MSc thesis in Computer Science

Correlation of Massively Distributed Scanners

Andy Chiu

August 2024

A thesis submitted to the Delft University of Technology in partial fulfillment of the requirements for the degree of Master of Science in Computer Science

Abstract

Cyber attacks have become increasingly more prominent and the associated cost to society is by several estimates reaching trillions of US dollars. A typical cyber attack goes through the several consecutive phases of the cyber kill chain. As a precursor for any attack, the malicious actor performs network reconnaissance in order to identify potential entry points through exploitable services connected to the internet. Therefore, early detection of reconnaissance by scanners can mitigate or entirely prevent future attacks. Modern intrusion detection systems are capable of blocking some scan attempts. However, more sophisticated and resourceful attackers are suspected to distribute their efforts over a large number of sources. This allows them to lower the individual scanrate while achieving the same throughput. Slow scanners are harder to detect, because they differentiate little from baseline noise levels. Additionally, a threat is severely underestimated if a large number collaborating scanners is not treated as a single entity. Therefore, the aim of this thesis is to infer such a coordinated relationship between scanners controlled by a single initiator.

We analyse the data from a network telescope with an observation of one year, much longer than previous work. Initial analysis led to the discovery of similar long-term activity patterns present in distributed scanners. These patterns can be used to uniquely identify a group which formed the basis for the correlation algorithm. We were successfully able to detect a large number of clusters employing various strategies in terms of size, scanrate and targeted services. Due to the absence of ground truth additional effort has been spent to validate potential clusters through other characteristics. We also demonstrate the utility of transforming the raw telescope data for cluster analysis through a case study of very slow scanners.

Contents

List of Figures

1 Introduction

In an increasingly information technology reliant society there is a need for defenders to obtain high situational awareness of malicious activity conducted in cyberspace. The reason being that a devastating cyber attack is often preceded by some form of reconnaissance. The purpose of gathering information about the target is to find the path of least resistance in order to achieve its goal like stealing sensitive data or full system compromise. More specifically in the context of the public internet, it is common to see a series of connection attemps known as scans originating from a small number of sources that enumerate all internet addresses in search for vulnerable devices or services. These vulnerabilities could be the result of bad security practises, misconfigurations, absence of security patches among other reasons. This enumeration of internet addresses is illustrated in Fig 1.1. The attacker is scanning a subset of all possible Internet Protocol (IP) addresses for open ports of interest. An analogy is calling every number combination using a phone. If a conversation is established then one can infer that someone owns that number, such information can then be used for phishing attacks.

Out of convenience even regular households tend to incorporate many ”smart” solutions, but consumers generally care more about functionality and are rather oblivious to the security hazards. As a result, insecure devices connected to the internet have become ubiquitous. Malicious actors are eager to find these using automated scanning tools for their own benefit. Typically, once any device is connected to the internet it will start receiving unsolicited scans from all around the world. Therefore, without proper device hardening attackers can completely take over to steal sensitive information, gain unauthorized access and even include the victim to a large pool of compromised hosts known as botnets.

Scanning as a means for conducting reconnaissance is the first phase in the so-called cyber kill chain depicted in Fig 1.2 which is a framework that describes the various phases of an

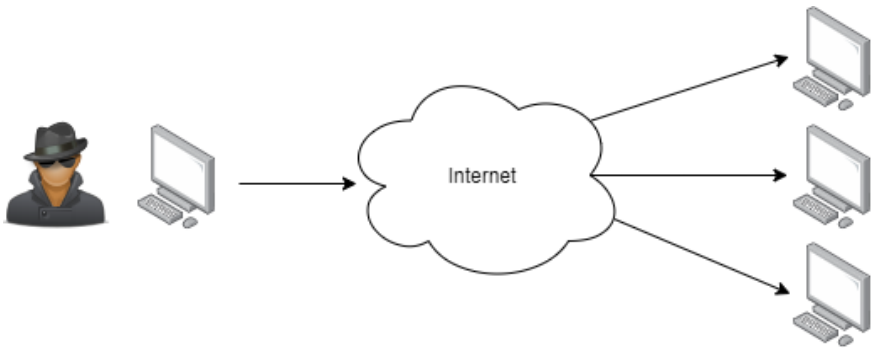

Figure 1.1: Internet wide port scanning

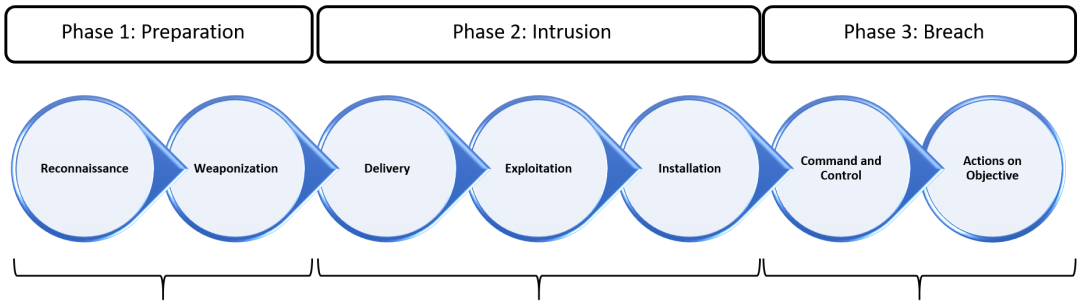

Figure 1.2: Phases of the Cyber Kill Chain

attack. Controls and defences can be installed at each phase to disrupt such attacks. The attacker is only successful if it can go through all the phases uninterrupted. This thesis focusses on the very first phase, namely reconnaissance. Hardening defences that inhibit the attackers from discovering potential entry points will contribute to a more secure network. The less intelligence an unauthorised entity can gain about the network the better. That is not to say that security can be achieved solely by obscurity, but it is one of the many measures that can be implemented throughout the kill chain to mitigate potential attacks. It is generally a good security practise to not only rely on a strong outer perimeter, but to provide defence in depth. Subsequent stages in the cyber kill chain identify how the attacker will compromise its target, gain control and finally reach its objective. This framework is applicable to a large variety of modern cyber attacks. The abstract stages might take different forms depending on the scenario, but still the framework remains useful in the identification, prevention and discussion regarding cyber intrusions activity.

Though not as prevalent as today, the practise of scanning has existed as long as the early days of the internet more than two decades ago. Since then, more resourceful and intelligent attackers have adopted sophisticated obfuscation strategies to remain undetected. One such approach is to divide the scans over multiple sources to achieve a lower scan rate for each individual scanner, see Fig 1.3. This method is effective at circumventing detection, because the scanner's generated traffic volume blends in with regular internet noise levels and thus avoids suspicion. Whereas a single source scanning at high speed will definitely stand out, multiple slow scanners will go unnoticed as current detection mechanisms are unable to correlate these to a single group. As a result, the larger and more advanced threats continue to escape the security communities' attention. Seemingly unrelated scanning sources may actually be part of a large campaign under the control of one entity. Without further research into the detection, correlation and employed strategies of massively distributed scanners we severely underestimate the capabilities of today's cybercriminals. Furthermore, the estimated risk in the current threat landscape would be inaccurate due to the inability to perceive distributed scanners. A better understanding of scanning behavior is beneficial to the design of new defence mechanisms, can provide early warning signals of incoming attacks and better cyber risk estimation. Preventing the attacker at the reconnaissance phase will mitigate the impact later on in the cyber kill chain or prevent an attack entirely.

Organizations of all sizes and even consumers have to deal with these scans in one way or another as literally every host connected to the internet is a potential target. The governmental department of defence has expanded its cyberspace presence to provide early threat intelligence of incoming attacks that may hurt the nation's safety or economy and are in

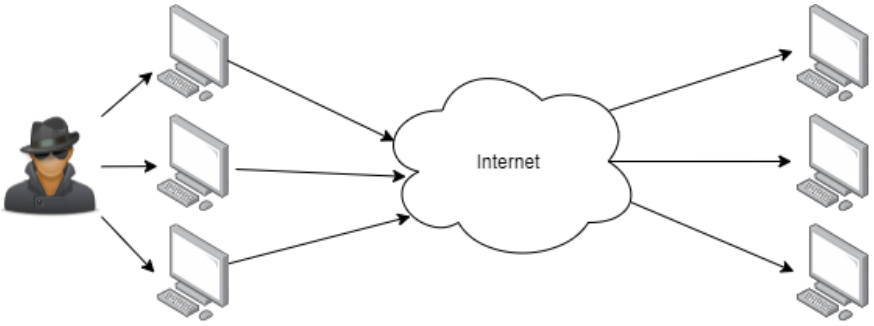

Figure 1.3: Attacker controlling multiple scanners

direct contact with many large organizations. Security researchers develop novel methods and analyse data to create a better understanding of scanners. Finally, there are companies providing security solutions to protect both organisations and consumers.

One way of observing internet-wide scanning activity is through a network telescope or also referred to as darknet which monitors the packets received from a range of unused IP addresses. Packets received at these addresses can all be considered suspicious, because they are unsolicited. The majority of these are scan traffic, but may also be the consequence of server misconfigurations and backscatter of ongoing attacks elsewhere with spoofed IP addresses. As such, a network telescope can provide the data to obtain valuable insights to trends occurring at internet scale by monitoring a only small subset of the entire IPv4 space.

The challenge is to infer knowledge about the attacker's methods from a large volume of data recorded at the telescope. While studies do mention the existence of distributed scans they often limit their work to single source scanners. The reason mostly likely being that it is hard to establish sufficient confidence that a group of scanners is indeed cooperating. Only the attacker knows for sure and it is not possible to prove collaboration among scanners from just the network telescope data. Furthermore, the sheer volume can make analysis computationally infeasible unless pre-filtering methods are used to reduce of the data size. For these reasons there is not much literature on the topic of distributed scan detection. Despite these challenges we are attempt to infer coordination among scanners at an acceptable level of certainty. We observe scanners from the same group to be highly similar in certain aspects of their scan behaviour. This observation provides the foundation for a new detection method that is able to correlate seemingly unrelated distributed scanners.

2 Research Questions

Main research question
How can distributed scanners be detected with sufficient confidence from prolonged observation?

These can be broken up into three sub-questions:

1. How can large-scale data captures be substantially compressed for scan traffic analysis?
2. How can we infer coordination among scanners that are part of the same group ?
3. To what extent can we confidently determine collaboration among scanners ?

3 Background

The purpose of this section is to provide fundamental background information that will help the reader understand the remainder of the thesis. General networking concepts are briefly touched followed by an another definitio nof scanning practises.

3.1 TCP/IP reference model

Nowadays we rely on communication over the internet as an necessity for a variety of tasks. Whether it be for work, some stress relieving moments of entertainment or to have a conversation with family, it should just work seamlessly and effortlessly. This is being made possible by standardizing a set of communication protocols to establish a common language among all network devices. The TCP/IP model is a set of such communication protocols and the core of modern internet. There is a subdivision of four layers, three less than the classical Open Source Interconnection(OSI)-model. The latter was designed to describe the functions of the communications system into smaller and simpler components, but remains merely a conceptual model. Figure 3.1 illustrates the different layers in both the OSI and TCP/IP reference model.

Each layer adds another level of abstraction and is responsible for its own part in the process of getting data from sender to receiver. Software applications used by the end-user wrap their data in the application layer. These include services like email, web browsing and file transfer provided by an external party, the relationship of both parties can typically be described by a client-server model. It is in this layer that actually contains the productive data of interest that must be sent. All subsequent layers encapsulate the data, adding meta-data which is necessary for the data to be delivered to the intended destination. The transport layer is responsible for host to host communication with direct interface to the application. Depending on the requirements and type of application, either Transmission Control Protocol(TCP) or User Datagram Protocol(UDP) is commonly selected. TCP provides functionality for reliable transmissions, but adds overhead and latency due to the stateful nature of the protocol. Both ends of TCP communication will need to formally establish two-way communication and track the reception of outstanding data packets as they are sent. In certain applications simpler communication with low latency is preferred at the risk of losing some packets in transit, in this scenario UDP is the better option. The interface to the transport layer is by communicating through ports that are opened by the operating system. All major operating systems support TCP and UDP. The internet layer enables sending a packet from one host to another. Host can reside in the local network or the public internet. Typically, this is done by forwarding packets through the shortest path along the internet routers distributed across the world. Each router along that path will know where to forward packets based on the 32-bit destination IP address, it will hop from one router to the next until the destination has been reached. Arriving at the link layer, which processes packets for direct communication between two adjacent networking devices. So there is a direct link between

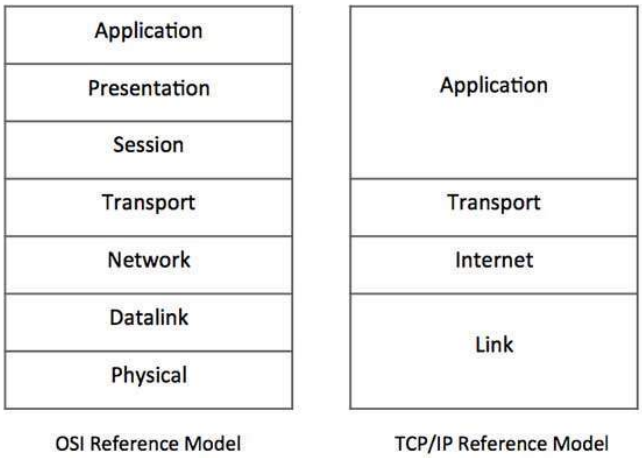

Figure 3.1: OSI and TCP reference model

two points with no other devices in between, using a transfer medium like a cable or electromagnetic waves. The link layer determines how much information can fit in one logical transmission unit also referred to as Maximum Transmission Unit(MTU). In effect, the data in upper layers have to be fragmented into sizeable chunks below the MTU. In the discussion on scanners both the internet and the transport layer are of particular interest, because attackers will modify the header values in the corresponding protocols to scan their targets. A target host that has a port open on a common port number is enough for the attacker to infer that a service of interest is accepting remote connection request. An additional step by more sophisticated scanners can be taken to verify what service is running by sending application specific queries or commands to the application layer. This can also reveal additional information like the version number if the attacker is looking to exploit those hosts running an older version known to contain vulnerabilities. However, this thesis analyses TCP scans received by a network telescope consisting of a range of unused IP addresses and thus an active TCP connection will never be established. From this perspective scanners are observed through one-way communication and no interaction will take place. Furthermore, there as many protocols in the application layer as there are actual applications so it would take great effort to correctly interpret this. Creating an understanding of how scanners search through a local network or subset of internet addresses for host running open ports can be done by analyzing network traffic at the internet and transport layer. For this reason, the IPv4 and TCP protocol are discussed next in more detail.

