. In this case, no records means that the IP address has not been registered with RIPE and no further ownership information can be derived from their database. From the 30 IPs with records, 23 have only one single record corresponding to a non-municipality organisation, 2 IPs have multiple records and all of them correspond to a non-municipality organisation, in 3 cases we can clearly see the transfer of the IP from a municipality network to a non-municipality one, 1 IP has only its old municipality record from 2017, and 1 IP corresponds to a municipality in Belgium. Although lacking a lot of information, this check shows that the classification made during our filtering step is largely valid (whenever information is available) and we can continue further with the analysis of the municipalities' IP addresses. Unfortunately, the available information is too sparse to make any useful conclusion on the ownership transfer timeline.

4.4.3. Services discovery

Focusing on the municipality-related IP addresses, the following step identifies which software services and products are mostly present on those networks. In order to do that, the records with valid product and version information have to be identified. This has been done by examining the `product` and `version` tags of the Shodan records. For both tags, we have considered two possibilities, the tag containing no information represented by a `nan` entry and the tag containing actual information about the service running on the host. For ease, we shall call the latter group records with `product/version` information. It is important to note that records related to the same IP address running the same non-`nan` product and version thereof on different ports have been considered to be one record. This is because it is quite common to run the same service on multiple ports, for example `http(s)`, where one of the ports is the non-secure port used only when a secure connection cannot be established. For `nan` products and versions no such distinction was made as we cannot know for sure the product and version

are the same. Using this distinctions and the fact that multiple products can be running on a single IP address but on different ports, we can see that one IP can be associated with multiple records. As such the number of records is greater than the number of found IP addresses. From all municipality IP addresses the distribution of values looks as follows:

- Total number of records: 6,755
- Records with `nan` product tag: 4,034
- Records with product information but `nan` version tag: 1,992
- Records with product and version information: 664

As we can see from the numbers, a lot of records do not have enough information for useful statistical analysis. From the difference between the total number of records and the sub-classified ones we also observe that 65 records with product and version information have the same product running on different ports on the same IP address. As mentioned before we merge these records into one and further consider it as a singular observation. Only 664 records provide full data on both what service and which version the host is running. It is those records that we further consider to analyse.

In the first step of analysis we investigates the different products present in those 664 records. Table 4.1 illustrates the number of records each product has and how many unique municipalities are running this service, for the top 12 most popular products. In all cases we observe that the number of unique municipalities running a given product is smaller than the number of associated records. This implies that municipalities run multiple services that use the same product on different IP addresses, for example.

Product	# Records	# Unique Municipalities
ntpd	178	64
Microsoft IIS httpd	125	68
Microsoft HTTPAPI httpd	100	44
DrayTek	63	34
OpenSSH	42	28
Apache	31	24
nginx	22	12
BGP	16	4
Dropbear sshd	15	11
lighttpd	9	8
Allegro RomPager	9	7
Jetty	6	5

Table 4.1: Distribution of records over top 12 most popular products

As it can be seen, unsurprisingly the most popular service is `ntpd` (Network Time Protocol daemon). One-third of the records have Microsoft server (IIS or HTTPAPI) active and 16 records have `BGP`. For the purpose of this study, those four products will not be considered for further analysis. The reasoning for `ntpd` and `BGP` is that those are well-established network protocols with fewer vulnerabilities. On the other hand, Microsoft products do not present an easy way to monitor version updates as they are commercial products. Finally, the `DaryTek` products have also been excluded, due to data abnormality later discovered in the historical data, where a considerable amount of products switch to a `DrayTek` router for a period of 1 to 2 days in September of 2022. This observation is assumed to be data poisoning and the router products have not been considered at all.

Using the rest of the products we can further observe the overlap between products used and municipalities. In Figure 4.6 we visualise that as a Super Venn diagram. One should read the figure as follows, each row with a name corresponds to a specific product. Each column corresponds to a subset of the observed municipalities, uniquely distinguished by the different products they are using displayed as the coloured row cells. The number of municipalities in each column is displayed on the bottom. In total there are 90 unique municipalities considered. On the right-hand side is the number of municipalities using this product. The number on the top shows how many products the subset of municipality(s) use.

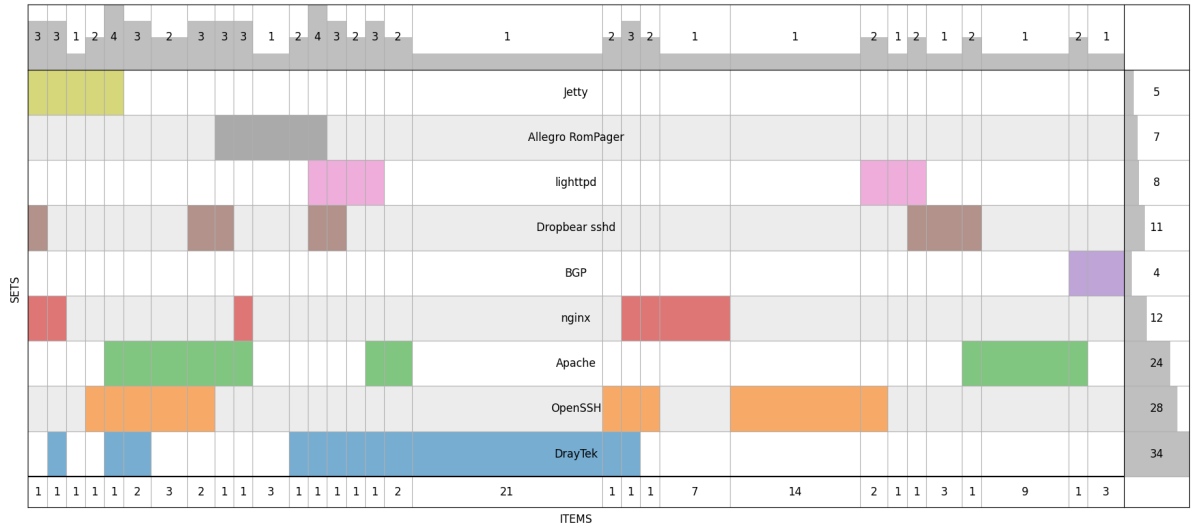


Figure 4.6: Super Venn diagram of products and municipalities

For example, we can see that 21 municipalities only have DrayTek as a product and 8 that have both OpenSSH and Apache. In general, we observe not a lot of overlap between different products and the corresponding municipalities, with the small exception of Apache and OpenSSH. This is easily explained, by the fact that most of those products are web servers and it is highly likely that municipalities only have one of those products on their networks.

5

Historic dataset

The descriptive statistics and visualisations from chapter 4 give us a good idea of the type and content of the data we have retrieved. However, looking at only the latest network scans does not provide any insights into the vulnerability management techniques that the municipalities employ. In order to observe those, we shall look into historic network scans.

This chapter explains the collected data in section 5.1, followed by some general analysis of the dataset in section 5.2. In order to give the reader a better understanding of the geographical coverage of the collected data in section 5.3 a brief explanation is provided. Following that, section 5.4 presents the findings of the product analysis using the historical data. Finally, we discuss the visible security best efforts in section 5.5.

5.1. Historic dataset description

As explained in earlier chapters, Shodan has the feature to request historic network scan data for a given IP address. This feature gives you all scans Shodan has performed for the given IP in the past and the network banner data it has retrieved. Using this we can retrieve information about the municipality networks from years ago, as long as Shodan has performed a scan. One important limitation of Shodan's historic scan is the presence of a retention period for newly discovered IPs. These IPs will only appear in the historic database 3 months after they have been first discovered by the Shodan scanners.

For the purpose of this study, this limitation does not pose any effects on the analysis to follow.

In May 2023, we performed the Shodan historic database query using all the discovered IP addresses from our network range discovery step. This data collection retrieved records corresponding to 15,652 unique IP addresses. In order to optimise the speed of the consecutive analysis, only the relevant Shodan tags have been saved as shown in Table 5.1. Using this we proceed to visualise and analyse the retrieved data.

Property Name	Type	Description	Required
`cpe`	`array of strings`	CPE information in the old, deprecated format	
`cpe23`	`array of strings`	CPE information in the 2.3 format	
`ip_str`	`string`	String representation of the IP address	
`org`	`string`	Name of the organisation that manages the IP	
`os`	`string`	Operating system	
`port`	`integer`	The port being scanned	Yes
`product`	`string`	Name of the software that powers the service	
`timestamp`	`string`	Date and time that the banner was collected in UTC time	Yes
`version`	`string`	The version of the product that powers the service	
`asn`	`string`	The ASN identifier the IP address is associated with	
`location`	`dictionary`	The location information of the service	

Table 5.1: Description of relevant Shodan banner tags for historic data analysis [1]

5.2. Longitudinal observations

As can be seen from the absolute number of unique IP addresses found in the scan data, there are more IPs present in the historic dataset compared to the scans of currently active networks. This shows that a big percentage of these historic IP addresses are no longer in use by the municipalities. To visualise this fact better, Figure 5.1 shows the unique IPs present in the historic network scans per year.

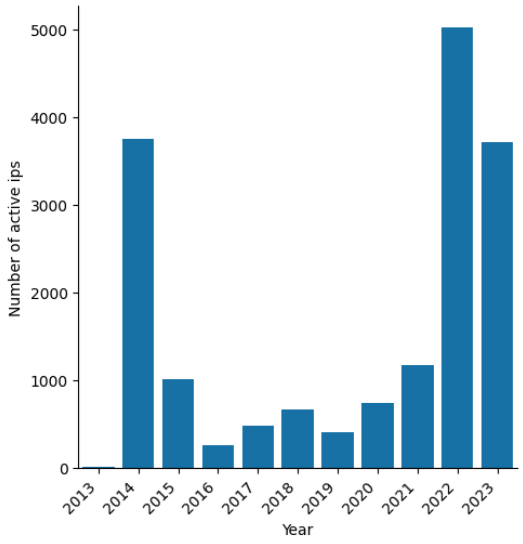


Figure 5.1: Histogram of unique IP addresses over the years

As can be seen the number of active IP addresses per year fluctuates, with two notable

[1] For the full list and description of tags visit: https://datapedia.shodan.io/

extremes. First, we can observe a sharp decrease in 2015 compared to the previous year. This could be explained by the mass migration of web services to cloud infrastructure around this period. Second, the prominent increase of active IPs in 2022 compared to the previous years. Without direct contact with the municipalities we cannot say for certain what caused this trend, however Figure 5.2 presents a histogram distribution of the most used ports in the year 2022. As we can see, the top ports in use are ones responsible for web services, thus this could mean that municipalities are increasingly conducting their work and providing their services online (e.g. submitting tax forms, registering on an address, etc.), however, at this stage, this is only speculation.

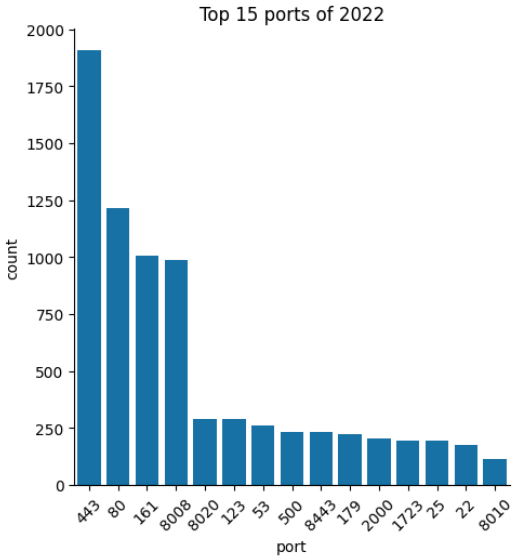


Figure 5.2: Histogram of most used ports in 2022

5.2.1. Active vs Inactive IP addresses

As mentioned before, some IP addresses are only used for a certain time in the past and are no longer active. Shodan itself makes this distinction whenever an IP does not respond to the scanner for 30 or more days. In this case, the IP address is no longer visible in the regular Shodan database and can only be accessed through the historic one. However, relying on the Shodan distinction of active vs. inactive addresses is not always exact. It is often seen that IP addresses “go down" for a period of time (usually a few months) and then come back online running the same services. As such this study’s distinction relies on the last date that the IP address was last seen by Shodan. If an IP’s last record on Shodan dates back to 2022 or previous years we classify this address as inactive, since it has been more than 5 months of inactivity. On the other hand, if an IP’s last record is from any time between January 2023 and May 2023 (the time the network scan data was retrieved) we believe that this address could still be functional. Using this classification, we proceed to show statistics on the lifetime of different IP addresses. In Figure 5.3 the distribution of the different lifetime durations in days separated based on the active vs. inactive classification is shown. One can observe that active IPs are also the ones that tend to be in use for longer periods of time. An inactive IP is on average online for 276 days compared to 358 days for still active addresses. These statistics show that a considerable number of IPs are only in use for short periods of time and are most likely no longer part of the core municipality network systems. Furthermore, when it comes to the lifetime of IP addresses, there are very few cases where systems stay online for more than 7 years, with a considerable preference for lifetime under 2 years.

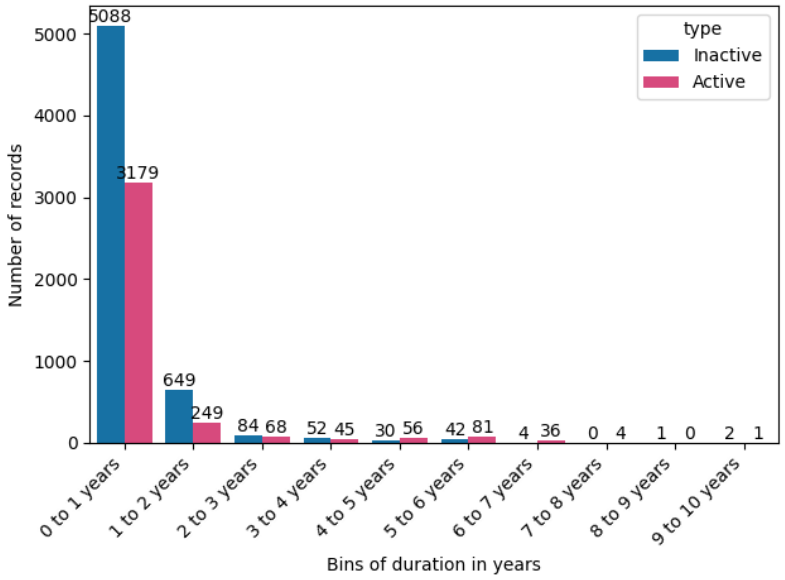


Figure 5.3: IP address' lifetime distribution

5.2.2. IP changes over the years

If we further take a look at the scan data and apply the same filtering techniques as in chapter 4 we would better observe the changes in ownership of IP addresses each year.

Figure 5.4 shows how the geographical location of the IP addresses changes each year. As it can be seen it is only in recent years, that certain IPs (previously) associated with municipality networks transfer their ownership to countries different from the Netherlands. In the observed cases this is due to reallocation of IP addresses, as none of the records with location outside of the Netherlands mentions Dutch municipality attributes in their network name of description.

However, just because an IP address is located in the Netherlands and was once associated with a municipality network, does not mean it stays that way, as we have seen in the previous chapter. Thus, it is interesting to see how many of the addresses stay as part of municipality assets. What can be derived from Figure 5.5 is that the majority of the IPs stay within municipal networks, and from the ones that do transfer, they tend to be assigned to for-profit companies.

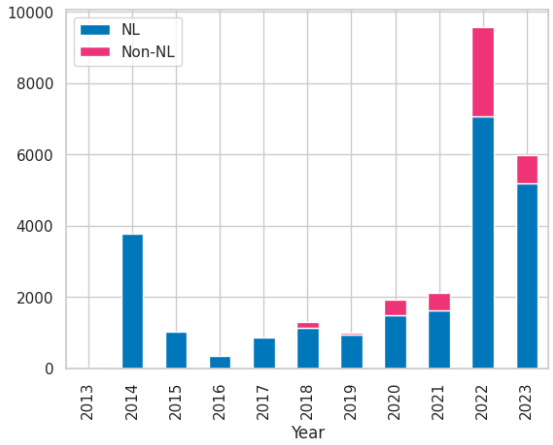


Figure 5.4: Distribution of geographical location over the years

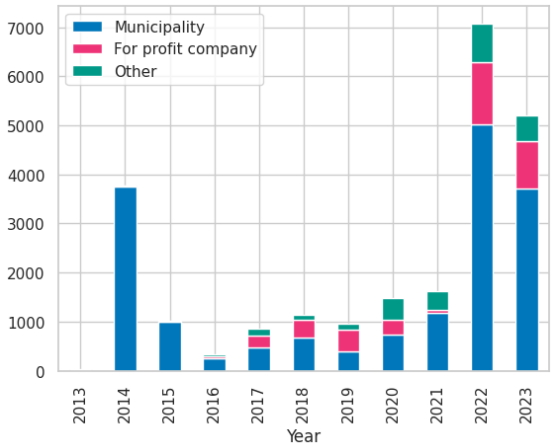


Figure 5.5: Distribution of organisation type over the years

The observations from these two graphs tell us that in recent years more and more IP addresses get reallocated to new networks, which no longer have any connection to a municipality. This shows that Dutch municipalities tend to use more often networks only for temporary purposes, “disposing" of them once they are no longer needed. This trend in itself could indicate the efforts put by municipalities to shrink their network attack surface, by disassociating from assets that are no longer needed.

Another interesting aspect of the changes in municipal networks is what proportion of the IP addresses are new, how many cease to be in use, and how many continue to be used for multiple years. In Figure 5.7 we observe that each year the proportion of new IPs (addresses that have not been used in the previous year) fluctuates. It is important to note that the numbers for 2023 are not final, as the year has not yet finished. An interesting observation is the fact that the amount of IPs that stay in use in-between two consecutive years (labelled “Old IN" in the figure and coloured in blue) has a steady increase. This can be explained by the fact that even though not static, municipality networks tend to stay consistent. This observation is also visible in Figure 5.6. There we observe the number of IPs that continue to be active in the next year, compared to the IPs that stop being active. Here again the final distribution of IPs for 2022 can only to be determined, once the year 2023 has passed. In the time period between 2016 and 2021, only a relatively small number of IP addresses ceased to be in use (coloured in red). To better understand the relation between the two figures it is useful to keep in mind that the IP addresses which are classified as “Continued" in Figure 5.6 are the same IPs which are classified as “Old IN" a year later in Figure 5.7. What is visible from combining the data from the two figures is that the number of IP addresses that stop being used in one year are never equal to the number of IP addresses considered as new the year after. With a few exceptions, the amount of new IPs tends to bigger than the amount of IPs which ceased to be in use the previous year.

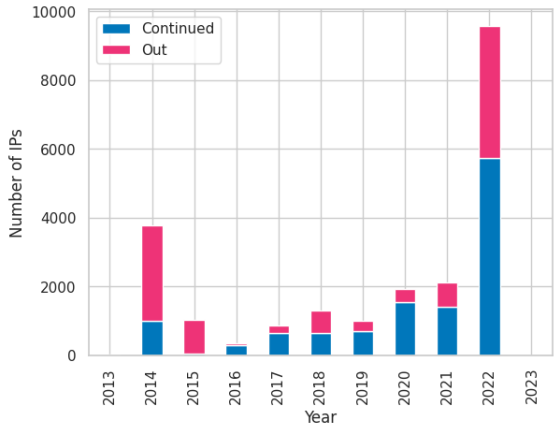


Figure 5.6: Distribution of the difference of IPs that stay active vs. become inactive over the years

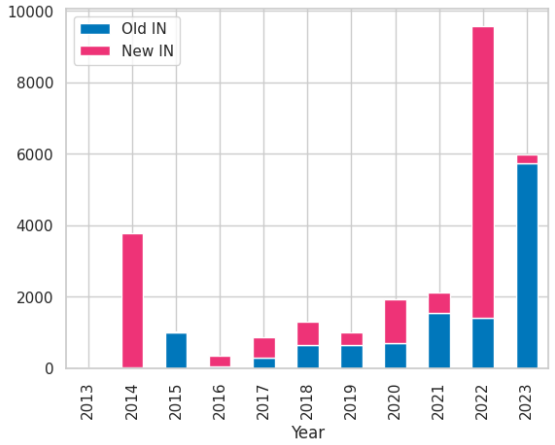


Figure 5.7: Distribution of the difference of new vs. old IPs over the years

5.3. IP address geographical coverage

For the rest of the longitudinal analysis, we shall focus on the records that correspond to municipality networks. As we mentioned before, in the historical network scan data we have entries corresponding to 15,652 unique IP addresses. Of those, 9,671 IPs belong to about 300 unique current and former municipalities, while the rest of the discovered IPs are no longer associated with a municipality network. For the time period between 2001 and 2023 in the

Netherlands there have been 567 unique municipalities, which means that the collected dataset has a coverage of 53% in this period. However, if we visualise the data we have on municipalities that are currently present, as shown in Figure 5.8, we can observe an increase in this percentage to 66. In the following figures, municipality areas coloured in red represent municipalities that are present in our dataset, whereas grey areas, are ones that we do not have the corresponding data for.

Unfortunately, the data gets even more scarce when we require more precise information. If we visualise the municipalities that provide service and version information, we see a sharp decline in coverage (Figure 5.9). These correspond to just about 45% of all current municipalities in the Netherlands.

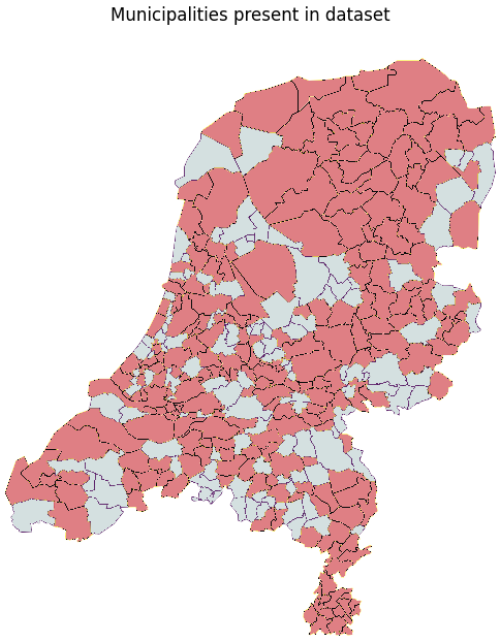


Figure 5.8: Geographical distribution of municipalities present in dataset

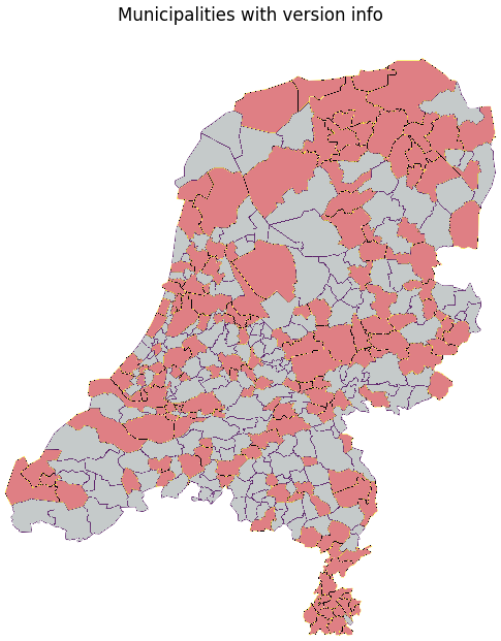


Figure 5.9: Geographical distribution of present municipalities with version information

5.4. Historic product analysis

Similar to the exploratory data analysis in chapter 4 we could look at what information about the product and version thereof is present in the historic network data scans retrieved from Shodan. Here again, the focus is on the IP addresses and their records that are related to a municipality network, as discovered in the previous sections. The Shodan tags for `product` and `version` have been examined and the same categories of disclosed information are used as in subsection 4.4.3. The distribution of records for the historic network scans looks as follows:

- Total number of records: 18,667
- Records with `nan` product tag: 14,006
- Records with product information but `nan` version tag: 3,047
- Records with product and version information: 1,614

Similar to before, the majority of entry points do not disclose any information about the service they are running or its version. From all records, only 1,614 are useful for further analysis.