IPv4 The Internet Protocol version 4 (IPv4) is used at the internet layer. It originated as a protocol designed by the United States Department of Defence implemented at the Advanced Research Projects Agency Network (ARPANET) in 1983. Its primary task is to provide logical addressing with 32 bits and route packets to their correct destination through a sequence of hops. With each hop, from one router to another, the packets gets closer to its destination. The protocol provides best effort to deliver packets, but does not guarantee and some packets will inevitably get damaged or lost. If guaranteed delivery is crucial to the correct operation of the intended application then this should be checked in the higher layer protocols such as TCP which will ask for re-transmissions of packets. While the successor IPv6 is gradually but slowly being deployed, IPv4 still remains the protocol of choice due to

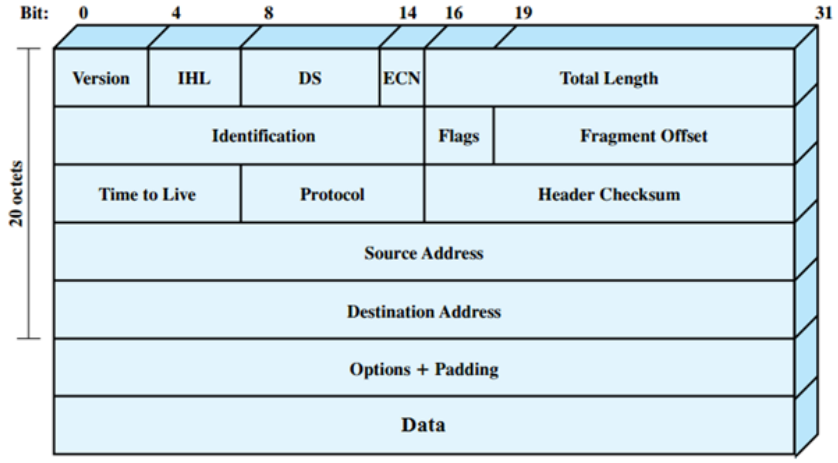

Figure 3.2: IPv4 Header

its wide adoption. Besides destination address there are other pieces of information present in the IPv4 header. The header is a fixed format collection of values that adheres to the protocol standard. It contains meta-data, information regarding the data itself and how it should be processed. The fields from the IPv4 header in Fig 3.2 will be briefly summarized.

From table 3.3 it can be seen that certain fields can be exploited by the attacker to achieve reconnaissance goals. Generally all possible destination addresses are scanned in order to maximize the chances of finding vulnerable hosts on the internet. The source address can even be spoofed to impersonate other computers if no packet filtering is in place. However, with scanning the attacker is particularly interested in replies from potential victims. The source address will likely to be of the scanner itself or the reporting server which has been designated to collect all replies. A single IP packet is sufficient to probe a target given that it does not get lost or corrupted. This leaves room to utilize the identification field for other purposes as long as it will not get rejected by the recipient. Some scanning tools like ZMAP Durumeric et al. [2017] assign a static value that allow network operators to easily identify scan attempts. The creators of ZMAP and other open source tools have no malicious intent. On the other hand, it is not in the cyber criminal's interest to announce its presence. This group is more likely to use the freed 16 bits for their own book keeping of outgoing packets when scanning at high speeds.

TCP protocol TCP is the transport layer protocol that provides communication as a reliable, ordered and error-checked data stream. It serves to exchange data in a traditional client-server model. The server offering a particular service listens on an open port for incoming connection request. A client that wishes to establish a connection will initiate a so-called three-way-handshake. The handshake comprises of three packets, two from the client and one from the server as is illustrated in Fig 3.4. For the first packet the client presents itself to the server by setting the SYN flag in the TCP packet which indicates a new connection request. Assuming the server has enough capacity it will respond by returning a packet with the both the SYN and ACK flags set. All that remains is the final acknowledgement through an ACK packet from the client to establish a connection where from hereon

Field Name	Length	Description
Version	4 bits	Specifies which IP version is used
Internet Header Length(IHL)	4 bits	Specifies the total header byte size as a multiple of 32
Differentiated Services(DS)	6 bits	What service is being used, takes effect only in differentiated service model
The Explicit Congestion Notification	2 bits	Used to detect and notify congestion ahead in the network
Total Length	16 bits	The total length of both header and data combined in bytes
Identification	16 bits	Used by the recipient to reassemble fragments of a message
Flags	3 bits	Two individual bits that control data fragmentation, the last one is reserved
Fragment Offset	13 bits	Specifies the location of a fragment in the packet
Time to Live	8 bits	Specifies how many hops this packet can make before being discarded
Protocol	8 bits	Specifies which protocol is used such as TCP or UDP
Header Checksum	16 bits	Used for checking errors in the header occurred during transmission
Source Address	32 bits	Contains the address of the sender
Destination Address	32 bits	Destination address where the packet should be delivered
Options	? bits	For specifying additional options or features that can be used by hosts that support them
Padding	0 - 31 bits	A sequence of zeros added to the options field to make a multiple of 32 bits

Figure 3.3: IPv4 Header fields

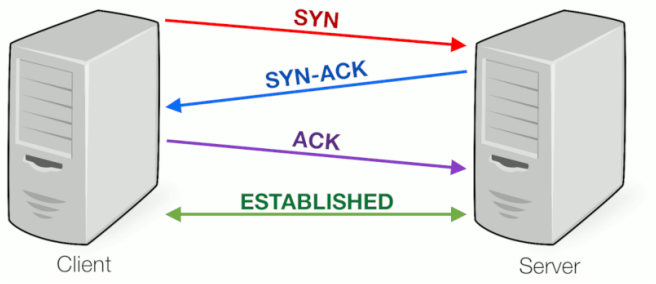

Figure 3.4: TCP Three-Way-Handshake

data can be exchanged in both directions. The contrast bewteen TCP and UDP is evident by the presence and absent of such a handshake. The latter does not manage states of active connection and misses important features that ensure reliable transmission.

As is the case in the Internet Layer and the corresponding IP protocol, TCP packets also contain header fields that are necessary for the correct operation of the protocol. The size of the header varies between 20 and 40 bytes depending on the presence of options, followed by the actual data to be transmitted. Fig 3.5 contains all fields of the header and their corresponding length in bits. 3.6 provides a brief description on all fields and we will later emphasizes those that are relevant in the discussion on reconnaissance.

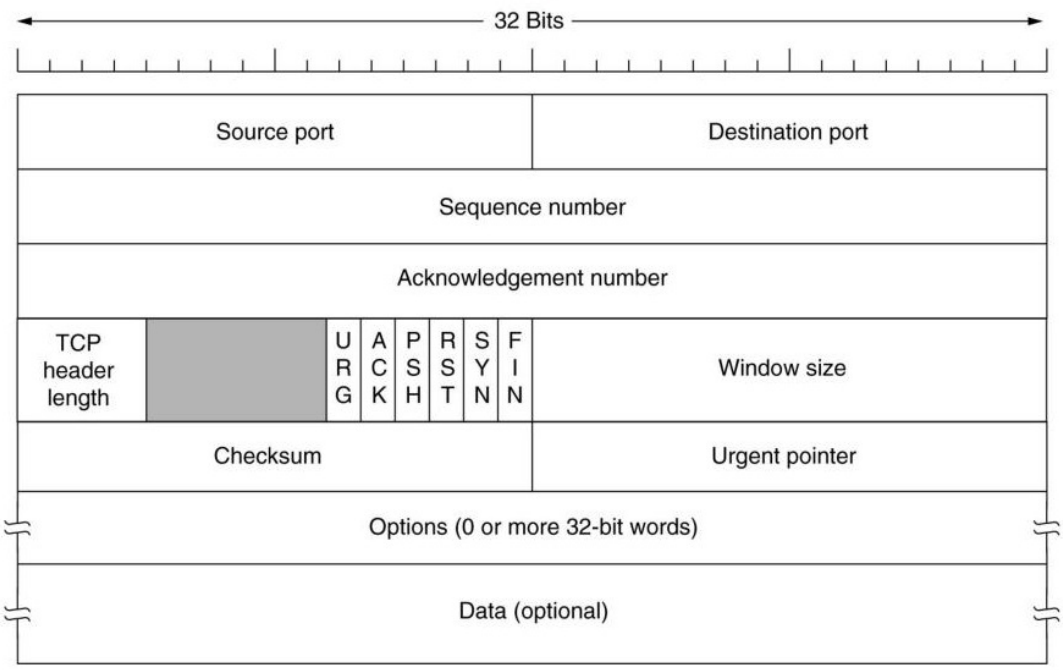

Figure 3.5: TCP Header

3.1.1 Scan types

Having provided the full TCP header specification, we proceed to discuss how attackers can manipulate certain fields to perform reconnaissance. Several scan types have been described in literature that can be characterised by the combination of set flags in the header and which connection states are visited.

TCP Connect One way to infer that a service is running at a particular port is by completing the full three-way-handshake. This is implies that some service is able to process that request and is ready to accept incoming connections. An attacker proceeds to discover any open ports of interest by enumeration, the three-way-hand shake is attempted on each of the ports where the successful attempts are reported back. This methods is effective, because TCP listener processes always follow the protocol by acknowledging incoming packets with the SYN flag set. However, there are two drawbacks to this approach. One is that three packets are required to infer an open port, much less efficient than the TCP half-open method discussed in the next paragraph. Second, completed handshakes leave traces at the server in the form of logs which will reveal the presence of scan attempts. Generally as an attacker it is beneficial to operate as stealthily as possible to avoid suspicion from the target. Defenders who become aware of incoming attacks might deploy additional security measures that will decrease the adversary's chances of success. The presence of logs also provide insight into the attacker's methods through post-attack analysis. A benefit of this scan type is its simplicity, no modification of TCP/IP stack is necessary as opposed to the half-open connection discussed next.

Field Name	Length	Description
Source Port	16 bits	Identifier used by the sender
Destination port	16 bits	Identifier used by the destination to listen for incoming connections
Sequence number	32 bits	a number to specify the order of sent packets
Acknowledgement number	32 bits	a number used to acknowledge the receipt of incoming packets
Data offset	4 bits	Specifies the length of the tcp header in 32-bit words, at the same time it also indicates where the data section starts
Reserved	6 bits	For future use and should be set to zero
URG	1 bit	One of the six control flags, this one indicates that the Urgent pointer field has a meaningful value
ACK	1 bit	To acknowledge a message from the sender
PSH	1 bit	Asks the application to send immediately and not wait for enough data to fill the entire TCP packet
RST	1 bit	Asks to abruptly terminate the TCP connection, used when encounting errors.
SYN	1 bit	Initiate a three-way-handshake to establish a connection
FIN	1 bit	Used to end the TCP connection by both parties
Window size	16 bits	Specifies the size of the receive window, notify the sender how much data can be received
Checksum	16 bits	Used for error checking the TCP header, payload and IP pseudo-header
Urgent pointer	16 bits	An offset from the sequence number to indicate the last urgent data byte
Options	0 - 320 bits	To include options that have up to three fields: Option-Kind, Option-Length and Option-Data
Padding	0 - 31 bits	a sequence of zeros to ensure that the header size is a multiple of 32 bits

Figure 3.6: TCP Header fields

TCP half-open / SYN Scan From an attacker's perspective where the goal is to find as many active hosts or services it is more efficient to just send many SYN packets to a large number of targets. Completing the full handshake is not necessary as the first acknowledgement from the server already announces its presence to the attacker. Therefore in the case of TCP scanning, a single probe means a packet with the SYN-flag set. This type of scanning is called half-open, because the server is left at an intermediate state where it is waiting for the final acknowledgement from the client. Resources at the server side have already been reserved, but the connection is never fully established. This is not considered normal behaviour since the client has unexpectedly closed the connection without notifying the server. High speed scanners are built by modifying the TCP/IP stack such that the final ACK from the client is never sent. A half-open connection often indicates malicious intent in the form of port scan reconnaissance or Denial of Service (DoS). Reserved resources are only freed after exceeding a time-out period. Sending a high enough burst of SYN packets will occupy all of the server's resources and eventually become unresponsive.

NULL, FIN and Xmas scan Other types of TCP scanning do exist, but are rarely encountered in the wild due to their unreliability in determining the presence of running services. The NULL, FIN and Xmas scan types exploit loopholes in the TCP protocol described in RFC 793. A closed port that receives packets without any of the SYN, RST or ACK set will result in a RST being sent back and no response if that port is open. These three scan types have in common that none of the SYN, RST and ACK flags are set, but differ by the presence of remaining flags. RFC 793 does not differentiate between these scan types meaning that they elicit the same response if the server is fully compliant with the TCP protocol. However, firewalls depending on the implementation may process them differently. In shielded network environments that allow only outbound connections, incoming SYN packets are dropped by the majority of firewalls. For this reason, one way of slipping through firewalls is to omit the SYN flag while still being able to perform limited reconnaissance. NULL scans have no flags set, a FIN scan sets just the FIN flag and lastly the Xmas scan sets the FIN, PSH and URG flags. In practise, not all operating systems follow RFC 793 to the letter. Microsoft Windows for example always responds with a RST regardless of the port being open or closed. The scan does work against most Unix-based systems. So if a host respond with RST it can either mean that the port is closed or that it deviates from the RFC 793 specification. If the running operating system is not known than one must infer this information through OS fingerprinting. However, that removes the benefit of the attacker to remain hidden since fingerprinting is performed by sending a series of packets and analyzing the returned responses for OS specific peculiarities. The absence of a server response when probing a port can also have an ambiguous meaning, firewalls or Intrusion Detection Systems(IDS) properly configured will simply drop the packet before it reaches its target. An attacker will have difficulty in determining whether a port is actually open, filtered or the packet may even be lost during transmission.

3.2 Network scanning

Network scanning is an information gathering process used to discover network elements such as active hosts, network services and users. It is commonly used by network administrators to monitor the current state of the network. Besides hosts there is also the network

topology, firewall policies and routing tables which can be difficult to maintain without proper documentation. To the network administrator, scanners are invaluable software tools that help assess the network and validate its proper behavior. This is done by sending a series of data packets to a specified IP address range and check whether the response or absence of response match the intended behaviour according to design. Comparing sent packets against the responses is one way to detect misconfigurations or gauge the state of the network. The same information is also of much interest to hackers which use scanners as a prelude for their attacks. The chances of a successful intrusion attempt depend on the hacker's ability to find an entry point. Therefore, the network's IP address space is enumerated to look for running services that contain known vulnerabilities. Upon successful breach the hacker will perform lateral movement for which it will once again rely on knowledge on the target network. Similar to a real threat is a security professional that is given the task to assess an organisation's defences by mimicking a hacker. Both the hacker and security professional's primary concern is to conceal their reconnaissance activities in order not to trigger Intrusion Detection Systems(IDS). Limiting the rate in which consecutive probes are sent is typically the way to avoid detection. A network administrator does not face this limitation as it is scanning its own network with the organisation's awareness, but hackers do have to resort to less obtrusive scan tactics. The chosen scan strategy by hackers is a trade-off between speed, accuracy, complexity and detection avoidance.

3.3 Port scanning

Port scanning is a more specific type of network reconnaissance, because it implies which protocols are used to conduct the scan. During a scan, packets are being sent to a target network in order deduce from the response whether there are any active services of interest. The notion of a port is associated with two transport layer protocols, TCP and UDP which have been discussed in the previous sections. Port scanning has become ubiquitous in today's cyber threat landscape and an important tool in the arsenal of adversaries as a preliminary phase before launching an actual attack. That is because the foundational protocols of the internet protocol suite are TCP and IP. Majority of applications such as browsing websites, sending E-mail, sharing files are made available through hosts with open ports that are accessible to the public internet. In a traditional client-server model, the server offering a particular service awaits incoming connection request from clients. The server responds to the client and establishes a connection via a three-way-handshake. A remotely accessible service is not inherently insecure, but human-error or deliberate placement of backdoors can potentially lead to vulnerabilities. The associated security risks can range from near harmless to a full system compromise. Upon discovery, a vulnerability is assigned a Common Vulnerability Scoring System (CVSS) by the Common Vulnerabilities and Exposures(CVE) to evaluate the threat level. Organisations that are affected should take proper measures e.g. by updating the software to mitigate the issue. There is a security risk between the time of disclosure, the moment when the vulnerability becomes public knowledge, and actually fixing the issue. Within this period there will be a significant increase of scan traffic directed towards the well-know port associated with the service. Attackers see new opportunities of capitalizing on the new vulnerability disclosure. One way to observe such large scale events is through network telescopes, a set of unused IP addresses, by monitoring suspicious traffic. The amount of traffic a certain port receives is mostly dependent on the vulnerability threat level and the adoption rate. Both of these factors contribute to increased benefits, mostly financial of nature, to the entity that exploits it. Every open port is a potential security risk

and the corresponding services need proper security hardening e.g. by keeping software up to date. The financial cost of performing internet-wide scans is negligible and relatively simple to setup. Port scan related traffic is increasing every year. Nowadays it will take less than a few minutes before a newly connected device to the internet receives its first SYN packet. Little knowledge is required by the introduction of open-source tools that can scan the entire IPv4 space in less than an hour on moderate hardware (quote). Complexity increases when a pool of distributed scanners require sophisticated control in order for them all contribute to a common reconnaissance goal in a stealthily manner. Therefore, a distinction is made between scanners that operate from a single host and those that are distributed among many hosts.

3.4 Single Source Scanners

The prerequisites for performing a scan are simple, just an active internet connection and a host running the software to perform the scan. A single attempt to query whether a port is open is also called a probe. In the case of TCP half-open scans, a probe is one packet with the SYN flag set from a source to target. The target comprises of a destination IP address and the protocol port number. An IP addresses is used to identify both the sender and recipient of the packet. However, due to the scarcity of IPv4 addresses several networking techniques such as DHCP Churn, Network Address Translation(NAT) and Carrier Grade NAT(CGNAT) have been deployed to prolong the slow transition to IPv6. This makes attribution of the scan origin less straightforward than if each host on the internet can be identified by a fixed IP address. For simplicity the ideal scenario of fixed addresses is assumed, but the possibility of multiple hosts sharing one IP address will be taken into account during the discussion on detection and attribution. Having now defined a target, port scan reconnaissance typically spans multiple targets. After all, a larger search space contains more active hosts and therefore increases the likelihood of discovering targets of interest. The set of targets that one is interested in characterizing is defined as the footprint. For internet wide scans this means that all IPv4 addresses are potential targets where a select number of ports are of interest. Being a two-dimensional space, IP addresses on one axis and destination port on the other, scanning a large set of ports is only feasible within a reasonable time for fewer IP addresses. For each probe the scanner will select a target from the footprint using a target selection algorithm, but it is not necessarily the case that all target will be visited at least once for full coverage. For example, a target selection algorithm that draws from the set at random can have overlapping targets and leave certain targets untouched. Overlap occurs when targets are visited more than once. On the hand, sequentially scanning the targets using the same number of packets will achieve full coverage and no overlap as can be seen in Fig 3.7.

In any case, a consecutive series of probes sent to targets within the footprint is called a scan. The number of probes within a certain time window determines the scan rate. High performance scanners are optimized to enumerate the search space in the smallest time window so no software restrictions are placed on the scan rate, probes are sent out in quick succession as fast as the hardware and network bandwidth limitations allow them to. However, the scanner must also be able to process replies from active targets. Some form of state management is necessary confirm that a reply is the result from an outgoing probe. Misconfigured hosts on the internet or backscatter can lead to unsolicited replies to the scanner. Therefore, scanners can not blindly assume that a reply implies the discovery of an active target. State can be kept internally within the scanner or encoded into packet

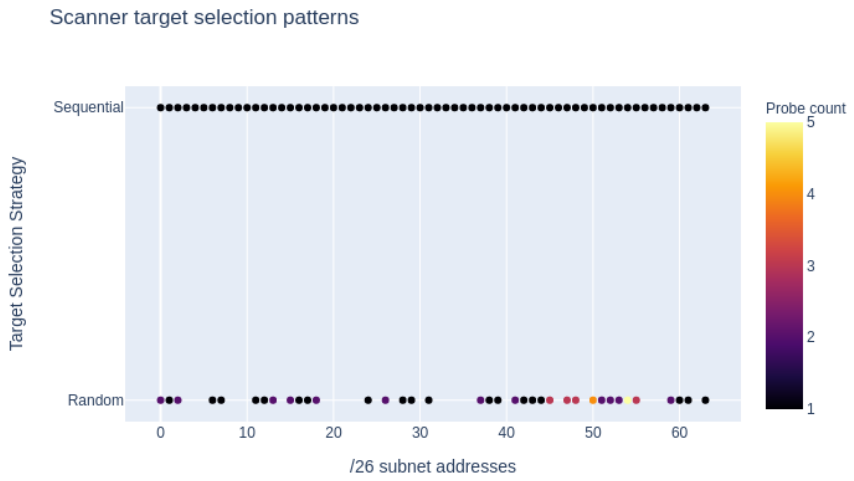

Figure 3.7: Target Selection Patterns

header fields without braking the correct operation of TCP/IP. The latter method is typically the choice of high performance scanners, because otherwise the scan rate will be limited by the amount of available memory. Each outstanding probe, meaning an outgoing probe awaiting response, will take up some memory until it can be release when either a reply is received or after some predefined time out value. A faster scan rate will lead to more outstanding probes which will quickly exhaust the available memory. A smart optimization is to encode all the necessary information into packet header fields. The scanner should be able to identify what was scanned from the reply. This technique exploits certain fields in both IP and TCP headers that remain mostly unchanged in the response, effectively using those bits to store data.

3.4.1 Goal

The main task of every scanner is to infer whether a given target is reachable through the internet, but the high-level goal of the campaign is the reason why the scanner is turned on in the first place. Owners of scanners can have various reasons for actively sending out probes. For instance, cyber criminals are motivated by financial incentives and thus benefit from finding hosts that they are able to compromise. A newly disclosed vulnerability for a particular service can lead to a temporary surge of scan activity on that port. This happened upon discovery of a severe vulnerability in MikroTik routers [?]. The routers were at risk of device hijacking and remote DNS cache poisoning through port 8291. From that moment an increasing number of campaigns were launched with the goal of finding the maximum number of hosts reachable through port 8291. Some scanners can be considered benign when the results are used for security research. ?? measures the adoption of HTTPS based on internet wide scans. Since the goal of a scan campaign is to obtain specific knowledge it also largely determines the footprint of potential targets. It is therefore also to some extent possible to infer the adversary's goal by analysing the packet destinations. Popular web based services if present on the host are expected to run at their default ports. So both the

range of port numbers and IPv4 destination addresses are able to reveal which services are targeted and what the scope of the scan campaign is. However, the footprint and the goal are not interchangeable terms. The goal includes the necessary motivation or objective to start scanning while the footprint is the desired knowledge than one wishes to obtain. Anybody else than the one responsible for the scan campaign can only make an educated guess on what that knowledge is used for in subsequent actions. Cross correlating the timing of scan campaigns and other internet wide events might provide some additional hints.

3.4.2 Footprint traversal patterns

The destination of any probe is identified by a pair of IP address and TCP port number. Therefore, the entire space in which targets can be selected is two dimensional. The target space is formed by the cartesian product of the 2^32 IP addresses and 2^16 port numbers. When performing network reconnaissance, the adversary typically is only interested in a small subset depending on the goal. That means reducing the range of IP addresses, selecting just several port numbers or a combination of the two. Traversing only as much of the target space as is necessary leads to more efficient scanner usage, saving both time and computational power. It would make little sense to target port numbers for which a particular interested service is not the default port. This is especially true when resources are scarce as is the case with a single scanner. In other words, the footprint is a subset of the entire scannable space that the adversary wants to characterise. Similar to confining land exploration in search of rare animals to a particular area known as their habitat. However, the footprint does not define how the individual targets should be traversed, that is the task of the scan algorithm. Knowing just what 23 information is interesting to the seeker reveals nothing yet about the scan strategy. Decisions have to be made regarding which target is scanned first and which one will be the next. In addition, scanners that do not operate at full speed have to wait for a certain period of time which is also part of the scan algorithm. From the literature there are common geometries that can be identified. These are are called horizontal, vertical and block scan which will be discussed in the next subsections in more detail.

horizontal Scan Horizontal scans are characterised by scanning just one port number to multiple hosts. This type of scan is very common when the goal is to find a large set of hosts that run a specific service. That service, if present will most likely run on a default port which is the only port number worth scanning by the adversary. Restricting to one port makes it feasible to complete an internet-wide scan, visiting each address in the IP space. Horizontal scans are very common once a new critical vulnerability has been discovered for specific internet-facing software that can provide financial gain when exploited.

Vertical scan As opposed to horizontal scans, vertical scan aim to discover any entry point to a particular host. This type of scan effectively sends multiple packets to one host, each time with a different port number. The end result is a map that reveals which internet-facing services are running. Subsequent steps can then be taken to determine the path of least resistance to execute a successful attack. Vertical scan are very focused, the adversary is showing exceptional interest to a specific host. Either the host itself or the responsible organisation has high value. This type of scan is also typically used by penetration testers, because their job is to take on the role of a cybercriminal to prematurely spot weaknesses in

employed defences. Every open port is a potential security hazard and the corresponding service will be checked for vulnerabilities that can be exploited. A full scan that enumerates all ports numbers would be the most thorough approach, but it is also very time consuming while providing diminishing returns. In addition, the scanner has a higher chance of being noticed due to the large number of consecutive probes. Therefore, limiting to the most frequently used port numbers for popular services can yield good results with less effort and noise. An analogy of a vertical scan would be a burglar trying to break into the house of a high-profile target. Going through the front door isn't likely going to be the easiest approach, because it typically receives more attention in terms of security than other areas. So in order to find the easiest way in, it is worthwhile to first properly explore the perimeters of the house to identify other entry points. After carefully observing each square meter he discovers that the lock of a basement window is outdated and trivial to pick with a recently released specialised tool. He then goes to purchase the tool from the black market which will serve as the entrance ticket. The key similarity here with a vertical scan is that every possibility to gain unauthorised access is considered. Investigate all potential entry points and opt for the path of least resistance. Other targets or not currently of interest, just this specific one. In contrast, horizontal scans are performed when the method of attack is already fixed. Equipped with this method, typically a recent vulnerability exploit, the network is scanned for any target for which this exploit is applicable. Suppose criminals developed an exploit for a smart lock which is easily recognised by its manufacturer logo, then driving through the neighbourhood can identify targets to break into.