Looking at the distribution of the top 12 most popular products in Table 5.2, the data once again shows similarity with the previous analysis. The main difference is in products like Apache and OpenSSH, which have been more used in the past among the municipality networks.

Product	# Records	# Unique Municipalities
ntpd	398	75
Microsoft IIS httpd	326	99
Microsoft HTTPAPI httpd	211	59
Apache	167	43
OpenSSH	99	46
DrayTek	75	35
nginx	48	21
Jetty	34	10
Dropbear sshd	28	16
BGP	24	5
Allegro RomPager	18	9
lighttpd	16	11

Table 5.2: Distribution of records over the top 12 most products

This validates the choice to focus on open-source products as they are not only relatively popular among the observed IP addresses, but also provide reliable information about software updates on a more granular level.

Taking into consideration the top 8 open source products we can once again visualise the overlap between different products that municipalities use. The Super Venn diagram as seen in Figure 5.10 represents this relation. As before, each row with a name corresponds to a specific product. Each column corresponds to a subset of the observed municipalities, uniquely distinguished by the different products they are using displayed as the coloured row cells. The number of municipalities in each column is displayed on the bottom. In total there are 92 unique municipalities considered. On the right-hand side is the number of municipalities using this product. The number on the top shows how many products the subset of municipality(s) use.

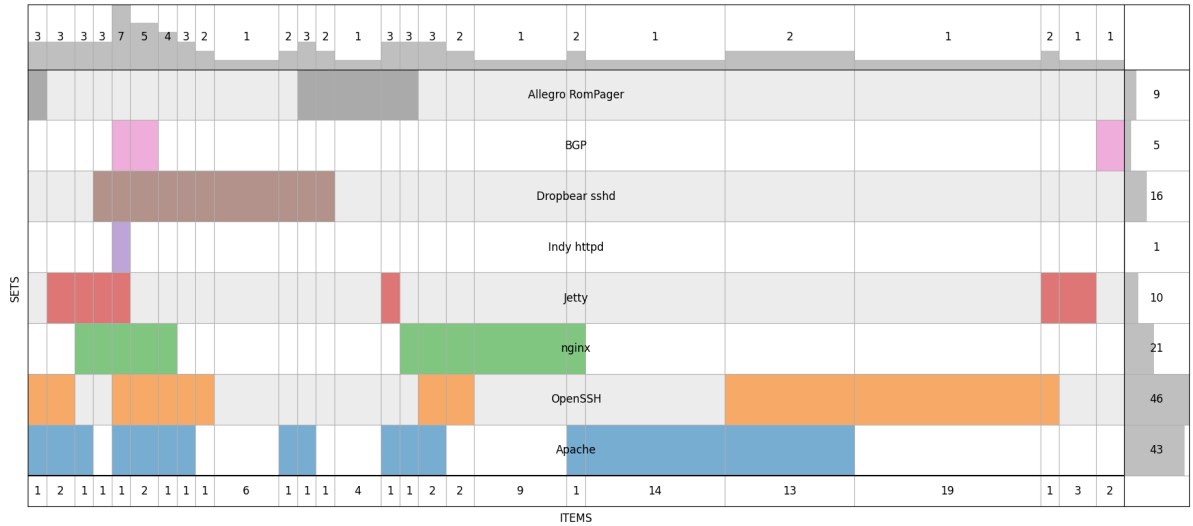


Figure 5.10: Super Venn diagram of products and municipalities using historical data

The observations we find show a considerable overlap between the municipalities running OpenSSH and Apache products. Nginx is also more popular among the historical set. When it comes to the rest of the services, the overlap visible is very minimal, supporting our previous findings, that municipalities prefer to use a singular web server product.

5.5. Security hygiene efforts

As seen from the statistics in the previous section, a big percentage of the available network scan records do not present information about the product and version thereof of the services they are running. This comes to show that some form of security hygiene is present on municipality networks. This means that system administrators put effort into obfuscating this information either by manually removing it or by having a security product that does that automatically. This could be seen as a step in lowering your security risk, as it poses difficulties for the adversary to retrieve this useful information easily.

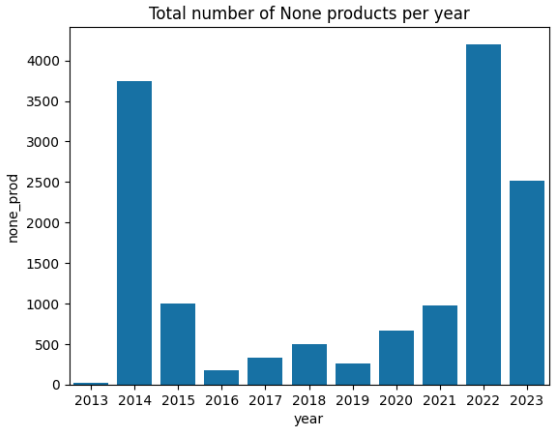


Figure 5.11: Number of entries with `nan` product tag over the years

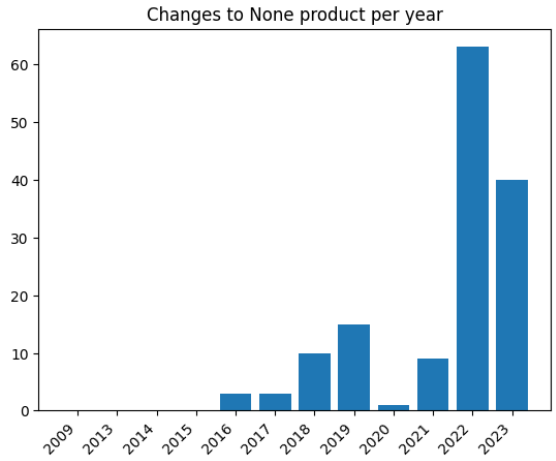


Figure 5.12: Number of entries that change to `nan` product tag over the years

In Figure 5.11 we observe that the number of records that have `nan` as their product tag value resembles the distribution of all entries over the years. This implies that it is standard practice for service information to be hidden from the network data. However, if we focus on the entries that did include actual information about their products, but later stripped that down, as shown in Figure 5.12, this trend has been increasing in the last years, meaning an active effort for better security hygiene is being made.

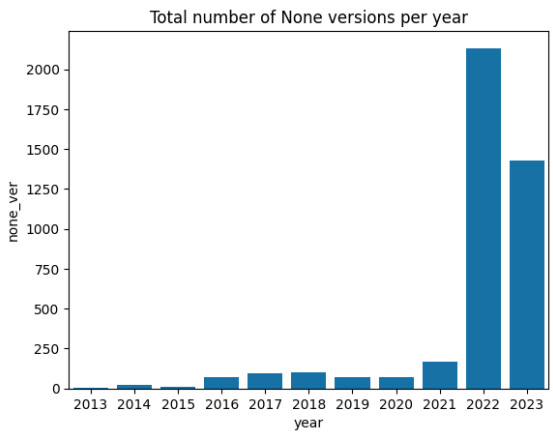


Figure 5.13: Number of entries with `nan` version tag over the years

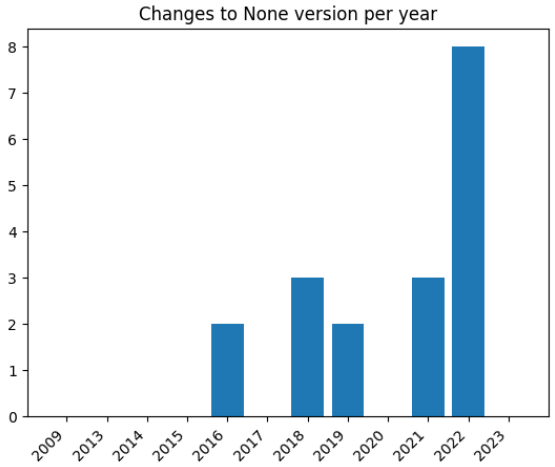


Figure 5.14: Number of entries that change to `nan` version tag over the years

Looking at the same statistics, but for the version Shodan tag, the graph looks different. Figure 5.13 shows that up to 2021 there have been very few records where product information was present but the version tag was `nan`. As of 2022, this has changed drastically and we can only imagine it will continue this way, by the already big number for 2023. A similar trend is also visible in Figure 5.14. There is an increase in the number of services that actively remove their version information from the network scan data. This once again shows the good efforts of municipalities to lower their exposure and remove or hide data that could easily be used by adversaries to facilitate future attacks.

6 Longitudinal Analysis

The next step of this study's analysis explores the longitudinal aspect of the research. In the previous chapters, we have seen some of the general trends in the data such as popular services and security hygiene efforts. In this chapter a more in-depth analysis is presented, involving observations of update strategies and vulnerability management.

We first explore some general statistics about the update data in section 6.1. After that, in section 6.2 an elaborate example of one IP address' vulnerability management is presented and explained. Following this we present the findings of the analysis of vulnerability management for the time period of 2021 to 2022 in section 6.3.

6.1. Descriptive analysis of update data

When talking about update behaviour and vulnerability management there are two ways we can measure that. The first is by observing the number of distinct product versions an IP has. With this observation any number greater than 1 shows that the IP address product has been updated. For example, if a given IP address running `Apache httpd` has the distinct versions of `2.4.52, 2.4.53, 2.4.54` in the records from Shodan, we can consider that this IP has been updated twice.

In Figure 6.1 the cumulative distribution function (CDF) of the percentage of observed version changes are compared to the percentage of municipalities that are responsible for those changes.

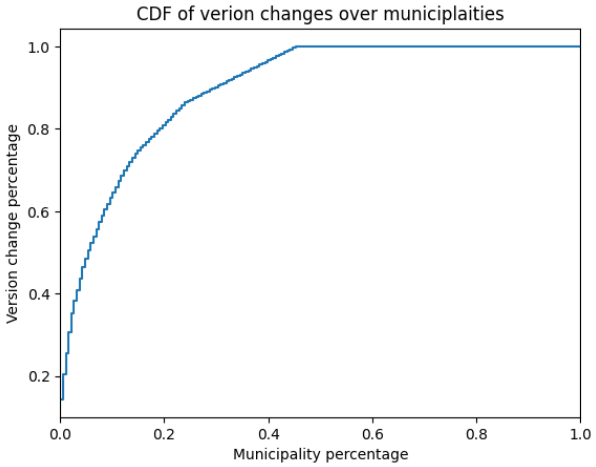


Figure 6.1: CDF of distinct version number over % of municipalities

What the figure reveals is twofold. Firstly, we see that approximately 20% of the municipalities present in the data are responsible for around 80% of the observed version changes. This shows that a small part of the present organisations are actively updating their services. Secondly, it is visible that 100% of the version changes are performed by approximately 45% of all municipalities. This means that more than half of the municipalities, for which we have version information, never show any updates on their systems.

However, it is important to note that solely using the number of version changes (updates) is not a good measurement of the vulnerability management of an organisation. It is possible that no updates have been observed for a service because the product is already running the latest version. Because of that, the second measurement we use is the number of days since the product's next version has been made available. In the rest of this report, we shall call this the outdatedness of a product measured in days or as it is often refereed as in literature the software age. To illustrate this better, if an IP address is running `Apache httpd` with version `2.4.53` on `20-09-2022`, while version `2.4.54` has been released on `08-06-2022` we say that this IP's outdatedness is 104 days. In other words, the outdatedness is the delta time in days between the date on which the next version was released and the last date on which the old version was running according to Shodan. One limitation of this approach is the fact that we do not have daily network scans from Shodan. The scanners complete a full crawl of the Internet every 7 to 10 days, meaning that we cannot precisely observe the date on which the software version has changed. Nevertheless, we can use this measurement as an approximation of when the update has taken place.

In order to use the above-explained technique, we have collected the software release dates for the most popular open-source products as explained in chapter 3 and chapter 5.

Using the software release dates and the records from Shodan about IP's product and version we can compute the average delay in days that it takes to update to a newer version. Table 6.1 displays these average delays for the considered products.

Product	Avergae delay
Plex	27
MiniServ	32
nginx	187
Jetty	232
OpenSSH	459
Apache	530
ProFTPD	2280

Table 6.1: Avg. delay in days until the version is updated

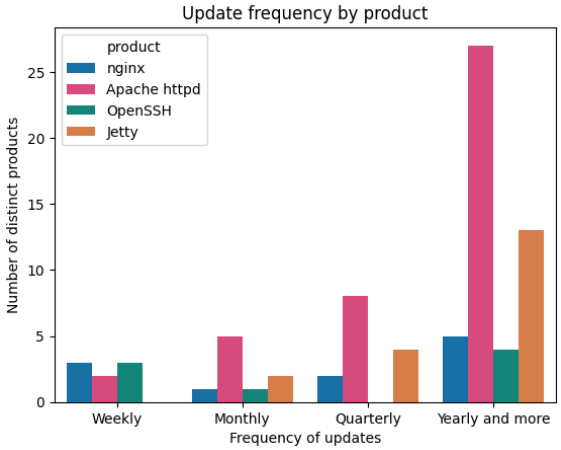


Figure 6.2: Distribution of update frequency by product

It is important to note that the data from `Plex, MiniServ, ProFTPD` is very scarce and based on singular observations. Thus, these numbers cannot be used for generalised statistics. From the rest, we can see that the more popular services like `Apache` and `OpenSSH` do suffer from larger update delays. Combining these delays of sometimes more than a year and the fact that these products release multiple versions each year, we can expect that a lot of systems seriously lag behind the most up-to-date versions of their software products.

If we further look into the distribution of the update delays over four distinct groups: "Weekly", "Monthly", "Quarterly", and "Yearly and more" we can better observe how often

different products get updated. What can be seen is that `Apache` and `Jetty` services are more likely to suffer from update delays in the magnitude of year(s), whereas `OpenSSH` and `nginx` seem more evenly distributed between all update frequencies.

To understand the trends in the absolute outdatedness in days the CDF can be constructed as shown in Figure 6.3. This CDF looks at the different software ages per product in our data set. As it can be seen, products such as `nginx` and `Jetty` tend to be kept more up-to-date and and when outdated the days since the available update are fewer compared to the more popular products such as `Apache` and `OpenSSH`. This graph is in accordance with the finding by West and Moore[28] and Demir et al. [6] and shows that roughly about 10% of all IP addresses with version information are running the latest version. Further, more than half of the IPs have products that are out of date for 3 or more years. There are some IPs that run software as old as 13 years. From this, we can conclude that municipality networks often run outdated software.

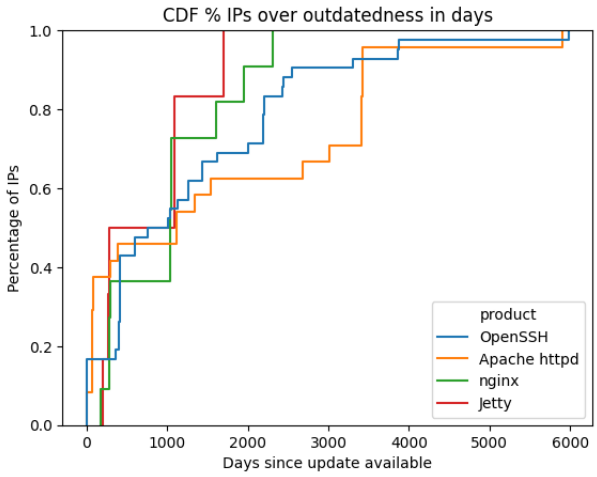


Figure 6.3: CDF of outdatedness in days over % of all IP addresses

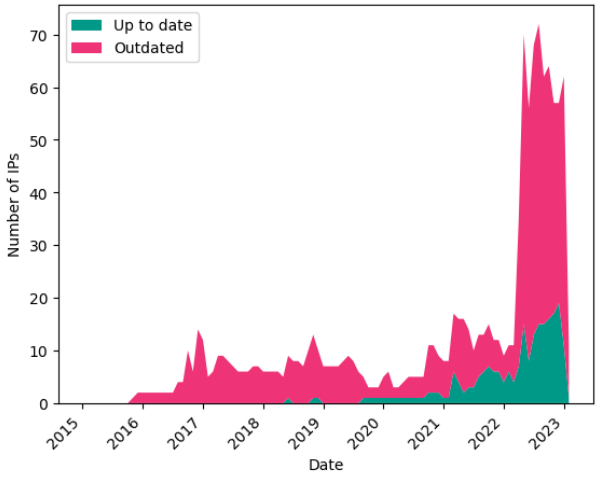


Figure 6.4: Distribution of outdated vs up-to-date products over the years

When a software product suffers from a considerable software age, one could wonder if there are other consequences apart from the security of the system. One such consequence is the inability to update a given outdated product because the version which is running is no longer supported and has reached its so called “End of Life" (EOL) state. Such systems no longer receive updates, including security ones. If we analyse the outdated products in our dataset we observe that 19% of the current systems run software versions that have reached their “End of Life" state. These systems can no longer be trivially updated to a supported version leaving them not only vulnerable to security threads, but also potential usability and compatibility issues. For the product of `Jetty`, versions with prefix `9.4.X` are expected to reach their EOL by the year 2025 and have now entered the end of community support. Past the year 2025 these products will no longer be updated and no longer receive security updates. This future development will affect 7% of the analysed current systems. This finding show that the considerable outdatedness of the municipal systems has resulted in almost “legacy" status of some products severely hindering their security.

We can further visualise the number of outdated products compared to that of up-to-date systems as seen in Figure 6.4. As can be seen, over the years the portion of services running an outdated open-source product has always been bigger than that of up-to-date ones. An interesting trend is the small decrease in outdated products towards the end of 2019. Around that period numerous security advisories from the NCSC have been issued with a severity score

of `high - high`, among others for products such as Microsoft [1], Intel CPU [2], and Citrix [3]. It is at the same time that the number of up-to-date products starts to gradually increase. This increase shows an effort to update systems to the newest software version, recognising the security implications of outdated products. However, as the figure shows in the last 2 years the amount of IP addresses running outdated products is rapidly increasing and doing so in a faster manner than the systems that are kept up-to-date. One possible explanation is the rapid increase of the number of IPs part of municipality networks as seen before coupled with the already lacking vigour of updating online systems. If these trends continue, this could have a snowball effect, leaving systems more vulnerable to adversaries.

Furthermore, some time should be spent on trends visible in the IP addresses that run outdated and up-to-date software respectively. From the 10% of IP addresses that have their software up-to-date, half of these IPs come from small networks of up to 5 IP addresses in size (this is measured by the unique number of IP addresses in the Shodan dataset, corresponding to the same municipality name) and have been active for up to 1 year. Unfortunately, once again the total number of data points is scarce, around 10, making this a descriptive statistic about our specific dataset and not necessarily reflecting the population behaviour of all municipalities.

The rest of the data points, corresponding to outdated systems are visualised in Figure 6.5 and Figure 6.6. In the first figure, we can see how the different IPs relate between their outdatedness in days and the total lifetime they have been active (again in days). The data points are randomly distributed along the different software ages for each duration, showing little correlation between these two attributes.

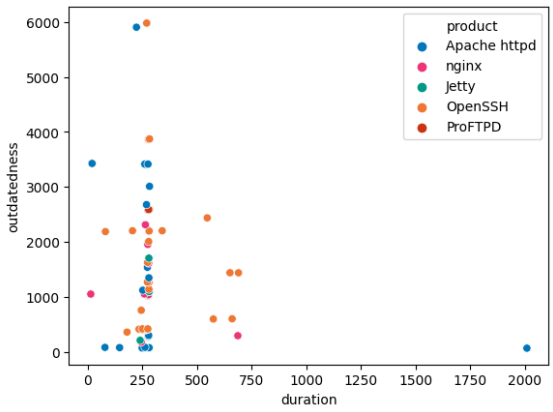


Figure 6.5: Distribution of outdatedness in days over the lifetime of the IP address

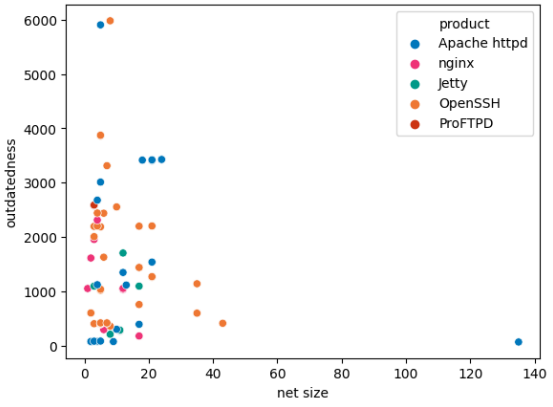


Figure 6.6: Distribution of outdatedness in days over the corresponding network's size

In the second plot, the outdatedness of IPs is compared to the size of the network each IP comes from. Here again, the points are mostly randomly distributed, especially for networks of size smaller than 30. In the bigger networks, we do see lower software age, but this could result from the under-representation of this category group.

In both plots, it is visible that only services running `OpenSSH` and `Apache` relate to outdatedness of over 3,000 days, which is in accordance with the statistics presented before. From these graph representations, we can see that there is no statistical significance of either the lifetime of an IP address or the size of the network it belongs to, which influences how outdated the service of the IP is.

If we go beyond the software's age and look at the amount and severity of associated

[1] `https://advisories.ncsc.nl/advisory?id=NCSC-2019-0381`
[2] `https://advisories.ncsc.nl/advisory?id=NCSC-2019-0380`
[3] `https://advisories.ncsc.nl/advisory?id=NCSC-2019-0979`

vulnerabilities and their exploitability we see interesting results. On average products have 22 unique CVEs associated with them, with an average of 2 having a CVSS score of `low`, 11 of `medium`, 6 of `high`, and 3 of `critical`. From all considered current entries, 62% of the systems run a product version which has a publicly disclosed exploit (verified or not) for at least one of its vulnerabilities. These findings are in accordance with previous work by Demir et al. [6] about the exploitability of web applications. If we only consider verified exploits, this number drops to 27%. Unfortunately the authors of that paper do not disclose the methodology they followed to collect and measure the existence of a vulnerability exploit and we tend to believe that they have considered any sort of published exploit as these numbers are closer to their findings of 60%. In reality, these numbers can be even higher, as not all found vulnerability exploits become publicly disclosed and a considerable amount of exploits are sold on underground marketplaces. In fact, previous work by Allodi and Massacci [1] suggests that vulnerability prioritisation techniques should at the bare minimum consider publicly disclosed vulnerability exploits, and for optimised results also take into consideration the exploits circulated on the dark markets.

When considering the CVSS score for those systems that have public exploits we see a particular increase in the number of critically scored CVEs. Exploitable systems have on average 37 CVEs of which 2 have a CVSS score of `low`, 20 of `medium`, 9 of `high`, and 6 of `critical`. This is a twofold increase compared to the average results presented earlier.

6.2. The case study of a ``good" municipality

To put things in perspective, we can look at a more granular example of a single IP address. To facilitate the further longitudinal analysis a few extra supplementary data points have been collected as explained in chapter 3.