Block scan Block scan or otherwise known as strobe scan is a pattern that targets multiple ports of multiple IP addresses. In this case, the adversary is interested in all host within a set of IP addresses. This might be the subnet of a particular organisation or extending up to the entire IPv4 space. The variety in destination port numbers can range from a handful to all of them. Therefore, a block scan is neither a horizontal nor a vertical scan. With the right prior knowledge, block scanning can yield the same results with less effort. Entire subnets belonging to security organisations or other unattractive parts of the IPv4 space are blacklisted by the scanners if there is nothing to gain from scanning them. Similarly, enumerating only those port numbers for which one has an exploit for the corresponding services leads to better utilisation of resources.

3.5 Distributed Scanners

Distributed scanners are a group of scanners under the control of a single actor. They can be viewed as a collection of single source scanners, but more than likely sharing specific characteristics due to similarity in the scanning tool and chosen strategy. The footprint of each individual scanner is a subset of the group footprint. The owner of such an infrastructure composed of multiple scanning sources is free to decide which targets from the search space are scanned by the sources. Besides selecting the targets, other characteristics including the scan rate, overlap, coverage, duration on both individual and group level are part of the scan strategy which will be discussed in greater detail in later sections. What sets distributed scanners apart from unrelated single source scanners is that they exhibit some degree of collaboration and synchronization performing reconnaissance. A controlling entity oversees the sources, delegating the process and collecting status reports. Example of

such form of collaboration is when all scanners are active during the same period. Advantages for launching scans from distributed hosts are numerous. For starters, multiple scans can be performed in parallel of which their combined scan rate can greatly exceed that of any individual scanner. A problem with high performance single sources scanners is the relative ease of detection. From the defender's perspective, any source that sends an exorbitant number of probes to the network compared to baseline noise levels will quickly be blocked from further attempts. Distributed scanners are able to evade threshold based detection methods by deploying numerous slow scanners while still achieving high combined scan rate. With enough sources one can set the scan rate low enough to hide the malicious purposes of the scan and prevent them from being blocked by the firewall. Another property to consider is the proximity in terms of geographical location and source IP address. Scanning from various network origins makes it harder for defenders to correlate participating scanners to the same group. On the other hand, sources from the same /24 subnet provide high confidence of a collaborated effort since such a small block of addresses often belongs to one organization. Individual scanners require a communication channel with the controller for receiving instructions. The topology of such a scan infrastructure can be centralized, peer-to-peer or of hybrid nature. Each type is a trade-off between complexity, resiliency and scalability. Scanners also contact a reporting server for sharing the results of completed scans. Finally, this work makes a distinction between self-propagating worms that upon infection are pre-programmed to look for new potential victims through internet-wide scans. While these are also scanners and show high similarity in their behaviour, they typically do not meet the requirement of being controlled by a single instigator. This work's definition includes a sense of deliberate intent for performing the scan.

3.6 Detection

Defenders monitor a network under their administration and bear the responsibility of handling incoming threats. Malicious scanners that repeatedly probe the network are an example of such a threat and must be dealt with accordingly. However, the identification of scanners is not straightforward when obfuscation methods are applied. Current state-of-the-art detection systems are based on thresholds where scan traffic from one source exceeding baseline levels are deemed malicious. As a consequence, slow scanners that remain below the threshold can go unnoticed. This approach does not achieve 100% accuracy in detection, but forcing adversaries to limit their scan rates is actually a positive result. After all, setting an upper bound for scan rates inhibits anyone to discover network characteristics in relatively short time. While it does not withhold any scanner from scanning slowly, port scan detection is effective as part of a larger defense-in-depth approach at mitigating the impact of intrusion attempts. Additional giveaways of scanners are patterns in the header fields of the TCP and IP protocol. For instance, scans performed with the popular tool ZMAP have a fixed IPID value of 54321. Within this never-ending game of cat and mouse between attackers and defenders, more sophisticated scanners have been developed that originate from multiple sources. To correctly gauge the scope of a distributed scan, all activity from individual scanners from that group must be attributed to a single controlling entity. However, the absence of a ground truth paired with the limited visibility makes this a very challenging task. Only the one responsible for the scan knows exactly how many sources are involved and the choice of strategy. A monitored network such as a network telescope can only shed light on whats happening in a small portion of the entire internet. That being said, the correlation of individual scanners is possible at a best effort basis by establishing reasonable

confidence. It is presumed that scanners within the same group share characteristic. That is because the reconnaissance goals and selected scan strategy is determined by the controlling entity and shared among all sources. To some extent, the intention of the scan will manifest itself as patterns that can be perceived by monitoring the network. Examples of such patterns are values within a probe and also patterns in packet arrival times. Furthermore, managing a large number of scan sources calls for an approach that capitalizes on the economies of scale. Therefore, it is likely that the distributed scanners are equipped with the same scan tool and configuration parameters. This relates to detection by reducing the problem of finding all scanners that belong to single group to the correlation of individual scanners that show high similarity in their scan characteristics. Chapter X will provide a more detailed discussion on which characteristics can be observed by a network telescope and the degree in which they contribute to increased confidence of scanner collaboration.

Single port scan detection algorithms are mainly designed for fast detection and prompt response in order to block further attempts at reconnaissance which are only consuming resources of the target network. The task is to classify traffic as either legitimate or scan traffic with low false-positive rate. Distributed port scan detection takes it a step further by estimating the scope or scale of a campaign. The size and employed tactics of a distributed group provides an indication of the attacker's sophistication level and/or its interest in the target network.

3.7 Obfuscation

A simple scanner can be configured to scan at full speed, as fast as the internet-uplink will allow. That would yield the fastest results, but this is not always desirable. Modern intrusion detection systems typically block scanners from further connection attempts. These defences are deployed to distinguish malicious traffic from benign. Sources that are deemed malicious are placed on a blacklist and subsequent packets will get dropped immediately upon reception instead of being processed. High- volume scan traffic is a nuisance from the perspective of the defender, because it consumes valuable network resources without providing any utility. Furthermore, it should be clear by now that scans can even pose a security hazard if there are any vulnerable services facing the internet. A blocked scanner will thwart any future connection attempts. Even if there actually is an active service listening at the target, SYN-packet will never reach its destination and elicit the SYN-ACK response according to the three-way-handshake. The scanner might even falsely interpret the absence of a reply as the target being inactive. Thus for the highest scan accuracy, it is in the interest of the adversary to blend into normal traffic. Fast scanners are particularly noisy and thus stand out from baseline traffic which lead to easier detection. Therefore, one typical approach for evading detection is to limit the scan rate. Any effort to make purposeful scanning resemble more like internet noise from accidental scans can be considered a form of obfuscation. The scan behavior is adapted to be more stealthy. This means that certain parameters of the scan tool are adjusted to meet this requirement. Obfuscation techniques are applied to avoid attention from a particular observant in mind. So far it has been discussed how scanners can avoid detection from an IDS. Besides evading detection an adversary can also put significant effort into hiding the goal of the scanner. It may not be desirable if network defenders are able to see what information is sought after. So instead of only scanning for the service of interest, occasionally also targetting other port numbers can serve as a decoy. This makes the scan look less threatining due its wider scope. In addition, defenders have a harder time

preparing for an incoming attack when their attention and resources are divided to cover multiple angles. For these reasons, prematurely alarming defenders about a potential entry point into the network is not a good idea. It's best to take advantage of the element of surprise to move unnoticed through all phases of the cyber kill chain. Also after a breach has took place, standard procedure includes post-attack analysis on how to improve defences to mitigate future attacks. Obfuscation of the attack makes it harder to reconstruct the modus operandi of the adversary which potentially leaves the network at a vulnerable state.

4 Related work

Single source port scan detection techniques have been developed as a first line of defense against external attacks. This category of port scan detection has received the most attention, because the presence of scanners indicate a potential threat and scans were typically launched from a single host. However, work on single sources scanners is not necessarily related to this thesis. The research question at hand is not how to distinguish benign from malicious probing attempts, because due to the nature of a network telescope it is already clear that the majority of incoming traffic serve no legitimate purpose. Therefore, no effort has to be spend on obtaining a labelled data set of scan traffic. Instead, the focus lies on determining which of the malicious scanners are potentially part of the same distributed group. A topic that has received very limited attention as of yet. A brief overview of single source port scan detection is provided where the authors can provide key insights into behavioral characteristics of scanners followed by the current state-of-the-art distributed port scan detection. Finally, research gaps are identified that will highlights potential areas of improvement to derive better detection methodologies.

4.1 Single Source Port Scan detection

In Dabbagh et al. [2011] they look at the imbalance between incoming SYN packets and outgoing replies from the network. Legitimate users are expected to know which host are available to contact which results in an acknowledgement for the majority of connection requests. On the other hand, the behavior of unsuccessfully probing many addresses is typical for a scanner. A positive sum of incoming probes subtracted by outgoing acknowledgements indicates within a reasonable time frame indicates an ongoing scan. It builds upon Jung et al. [2004] which is the current state-of-the-art algorithm named Threshold Random Walk (TRW). Its implementation has found way in many modern Network Intrusion Detection Systems(NIDS).

4.2 Distributed Port scan detection

While scanning has been an old technique to perform reconnaissance, only limited research has been dedicated to distributed scanners. An early work from Gates [2006] presented an adversary model based on the information it is trying to obtain from the targeted network, also called the scan footprint. A framework is provided in which different adversaries can be described and compared with by their scan footprint. It is assumed that the adversary's intention is to cover the target space efficiently with the least number of probes which means that no destination address and port number combination will receive a probe more than once. The author then reduces the problem of finding distributed hosts to a set covering problem, but this NP-complete problem is only feasible for small scale evaluations due its

computational complexity and the technique requires that the group hit at least 95% of all addresses of the monitored subnet.
In Robertson et al. [2003] the authors define distributed port scans as a group of scanners with source addresses that lie in close proximity to each other. They assume that an attacker is more likely to use scanners originating from the same subnet rather than scattered across the IPv4 address space. Certainly this is a reasonable assumption to make since IP addresses are assigned in blocks to an organization. Therefore, it is likely that scanners from the same subnet are controlled by a single instigator owning that block of ip addresses. A higher prefix of the observed scanner subnet increases the confidence that the scanners are collaborating. The algorithm is easy to implement and can yield good result with few false positives. However, the used definition is very narrow due to its strong assumption of source address proximity which only represent a subset of the large variety in strategies employed by distributed scanners.
Another definition is given in Yegneswaran et al. [2003]: Coordinated scans are scans from at least 5 different sources that target a particular port in the same /24 subnet within a one hour window. They found evidence of coordinated scanners by observing similar on-off behaviour. A group of scanners were active during the same days in a one month period. No systematic method is provided to effectively detect collaboration among scanners. With just three examples the authors conclude that such attacks are very common and that collaborative clusters can be effectively isolated.
In Feng [2013] All TCP and UDP packets are continuously monitored in a /24 network telescope. If the number of scanners that target a port within one ten minutes exceeds a certain threshold, then they are considered one group. This needs clean training data to determine the threshold, but can be updated during operation. For popular ports the method is likely to yield many false positives, because only a small observation period of 10 minutes is used to characterize a scanner. With the large volume of generated scan traffic, there is a non-negligible probability that unrelated scanners will hit the network telescope within the same time window.
Griffioen and Doerr [2020a] were able to identify and detect distributed scanners based on commonalities in packet header fields. The proposed detection method leverages the fact that high-performance scanners embed information in the packet header. Due to the economies of scale, scanners within a group are likely to use the same scan tool. Probes generated by a scan tool typically have information embedded in the same header fields in a similar pattern. The scan rate does not influence detection which even allows groups of very slow scanners to be revealed. Not only have common port scanning tools been identified, but it has also lead to the discovery of several new custom made tools that were previously unknown in the literature.
The approach in Haas et al. [2020] is based on the key insight that scan activity from the same attacker exposes similar properties, even when accomplished by a coordinated scan using multiple nodes. Therefore, they have identified ten key features to characterize scanners. Pairwise distances are calculated between each pair of scanners and then fed into a hierarchical clustering algorithm. However the temporal scope of their data set, only 15 minutes, is rather limited and this already excludes slow scanners. Furthermore, they evaluated their method using a minimum threshold of 100 packets for each scanner. In combination with the relatively short time window, this will retain only the noisiest scanners in the data set. The authors do not provide any evaluation for their results nor do they take into account any false positives. Similar to the proposed method in this thesis, Yao et al. [2013] and Lv et al. [2014] derived features from scanner time series characteristics after which they can be clustered using the minimum spanning tree algorithm. The experiment is rather small scale and the distributed scanner traffic are captured from a controlled environment mixed

with background noise. Public scanning tools are used to control 16 coordinating scanners of which the traffic is used for evaluating the method. In Jing Yang, Liming Wang, Zhen Xu, Jigang Wang [2019] they observed there is significant spatial and temporal similarity between scanners from the same campaign. Knowing this, they developed a hierarchical correlation algorithm that starts by comparing individual semantic behavior followed by temporal-spatial correlation. The data set was collected from a web hosting service provider spanning one week. In the absence of ground they had to manually verify which malicious traces belonging to distributed scanners. Bhuyan et al. [2012] proposes an outlier based detection method. Profiles of normal behavior first need to be established which can then be utilised to identify anomalous data points. Anomalies that are clustered in the same region are considered being part of one distributed campaign. An extensive list of packet features are listed, incorporating them all would be too computationally expensive. Therefore, a subset of features are selected by means of principal component analysis which can then be fed to the fuzzy C-means algorithm. They took special care during labelling, because the data set was used for both training and testing. It consists of real-life data mixed with data obtained from their university test bed. As is the case with most other work, coordinated scans were captured in a controlled environment and labelled accordingly.