In this case, we are observing an IP corresponding to a municipality network we shall call “Municipality A". The particular IP address has been active for a bit more than 2 years and belongs to a small network of just 4 addresses as observed in the Shodan dataset. The physical municipality using this network is a medium-large one as defined in Table 6.2, that has been formed after a merger between a few smaller municipalities. The specific IP address in this case study is running a web service using `Apache httpd` as the product.

Municipality classification	Population size range
Large	300,000+
Medium-Large	100,000 to 300,000
Medium	50,000 to 100,000
Small-Medium	20,000 to 50,000
Small	$<$ 20,000

Table 6.2: Municipality size distinction

In Figure 6.7 the timeline representation of the vulnerability management of “Municipality A" is shown. In this example, we are observing the time period of 2021 and 2022. What is visible in the timeline plot is the different periods in which the different versions of the `Apache` software were in use seen as dark blue horizontal bars, the release dates of the next software version as illustrated by the black vertical lines on the corresponding date, and the NCSC’s security advisories relevant for the given version of the product represented by a vertical line in the colour relevant to the advisory’s severity score on the corresponding date of publication. In this study, we assume that the NCSC security advisories are immediately received by the municipalities on the date they have been published online. In reality, this time can vary anywhere between immediate and 10 days of delay, but for the ease of analysis, we make this assumption. The severity score of an advisory consists of two parts `opportunity` and `damage`,

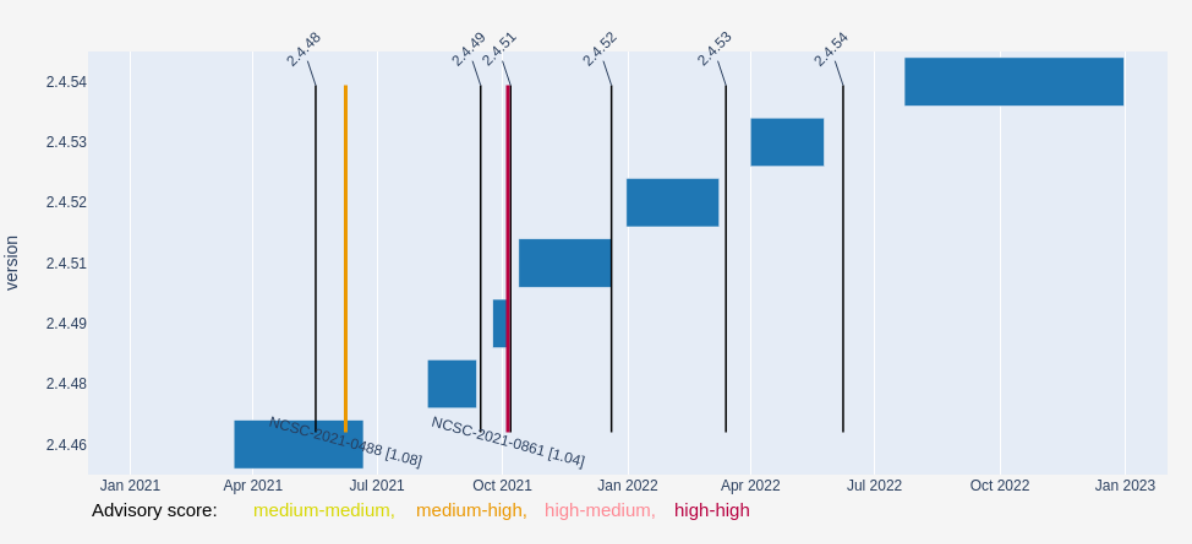


Figure 6.7: Vulnerability management timeline “Municipality A"

where the opportunity is the metric measuring how easily a vulnerability can be abused, and the damage measures what level of consequences and damage the abused vulnerability can pose. Both metrics can have a value of low, medium, or high. As we can see between two consecutive versions of the plot there is a varying period of missing data. This is an artefact of the IP address going offline and not being available to the Shodan scanners.

As it can be seen this IP address has had 7 different versions of the `Apache httpd` software in the given time interval. The first observed version is `2.4.46` which has been in use between the end of March 2021 and the end of June 2021. In this sub-time period, we also observe two external events. First, in May 2021 a newer version of the product was released (version `2.4.48`), visualised by a black vertical line and annotated with the version's ID on top. Second, in June 2021 the NCSC published a security advisory addressing a vulnerability in the currently running system's version. The advisory is given a score of `medium - high` as indicated by the orange colour of the vertical line which has the advisory's ID on the bottom. As we can see a few days later the service is no longer online.

In this particular case study, it is visible that the system is always running the newest version of the product with the exception of a small period in May and June of 2021. This shows the preventative vulnerability management behaviour of the system's administrator, as updating your system whenever a new release is available is considered a good security practice. It is also good to mention the difference between the delay of the update after the two security advisories. It is visible that the advisory of October 2021, which has a severity score of `high - high`, has triggered an almost immediate reaction of taking the IP offline and performing a software update. In contrast, the advisory of June 2021 with the severity of `medium - high` has seen a larger waiting period before the service has gone offline.

Another way to visualise this timeline is to compare it to the time periods (if any) during which the IP address could have been exploited. As it's seen from research, not all vulnerabilities correspond to an available exploit and not all exploits are developed in the same time frame [14]. As such it is important to evaluate the threat of exploitability associated to municipal networks. To do that we compare the CVEs each version possesses to the publicly disclosed vulnerability exploits as gathered earlier. We consider an IP address to be exploitable if, for a time period after the exploit's publication date, the vulnerable product version is still in use. We further distinguish between “Verified Exploits" and “Non-Verified Exploits", the former

having the `verified` tag present in the ExploitDB database.

For our “Municipality A" IP address, this timeline comparison plot is presented in Figure 6.8.

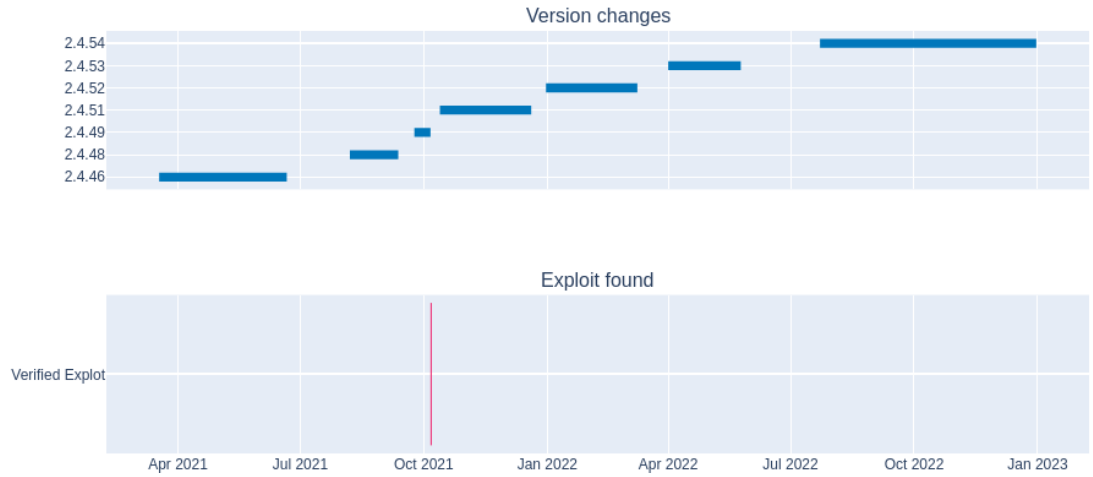


Figure 6.8: Vulnerability management timeline “Municipality A" compared to exploit timeline

As can be seen, in the two-year time period we are analysing, the IP address has only been exploitable for a few days in October 2021. This exploitability is related to the same CVE for which the NCSC has published the `high - high` security advisory. Because of the quick reaction after this advisory as described before, the service has mitigated long-term exploitability exposure. This comes to show that measuring vulnerability management only by using the raw number of present CVEs, could grossly overestimate the security landscape. In this case, just looking at the number of associated CVEs would have given us an average of 13 CVEs at any given time, but in reality, the IP address has only been vulnerable to a publicly available vulnerability exploit for those few days in October 2021. This statement, however, should not be an incentive to postpone security updates that patch known software vulnerabilities.

6.3. The case study of 2021 and 2022

For the longitudinal vulnerability management analysis we shall look at a representative time period where enough data points are present. If we look at the absolute number of version changes per year for the popular open source products as displayed in Figure 6.9, we can see that in 2021 and 2022 this number is the highest. As such in this section, we shall proceed to perform the analysis using this time period.

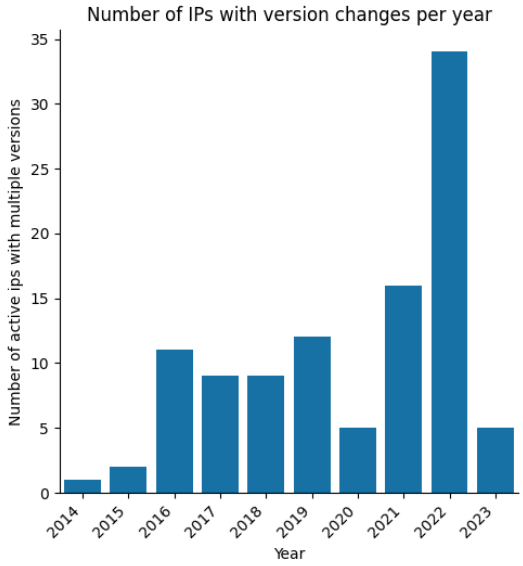


Figure 6.9: Number of version changes per year

The initial selection criteria for IP addresses to be considered for this longitudinal analysis included the following:

1. The IP should have been active before 1st January 2021
2. The IP should have been active after 31st December 2022
3. The IP should have had update behaviour related to at least one of the most popular open source products (multiple unique versions in the Shodan `version` tag)

Looking at all IPs that satisfy the above criteria renders a total of 3 unique IP addresses. This is less than satisfactory for a longitudinal analysis and comparison. Moving forward, without the loss of generality, we have substituted requirements 1 and 2 with the following:

1. The IP should be active in the time period of 2021 and 2022.

This new rule increases our data point size to 31. We could safely do this substitution of rules, due to our previous findings, showing that the lifetime of an IP address has no significant influence on the software age of the underlying product.

The analysed 30 IP address-product combinations belong to 19 unique municipalities in the Netherlands. Within those, there is a variety between size, merged ones, and geographical location. Table 6.3 represents an anonymised description of the considered municipalities.

As it is visible, most of the IP addresses come from small networks and have been active for a year or less. When it comes to the physical municipality characteristics, the groups of “Large" and “Small" as defined in Table 6.2 are not represented in this dataset. Furthermore, we have a few representatives of the so-called “merged" municipalities. These municipalities have been formed by combining the administrative forces of two or more smaller organisations.

The products these municipalities run range from the popular `Apache httpd` web servers and `OpenSSH`, to less popular open source products such as `Plex` and `ProFTPD`. When it comes to the average delay of applying software updates there are huge differences between IP addresses and products, ranging from 3 days all the way to 2280 days. If we compare the average delay in updating between products in Table 6.4, we see that the statistics for the

Municipality identifier	Municipality description	Size of network	Lifetime of IP
Municipality A	Medium-large, merged	≤ 10 IPs	2+ years
Municipality B	Medium-large	≤ 10 IPs	≤ 1 year
Municipality C	Medium-large	51 to 100 IPs	5+, 3+ years
Municipality D	Medium-large	11 to 50 IPs	4+ years
Municipality E	Medium	100+ IPs	9+, 2+, 5+, 1+ years
Municipality F	Medium	11 to 50 IPs	1+ years
Municipality G	Small-medium	≤ 10 IPs	≤ 1 year
Municipality H	Medium, merged	≤ 10 IPs	≤ 1 year
Municipality I	Medium	≤ 10 IPs	≤ 1 year
Municipality J	Small-medium	≤ 10 IPs	2+ years
Municipality K	Medium	≤ 10 IPs	≤ 1 year
Municipality L	Medium, merged	11 to 50 IPs	≤ 1 year
Municipality M	Medium-large	11 to 50 IPs	≤ 1, ≤ 1, ≤ 1 year
Municipality N	Small-medium	≤ 10 IPs	≤ 1 year
Municipality O	Medium-large	11 to 50 IPs	≤ 1 year
Municipality P	Small-medium	≤ 10 IPs	≤ 1 year
Municipality Q	Small-medium, merged	11 to 50 IPs	≤ 1, ≤ 1 year
Municipality R	Small-medium	≤ 10 IPs	≤ 1, ≤ 1 year
Municipality S	Medium-large	≤ 10 IPs	≤ 1 year

Table 6.3: Description of considered municipalities in the time period 2021 and 2022

selected municipalities are similar to those presented before corresponding to all data records.

Product	Average delay
Plex	27
MiniServ	32
nginx	52
Jetty	68
Apache	173
OpenSSH	522
ProFTPD	2280

Table 6.4: Avg. delay in days until the version is updated

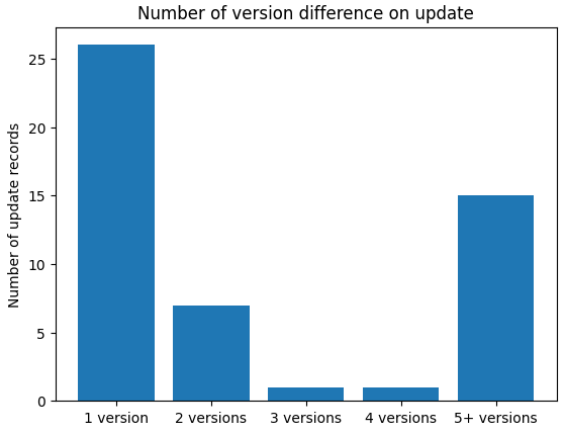


Figure 6.10: Version differences of observed records

The only two main differences are that in the time period of 2021 to 2022, `Apache` products take on average less time to update compared to `OpenSSH`, which was the other way around when analysing the whole dataset, and the fact that `OpenSSH` products, in general, take more days on average in this shorter period compared to before. This means, that some of the municipalities, which take very long to update their `OpenSSH` products are present in our subset of selected organisations for longitudinal analysis.

We can also look at the number of version differences organisations update to. This we define as the number of unique versions between the current version of the product and the updated version. For example, if an `Apache` product goes from version `2.4.52` to versions `2.4.53`, this will count as 1 version difference, whereas if the product goes from versions `2.4.52` to versions `2.4.57` this will be 5 versions difference. In Figure 6.10 these version differences

are presented. It is important to note the limitation here that the data from Shodan is on a weekly basis, meaning that updates cannot be observed if they happen between two network scans. As such the group of 2 version differences could in practice be smaller than currently presented. Nevertheless, the data shows that municipalities usually update their software on a small granular level, by applying singular version changes the most. The second most popular category however is that of updating to 5 or more versions at once. This could explain the long delays of updating as observed previously, and is a trend that could snowball out of control if a product could no longer be updated because the organisation has missed too many smaller updates.

Another way to look at the update delay time is from the perspective of the organisations. When grouped by their “Municipality description" size we observe that there are considerable differences in how many days the organisation wait on average before updating their products. As seen in Table 6.5 the fastest to update are organisation from the category of `Medium - large`. It is important to note that the considerably larger wait time for the organisations of `small - medium` size is highly influenced by the fact that 2 out of the 3 huge outliers in update waits fall under this category. If those where not considered it is in fact this class of municipalities that update most frequently with an average delay of just 49 days. These results suggest that the size of the municipality, has an influence on how fast updates are applied, with medium sized organisation performing on avergae the worst.

Municipality size	Avg. delay
Small-medium	611
Medium	272
Medium-large	113

Table 6.5: Avg. delay in days until the version is updated per organisation type

To observe these updates, the timeline plots have been created as seen in the previous section. Here we will discuss a few to support our analysis and the full list of timeline visualisations can be found in Appendix C. We have already seen an example of good vulnerability management by regular software updates in section 6.2. Unfortunately, this is a rare observation in the dataset and greatly depends on the product and the organisation that is running it. What is visible is that different municipalities express different update behaviours. Some show efforts to keep their systems up-to-date, whereas others lack in this aspect. Within the same municipality, there are also observable differences. Sometimes the update patterns differ between different products, but also there are differences within the same product on different IP addresses. In fact, only Municipalities M and Q show identical intra-organisational behaviour within their systems that run the same product.

An example of differences within a municipality with different products is shown in Figure 6.11 and Figure 6.12. In this case, it is visible that the municipality has updated its `OpenSSH` system within days of the update release. On the other hand, the same municipality exhibits considerable delays when it comes to their `Apache` product. As we can see in the time period between January 2022 and July 2022, the 2 other used versions of the product were made available, as well as 2 security advisories published by the NCSC concerning the then-running version. Neither of those 4 events seems to have an effect on the system vulnerability management at the time. It is also important to note that there have been a further 4 version of `Apache` that are not visualised in the figure as they have completely been ignored by the municipality. As such the choice for which version to update to seems as good as random from an outsider perspective and is most likely related to internal dependencies. This observation could also be explained by acknowledging that `OpenSSH` is a secure networking product and as

such could have a higher priority when it comes to software updates. This behaviour makes it hard to generalise the vulnerability management of a municipality as a whole.

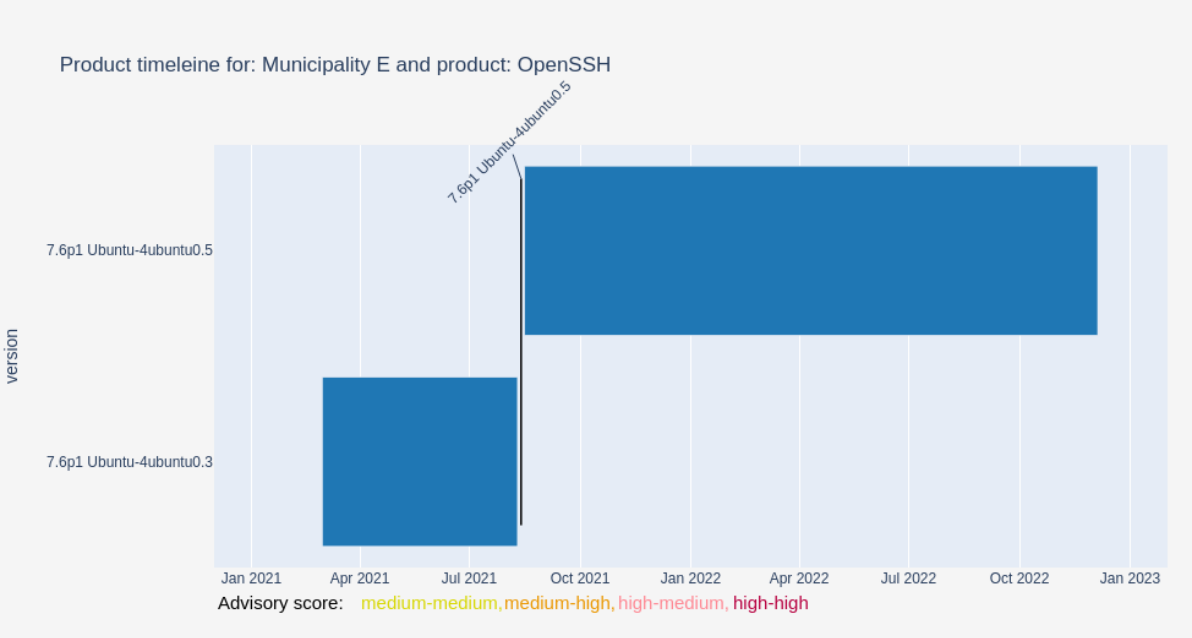


Figure 6.11: Vulnerability management timeline “Municipality E"

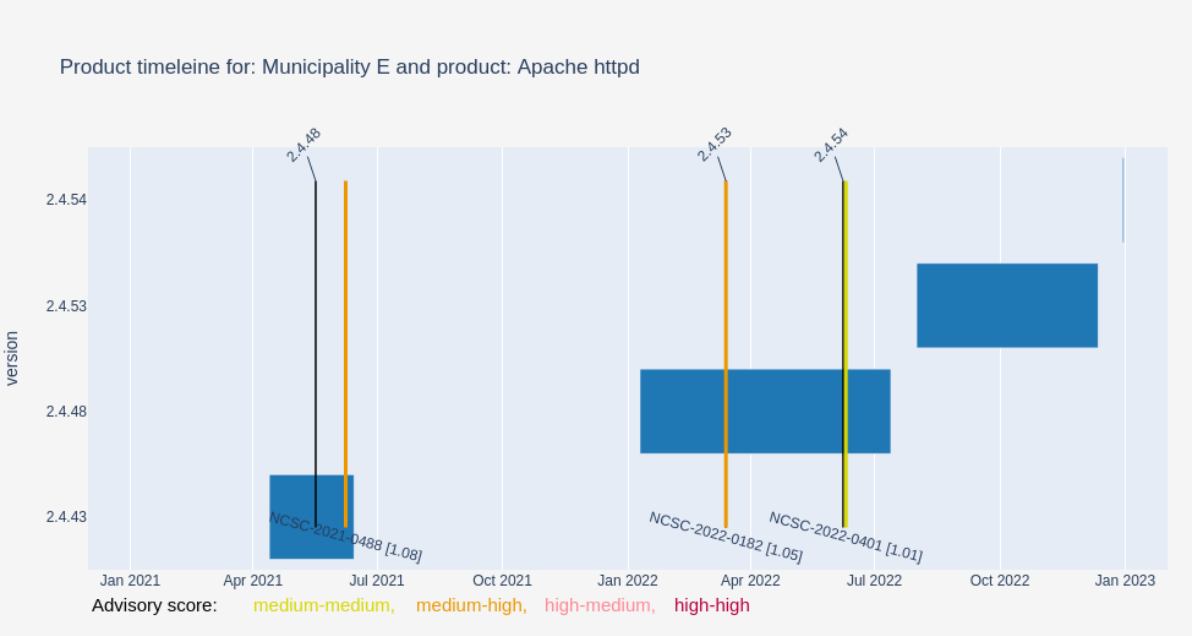


Figure 6.12: Vulnerability management timeline “Municipality E"

Another example is visualised in Figure 6.13 and Figure 6.14, where the differences are visible within the same municipality and product. In this case, we observe the vulnerability management of Municipality C on their `Apache` products. The biggest difference between the two timelines is in the fact that in Figure 6.13 the versions observed are already outdated by more than two years. This is in sharp contrast to Figure 6.14 where starting from September

2021, the versions are never more than a few months outdated. Another interesting observation is the influence of the security advisories or lack thereof. It is very clear that advisory with the severity of `high - high` triggers an almost immediate effect, while all others seem to be ignored. In this case, we do not have any explanation for the observed differences.