4.3 Research gaps

This literature overview indicates that the detection of distributed groups is a challenging task. Despite that the phenomena of distributed scanners have been known for decades, there is relatively little work dedicated to its detection. Correlating which out of the possible 4 billion scanners are collaborating quickly gets unfeasible unless some assumptions are made that reduce the search space. However, these assumptions limit the type of scanners that can be found. As a consequence, it will paint an incomplete picture of the threat that distributed scanners pose in the wild. The mentioned works have not extensively explored the relationships and commonalities between distributed scanners. There are potentially many shared behavioral characteristics that are useful in effectively correlating them into groups. In the absence of ground truth, the identification of more features that characterize a distributed scan can provide additional evidence to confirm a cooperative relationship between scanners. Furthermore, earlier work tend to stick to small scale evaluations (e.g. one week or less) due to lack of data or computational feasibility. The algorithms employed typically belong to the class of unsupervised learning techniques which may explain the smaller observation period. The challenge of not having a labelled data set containing distributed scanners forces research to take the approach of generating such traffic in a controlled environment. Thus introducing severe bias when evaluating a detection method against such data. Furthermore, there was no intention to study the characteristics of distributed scanners in the wild. The discussed work do not go beyond devising a method for detection. Therefore, there was never any intention to investigate scan strategies that come in varying levels of coordination and their prevalence in the current threat landscape.

5 Approach

The initial approach starts with recognising characteristics about the problem that need to be taken into consideration when devising an effective approach. Such identification is best performed early on to narrow down viable solutions to, as yet, the abstract problem of correlating distributed scanners. There is no perfect solution due to the absence of ground truth and single methodology is able to capture all types of distributed scanners employing various strategies. However, we aim to make the correlation method applicable to the majority of groups. This work will follow a similarity-based approach where high-entropy features take a central role in detecting as many groups of distributed scanners while reducing the number of false-positives.

5.1 Considerations inherent to the detection problem

5.1.1 Absence of ground truth

The available dataset consists of incoming packets received at the university network telescope. With the right analysis new insights can be gained. However, as the dataset is unlabeled it becomes a challenge to evaluate the results. In supervised machine learning, performance can be expressed in common statistical metrics due to the presence of ground truth. Data samples have been first manually verified by field experts to provide corresponding labels. Currently there are no public datasets where scanners are labeled for a particular group. Manual verification is also a very labor intensive process which is only feasible for very small scale data. To make things more complex, there are no fixed number of groups so the number of possible labels potentially equals the number of available IP addresses where each group is of size 1. The correlation of distributed scanners shows more resemblance to a clustering problem which falls under the unsupervised learning category. Therefore, there is no guarantee that a group of scanners is indeed cooperating even if several pieces of evidence might suggest so. Through passive observation of network traffic one can never provide certainty. In this work a a best effort attempt is made to establish sufficient confidence through several indicators as an acceptable confirmation of collaboration. Special care will be taken in the evaluation section where more validation methods will be provided to properly address the challenge of missing ground truth.

5.1.2 Assumptions

With a large dataset containing millions of scanners, it becomes a daunting task to find a (small) subset that is taking part in a distributed group. An analogy would be finding a needle in a haystack. In addition to the large search space there is no standard way of identifying coordination between scanners. That is because a scanner can be implemented in

different ways and adopt various strategies depending on the strategy chosen by the initiator. To the eyes of a passive observer of the resulting scans, the individual probe packets from different sources typically do not contain an identifier that is unique to a particular group. Some open-source implementations do embed certain pieces of information in header fields which can be used to identify the usage of a specific toolchain Griffioen and Doerr [2020a]. However, an adversary who makes a deliberate attempt to obfuscate a coordinated effort will be able to make individual scanners appear as if they were completely unrelated while in fact they are actually part of the same campaign. In order to infer relationships between scanners some assumptions regarding scanner behavior are necessary. Assumptions help simplify the problem which can lead to an acceptable solution. This simplification does come with the risk of not accurately representing all samples in the dataset. For instance, in Robertson et al. [2003] the authors assume that IP addresses in the same / 24 subnet are used for coordinated scans. An algorithm to retrieve such groups would be simple to implement with short run-time. Few would argue that scanners originating from the same /24 subnet are controlled by a single entity, but this can be considered a very hard assumption to make which does not account for the majority of distributed scanners. Since this work attempts to perform a more comprehensive study on distributed scanners its assumptions must exclude as little of the entire set of distributed scanners as possible while retaining the accuracy and computational feasibility of detection to answer the research question. The most general assumption to make is that coordinating scanners exhibit some degree of similarity. Developing, deploying and controlling a large number of scanners is more complex than its solo counterpart. For this reason, we expect adversaries to leverage the economies of scale during a large scan campaign. Multiple scanners adopting the same strategy or having a similar behavioural pattern opens up possibilities to infer a coordinated effort. However, a characteristic that is shared within one group might not be applicable for another. Therefore, correlation based on different similar characteristics will each yield different subsets of the entire set of distributed scanners.

5.2 Desired Result

The data analysis performed in this research should make an effort to satisfy certain requirements of the results. Firstly, the aim is to achieve a high true positive rate. In other words, as many of the distributed scanners present in the dataset should be detected. However, an even larger emphasis is placed on reducing false negatives. In order to perform post-analysis on distributed scanners it is paramount that scanners are not incorrectly assigned to a group. This work attempts to create the first labelled dataset containing distributed scanners. A dirty dataset would impede drawing any conclusions when used as the basis for studying this topic. For this reason, the preferred methodology is one that minimizes the false positive rate at the possible expense of missing out some groups. Various strategies can be employed by scanner, but a particular interest is taken towards those that operate at very slow speeds. Slow scanners have received relative little attention from the research community partly because it is a tactic to achieve obfuscation by arguably more sophisticated adversaries. Studying real-world scanners adopting this tactic within a distributed setting has never been done before.

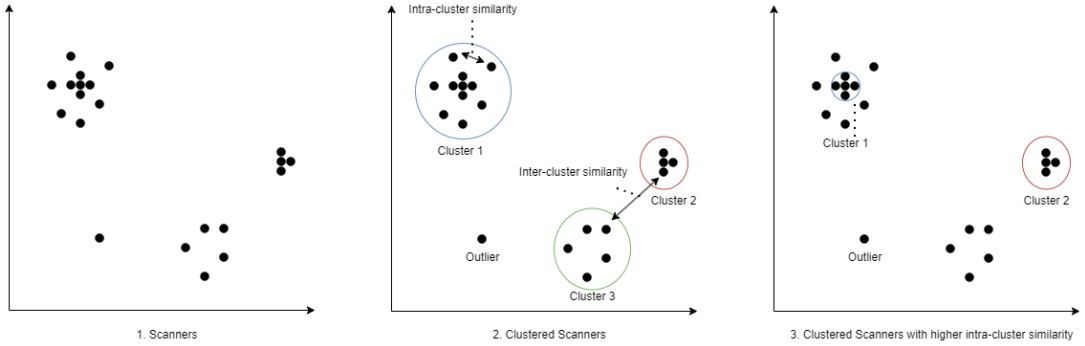

Figure 5.1: Process of clustering scanners with increasing intra-similarity thresholds

5.3 Exploring similarities

5.3.1 Similarity

To reiterate, the aim is to achieve a very low false-positive rate on the final result of identified groups. Recall that there is also the limitation of missing ground truth. The implication is that any potential correlation method can not be verified against real-world data. One way to mitigate this issue is find supporting evidence that the scanners are cooperating beyond reasonable doubt. By assuming that distributed scanners exhibit some form of similarity, the problem can be reformulated as finding observable features or characteristics that are highly similar between scanners belonging to the same group and have relative low similarity with scanners not part of the group. This bears high resemblance to a general clustering problem as shown in fig 5.1 which will be adapted for grouping scanners.

Initially the scanners are represented just as data points, the value of each data point has a distance to any other data point calculated using a particular distance metric. Clustering is the process of grouping together data points with high intra-cluster similarity and low intra-cluster similarity. Data points from the same class should have low pair-wise distance while being relatively far away from unrelated data points. Not every scanner is part of a group. In fact, the expectation is that the majority of scanner operate solo due to the lower level of complexity and resource requirements for its operation. In clustering there are labelled as outliers or noise in the end result. Clustering involves parameter configuration of the corresponding algorithm that will hopefully lead to more accurate clusters. For example, increasing the threshold for intra-cluster similarity means that certain data points that previously were part of a cluster will now be labelled as noise due to its relatively large distance to the nearest cluster. As a consequence, the end result will contain fewer false-positives as only the most dense clusters remain but some scanners will be incorrectly considered as solo scanners. Such parameter configuration is all about making trade-offs in order to find a sweet spot in the balance of performance metrics. Having said this, the first step to successful clustering is to transform scanners into their data point representation, also called a model. The performance of any algorithm highly depends on the separability of the to be identified clusters that meet the earlier discussed criteria of high intra-cluster similarity and low inter-cluster similarity. A bad example would be to cluster based solely on the number of packets received by source IP within a give time frame. While this approach might be able to distinguish slow from very fast scanners, the concentration of data points within a

small range of values with respect to the vast number of scanners in the dataset will make it impossible to identify groups of independently operating scanners. using a model where the features have low discriminative power will make every scanner appear as if it were the same. On the other hand, a high-dimensional model that is too complex will suffer from the curse of dimensionality where the distances between data points become too large to detect any structure or pattern. In such a case, the distances even for true clusters become so great that all data points seem unrelated. Therefore, choosing features for building a model should involve evaluating their usefulness in contributing to cluster separability.

5.3.2 Entropy

A good feature provides information about the corresponding event or object of study that is hard to predict. Entropy as defined by Shannon expresses the average amount of information present in a variable when considering all possible values. In otherwords, entropy quantifies the degree of spread over the variable's values. High entropy implies uncertainty due to the data being spread out while low entropy has many of the data points concentrated around a relatively small number of outcomes. Thus low entropy provides less information, because the outcome is more predictable even if there are potentially many possibilities. For instance, a six-sided die has 6 possible outcomes of equal probability. Assuming a uniform distribution this maximises the entropy, because the outcome is completely random. In comparison, a die with eight sides has higher entropy if thought about it in terms of uncertainty and predictability. The chances of guessing the correct outcome is lower in the case of the eight-sided die. However, if this die were modified such that it always would land on the value 7 then it would be considered very deterministic and predictable. The outcome would be little surprising and as a consequence have low entropy providing little information. The minimal and maximal entropy examples are opposite extremes to elaborate the concept. In this work, entropy will be the prime criteria for judging whether potential features are suited to identify distributed scanners. Scanners that belong to the same group are expected to be similar. A high entropy feature provides a large search space which will make it easier to identify groups from the noise of solo scanners. Scanners belonging to the same group will form clusters around the same value that have a relative large distance to other data points. Thus making it easier to spot order within the chaos of many possibilities. The probability that data points happen to be near eachother in close proximity by coincidence decreases as the entropy increases. Thus a high-entropy feature is able to detected distributed scanners while also achieving a low false-positive rate. However, the feature should capture a distinct behavior of the corresponding group. Entropy in the context of correlating similar scanners is only useful if the event can be used to fingerprint a common group behaviour. We will continue to discuss a powerful feature that captures long-term scanner behaviour after describing the dataset obtained from a network telescope.

6 Network telescope dataset

A network telescope is a system for studying internet-wide phenomena. In literature, it is also sometimes referred to as a darknet. The main purpose is to store and procecss incoming packets that arrive at any of the telescope's IP addresses. A telescope's resolution is defined by the number of designated IP addresses and ultimately determines what portion of the internet can be perceived. The resolution or size is typically expressed as a prefix length like a /16 or /24 subnet. Thus, a larger telescope has more 'sensing' capabilities, because more of the entire IPv4 space is being monitored. Telescope addresses are unused, meaning these addresses are not assigned to regular active hosts that run services neither do they respond to any communication from external entities on the internet. Therefore, any incoming data packets directed towards the telescope's addresses are unsolicited and can be considered suspicious. There are several causes for receiving unsolicited packets. One of them include DDoS backscatter in which attackers flood a victim with traffic using spoofed addresses as the sender. Typically the spoofed source address is selected at random which may coincidentally be one of the telescope's addresses. The victim's resources is quickly being exhausted when it attempts to send a reply back to an enormous amount of spoofed connection request packets. So the telescope may be witness to ongoing DDoS backscatter attacks against a particular victim of which it is receiving unsolicited reply packets from. Another observable event are random scanners that aim to find any hosts on the internet listening to a particular port. Selecting targets at random may appear like an inefficient approach to enumerate all possible addresses, but sending a large number of packets is low-cost and fast. Internet-wide scanners aim to traverse the entire internet which allows even the smallest network telescopes to observe this type of event. A telescope with a larger resolution has a higher probability to witness small scale events that operate more locally. With this in mind, the data from a telescope is invaluable for the study of scanners. This thesis research utilises such a telescope that monitors relative large subnets associated with the TU Delft. Exactly What type of data is captured and the complete setup will be presented in more detail in subsequent sections. This is followed by a demonstration about what insights and knowledge related to scanners can be extracted from the raw capture data. We hypothesize that a longer observation period may reveal new insights regarding the tactics and behavioral characteristics of scanners that were not previously visible in smaller scale experiments. No other work has previously analysed long-term data at such scale which this thesis is set to explore. However, some pre-processing steps of the data are necessary to reduce computational requirements. For this reason we retain only what is relevant for the purpose of correlating distributed scanners.

6.1 TU Delft Setup

6.1.1 Telescope Resolution

The TU Delft operates a relatively large network telescope with a resolution of two /16 subnets. A /16 subnet contains 2 to the power of 16 addresses, the total number of addresses amounts to 131072. However, that is the theoretical maximum number of addresses allocated to TU Delft and all its sub-departments, not just the Cyber Security group. So a significant portion is designated for other purposes related to daily operation of the university. In order to have a more accurate view of the telescope's size, we record all distinct IP addresses that were able to receive a minimum number of packets over a prolonged observation period. We used a sample of 1.6 million scanners that have sent 927 million packets over the course of one year. If addresses are chosen at random then one would expect that each destination would receive roughly 7000 packets. However, that is only if the telescope's resolution actually consists of two full /16 subnets. The number of packets per address is likely to be much higher if there are less addresses to account for the same number of packets. Thus, a threshold of at least 1000 packets is set to determine whether the address is being monitored by the telescope. 41609 addresses meet this criteria in the 131.072.0.0/16 subnet and 10469 in the 131.180.0.0/16 subnet. This is less than half of the initially assumed size, slightly less than a full /16 subnet. Determining the telescope's true address count is essential for accurate extrapolation of any findings from telescope data to the entire IPv4 space. In fig 6.1 and fig 6.2 illustrate where there are gaps in the two /16 subnets from the perspective of the telescope.

6.1.2 Observation period

The network telescope has been in operation for several years. For long-term analysis a period of approximately one year is chosen which is significantly longer than previous studies. More specifically, the first day of this dataset is march 2nd 2018 and ends at march 18th 2019. Within this period, the telescope did not have 100% uptime due to maintenance or outages. Still, there are 360 full days worth of data which is more than sufficient for any type of analysis.

6.1.3 Data and storage

Operating a telescope for an extended period of time can quickly occupy a vast amount of storage space. The telescope does not apply any sampling technique, but instead stores every incoming packet in its raw format. Metadata from both the transport layer and the network layer are present which have been described in section reffig:ipv4-header. Additionally, the packet payload (if present) is also included which allows for application specific analysis. The data is stored continously in PCAP files of 200MB each. As the rate of incoming traffic may vary from moment to moment, each file may approximately contain between 5 and 20 minutes of telescope data.

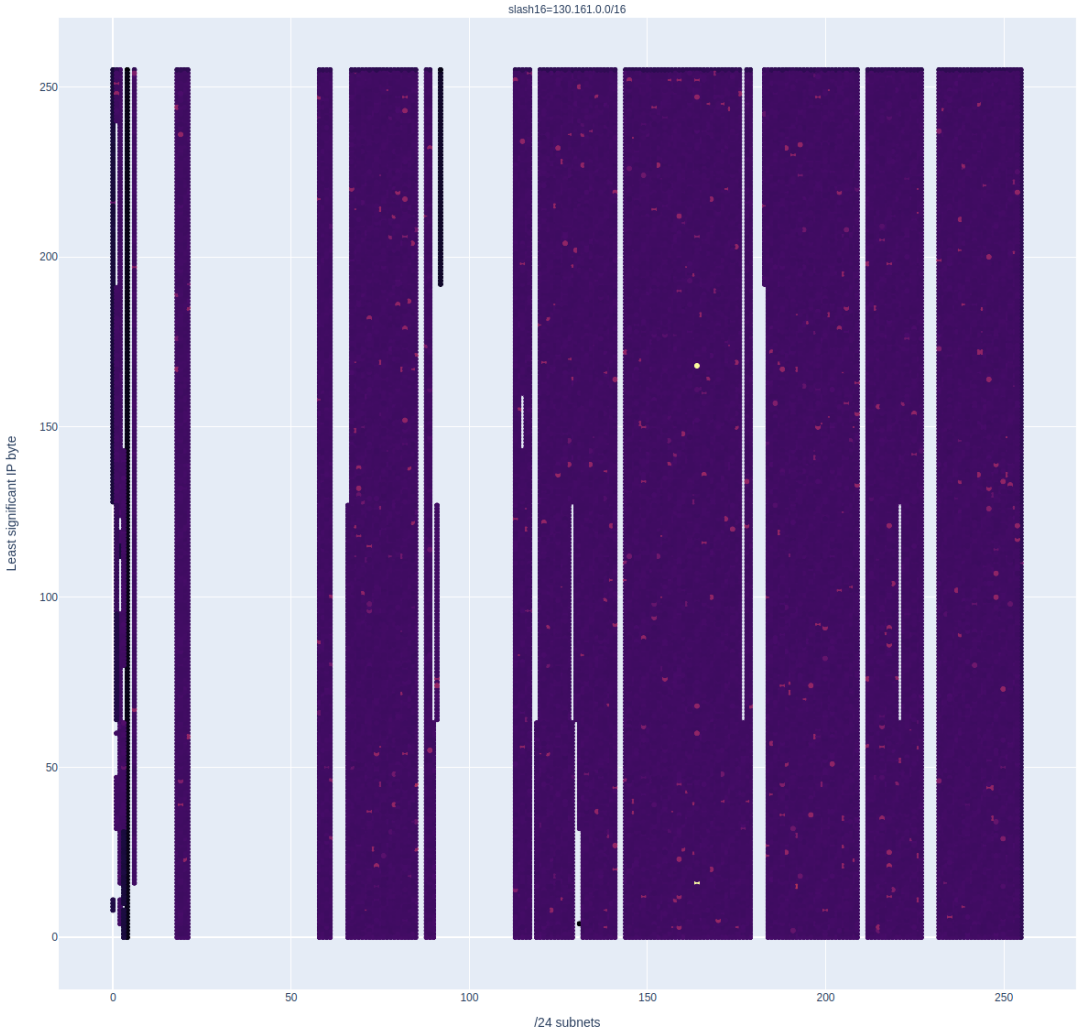

Figure 6.1: Heatmap of 130.161.0.0/16 incoming packets

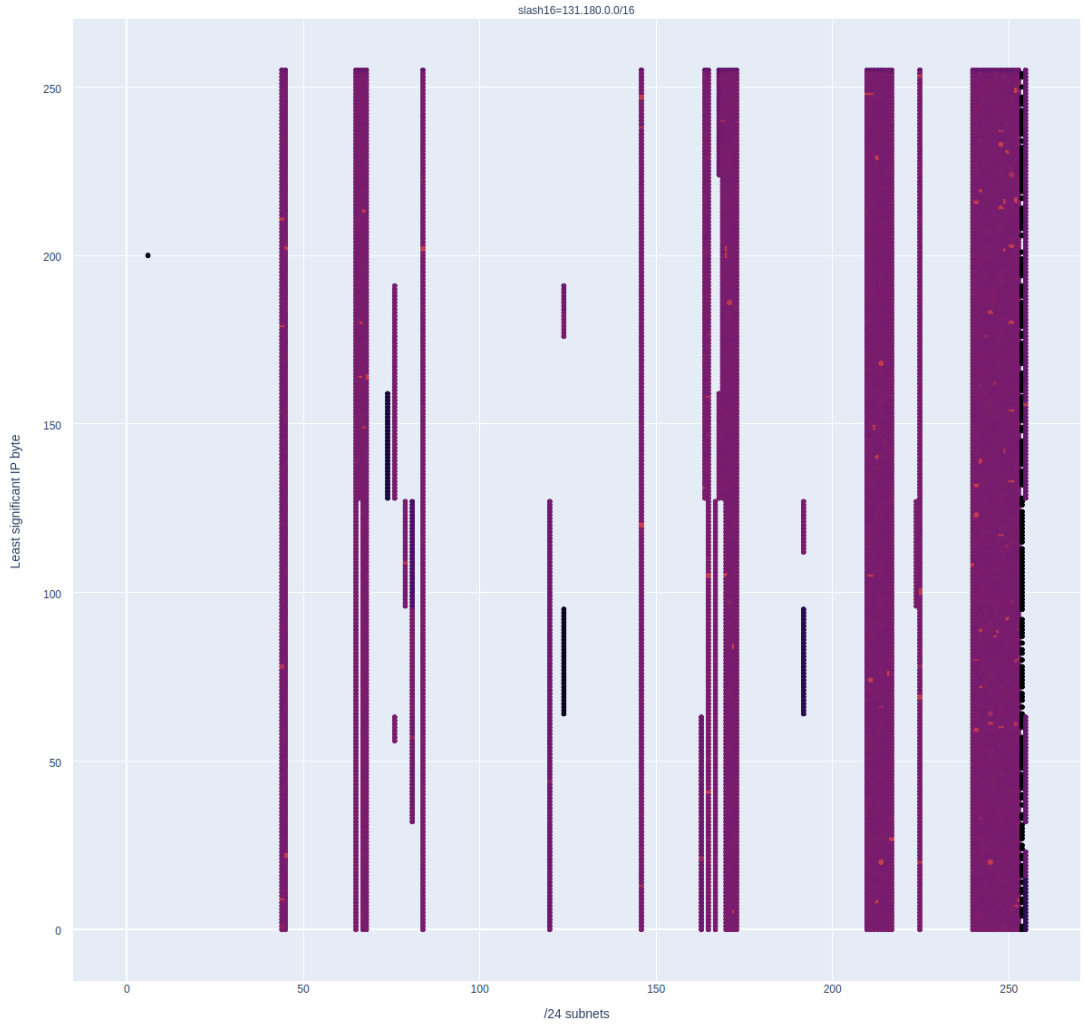

Figure 6.2: Heatmap of 131.180.0.0/16 incoming packets

6.2 Observable characteristics at different data resolutions

Having access to unfiltered data makes it becomes possible to study any specific scanner that has been active within the observation period. However more data can be a blessing and a curse. Parsing terabytes of traffic data requires heavy computation which is only performed after careful consideration. Therefore, it is paramount to first identify the type of knowledge that one wishes to extract from such an analysis. The correlation of distributed scanners is like the equivalent of finding a needle in a haystack. A very tedious task at which computers excel in, but even the fastest computer has its limitations. This section briefly covers which scanner characteristics were observed during initial exploration. When considering raw packets in isolation, the meta-data present in both IP and TCP headers provide basic information regarding protocol, source and target. However, when multiple packets from the same origin are viewed in conjunction then a more comprehensive image of the scanner's tactics reveals itself. This is especially true when extending analysis from a single packet up to an entire year in order to observe long-term behaviour. The task of correlating distributed scanners in the absence of ground truth is only possible when we are able to distinguish scanners belonging to one group from all the other scanners in the dataset. Thus, these distributed scanners must possess certain characteristics which others do not to enable such identification. In machine learning terminology, such characteristics are also called features. Short-term analysis performed in related work provides a narrow perspective of scanners due to the lack of data on a temporal scale. Having access to more data as in the current case of the TU Delft telescope provides a unique opportunity to explore both short-term but in particular long-term data. This exploration starts with a discussion about what type of characteristics are expected to be found that are unique to their respective observation periods and how they can be utilised to detect distributed scanners.

6.2.1 Short-term analysis

Analysis in the short-term is considered here as a time window smaller than one hour. The total number of incoming probes resulting from a scan will vary depending on the rate of the scanner and its traversal pattern across the entire IPv4 space. To reiterate, a telescope only sees a small portion of the internet and thus only a fraction of the probes will arrive when assuming that destinations are selected at random. Typically one hour starting from the first arriving packet can provide a lot of details regarding a moderately paced scanner. That should reveal which services and their corresponding ports are of interest to the initiator. However, multiple scanners targeting similar ports do not provide sufficient evidence for cooperation, because the most popular destination ports account for the vast majority of scan traffic. Unless the distributed scanners collectively target a very unconventional port there will be many unrelated scanners that share the feature of specific target ports. Another useful indicator is the scan rate which can be determined by counting the maximum number of packets arriving within a predefined time window. Some extrapolation is required to estimate the true scan rate since the telescope only partially covers the entire IPv4 space. Furthermore, if the addresses are selected at random then there will be some variance in the perceived scan rate by the telescope. This becomes particularly problematic with lower scan rates which results in fewer data points to obtain a good estimate. One idea is to form clusters of similar scanners in terms of their scan rate. However, apart from the difficulty of extrapolating the true scan rate there is likely not enough variability to make this a reliable feature. However, some have attempted to detect distributed scanners by including the

scan rate as feature with acceptable results. Here an argument is made that while such a feature has some discriminative power, the scan rate is too ambiguous due to the sheer number of scanners that contact the telescope. Extending the observation period to a full hour will allow for even more data points to estimate the aforementioned characteristics. In addition, there is a higher probability that multiple bursts occur. Such bursts are separated by moments of inactivity and if these breaks are of consistent duration then that could be a useful feature.