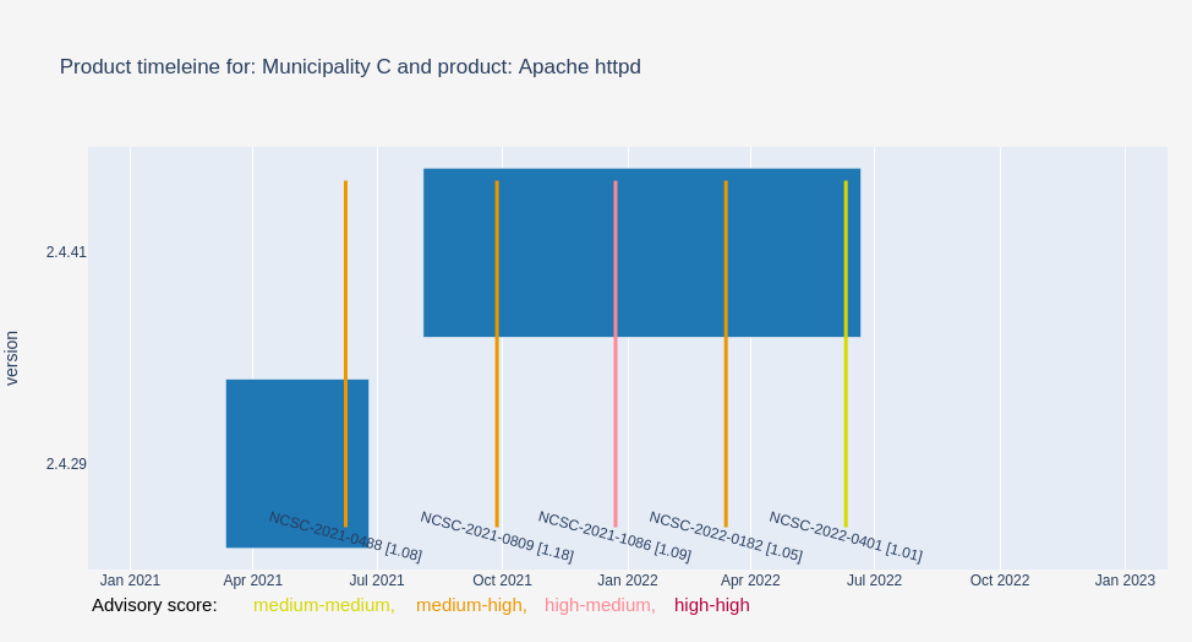


Figure 6.13: Vulnerability management timeline “Municipality C"

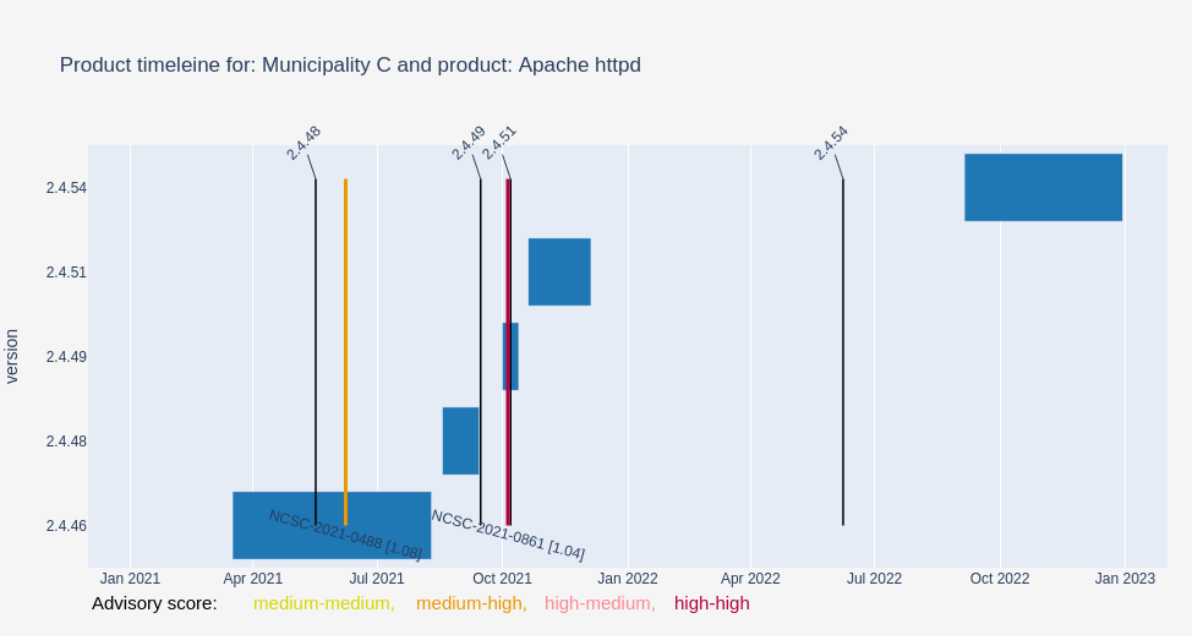


Figure 6.14: Vulnerability management timeline “Municipality C"

A further measurement of the vulnerability of Dutch municipalities we can observe is the number of CVEs associated with the systems they are running. During the CVE retrieval step, it was discovered that not all products have the same amount of vulnerabilities (CVEs).

The less popular products such as `nginx` have relatively fewer publicly disclosed vulnerabilities. In Figure 6.15 we observe these differences which show that the most popular products are also the ones that have the bigger number of CVEs per version on average. Furthermore, all products tend to have more CVEs with CVSS scores of `medium` and `high`, compared to `low` and `critical`. It is only the `Apache` and `ProFTPD` products that are usually connected to `critical` CVEs.

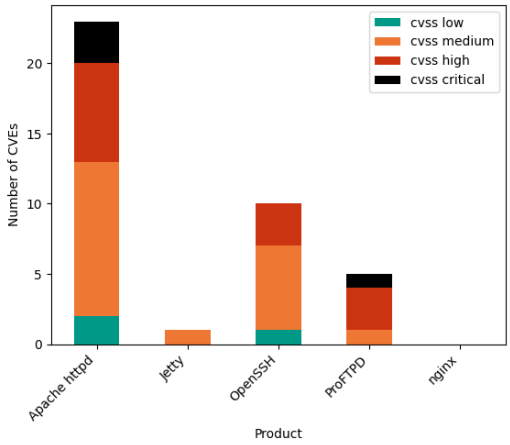


Figure 6.15: Distribution of average number of CVEs per product

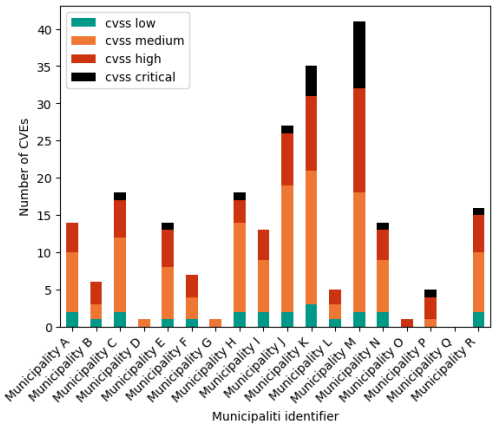


Figure 6.16: Distribution of average number and CVSS score of CVEs per municipality

When it comes to the distribution of CVEs and their CVSS scores over the analysed municipalities, Figure 6.16 displays these statistics. From the figure we can observe that vulnerabilities with a CVSS score of `medium` and `high` are the ones present the most on average. The amount of CVEs with score `low` seems to be constant among the different municipalities, while only 50% of the municipalities have at least 1 CVE with `critical` score on average. With the exception of Municipalities K and M, the number of `critical` CVEs is also uniform among different organisations. These results come to show that all of the considered municipalities carry a certain degree of vulnerability on their systems.

If we take a look on the average % of mitigated CVEs after update we get the following results as seen in Table 6.6.

Update to latest version	CVSS low	CVSS medium	CVSS high	CVSS critical
False	12%	13%	22%	8%
True	3%	11%	16%	21%

Table 6.6: Avg. % of mitigate CVEs after software update

As the results suggest, updating to a newer software version has a positive effect on the amount of CVEs and their CVSS scores. What is more, updating to the newest version available, has an almot three fold mitigation effect specifically of the CVEs with a score of `critical`.

However, as mentioned before, basing the security of an organisation only on the number of CVEs its system has could lead to overestimations. As such we shall also look into how exploitable these vulnerabilities are. As it turns out, from the 30 unique systems we have observed, only 6 have been exploitable at any given period of time between 2021 and 2022. 2 of those 6 systems have the product of `Apache` and have been exploitable due to the vulnerability disclosed in `CVE-2021-41773` for 1 and 3 days respectively. The other 4 systems have been running versions of `OpenSSH` with an exploitable vulnerability as disclosed by ExploitDB or CISA. Two examples are visible in Figure 6.17 and Figure 6.18 while all exploitability visualisations are

also available in Appendix C.

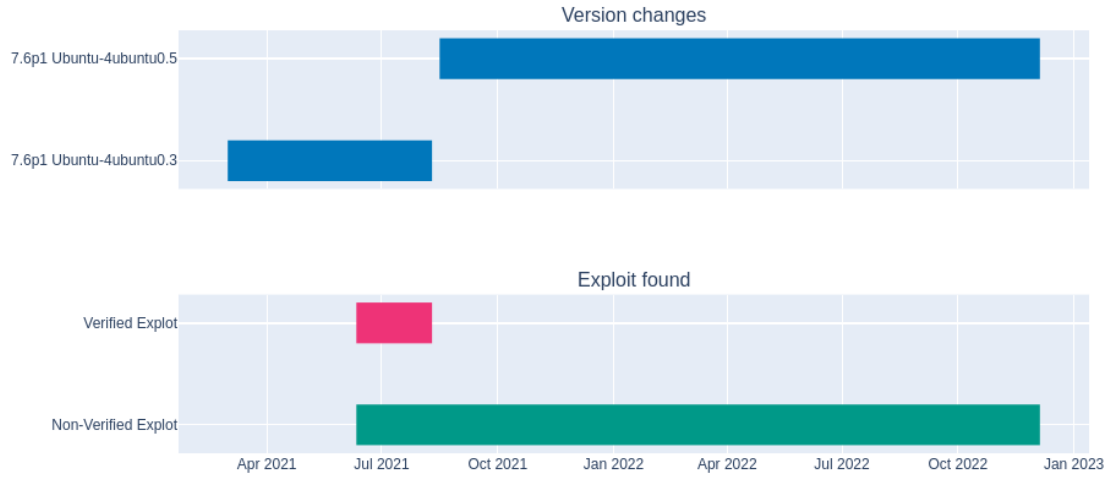


Figure 6.17: Vulnerability management timeline “Municipality E" compared to exploit timeline

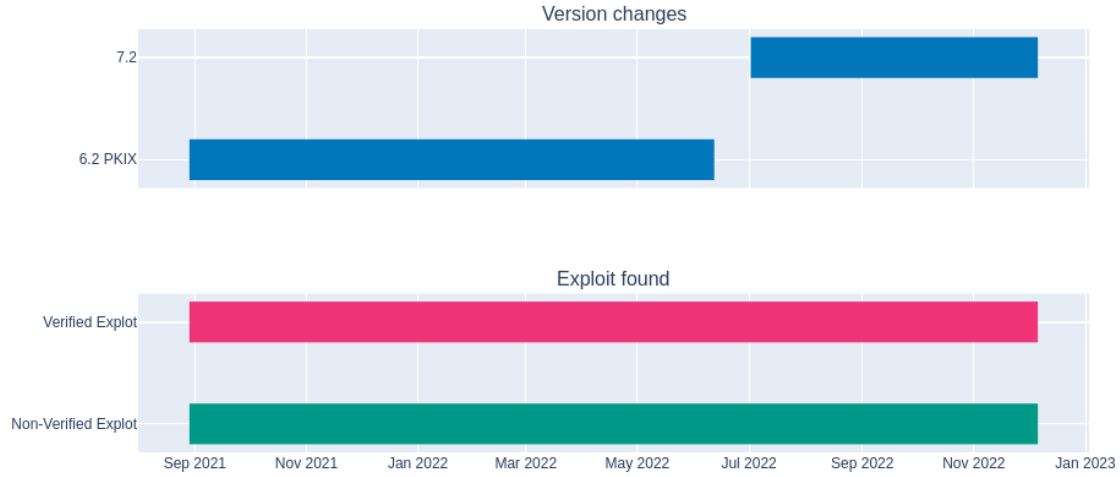


Figure 6.18: Vulnerability management timeline “Municipality J" compared to exploit timeline

What we observe in these two examples is not all software updates mitigate exploitable vulnerabilities. We can see that in Figure 6.17 the change of versions effectively removes the risk associated with the verified exploitability, while still leaving the possibility for exploitation using the unverified one. On the other hand in Figure 6.18 the update in version has not changed the security exploitability of the system, due to the outdatedness of the system's version.
These types of analyses show that software updates are good starting points for vulnerability mitigation, however, they only have positive effects on the critical and exploitable vulnerabilities if the updates are to the latest version of the product.

6.4. Looking beyond 2 years

In order to put the findings from the previous section into perspective we have decided to carry out a final case study, looking at 3 IP addresses for a time period spanning over multiple years. Doing so shows that the observations presented before are not outlier behaviour, but consistent over time.

In this case study we have considered 3 IPs each corresponding to Municipalities C, D, and E as listed before. Only the IP address part of Municipality C has been observed in the previous step. The other two addresses have not been analysed before due to their lack of data for the years 2021 and 2022. These 3 new addresses are the ones considered for a longer case study, as they are the only ones that have data spanning over multiple years outside of the previously analysed 2 year period. The update timeline visualisations of the considered municipalities are presented in Figure 6.19, Figure 6.20, and Figure 6.21 respectively.
What is immediately visible, is that the poor vulnerability management we have observed for Municipality C before, is in fact a longer practice that has not changed much throughout the years. We can see that the updates as observed in 2021 are not influenced by an available software update as these versions are already considerably outdated by the time they have been put in use. Once again the NCSC advisory has no observable effect, explaining the observations about the other advisories present later in the timeline. As such we can conclude that the observed trend for Municipality C is in accordance with its behaviour throughout the years and is not dependent on the specific time period selected previously.

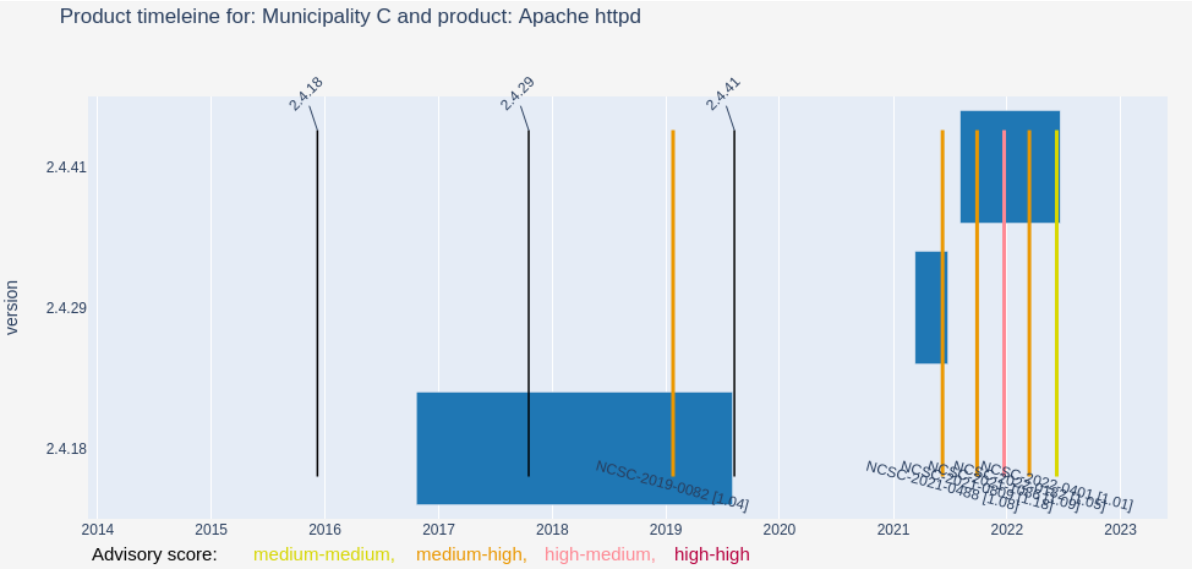


Figure 6.19: Vulnerability management timeline “Municipality C" for multiple years

When analysing the timeline for Municipality D we can see that updates are more frequently applied for this product. However, these updates are always applied with a certain delay of at least a few months and the difference between versions is on average a jump of 6 versions. Even though this shows a bigger effort in updating the product than observed in the previous case, once again we see that the newest versions are not considered. If we compare this pattern to what has been presented before for this municipality and product, the results are similar in frequency of updating, however in resent years Municipality D has been updating its other `Jetty` product with smaller version jumps and to versions closer to the newest one available. This shows an improvement of their vulnerability management.

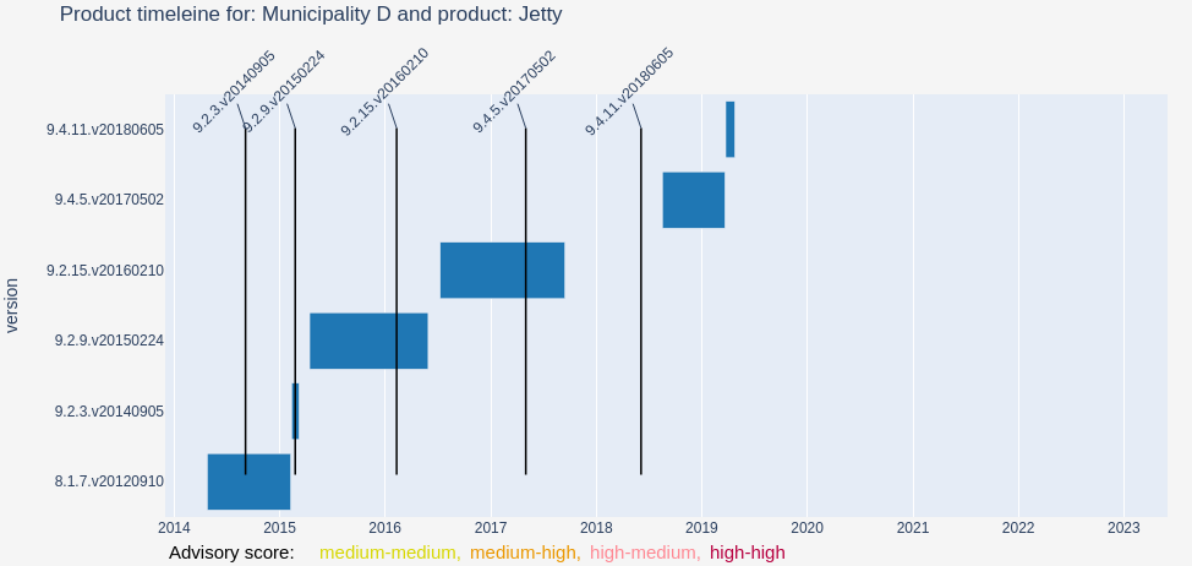


Figure 6.20: Vulnerability management timeline “Municipality D" for multiple years

The final considered case is that of Municipality E running a `nginx` product. This product has not been observed in the previous case study as part of the products for Municipality E because as it is visible from the timeline, no updates have been preformed in the last 3 years. What is visible is that the release of version `1.19.1` clearly triggers the update of the product. Before that there have been numerous product update releases that have been ignored, leaving the service outdated by 6.5 years until the observed version change. This behaviour of ignoring update releases continues after the change of version as well, as in the time period for which the service has been online there have been further 19 updates. It is also clear that the NCSC advisory has no effect on the vulnerability management either. This follows similar patterns as all other products for this municipality as observed in the previous case study, although generalising conclusions should be treated with caution as we have seen that sometimes different products get different vulnerability management treatments.

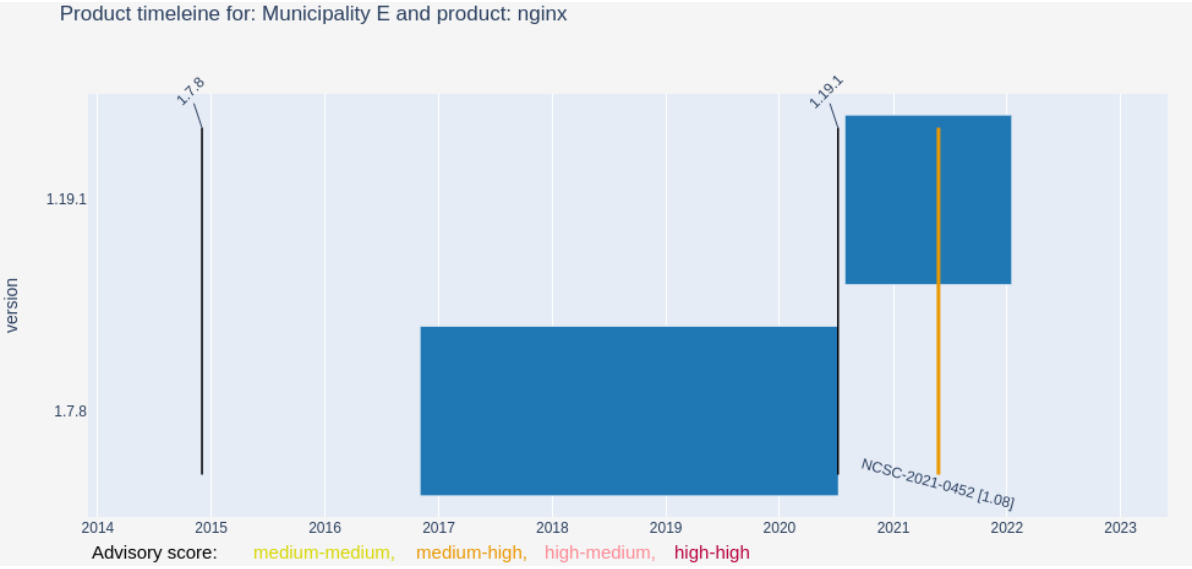


Figure 6.21: Vulnerability management timeline “Municipality D" for multiple years

The final comparison to the previous results is the lack of active exploits to the vulnerabilities that these 3 considered system posses. This shows that even though the vulnerability management is lacking on the best practices of software updates, these systems could not be exploited by a publicly disclosed exploit for one of the product specific vulnerabilities. Nevertheless, as explained earlier, the lack of public exploits should not be considered a reason for not keeping up with updates and mitigation risks.
What is visible from these 3 additional case studies is that the observed behaviours in section 6.3 are not abnormalities caused by external factors. The vulnerability management of Dutch municipalities has always presented a certain level of lack of timely software updates and keeping the systems up to date. Moreover, it was validated that the NCSC advisories have little to no effect on the speed of updates, unless they have a severity score of `high - high`.

7

Discussion and Conclusions

The analysis from the previous chapters provides a broad picture of the available data and observed vulnerability management of the Dutch municipalities. In this chapter, we further elaborate on our findings in section 7.1 and show our concluding remarks in section 7.2

7.1. Discussion

The findings of this research can be categorised into two main areas. Firstly, we shall discuss what is publicly observable using Open source intelligence and what this information can be useful for. Secondly, conclusions can be made about the general vulnerability management of the Dutch municipalities as analysed before. These two main topics are further outlined in this section.

7.1.1. Open source data collection

One of the main characteristics of this research is its outsider perspective. The whole process from data collection to vulnerability management analysis has been conducted using entirely open source data. This approach comes with its limitations in the face of missing ground truth data. However, this gives a good perspective of what is and is not observable without active involvement with the organisations.

To put things into perspective and answer our first sub-research question regarding what open source data is available we can follow the data discovery path as explained earlier in Figure 4.5 in chapter 4. These steps were performed for both the exploratory data analysis in chapter 4 as well as the historical analysis in chapter 6. In both cases the first two data points of “Found ranges" and “Found IP addresses" are the same. In total we have discovered 4,455 `InetNums` from the Ripe database and 4,230 CIDR ranges from MaxMind and Hurricane Electric combined. If we unfold the IP addresses that belong to these ranges we have 383,831 unique IPs that have been assigned to a municipality at one point in time in the past 23 years.