6.2.2 Medium to long-term analysis

Going beyond one hour to one day for instance some other intricacies of the scanner's algorithm become apparent. A scanner might be configured in a way such that it only operates at specific hours of the day. With short-term analysis this degree of scheduling would go unnoticed. During experimentation with the data several instances were encountered that even implemented scheduling on a daily basis. A schedule can be considered as being part of the scanner's configuration. Consequently, it follows an automated activity pattern as the result of an algorithm. In our current example, this implies that not this week, but at some other point in time in the past the scanner was manually started. Every scan campaign starts with an individual or organisation who wishes to (partially) map the internet. The actual implementation and configuration of the scanner incorporates the goals and tactics of its initiator. While obvious, it is worth noting that scans resulting from scheduling arise without manual intervention. Therefore, they belong to the same scan campaign that was initiated with a clear knowledge goal. Both temporal and spatial (destination IP and port) patterns do not change within this informal definition of a scan campaign. A deliberate action or manual intervention from the initiator marks the start of a new campaign. We can only infer the intention and tactics of the initiator from the incoming data, but never be certain in the absence of ground truth. An extended period of inactivity that lasts several magnitudes longer than what is expected on average is very likely the result of manually powering down the scanner. Such phenomena can typically be observed at very long-term analysis e.g. one year. Scanners are often intended to work autonomously until a certain condition is met and a longer observation period has a high probability to record the end of a sequence of consecutive active days or a full campaign. Temporal activity patterns of a scanner become more unique at increasing scale, because the corresponding entropy of all possibility activity patterns increases exponentially. Scanners active on exactly the same days in a month provides can be used as a fingerprint. Extending that observation period to a full year would provide even more confidence when a group of similarly active scanners is found. Especially when the pattern is able to capture multiple scan campaigns. To conclude, incorporating more telescope data up to the point where multiple scan campaigns can be observed can provide more powerful features than present in short-term data for detecting distributed scanners.

6.3 Data pre-processing for scalability

We previously discussed discussed that more data in the temporal axis is beneficial for understanding scanner long-term behavior. However, this comes at great costs in terms of both computational and spacial complexity. This problem can be mitigated by transforming the raw telescope data to another format that is more space efficient and also faster to query

repeatedly. A year of telescope data was selected to be analysed. The total disk storage required for this data is 11TB, which is approximately the amount of unsolicited traffic the network telescope receives each year. Due to the scale of the data it was necessary to investigate which programming language and its corresponding libraries would provide short run time for reading all files. For this research's specific purpose of correlating distributed scanners, the telescope data can be greatly reduced in size by removing all packets unrelated to scan activity. Scans can be recognised as TCP-SYN packets. In addition, a trimming step will be performed which removes other unnecessary TCP and IP header values. All extracted data deemed relevant was made more compact by means of aggregation and more efficient type of data structures were created. Each of them containing different properties that are more suited to particular queries e.g. membership testing, retrieve scanners active in a specific period, total packets received and more. The goal is to parse the raw telescope data only once and continue data analysis on the newly created data structures which are more smaller in size and faster to query.

6.3.1 Parse data

Preliminary analysis was performed in the Python programming language, because the code could be written relatively quick and easy for experimenting with the data. While this approach was feasible for short observation periods, scalability would become an issue for increasing data size. Python's bad performance can be attributed to the code being interpreted whereas the C programming language compiles the code to machine language in advance which can be executed by the CPU directly. Also memory management has to be done manually, allowing better control for optimising efficiency . Therefore, the most heavy duty processing of parsing all telescope data will be done in the C language. In total, 360 days of telescope data would have to be parsed within acceptable run time. Since this is such an expensive operation in terms of CPU cycles, ideally only one pass should be performed. The objective is to create new representations of the data in a more compact format for future analysis. Subsequent data analysis will be written in Python for it's flexibility and high-level programming features at the cost of performance loss.

6.3.2 Trim dataset

The network telescope records all incoming traffic without any modifications to the data. Consequently, not all recorded packets are actually useful within the context of studying scanners. Other unsolicited packets include internet backscatter and packets due to host misconfigurations. Therefore, it makes sense to only retain information from packets that are considered deliberate scan attempts. Previously in 3.1 the different scan types were discussed. The most common scan technique is the TCP half-open or also referred to as a SYN scan. Such a scan is easily identified upon inspection of a packet's TCP header. TCP SYN packets typically do not contain a packet payload, because an actual two-way communication has yet to be established. This is in contrast to internet backscatter where a victim is flooded with requests from spoofed addresses. Depending on the type of service running at the victim, replies may contain a payload which make them several orders of magnitude larger than a scan. Thus removing packets unrelated to scans will contribute to achieving storage space reduction. Upon closer look at the IP and TCP headers, only a handful of values provide useful information regarding the scanner's behavior. The majority of the header

values serve to ensure correct operation of the TCP protocol which involves state management, integrity checks and other options. From both headers only the IP source address and the destination port is retained together with the timestamp of the arriving packet. This is the bare minimum to allow both spatial and temporal analysis with the source IP as the scanner's identifier. There were some other values that could potentially be useful, but were not included due to space considerations. For example, the IP destination address provides information regarding the target traversal pattern. However, that would lead to storing an additional 32 bits per packet. Typically, scanners randomly select a destination IP address. Such a strategy can be considered naive but effective in traversing the IPv4 space. Assuming most scanners even within the same distributed group generate random targets, the destination IP address is not very useful as a common fingerprint in determining collaboration between scanners. Some toolchains and very fast scanners encrypt information in header fields like the IP Identification, TCP source port and TCP Sequence number. Work has been done to identify specific toolchains based on fingerprinting Griffioen and Doerr [2020a]. Typically, these values are also randomised by implementations of the protocol unless modified intentionally. Therefore, it does not justify the increased storage space from recording these type of information to accommodate niche strategies. Furthermore, deployment using the same toolchain does not necessarily imply cooperation between scanners. In particular open source scanner software are readily available for anyone to use.

6.3.3 Data aggregation

In essence, what is being retained from individual scan packets are just the IP source address and the TCP destination port. Leaving everything else out will greatly reduce the required storage space. Data aggregation will take the reduction one step further. The idea is based on the repetitive behavior of scanners. Typically, scanners performing internet-wide scans are targeting just one or a handful of TCP destination ports in quick succession. Different IP destination addresses are (randomly) selected to achieve good coverage of the IPv4 space for a limited set of TCP ports. In other words, scanners are searching for any host running the services of interest. In the case of fast scanners the network telescope will typically see many incoming packets within a short time window. If it is expected that the same ports are targeted, then storing the IP source address and the TCP destination port on a per packet basis would not provide much additional information. The data size would be linear to the number of incoming packets. For very slow scanners this might not pose a storage problem, but high-speed scanners can potentially contact the telescope at megabits speed or faster. Therefore the approach is taken to create summaries of a scanner's activity that describe its target ports and the corresponding number of attempts.

Data was aggregated for every PCAP file recorded by the telescope. Recall that these files are of varying length between 5 and 20 minutes depending on the volume of unsolicited traffic at the moment of capture. Time windows of such lengths are a good choice, because campaigns are expected to run longer than that without any change in strategy. Counting the number of received packets provides a rough estimate on the scan rate. Selecting Wider time windows would lead to coarser summaries which are less detailed. A scanner could take breaks between bursts or only be active a fraction of the chosen time window. Another beneficial property of summarising the existing PCAP files is to easily locate the original packet information if there ever is a need to analyse a scanner in more detail down to the raw packet level. The naming conventions of the summaries will contain a recording times-

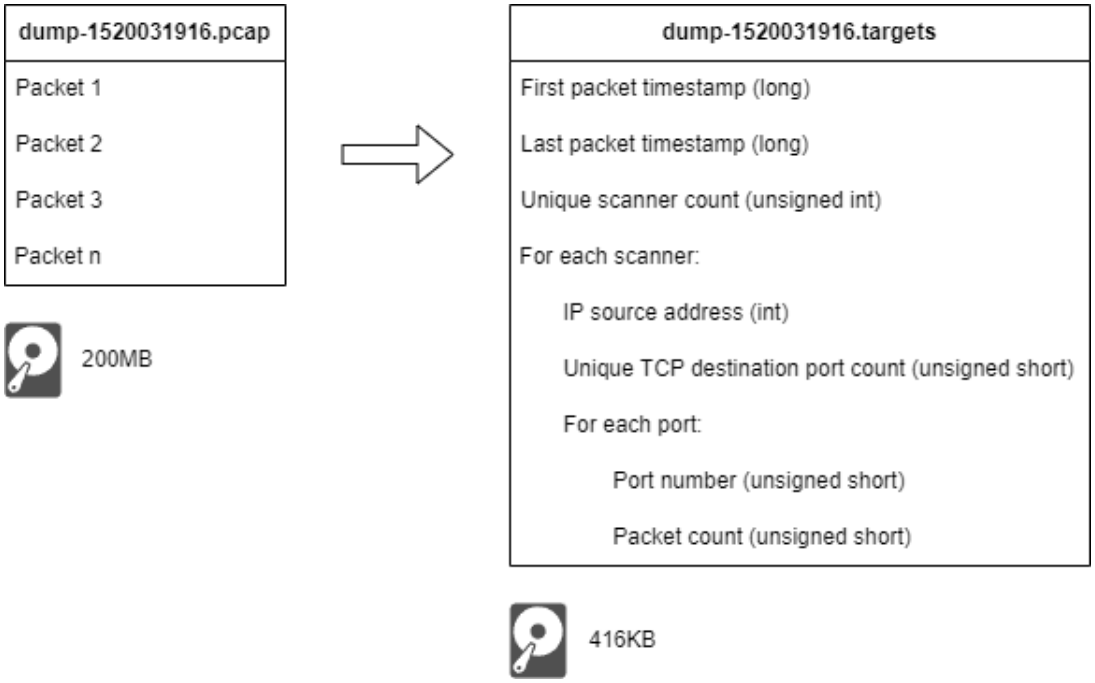

Figure 6.3: Transformation of raw telescope data in pcap format to summaries describing the scanners present and their target ports

tamp of the first incoming packet which matches the original PCAP file. Fig 6.3 illustrates the resulting summary after both trimming and aggregation of the original data.

The summary starts with stating the timestamp of the first and last packet of the original file in order to determine the exact time window of the capture. This is followed by a count of unique scanners identified by their source IP address. Every summary will contain at least these three pieces of meta-data regardless of the number actual packets received. What follows is a per scanner based summary. A scanner is described by its source IP address and a record of how many unique TCP ports it has contacted during the capture period. For each port, both the port number and the corresponding number of attempts to that port will be recorded. In order to achieve optimal space efficiency, the smallest data type that can accommodate any piece of information has been chosen. For example, port numbers range from 0 to 65535 which would precisely fit in an unsigned short data type. Special care was taken to also store the packet count into an unsigned short even though, while uncommon, the counter can potentially overflow if the scanner operates at high enough speeds. A count overflow to a specific port would indeed imply a very fast scanner which is sufficient information for the purpose of this study. Selecting a data type that is larger e.g. 32-bits integer would greatly increase the required storage requirements for all scanners in order to accurately represent these outliers. Generally very fast scanners pose a smaller threat in the sense that they are already easily detected by existing methods. In particular the slow scanners that show a high degree of sophistication are of more interest. To indicate an overflow, the counter will be assigned a special 0 value. The value 0 is unused, because there is only a record of a TCP port and its counter if it at least one probe was received.

To support long-term analysis, summaries of coarser time windows where also created. These will provide a lower resolution perspective of the telescope data, effectively zooming out of the data. A lower resolution perspective is especially suitable for observing a scanner's long-term behavior, because it enables temporal analysis on longer observation periods with less processing time. For this reason, daily summaries were created that can be used in conjunction with the higher-resolution summaries. The former follows the same data format for a longer time window, resulting in stronger data compression. Continuing the previous example in fig 6.3, a day of unmodified telescope data is stored using 37.5GB. The corresponding high-resolution summaries for that day combined are only 72MB in size. After the second low-resolution aggregation step, a single daily summary is 2.1MB, a near 18000 times size reduction compared to the original data. Having views of varying resolutions will allow hierarchical search algorithms to reduce the run-time exponentially. The presence of certain scanners within a year's time can be quickly determined on a coarser level. If the need arises to investigate scanners of interest in more detail then increasingly detailed views can be consulted down to the raw packet level.

7 long-term activity feature

7.1 Observed scanner activity

In the previous section we described how the raw telescope data was transformed into efficient summaries which enables fast membership testing in daily resolution and pcap resolution. This opens up the possibility to easily visualise a scanner's activity during the entire observation period of 1 year. In manual exploration of the data we encountered several examples of potentially collaborating clusters where such a tool proved to be insightful. Initial experimentation included trying out several features such as packet count, scan rate, interarrival times and evualating their efficacy in detecting distributed scanners. Detected clusters belonging to the same /24 subnet provided quick confirmation. Additionally, plotting the long-term activity of every scanner in such clusters often revealed that distributed scanners operate simultaneously. In other words, scanners all appear to be active and scanning or entirely absent for any given day within the observation period. Figure 7.1 illustrates a detected cluster with exactly the same activity pattern. A blue dot in the chart represents recorded activity of at least one packet on a specific day from the corresponding scanner. This example depicts a cluster which operates on a weekly schedule most of the time from March 2018 till August 2018. Lacking any labels in this research phase, we can treat the source address being in the same /24 subnet as the best available alternative to having ground truth. We continue to observe a high correlation with matching activity patterns and scanners from the same subnet. The latter indicator by itself is already able to instantly confirm the presence of confirmation by visual inspection. In figure 7.2 we witnessed another group of scanners where the source IP addresses are not in close proximity. That is, they do not reside within the same /24 or not even /16 subnet. Traditionally, IP address proximity was the most relied upon characteristic to infer collaboration between scanners. However, visualisation of long-term activity reveals a common feature among scanners which can serve as a fingerprint to identify a group regardless of the source addresses.

The strength of this evidence lies in the fact that there are 360 seperate days for which there are two observations possible: Either a scanner has sent one more probes or none at all. Both the collective presence and the absence of an entire cluster provides additional confidence to the correct identification of a group. The statistical probability of such an event where multiple scanners have exactly the same operating days is very low. Even without quantification of the probability, intuitively one could confidently say that they are in fact collaborating. We continued to discover multiple instances in which the potential cluster as a whole followed a distinctive pattern throughout the year. These observation were key supporting evidences in the formulation of a hypothesis that distributed scanners share the same long-term activity pattern. Thus, we consider this characteristic as a powerful feature that can be used to effectively correlate scanners to their respective group and seperate them from other data points.