If we focus on the retrieved information about currently active service we have observed a significant drop in currently used IP addresses to 6,354. That corresponds to 1.7% of all discovered IPs. This further decreases once we filter out addresses located outside of the Netherlands, belonging to for-profit companies, and ones where not enough information was present to be considered municipality owned. After the filtration step we are left with only 3,866 IPs corresponding to 285 municipalities. For each of those IP addresses we have analysed the network scan records as retrieved from Shodan, which sum up to 6,755 records. This number shows that usually IPs are associated with multiple services running on different ports. For the purpose of vulnerability management analysis we have further removed records that are missing product or version information. This step has produced the biggest decrease so far, as after it

we were left with only 664 records. From those we have further excluded commercial products, and products with not enough update behaviour or records. The final number of records we have used for our analysis is 106 corresponding to only 76 IPs from 52 municipalities. From those records, 66 systems are exploitable, which are linked to 52 IPs from 35 municipalities using publicly disclosed vulnerability exploits. If we put this number in perspective to the initially discovered IP addresses, the exploitable IPs are less than 0.8% of the currently active municipality IPs. This data discovery and filtering path comes to show that open source data quickly decreases the more information we require from the records.
The numbers for the historic data analysis follow a similar curve of decline and are case specific for the years selected for analysis. We present the statistics for the above explained numbers in Table 7.1.

	Total Active	Municipality owned	With product and version information	Analysed	Exploitable
IP addresses	6,354	3,866 (61%)	623 (10%)	76 (1.2%)	52 (0.8%)
Shodan records	N.A.	N.A.	664	106	66
Municipalities	342	285 (83%)	164 (48%)	52 (15%)	35 (10%)

Table 7.1: Historic data filtration numbers and percentage of total

7.1.2. Open source data usage

It is also important to discuss which parties could make use of open source intelligence as performed in this study. The two main candidates are cyber adversaries that try to cover their presence and intentions and security advisory bodies, interested in monitoring the networks of their contingencies.
When it comes to attacker characteristics we can distinguish between two particular interests: targeted adversaries and opportunistic ones. One of their main differences is in the selected targets. Targeted attackers, start with identifying a target or a victim they want to attack, while opportunistic attackers select an attack vector they want to use and try to see which victims could be attacked. If we translate this to the context of vulnerability exploitation, the targeted adversaries will fist identify their victims and then search for present vulnerabilities, while the opportunistic ones will start by identifying a vulnerability they want to exploit and then search for targets that have the selected vulnerability. From this it becomes clear that their modus operandi differ in terms of reconnaissance.
Cyber attacks always start with some sort of reconnaissance. Different previous studies, such as the works of Di Tizio et al. [7] and Mazurczyk and Caviglione [18], show that even though social engineering is by far the most employed attack vector, vulnerability exploitation is still widely used by adversaries. In particular, Advanced Persistent Threads (APTs) do usually employ publicly known vulnerabilities [7] in their attacks. These studies further claim that network scanning services such as Censys and Shodan are a “quick and easy" way to collect vulnerability data. Other studies of the cyber crime ecosystem [2], show that indeed these services are widely used for information gathering, especially in the world of Intenret of Things (IoT). Depending on the adversaries intent, however, this open source data could yield different results.
Focusing on targeted attackers, we can expect similar methodology to the one employed in this research. As it was shown in the previous chapters, the open source information available, related to Dutch municipality networks in this case, provides very minimal useful data on the

specific services that are running on the networks and their associated vulnerabilities. It was shown that this form of passive OSINT is a tedious approach which could yield very minimal results. As such adversaries would need other paths of vulnerability searching, such as active network scans, techniques which could reveal their intentions. In this sense, the data set size limitation of this study is a real-life advantage for the observed municipalities.
On the other hand, opportunistic adversaries focus their search efforts on which networks have a specific vulnerability. This approach is widely covered in literature as well, when analysing crucial infrastructure and IoT devices ([10], [21], [2]). Although different from the presented methodology, this form of reconnaissance would still produce minimal results in the context of Dutch municipal systems. This claim lays in the fact that still a large number of the network assets do not disclose product and version information, meaning that the adversaries will not consider them for an attack. In this light, the cyber security hygiene as observed in section 5.5 is a good first step in minimising an organisation's attack surface against opportunistic adversaries.
As such, our finding oppose the previous notions that network scan data as retrieved from Shodan is a quick and easy way for attackers to perform reconnaissance. It is true that services like Shodan enable a wider range of adversaries to collect network data without getting detected, however the acquired data is limited in its volume and depending on the intent not proportional to the required time and efforts for its retrieval.
Another important aspect to consider are security advisory bodies and response teams such as the NCSC and national CERT, which might want to use open-source data to monitor the vulnerability landscape of governmental networks. Unfortunately for them, they will be faced with the same limitations as this research. Although, they would have the ground truth data on the exact municipality network ranges, the available passive networks scan data from services like Shodan would not disclose all of the products and versions that are currently running on these networks. Performing active network scans would suffer from the same scarcity of available data, rendering it useless in this case. For legitimate parties such as the NCSC collaborative work would be the only feasible solution, if precise monitoring is wanted.

7.1.3. Vulnerability posture of municipalities

When it comes to what the actual data tells about the vulnerability management of Dutch municipalities, things don't always look great. We have observed that only around 10% of the municipalities have their systems up to date. This issue is further worsened by the fact that of those organisations that have outdated products, about half have products that have been outdated for 3 or more years. Both systems kept up to date and greatly aged software suffer from a certain amount of software vulnerabilities. We have observed that on average municipalities are associated with a considerable amount of `medium` and `high` severity CVEs. For the systems for which vulnerability exploitations have been publicly disclosed, we observe an increase in the average number of `critical` vulnerabilities as well. Moreover, we can see that the rate of increase in the number of outdated products is way bigger than that of up-to-date systems, which will only further worsen the situation in the future.
A negative trend we have observed is the increasing number of systems susceptible to publicly disclosed vulnerability exploits. As we have seen in our case study for the years 2021 and 2022 only around 20% have been vulnerable at a given time. Unfortunately this number has increased to just above 60% for the systems that have been active as of this year. These findings are not surprising in the light of the great software age of most used products and the lag of applying software updates as seen before. It is however important to praise the efforts of some municipalities to patch their systems and effectively remove the exploit possibility as seen in some rare cases. These efforts are in line with previous works, that show that

patching vulnerabilities with public exploits is a better strategy for vulnerability management prioritisation than solemnly relying on the CVSS score [1]. A future improvement in this direction could be the use of dark market exploits in the vulnerability prioritisation as advised in that study, a step strongly encouraged, however not feasible to be undertaken by majority of the observed organisations.
As seen from all the timelines, there is no united update behaviour between the analysed municipalities. Strategies differ not only between different organisations but also within a municipality's own network. This is a particular disadvantage for smaller municipalities or ones that struggle to keep their systems updated, as they could benefit from collaborations with other organisations that present rigorous vulnerability management. Such collaborations are not something new as examples are already present in some regions of the Netherlands[1], however more and better interaction and partnership could stimulate knowledge sharing and enrich the vulnerability management stance of Dutch municipalities. A further collaboration that already exists is with the NCSC. As observed in the case studies, the advisories sent by the NCSC often get ignored unless they have a severity score of `high - high`. This behaviour could lead to severe accidents if crucial vulnerabilities stay unpatched, leading to great losses for the organisations.

7.2. Conclusions

In this research, we have investigated the vulnerability management of Dutch municipality IT networks using open-source data and intelligence. In order to do that we have answered the main and sub-research questions, concerning what information can be collected using publicly available data and what factors influence the vulnerability management of the analysed organisations.
It was shown that by using open source intelligence a lot of data can be collected related to municipality networks. However, when it comes to vulnerability data the available data points are way fewer. This shows that using solely publicly available information is not enough to monitor governmental networks and in the case of adversarial preparations this information will not always provide a clear picture of the security landscape and underlying vulnerabilities. From the available information, we have also analysed what factors are responsible and influence the vulnerability management of the organisations. Our findings suggest that there are differences in update behaviour not only between different municipalities but also when it comes to different products. In some cases software update availability is seen to trigger update practices, however, this is not always the case and in the majority of systems the software age of the products has a magnitude of years. On the other hand, security advisories published and broadcasted by the NCSC should in theory contribute to faster vulnerability patching and mitigation. In practice, this only holds true for advisories with a severity score of `high - high`. The rest of the advisories with lower severity are usually ignored by the municipalities. This practice in some cases lead to a piling of multiple advisories and vulnerabilities associated with them. Finally, we could not find any statistical significance in the influence of network size or an IP's lifetime on the vulnerability management concerning the underlying system. This is in part due to the scarcity of our data set, but also the software age seems to be high for products and systems with varying characteristics in general.
Taking into consideration the findings corresponding to our two sub-research questions, we can conclude that the current security practices as observed using publicly available data rarely show best effort to keep products up to date. More often than not, software updates present reactive behaviour in response to crucial vulnerabilities, rather than proactive practice of good security mitigation. As such it is important to emphasise the need for better vulnerability management

[1] `https://www.kempengemeenten.nl/`

within Dutch municipal ICT networks and enable the possibility for knowledge sharing and collaboration with organisations that already possess expertise in better vulnerability behaviour. Better integration of the security advisories of the NCSC is also highly encouraged in the path for a better vulnerability management.

References

[1] Luca Allodi and Fabio Massacci. "Comparing Vulnerability Severity and Exploits Using Case-Control Studies". In: ACM Trans. Inf. Syst. Secur. 17.1 (Aug. 2014). ISSN: 1094-9224. DOI: 10.1145/2630069. URL: https://doi-org.tudelft.idm.oclc.org/10.1145/2630069.

[2] Maria Bada and Ildiko Pete. "An exploration of the cybercrime ecosystem around Shodan". In: 2020 7th International Conference on Internet of Things: Systems, Management and Security (IOTSMS). 2020, pp. 1–8. DOI: 10.1109/IOTSMS52051.2020.9340224.

[3] Christopher Bennett, AbdelRahman Abdou, and Paul C. van Oorschot. Empirical scanning analysis of Censys and shodan. May 2021. URL: https://www.ndss-symposium.org/ndss-paper/auto-draft-144/.

[4] BGP (Border Gateway Protocol). 2013. URL: https://web.archive.org/web/20130928115120/http://www.orbit-computer-solutions.com/BGP.php.

[5] Censys. URL: https://censys.io/.

[6] Nurullah Demir et al. "Our (in)secure web: Understanding update behavior of websites and its impact on security". In: Passive and Active Measurement (2021), pp. 76–92. DOI: 10.1007/978-3-030-72582-2_5.

[7] Giorgio Di Tizio, Michele Armellini, and Fabio Massacci. Software updates strategies: A quantitative evaluation against advanced persistent threats. May 2022. URL: https://doi.org/10.48550/arXiv.2205.07759.

[8] Nesara Dissanayake et al. "Software security patch management - A systematic literature review of challenges, approaches, tools and practices". In: Information and Software Technology 144 (2022), p. 106771. DOI: 10.1016/j.infsof.2021.106771.

[9] Zakir Durumeric et al. "The matter of Heartbleed". In: Proceedings of the 2014 Conference on Internet Measurement Conference (2014). DOI: 10.1145/2663716.2663755.

[10] P. D. Francik, T. D. Ashley, and M. E. Poplawski. Connecting the dots: An assessment of cyber-risks in networked building and Municipal Infrastructure Systems. Feb. 2023. URL: https://www.pnnl.gov/publications/connecting-dots-assessment-cyber-risks-networked-building-and-municipal-infrastructure.

[11] Gemeentelijke Herindeling. Dec. 2022. URL: https://www.rijksoverheid.nl/onderwerpen/gemeenten/gemeentelijke-herindeling.

[12] Gemeentelijke Indelingen per jaar. Feb. 2023. URL: https://www.cbs.nl/nl-nl/onze-diensten/methoden/classificaties/overig/gemeentelijke-indelingen-per-jaar.

[13] Thomas Gerace and Huseyin Cavusoglu. The critical elements of Patch Management: Proceedings of the 33rd Annual ACM SIGUCCS conference on user services. Nov. 2005. URL: https://dl.acm.org/doi/10.1145/1099435.1099457.

[14] Mehran Bozorgi UCSD / Google et al. Beyond heuristics: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and data mining. July 2010. URL: https://dl.acm.org/doi/10.1145/1835804.1835821.

[15] Hurricane Electric BGP Toolkit. URL: https://bgp.he.net/.

[16] Elissa M. Redmiles University of Maryland et al. How I learned to be secure: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. Oct. 2016. URL: https://dl.acm.org/doi/10.1145/2976749.2978307.

[17] MaxMind. URL: https://www.maxmind.com/en/home.

[18] Wojciech Mazurczyk and Luca Caviglione. "Cyber Reconnaissance Techniques". In: Commun. ACM 64.3 (Feb. 2021), pp. 86–95. ISSN: 0001-0782. DOI: 10.1145/3418293. URL: https://doi.org/10.1145/3418293.

[19] NIST Glossary. URL: https://csrc.nist.gov/glossary.

[20] Jamie O'Hare, Rich Macfarlane, and Owen Lo. "Identifying Vulnerabilities Using Internet-Wide Scanning Data". In: 2019 IEEE 12th International Conference on Global Security, Safety and Sustainability (ICGS3). 2019, pp. 1–10. DOI: 10.1109/ICGS3.2019.8688018.

[21] Mark Patton et al. "Uninvited Connections: A Study of Vulnerable Devices on the Internet of Things (IoT)". In: 2014 IEEE Joint Intelligence and Security Informatics Conference. 2014, pp. 232–235. DOI: 10.1109/JISIC.2014.43.

[22] RIPE Network Coordination Centre. URL: https://www.ripe.net/.

[23] Shodan. URL: https://www.shodan.io/.

[24] Centraal Bureau voor de Statistiek. Gemeentelijke Indelingen per jaar. Feb. 2023. URL: https://www.cbs.nl/nl-nl/onze-diensten/methoden/classificaties/overig/gemeentelijke-indelingen-per-jaar/.

[25] Christian Tiefenau and Maximilian Häring. Security, availability, and multiple information sources: Exploring update behavior of system administrators. 2020. URL: https://www.usenix.org/conference/soups2020/presentation/tiefenau.

[26] NCSC UK. The problems with patching. 2019. URL: https://www.ncsc.gov.uk/blog-post/the-problems-with-patching.

[27] Ángel Jesús Varela-Vaca et al. "Feature models to boost the Vulnerability Management Process". In: Journal of Systems and Software 195 (2023), p. 111541. DOI: 10.1016/j.jss.2022.111541.

[28] Jonathan Codi West and Tyler Moore. "Longitudinal Study of internet-facing openssh update patterns". In: Passive and Active Measurement (2022), pp. 675–689. DOI: 10.1007/978-3-030-98785-5_30.

A

Municipalities' names and changes

A.1. Municipalities' names between 2001 and 2023

- 's graveland
- 's gravendeel
- 's gravenhage
- 's gravenzande
- 's hertogenbosch
- aa en hunze
- aalburg
- aalsmeer
- aalten
- abcoude
- achtkarspelen
- akersloot
- alblasserdam
- albrandswaard
- alkemade
- alkmaar
- almelo
- almere
- alphen aan den rijn
- alphen chaam
- altena
- ambt montfort
- ameland
- amerongen
- amersfoort
- amstelveen
- amsterdam
- andijk
- angerlo
- anna paulowna
- apeldoorn
- appingedam
- arcen en velden
- arnhem
- assen
- asten
- axel
- baarle nassau
- baarn
- barendrecht
- barneveld
- bathmen
- bedum
- beek
- beekdaelen
- beemster
- beesel
- bellingwedde
- bemmel
- bennebroek
- berg en dal
- bergambacht
- bergeijk
- bergen (limburg)
- bergen (noord holland)
- bergen op zoom
- bergh
- bergschenhoek
- berkel en rodenrijs
- berkelland
- bernheze
- bernisse
- best
- beuningen
- beverwijk
- binnenmaas
- bladel
- blaricum
- bleiswijk
- bloemendaal
- boarnsterhim
- bodegraven
- bodegraven reeuwijk
- boekel
- bolsward
- borculo
- borger odoorn
- borne
- borsele
- boskoop
- boxmeer
- boxtel
- breda
- breukelen
- brielle
- bronckhorst
- brummen
- brunssum
- bunnik
- bunschoten
- buren
- bussum
- capelle aan den ijssel
- castricum
- coevorden
- cranendonck
- cromstrijen
- cuijk
- culemborg
- dalfsen
- dantumadeel
- dantumadiel
- de bilt
- de friese meren
- de fryske marren

- de lier
- de marne
- de ronde venen
- de wolden
- delft
- delfzijl
- den helder
- denekamp
- deurne
- deventer
- didam
- diemen
- dijk en waard
- dinkelland
- dinxperlo
- dirksland
- dodewaard
- doesburg
- doetinchem
- dongen
- dongeradeel
- doorn
- dordrecht
- drechterland
- driebergen rijsenburg
- drimmelen
- dronten
- druten
- duiven
- echt
- echt susteren
- echteld
- edam volendam
- ede
- eemnes
- eemsdelta
- eemsmond
- eersel
- eibergen
- eijsden
- eijsden margraten
- eindhoven
- elburg
- emmen
- enkhuizen
- enschede
- epe
- ermelo
- etten leur
- ferwerderadiel
- franekeradeel
- gaasterlan sleat
- geertruidenberg
- geldermalsen
- geldrop
- geldrop mierlo
- gemert bakel
- gendringen
- gennep
- giessenlanden
- gilze en rijen
- goedereede
- goeree overflakkee
- goes
- goirle
- gooise meren
- gorinchem
- gorssel
- gouda
- graafstroom
- graft de rijp
- grave
- groenlo
- groesbeek
- groningen
- grootegast
- gulpen wittem
- haaksbergen
- haaren
- haarlem
- haarlemmerliede en spaarnwoude
- haarlemmermeer
- haelen
- halderberge
- hardenberg
- harderwijk
- hardinxveld giessendam
- haren
- harenkarspel
- harlingen
- hattem
- heel
- heemskerk
- heemstede
- heerde
- heerenveen
- heerhugowaard
- heerjansdam
- heerlen
- heeze leende
- heiloo
- helden
- hellendoorn
- hellevoetsluis
- helmond
- hendrik ido ambacht
- hengelo
- hengelo (gld)
- het bildt
- het hogeland
- heumen
- heusden
- heythuysen
- hillegom
- hilvarenbeek
- hilversum
- hoeksche waard
- hof van twente
- hollands kroon
- hontenisse
- hoogeveen
- hoogezand sappemeer
- hoorn
- horst aan de maas
- houten
- huizen
- hulst
- hummelo en keppel
- hunsel
- ijsselstein
- jacobswoude
- kaag en braassem
- kampen
- kapelle
- katwijk
- kerkrade
- kessel
- kesteren
- koggenland
- kollumerland en nieuwkruisland
- korendijk
- krimpen aan den ijssel
- krimpenerwaard
- laarbeek
- land van cuijk
- landerd
- landgraaf
- landsmeer
- langedijk
- lansingerland
- laren
- leek
- leerdam
- leersum
- leeuwarden
- leeuwarderadeel
- leiden
- leiderdorp
- leidschendam

- leidschendam voorburg
- lelystad
- lemsterland
- leudal
- leusden
- lichtenvoorde
- liemeer
- liesveld
- limmen
- lingewaal
- lingewaard
- lisse
- lith
- littenseradiel
- lochem
- loenen
- loon op zand
- loosdrecht
- lopik
- loppersum
- losser
- maarn
- maarssen
- maasbracht
- maasbree
- maasdonk
- maasdriel
- maasgouw
- maashorst
- maasland
- maassluis
- maastricht
- margraten
- marum
- medemblik
- meerlo wanssum
- meerssen
- meierijstad
- meijel
- menaldumadeel
- menameradiel
- menterwolde
- meppel
- middelburg
- middelharnis
- midden delfland
- midden drenthe
- midden groningen
- mierlo
- mill en sint hubert
- millingen aan de rijn
- moerdijk
- molenlanden
- molenwaard
- monster
- montferland
- montfoort
- mook en middelaar
- moordrecht
- muiden
- naaldwijk
- naarden
- neder betuwe
- nederhorst den berg
- nederlek
- nederweert
- neede
- neerijnen
- niedorp
- nieuw lekkerland
- nieuwegein
- nieuwerkerk aan den ijssel
- nieuwkoop
- nijefurd
- nijkerk
- nijmegen
- nissewaard
- noardeast fryslan
- noord beveland
- noordenveld
- noorder koggenland
- noordoostpolder
- noordwijk
- noordwijkerhout
- nootdorp
- nuenen, gerwen en nederwetten
- nunspeet
- nuth
- obdam
- oegstgeest
- oirschot
- oisterwijk
- oldambt
- oldebroek
- oldenzaal
- olst
- olst wijhe
- ommen
- onderbanken
- oost gelre
- oostburg
- oosterhout
- oostflakkee
- ooststellingwerf
- oostzaan
- opmeer
- opsterland
- oss
- oud beijerland
- oude ijsselstreek
- ouder amstel
- ouderkerk
- oudewater
- overbetuwe
- papendrecht
- peel en maas
- pekela
- pijnacker
- pijnacker nootdorp
- purmerend
- putten
- raalte
- ravenstein
- reeuwijk
- reiderland
- reimerswaal
- renkum
- renswoude
- reusel de mierden
- rheden
- rhenen
- ridderkerk
- rijnsburg
- rijnwaarden
- rijnwoude
- rijssen
- rijssen holten
- rijswijk
- roerdalen
- roermond
- roggel en neer
- roosendaal
- rotterdam
- rozenburg
- rozendaal
- rucphen
- ruurlo
- sas van gent
- sassenheim
- schagen
- scheemda
- schermer
- scherpenzeel
- schiedam
- schiermonnikoog
- schijndel
- schinnen
- schipluiden

- schoonhoven
- schouwen duiveland
- sevenum
- simpelveld
- sint anthonis
- sint michielsgestel
- sint oedenrode
- sittard geleen
- skarsterlan
- sliedrecht
- slochteren
- sluis
- sluis aardenburg
- smallingerland
- sneek
- soest
- someren
- son en breugel
- spijkenisse
- stadskanaal
- staphorst
- stede broec
- steenbergen
- steenderen
- steenwijk
- steenwijkerland
- stein
- stichtse vecht
- strijen
- sudwest fryslan
- susteren
- swalmen
- ten boer
- ter aar
- terneuzen
- terschelling
- texel
- teylingen
- tholen
- thorn
- tiel
- tilburg
- tubbergen
- twenterand
- tynaarlo
- tytsjerksteradiel
- ubbergen
- uden
- uitgeest
- uithoorn
- urk
- utrecht
- utrechtse heuvelrug
- vaals
- valkenburg
- valkenburg aan de geul
- valkenswaard
- veendam
- veenendaal
- veere
- veghel
- veldhoven
- velsen
- venhuizen
- venlo
- venray
- vianen
- vijfheerenlanden
- vlaardingen
- vlagtwedde
- vlieland
- vlissingen
- vlist
- voerendaal
- voorburg
- voorhout
- voorne aan zee
- voorschoten
- voorst
- vorden
- vriezenveen
- vught
- waadhoeke
- waalre
- waalwijk
- waddinxveen
- wageningen
- warmond
- warnsveld
- wassenaar
- wateringen
- waterland
- weert
- weesp
- wehl
- werkendam
- wervershoof
- west betuwe
- west maas en waal
- wester koggenland
- westerkwartier
- westerveld
- westervoort
- westerwolde
- westland
- weststellingwerf
- westvoorne
- wierden
- wieringen
- wieringermeer
- wijchen
- wijdemeren
- wijk bij duurstede
- winschoten
- winsum
- winterswijk
- wisch
- woensdrecht
- woerden
- wognum
- wonseradeel
- wormerland
- woudenberg
- woudrichem
- wymbritseradiel
- zaanstad
- zaltbommel
- zandvoort
- zederik
- zeevang
- zeewolde
- zeist
- zelhem
- zevenaar
- zevenhuizen moerkapelle
- zijpe
- zoetermeer
- zoeterwoude
- zuidhorn
- zuidplas
- zundert
- zutphen
- zwartewaterland
- zwijndrecht
- zwolle

A.2. Municipality changes between 2002 and 2023

Year	New municipality	Old municipality(s)
2002	Pijnacker-Nootdorp	Nootdorp, Pijnacker
	Leidschendam-Voorburg	Leidschendam, Voorburg
	Wijdemeren	's-Graveland, Nederhorst den Berg, Loosdrecht
	Castricum	Castricum, Akersloot, Limmen
	Kesteren	Kesteren, Dodewaard, Echteld
2003	Olst-Wijhe	Olst
	Dinkelland	Denekamp
	Twenterand	Vriezenveen
	Lingewaard	Bemmel
	Steenwijkerland	Steenwijk
	Zwijndrecht	Heerjansdam, Zwijndrecht
	Echt-Susteren	Echt, Susteren
	Oss	Oss, Ravenstein
	Terneuzen	Sas van Gent, Axel, Terneuzen
	Hulst	Hulst, Hontenisse
	Sluis	Sluis-Aardenburg, Oostburg
2004	Neder-Betuwe	Kesteren
	Rijssen-Holten	Rijssen
	Geldrop-Mierlo	Geldrop, Mierlo
	Westland	's-Gravenzande, De Lier, Monster, Naaldwijk, Wateringen
	Midden-Delfland	Maasland, Schipluiden
2005	Groenlo	Groenlo, Lichtenvoorde
	Aalten	Aalten, Dinxperlo
	Oude IJsselstreek	Gendringen, Wisch
	Montferland	Bergh, Didam
	Zevenaar	Angerlo, Zevenaar
	Bronckhorst	Hummelo en Keppel, Hengelo (gld), Steenderen, Vorden, Zelhem
	Berkelland	Borculo, Eibergen, Neede, Ruurlo
	Lochem	Gorssel, Lochem
	Zutphen	Warnsveld, Zutphen
	Doetinchem	Doetinchem, Wehl
	Deventer	Bathmen, Deventer
2006	Drechterland	Drechterland, Venhuizen
	Katwijk	Katwijk, Rijnsburg, Valkenburg
	Teylingen	Warmond, Sassenheim, Voorhout
	Utrechtse Heuvelrug	Maarn, Amerongen, Leersum, Driebergen-Rijsenburg, Doorn
2007	Oost Gelre	Groenlo
	Roermond	Swalmen, Roermond
	Roerdalen	Roerdalen, Ambt Montfort
	Maasgouw	Heel, Thorn, Maasbracht
	Leudal	Haelen, Hunsel, Heythuysen, Roggel en Neer

	Koggenland	Wester-Koggenland, Obdam
	Medemblik	Medemblik, Wognum, Noorder-Koggenland
	Lansingerland	Bergschenhoek, Berkel en Rodenrijs, Bleiswijk
	Binnenmaas	Binnenmaas, 's-Gravendeel
	Nieuwkoop	Liemeer, Nieuwkoop, Ter Aar
2008		
2009	Kaag en Braassem	Alkemade, Jacobswoude
	Bloemendaal	Bloemendaal, Bennebroek
	Dantumadiel	Dantumadeel
2010	Zuidplas	Moordrecht, Nieuwerkerk aan den IJssel, Zevenhuizen-Moerkapelle
	Oldambt	Reiderland, Scheemda, Winschoten
	Peel en Maas	Helden, Kessel, Maasbree, Meijel
	Horst aan de Maas	Horst aan de Maas, Sevenum, Meerlo-Wanssum
	Venlo	Arcen en Velden, Venlo
2011	Rotterdam	Rozenburg, Rotterdam
	Menameradiel	Menaldumadeel
	Sudwest-Fryslan	Bolsward, Nijefurd, Sneek, Wonseradeel, Wymbritseradiel
	Medemblik	Andijk, Medemblik, Wervershoof
	Bodegraven-Reeuwijk	Bodegraven, Reeuwijk
	Oss	Lith, Oss
	Eijsden-Margraten	Eijsden, Margraten
	De Ronde Venen	Abcoude, De Ronde Venen
	Stichtse Vecht	Breukelen, Loenen, Maarssen
2012	Hollands Kroon	Anna Paulowna, Niedorp, Wieringen, Wieringermeer
2013	Schagen	Harenkarspel, Schagen, Zijpe
	Goeree-Overflakkee	Dirksland, Goedereede, Middelharnis, Oostflakkee
	Molenwaard	Graafstroom, Liesveld, Nieuw-Lekkerland
2014	Alphen aan den Rijn	Alphen aan den Rijn, Boskoop, Rijnwoude
	De Friese Meren	Gaasterlan-Sleat, Lemsterland, Skarsterlan, Boarnsterhim
2015	Alkmaar	Alkmaar, Graft-De Rijp, Schermer
	Groesbeek	Groesbeek, Millingen aan de Rijn, Ubbergen
	Nissewaard	Bernisse, Spijkenisse
	Krimpenerwaard	Bergambacht, Nederlek, Ouderkerk, Schoonhoven, Vlist
	Oss	Maasdonk, Oss
2016	De Fryske Marren	De Friese Meren
	Berg en Dal	Groesbeek
	Gooise Meren	Bussum, Muiden, Naarden

	Edam-Volendam	Edam-Volendam, Zeevang
2017	Meierijstad	Schijndel, Sint-Oedenrode, Veghel
2018	Westerwolde	Bellingwedde, Vlagtwedde
	Midden-Groningen	Hoogezand-Sappemeer, Slochteren, Menterwolde
	Waadhoeke	het Bildt, Franekeradeel, Menameradiel, Littenseradiel
	Leeuwarden	Leeuwarden, Leeuwarderadeel
	Zevenaar	Zevenaar, Rijnwaarden
2019	Het Hogeland	Bedum, Eemsmond, De Marne, Winsum
	Groningen	Ten Boer, Groningen, Haren
	Westerkwartier	Grootegast, Leek, Marum, Zuidhorn
	Noardeast-Fryslan	Dongeradeel, Kollumerland en Nieuwkruisland, Ferwerderadiel
	West Betuwe	Geldermalsen, Neerijnen, Lingewaal
	Haarlemmermeer	Haarlemmerliede en Spaarnwoude, Haarlemmermeer
	Vijfheerenlanden	Leerdam, Vianen, Zederik
	Noordwijk	Noordwijk, Noordwijkerhout
	Hoeksche Waard	Oud-Beijerland, Binnenmaas, Korendijk, Cromstrijen, Strijen
	Molenlanden	Giessenlanden, Molenwaard
	Altena	Aalburg, Werkendam, Woudrichem
	Beekdaelen	Onderbanken, Nuth, Schinnen
2020		
2021	Eemsdelta	Appingedam, Delfzijl, Loppersum
	Oisterwijk	Oisterwijk, Haaren
2022	Purmerend	Beemster, Purmerend
	Dijk en Waard	Heerhugowaard, Langedijk
	Maashorst	Landerd, Uden
	Land van Cuijk	Boxmeer, Cuijk, Grave, Mill en Sint Hubert, Sint Anthonis
	Amsterdam	Amsterdam, Weesp
2023	Voorne aan Zee	Brielle, Hellevoetsluis, Westvoorne

Table A.1: Changes of municipalities per year

B

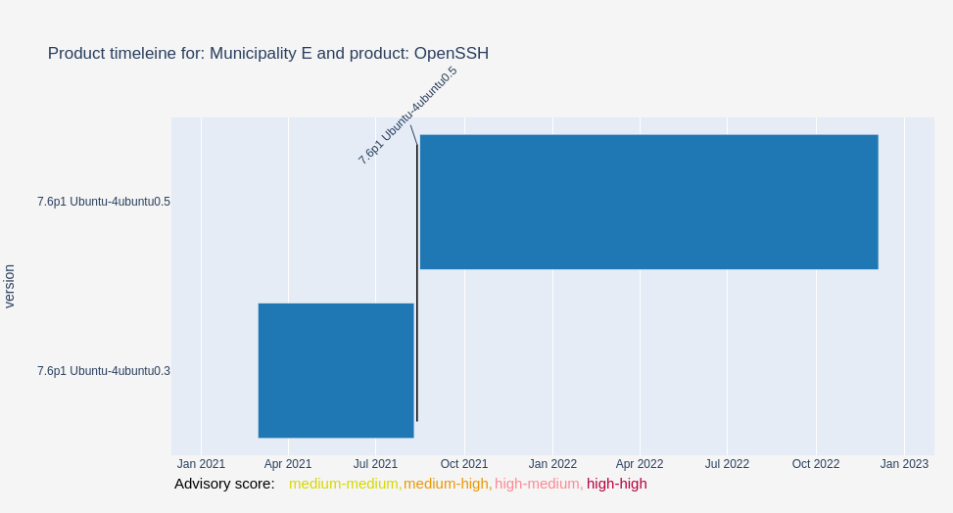
Software release information resources

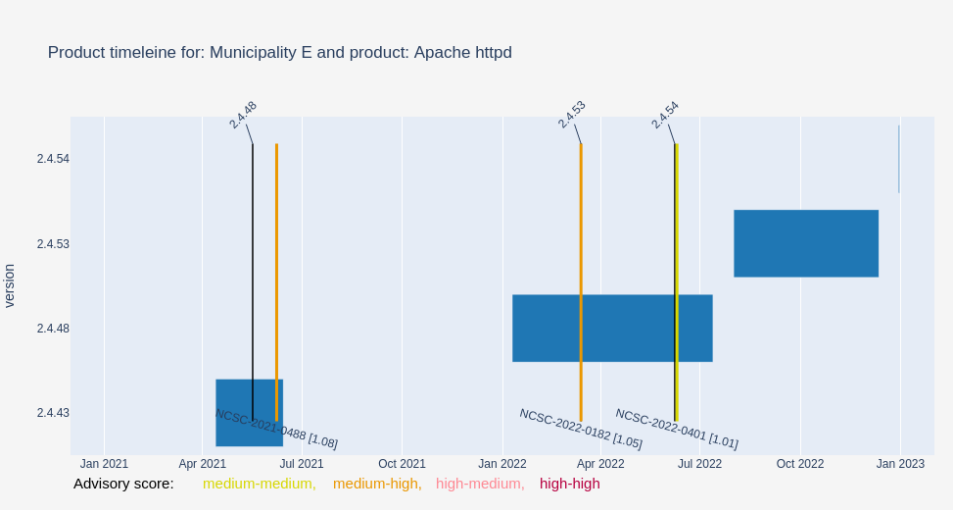
Software	Resource
Apache httpd	`https://github.com/apache/httpd/tags` `https://archive.apache.org/dist/httpd`
Jetty	`https://github.com/eclipse/jetty.project/releases` `https://www.eclipse.org//lists/jetty-announce/maillist.html`
OpenSSH	`https://launchpad.net`
Nginx	`http://hg.nginx.org/nginx/log`
Webmin (MiniServ)	`https://github.com/webmin/webmin/releases`
ProFTP	`https://sourceforge.net/p/proftp/mailman/proftp-announce`
Plex	`https://forums.plex.tv/t/plex-media-server/30447`

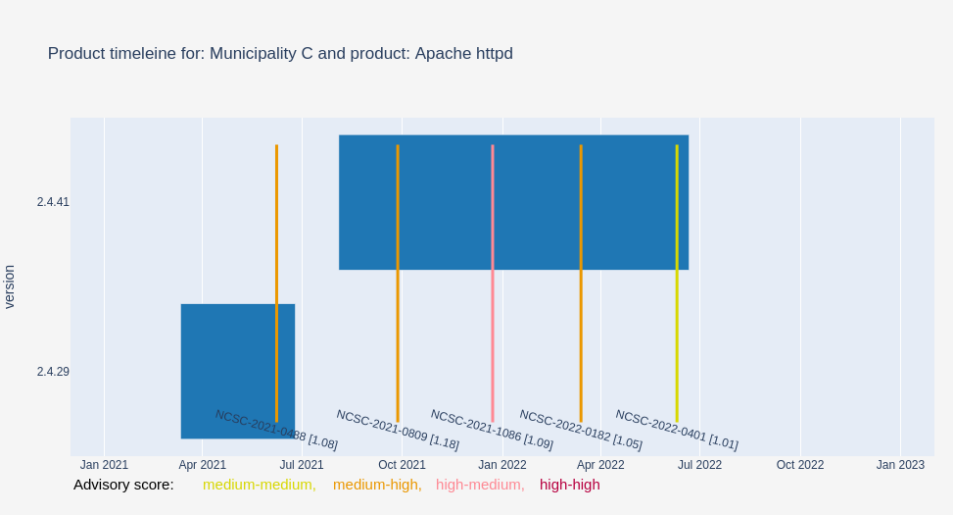
C

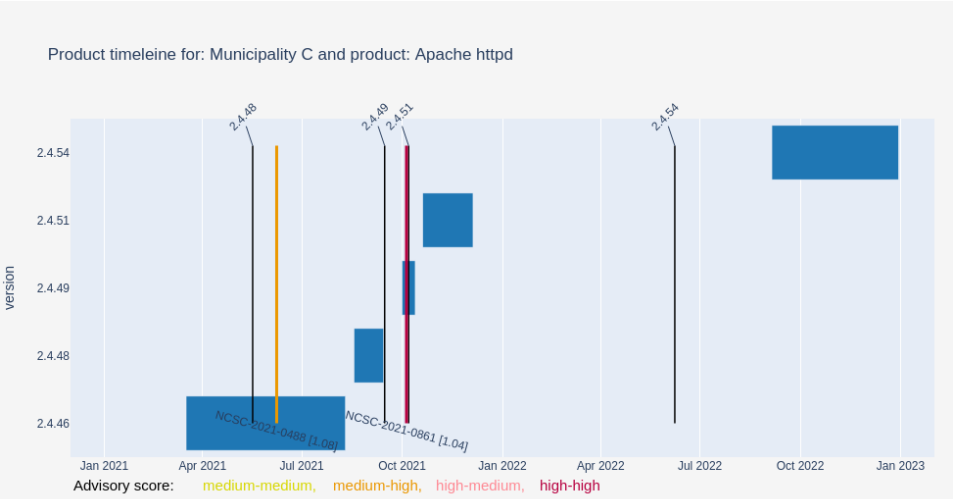
IP timelines for municipalities in 2021 and 2022

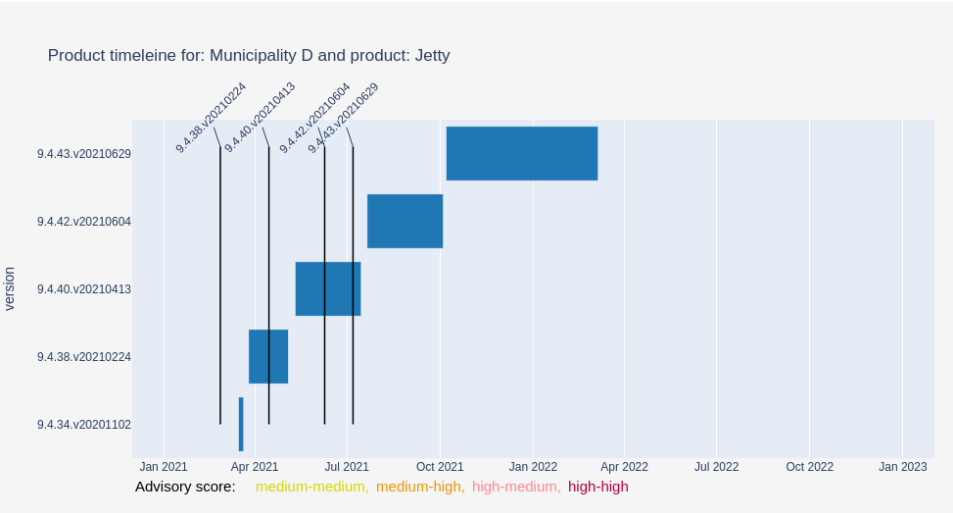
C.1. Update timelines

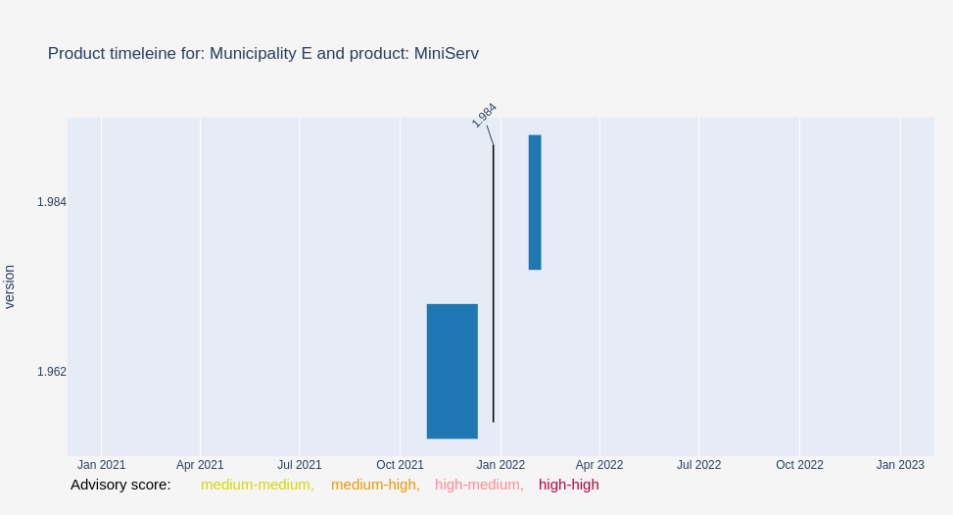


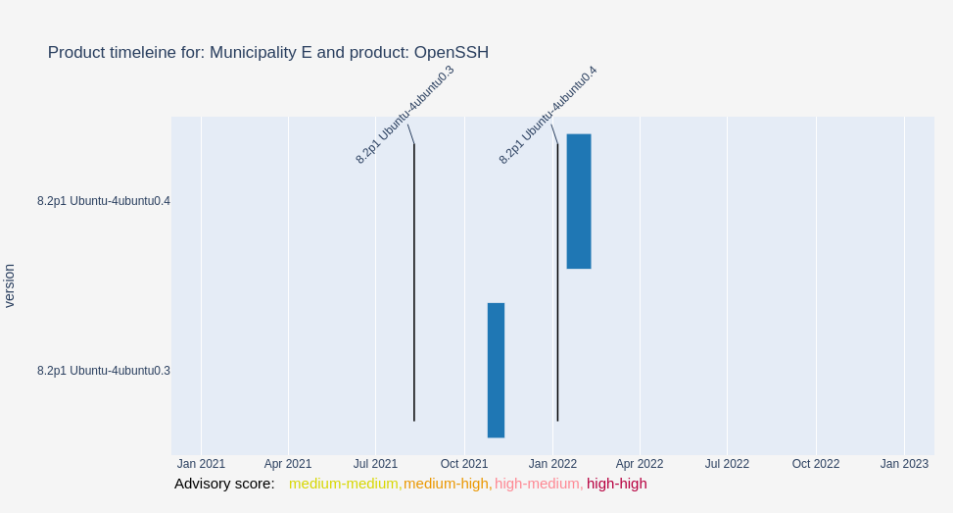


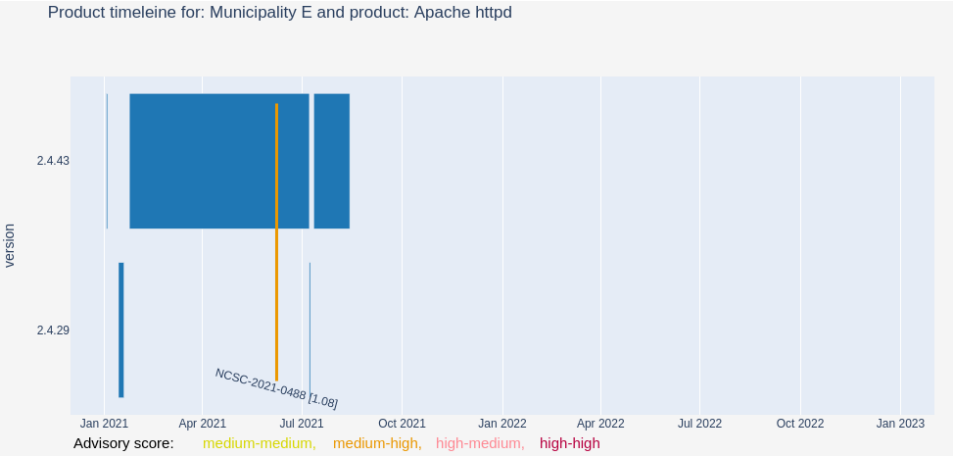

Product timeleine for: Municipality C and product: Apache httpd
version
2.4.41
2.4.29
NCSC-2021-0488 [1.08]
NCSC-2021-0809 [1.18]
NCSC-2021-1086 [1.09]
NCSC-2022-0182 [1.05]
NCSC-2022-0401 [1.01]
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

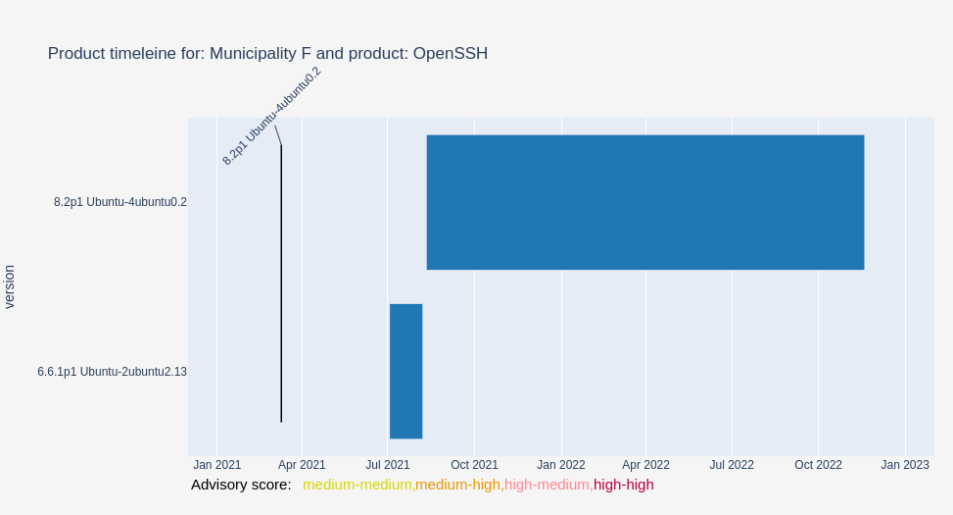

Product timeleine for: Municipality C and product: Apache httpd
version
2.4.54
2.4.51
2.4.49
2.4.48
2.4.46
2.4.48
2.4.49
2.4.51
2.4.54
NCSC-2021-0488 [1.08]
NCSC-2021-0861 [1.04]
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

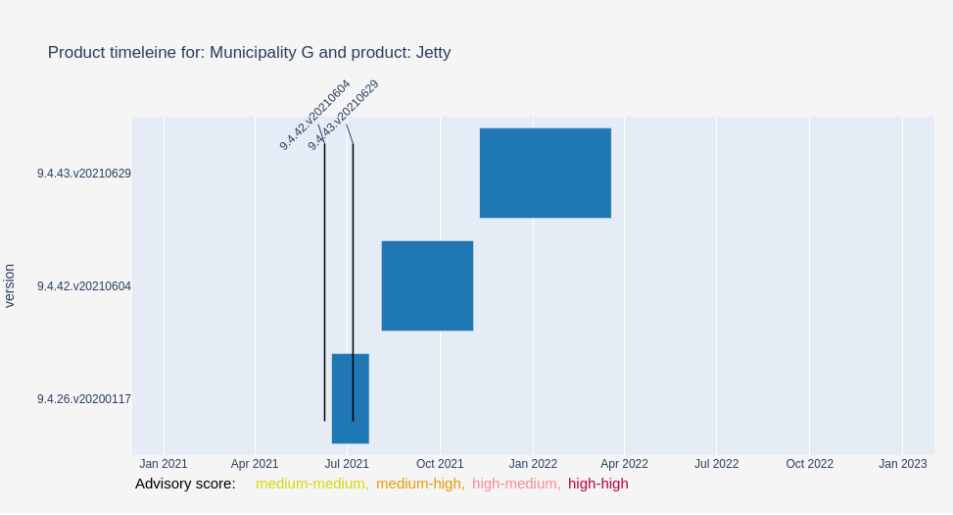

Product timeleine for: Municipality D and product: Jetty
version
9.4.43.v20210629
9.4.42.v20210604
9.4.40.v20210413
9.4.38.v20210224
9.4.34.v20201102
9.4.38.v20210224
9.4.40.v20210413
9.4.42.v20210604
9.4.43.v20210629
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

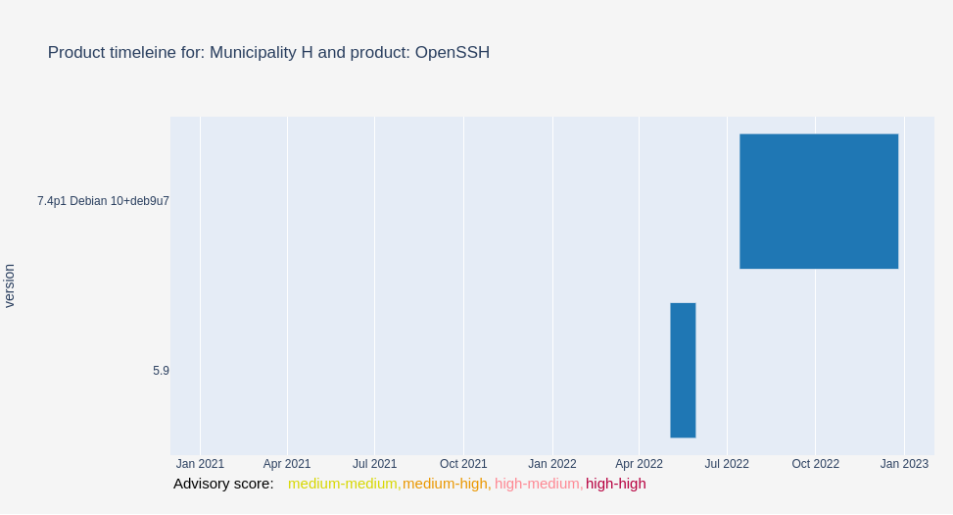

Product timeleine for: Municipality E and product: MiniServ
1.984
1.984
1.962
version
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

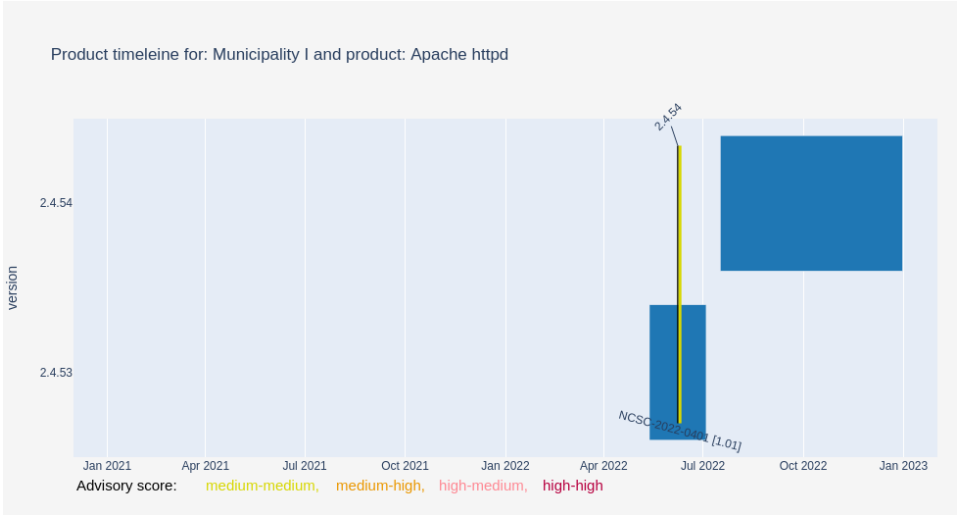

Product timeleine for: Municipality E and product: OpenSSH
8.2p1 Ubuntu-4ubuntu0.3
8.2p1 Ubuntu-4ubuntu0.4
8.2p1 Ubuntu-4ubuntu0.4
8.2p1 Ubuntu-4ubuntu0.3
version
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

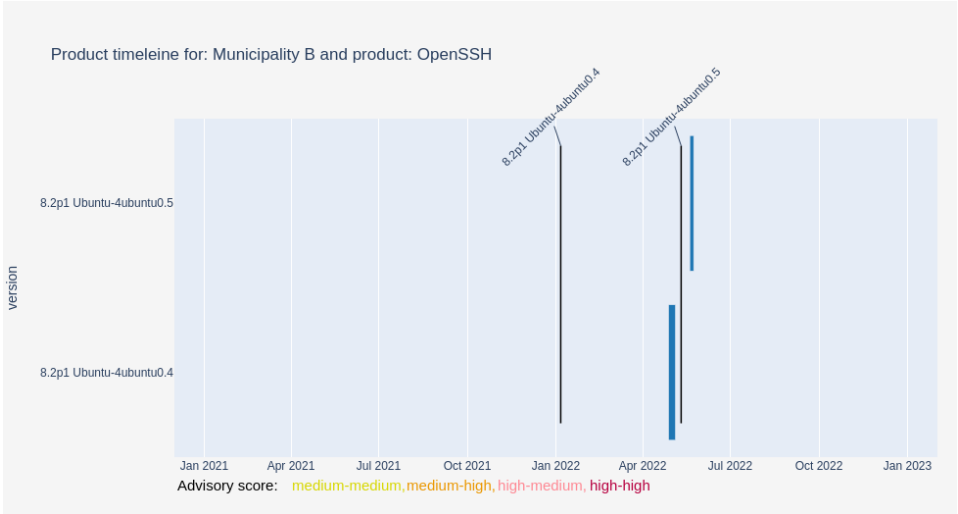

Product timeleine for: Municipality E and product: Apache httpd
2.4.43
2.4.29
NCSC-2021-0488 [1.08]
version
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

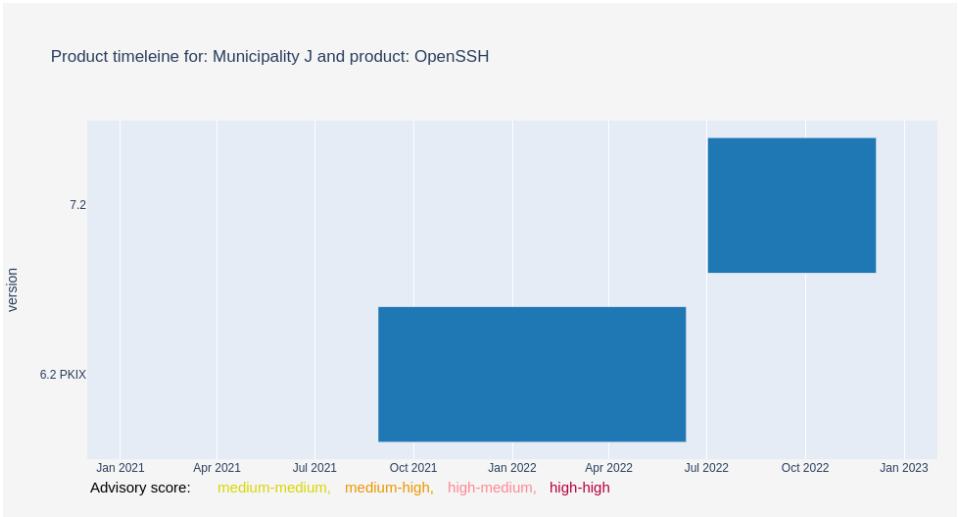

Product timeleine for: Municipality F and product: OpenSSH
8.2p1 Ubuntu-4ubuntu0.2
8.2p1 Ubuntu-4ubuntu0.2
6.6.1p1 Ubuntu-2ubuntu2.13
version
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium,medium-high,high-medium,high-high

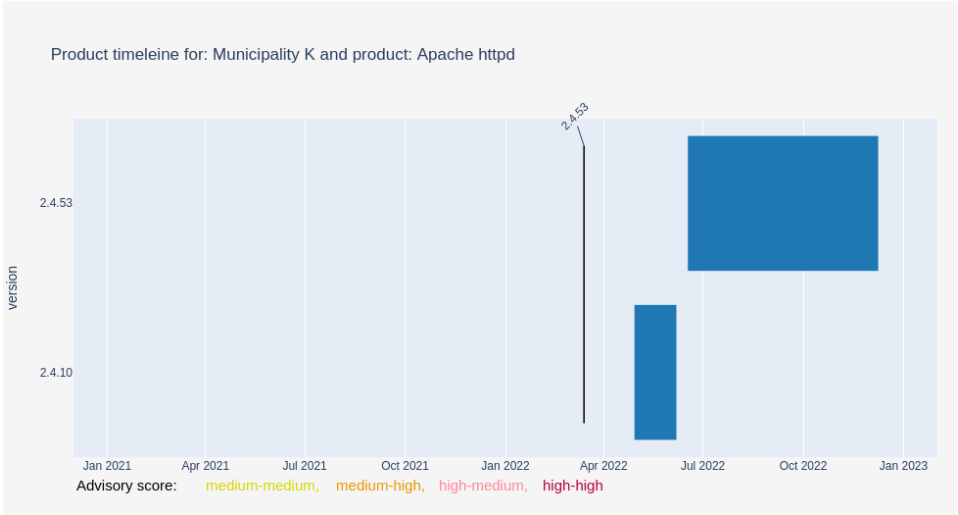

Product timeleine for: Municipality G and product: Jetty
9.4.42.v20210604
9.4.43.v20210629
9.4.43.v20210629
9.4.42.v20210604
9.4.26.v20200117
version
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

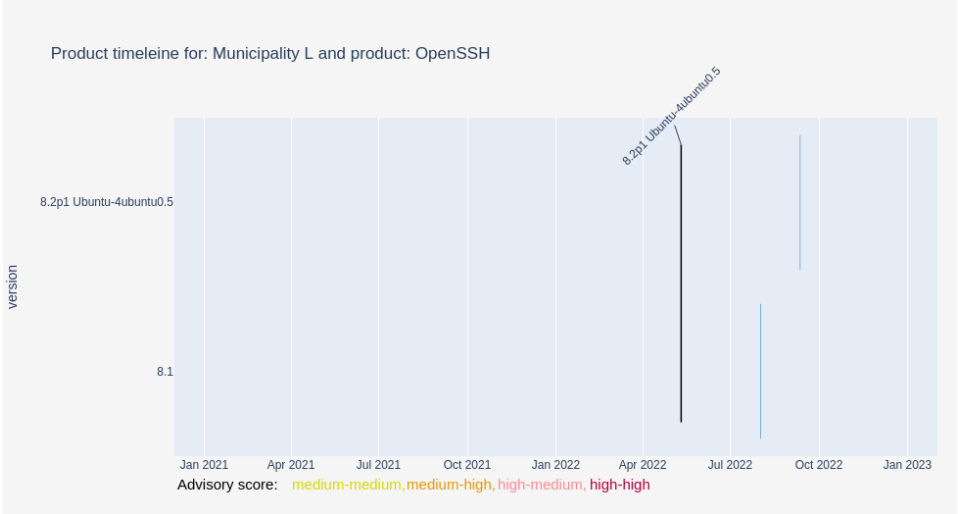

Product timeleine for: Municipality H and product: OpenSSH
7.4p1 Debian 10+deb9u7
5.9
version
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium,medium-high, high-medium, high-high

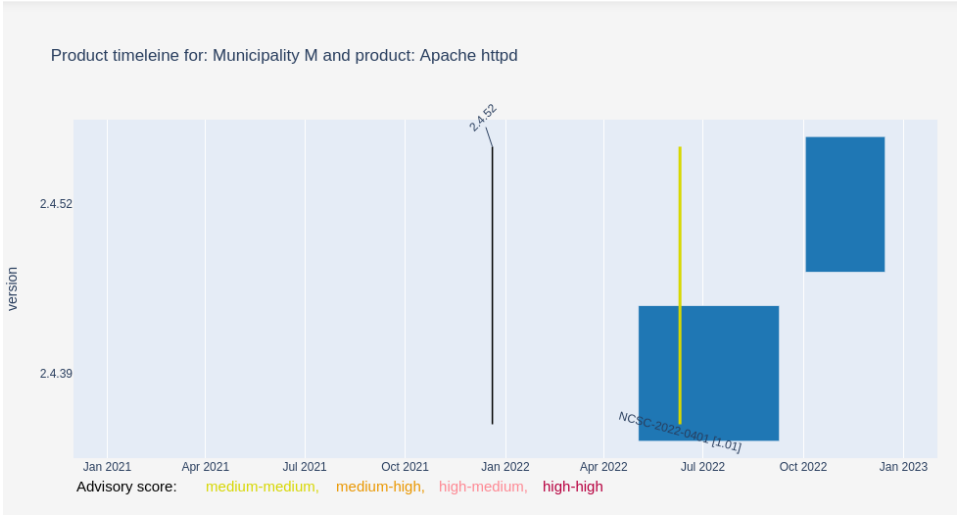

Product timeleine for: Municipality I and product: Apache httpd
version
2.4.54
2.4.53
2.4.54
NCSC-2022-0404 [1.01]
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

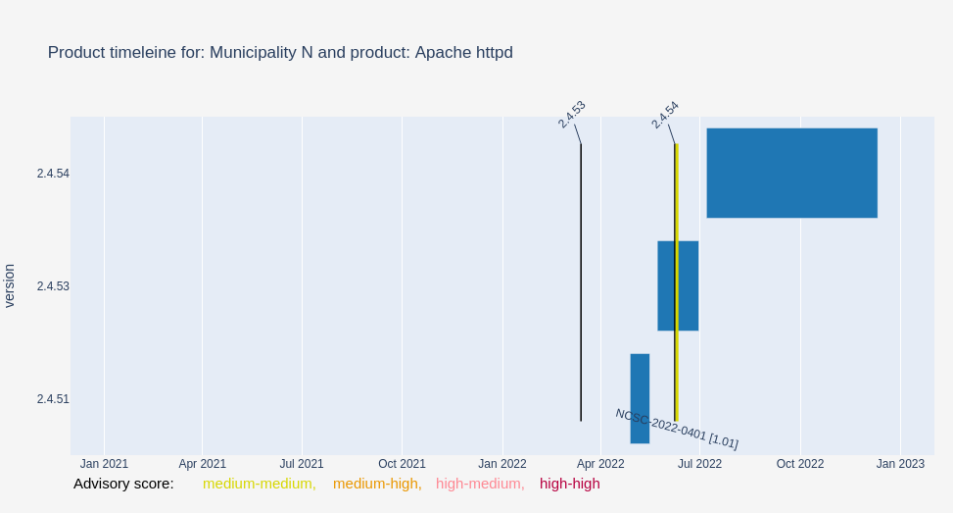

Product timeleine for: Municipality B and product: OpenSSH
version
8.2p1 Ubuntu-4ubuntu0.5
8.2p1 Ubuntu-4ubuntu0.4
8.2p1 Ubuntu-4ubuntu0.4
8.2p1 Ubuntu-4ubuntu0.5
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

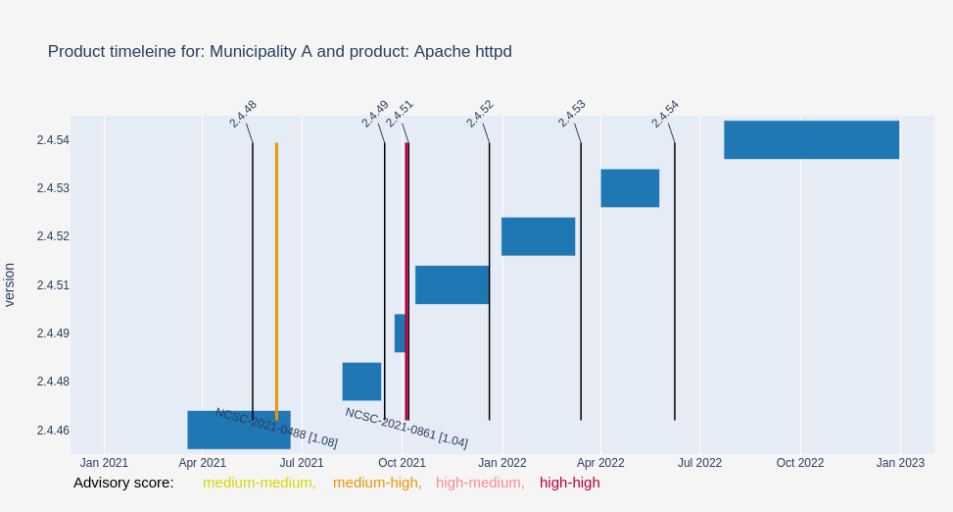

Product timeleine for: Municipality J and product: OpenSSH
version
7.2
6.2 PKIX
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

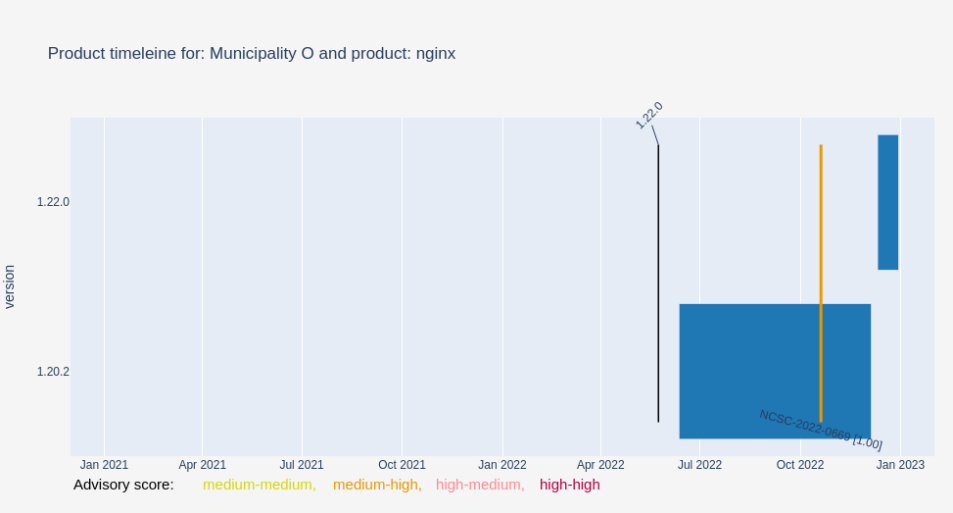

Product timeleine for: Municipality K and product: Apache httpd
2.4.53
version
2.4.53
2.4.10
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

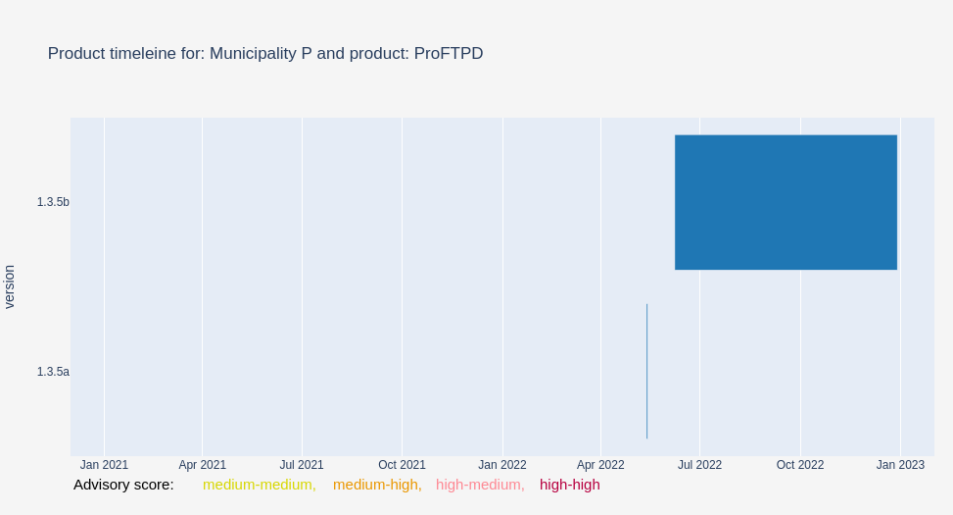

Product timeleine for: Municipality L and product: OpenSSH
8.2p1 Ubuntu-4ubuntu0.5
version
8.2p1 Ubuntu-4ubuntu0.5
8.1
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

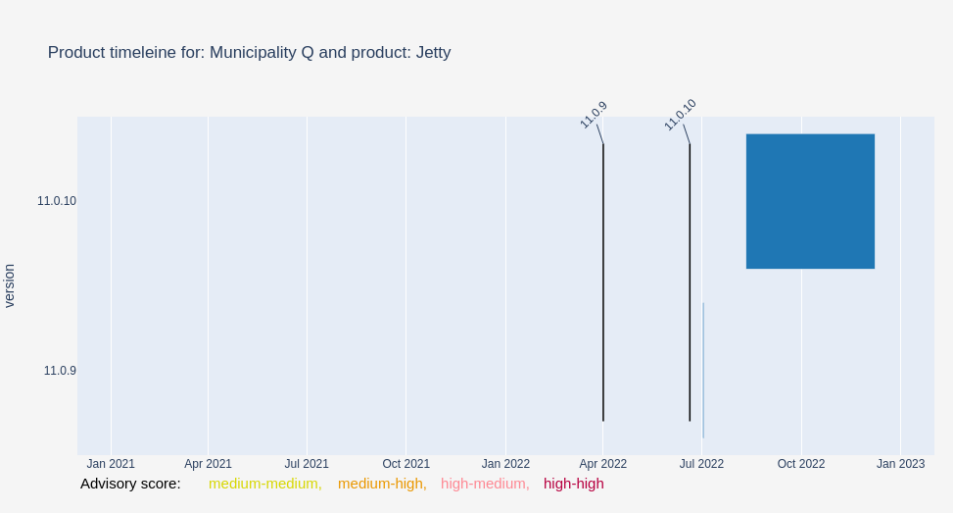

Product timeleine for: Municipality M and product: Apache httpd
2.4.52
version
2.4.52
2.4.39
NCSC-2022-0401 [1.01]
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

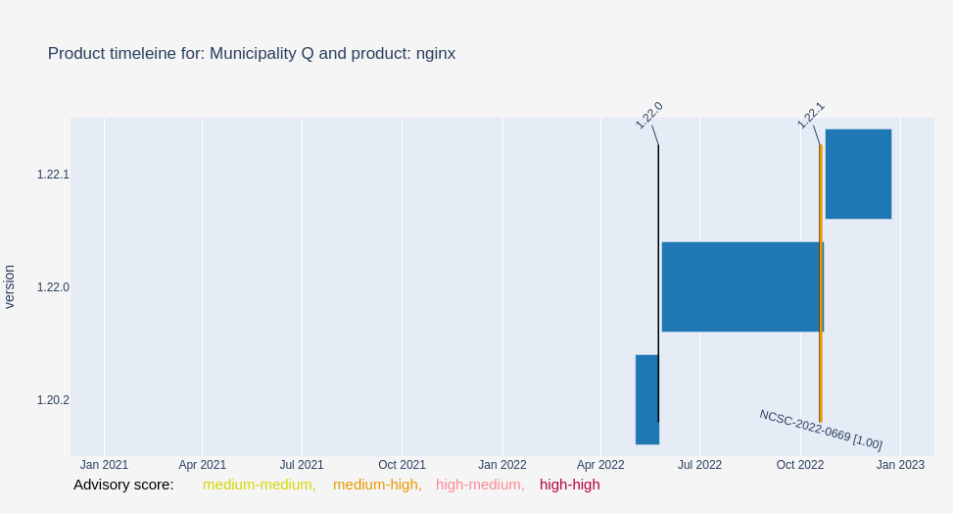

Product timeleine for: Municipality N and product: Apache httpd
2.4.53
2.4.54
2.4.54
2.4.53
2.4.51
version
NCSC-2022-0401 [1.01]
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

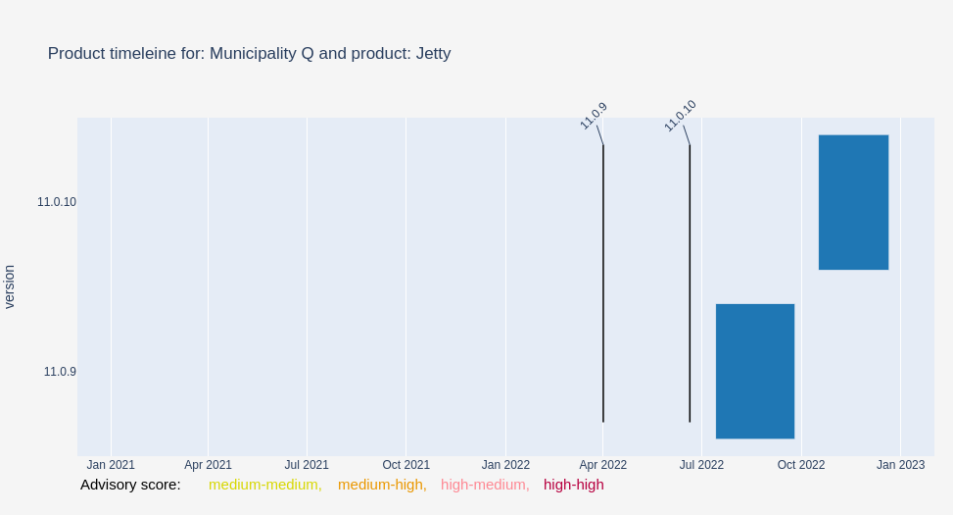

Product timeleine for: Municipality A and product: Apache httpd
2.4.48
2.4.49
2.4.51
2.4.52
2.4.53
2.4.54
2.4.54
2.4.53
2.4.52
2.4.51
2.4.49
2.4.48
2.4.46
version
NCSC-2021-0488 [1.08]
NCSC-2021-0861 [1.04]
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

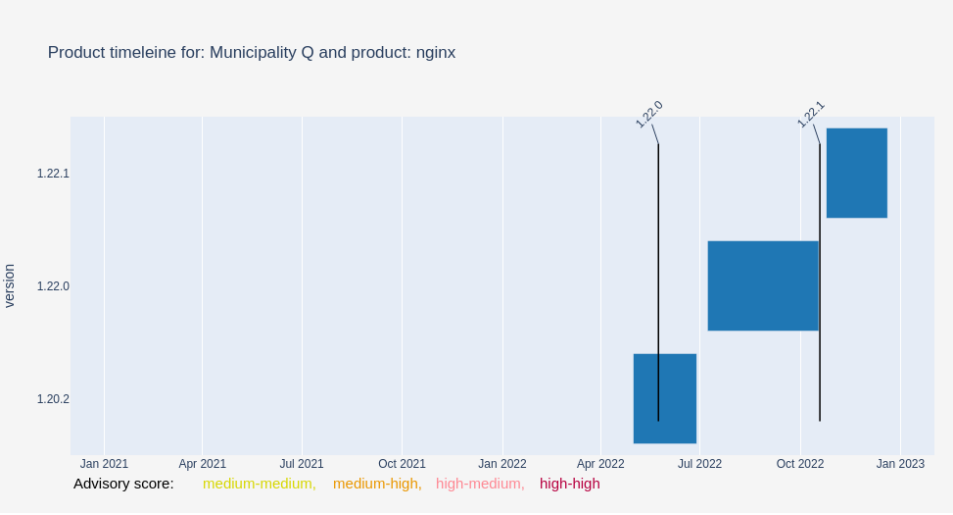

Product timeleine for: Municipality O and product: nginx
1.22.0
1.22.0
1.20.2
version
NCSC-2022-0669 [1.00]
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

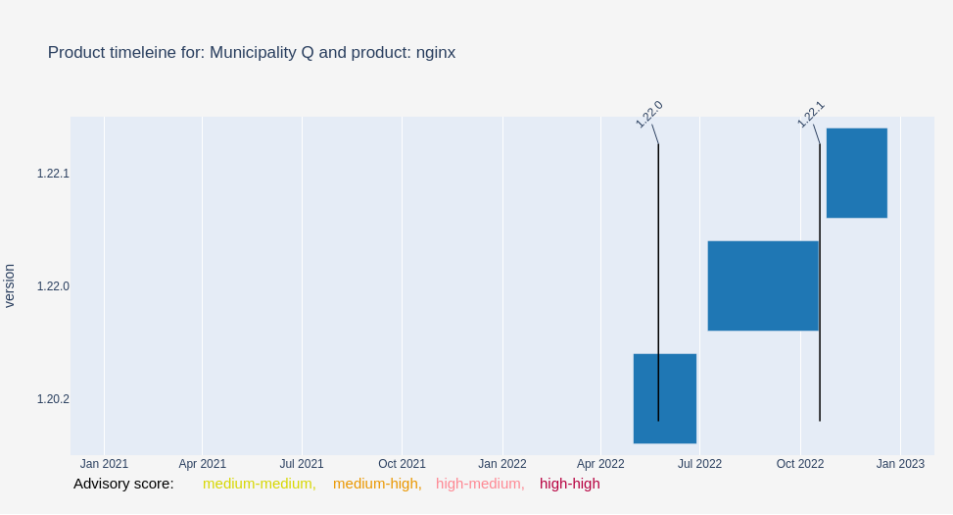

Product timeleine for: Municipality P and product: ProFTPD
version
1.3.5b
1.3.5a
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

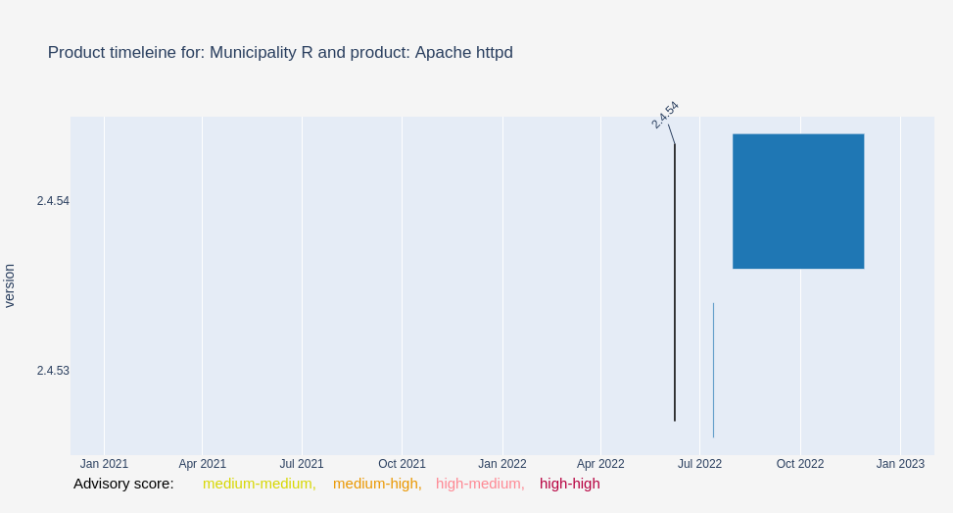

Product timeleine for: Municipality Q and product: Jetty
version
11.0.10
11.0.9
11.0.9
11.0.10
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

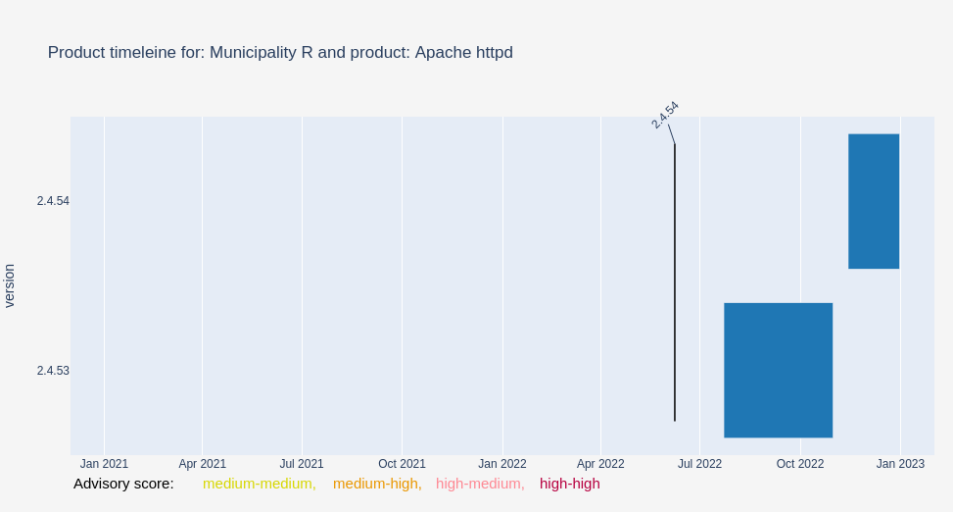

Product timeleine for: Municipality Q and product: nginx
version
1.22.1
1.22.0
1.20.2
1.22.0
1.22.1
NCSC-2022-0669 [1.00]
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

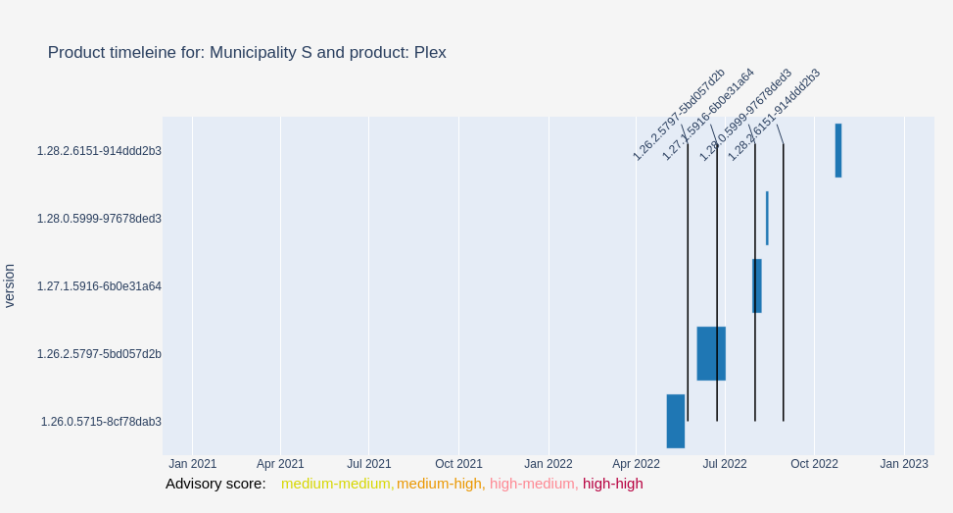

Product timeleine for: Municipality Q and product: Jetty
11.0.9
11.0.10
version
11.0.10
11.0.9
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

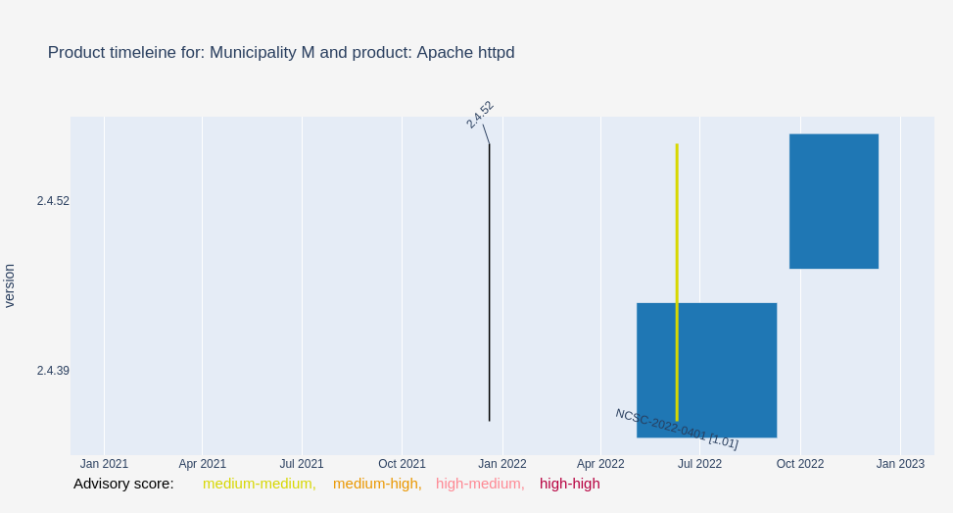

Product timeleine for: Municipality Q and product: nginx
1.22.0
1.22.1
version
1.22.1
1.22.0
1.20.2
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

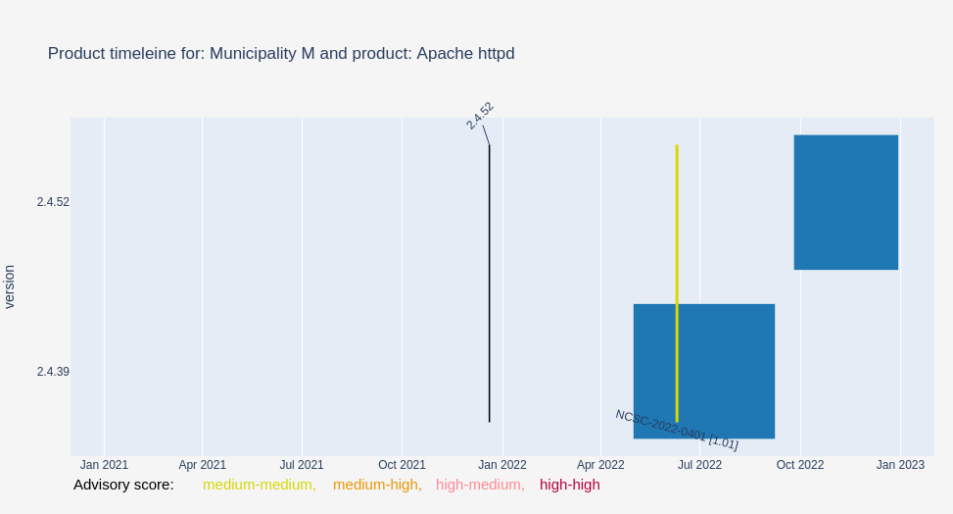

Product timeleine for: Municipality Q and product: nginx
1.22.0
1.22.1
version
1.22.1
1.22.0
1.20.2
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high


Product timeleine for: Municipality R and product: Apache httpd
2.4.54
version
2.4.54
2.4.53
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

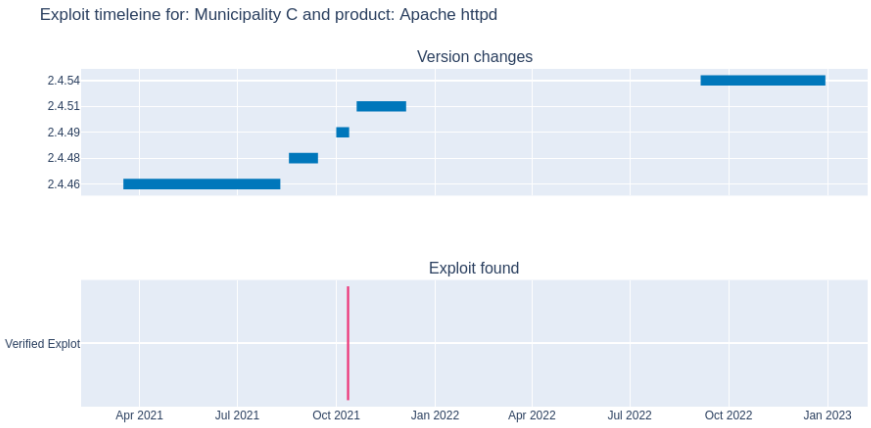

Product timeleine for: Municipality R and product: Apache httpd
2.4.54
version
2.4.54
2.4.53
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

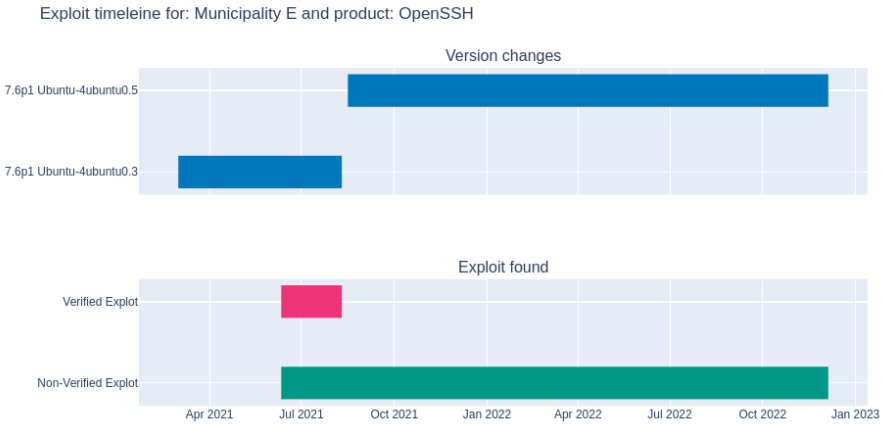

Product timeleine for: Municipality S and product: Plex
1.26.2.5797-5bd057d2b
1.27.1.5916-6b0e31a64
1.28.0.5999-97678ded3
1.28.2.6151-914ddd2b3
version
1.28.2.6151-914ddd2b3
1.28.0.5999-97678ded3
1.27.1.5916-6b0e31a64
1.26.2.5797-5bd057d2b
1.26.0.5715-8cf78dab3
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

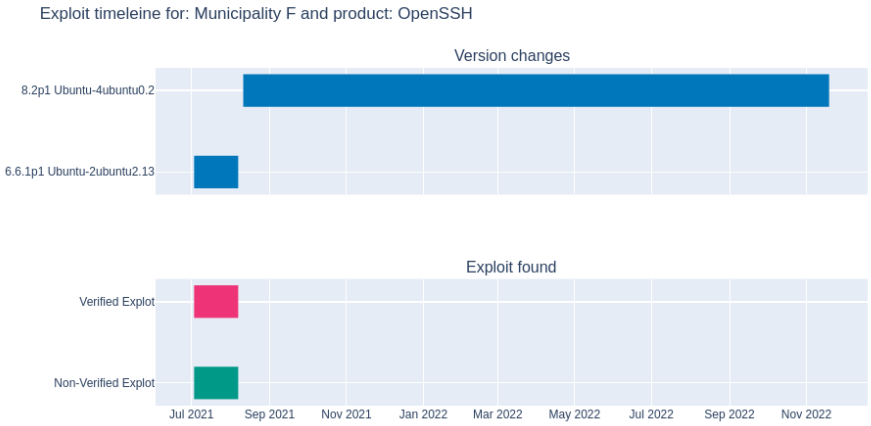

Product timeleine for: Municipality M and product: Apache httpd
2.4.52
version
2.4.52
2.4.39
NCSC-2022-0401 [1.01]
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high


Product timeleine for: Municipality M and product: Apache httpd
2.4.52
version
2.4.52
2.4.39
NCSC-2022-0401 [1.01]
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Advisory score: medium-medium, medium-high, high-medium, high-high

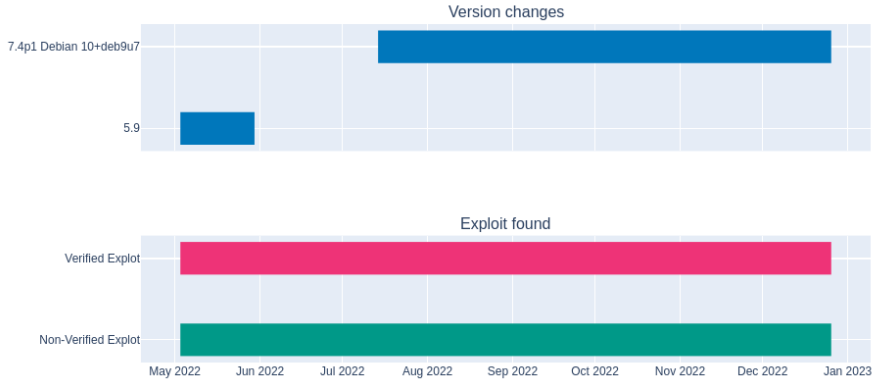
C.2. Exploit timelines



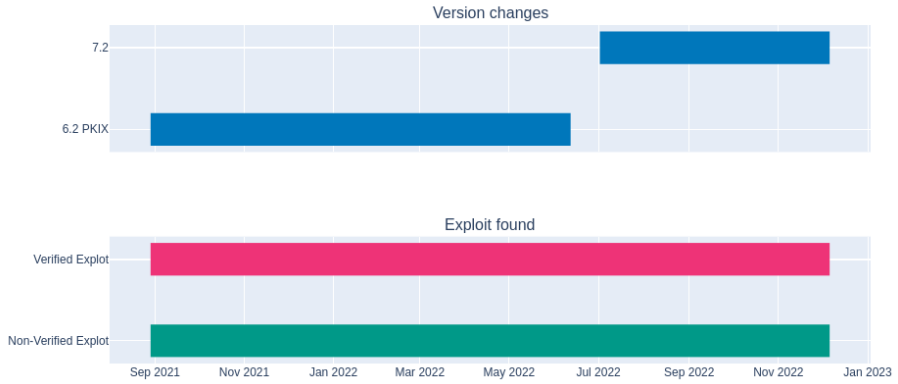




Exploit timeleine for: Municipality H and product: OpenSSH



Exploit timeleine for: Municipality J and product: OpenSSH



Exploit timeleine for: Municipality A and product: Apache httpd