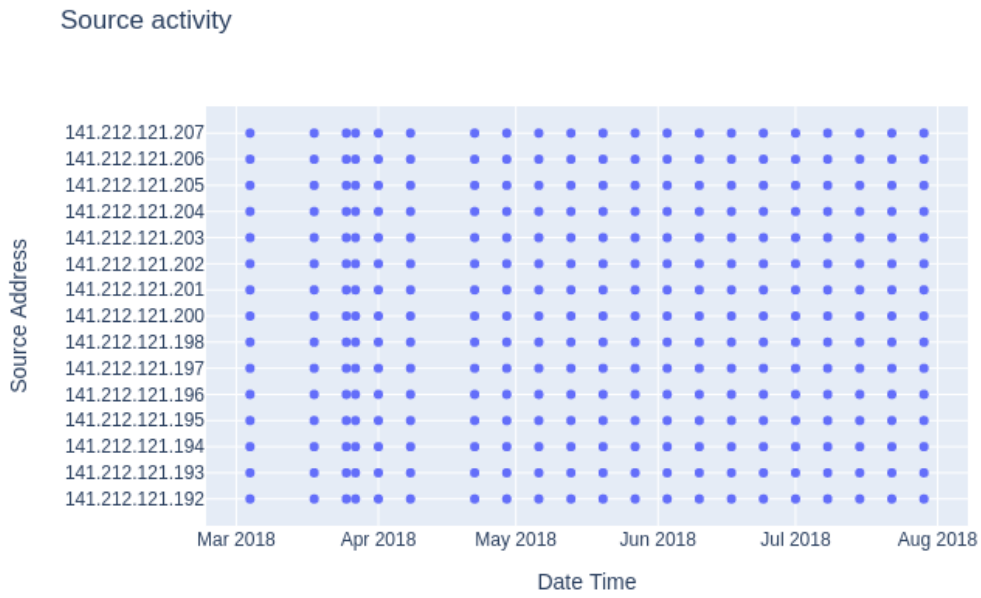

Figure 7.1: Example of patterns in scan activity found during data exploration

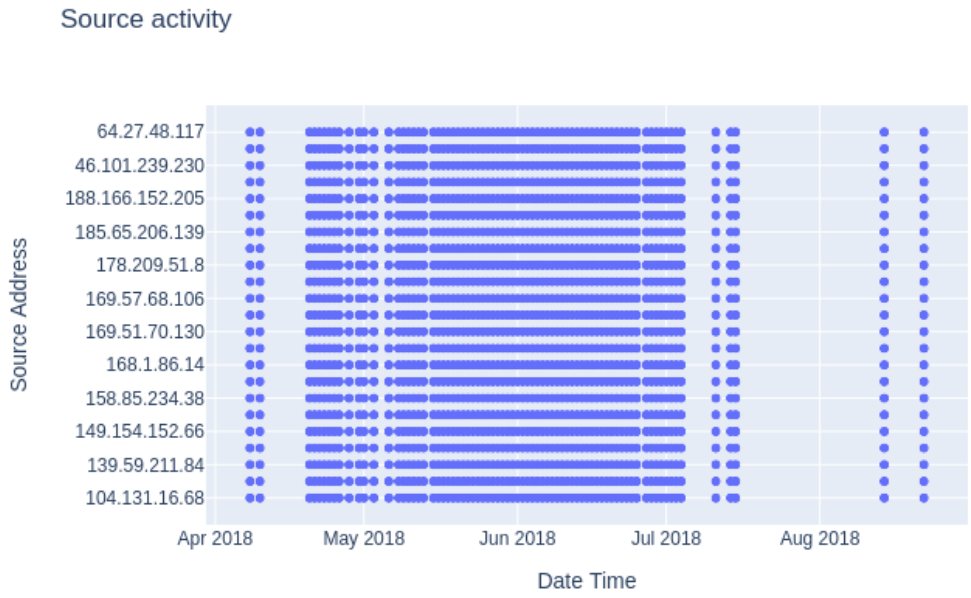

Figure 7.2: Scanner with similar activity not within same subnet

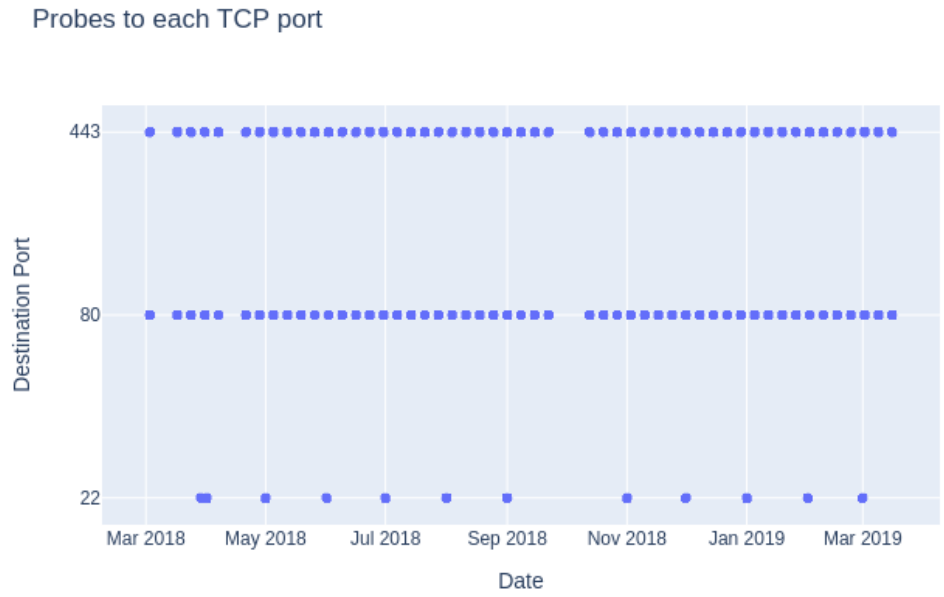

Figure 7.3: Instance of a verified cluster and its targeted TCP ports over time

7.2 Inluding target ports in activity

Currently we have only discussed binary activity patterns where a scanner can either be active or inactive in a specific time window. In addition, the previously created daily summaries also recorded how many packet were sent to a port number. Which port numbers a group as whole is targeting reveals what information is of interest to the initiator of the scan. Port scanning as a form of network reconnaissance involves mapping the network of interest and obtaining knowledge about running services. Therefore, we can expect the individual scanners to operate in line with a common goal. Fig 7.3 depicts a moderately sized cluster for which collaboration has already been verified by means of both subnet proximity and matching binary activity sequence. By visualising the targeted ports in daily intervals we can infer the purpose of such a scan campaign. The services associated with port 80, 443 and 22 are HTTP, HTTPS and SSH respectively.

For the purpose of detecting distributed scanners we do not intend to dive deeper into why anyone is interested in specific target ports and their corresponding services. However, these observations do provide evidence that not only do clusters operate on the same days, but they also are also instructed to simultaneously target a common set of ports. It's these shared characteristics between scanners that will enable group detection beyond and establish confidence beyond reasonable doubt. Thus, including target ports in scanner activity increases the entropy. Depending on the uniqueness of any activity sequence we can confidently say that similar scanners in terms of activity are related. In figure 7.4 another cluster instance is shown with a much more complex target pattern. In light of such evidence one can hardly argue the presence of a coordination.

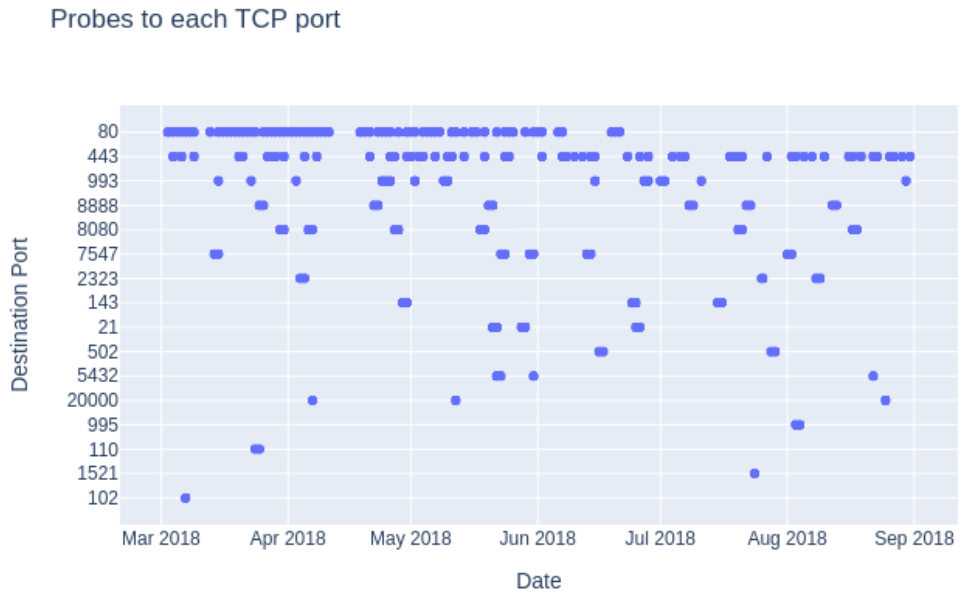

Figure 7.4: Instance of a verified cluster with more complex target scheduling and larger set of interested ports

7.3 selecting activity as feature

For every scanner encountered during the 360 day observation period a sequence of bits will be created. Each bit indicates whether the scanner was seen active during the corresponding day. A scanner is considered active when at least incoming probe arrives at the network telescope. Therefore, the slowest rate this model can accommodate is 1 probe per day which should be applicable to the majority of slow scanners. Modern IDS have a much higher threshold configured in the range of multiple probes within 5 minutes for their scan detection. For the adversary it would seem unnecessary to sacrifice any more speed when the scanner is already blending in with normal traffic. The study performed here is the first to track scanners for nearly a whole year. Therefore, any adversary with a strong intention to obfuscate his reconnaissance efforts is unlikely to have accounted for such analysis. Furthermore, there is a difference in attempting to evade real-time detection and going unnoticed during post-analysis. This work falls under the latter category which can afford a much longer observation period due the absence of some time constraints present in real-time detection. Scanners typically are more concerned with completing their reconnaissance objective and thus it is sufficient to prevent being blocked during an ongoing scan.

7.3.1 Data distribution

With 360 measurements in a year, there exists many combinations of binary activity sequences or also called bitstrings. Although each solo scanner or group exhibits some degree

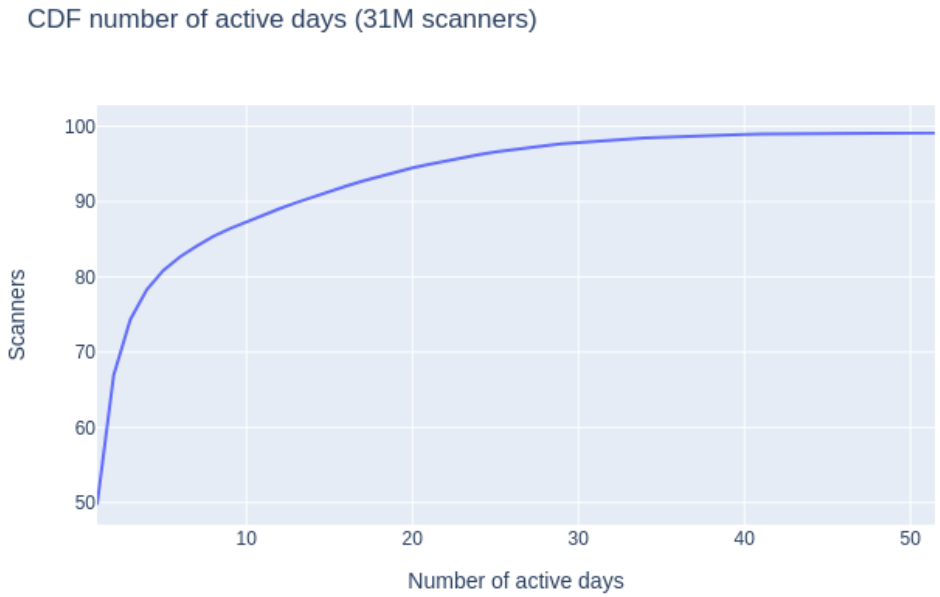

Figure 7.5: Cumulative distribution function of active days per scanner

of uniqueness with respect to its strategy, some notable trends can be observed. The first analysis seeks to answer how many days a scanner is active in general during the period of one year. In order to unambiguously identify groups by means of their activity pattern, there should be at least a certain amount of active days to allow for sufficient combinations of bitstrings. The number of possible combinations can be calculated using the following equation

$$C(n,r) = n!/r! * (n-r)! \tag{7.1}$$

This is a general equation from set theory to find the total number of combinations of size r from a set of size n. Application to the current context translates to determining how many variations in bitstring are possible when a scanner is active for a certain amount of days. More possible combinations make it less likely that unrelated scanners will have a similar bitstring. The maximum number of combinations is achieved when $r = n/2$ Thus, for identified groups of which the scanners approach 180 active days it can be said with increasing confidence that these are indeed collaborating. At on extreme, scanners have only one active day and it does not require much elaboration why groups formed by the same logic do not make a strong case.

Fig 7.5 and fig 7.6 depict the distribution of active days from all 31 million scanners present in the dataset. What these figures tell us is that the majority of scanners are only active for a fraction of the year. With 95 percent of the scanners being less than 20 days active, this is not an ideal situation considering the scenario with maximum combinations was determined at

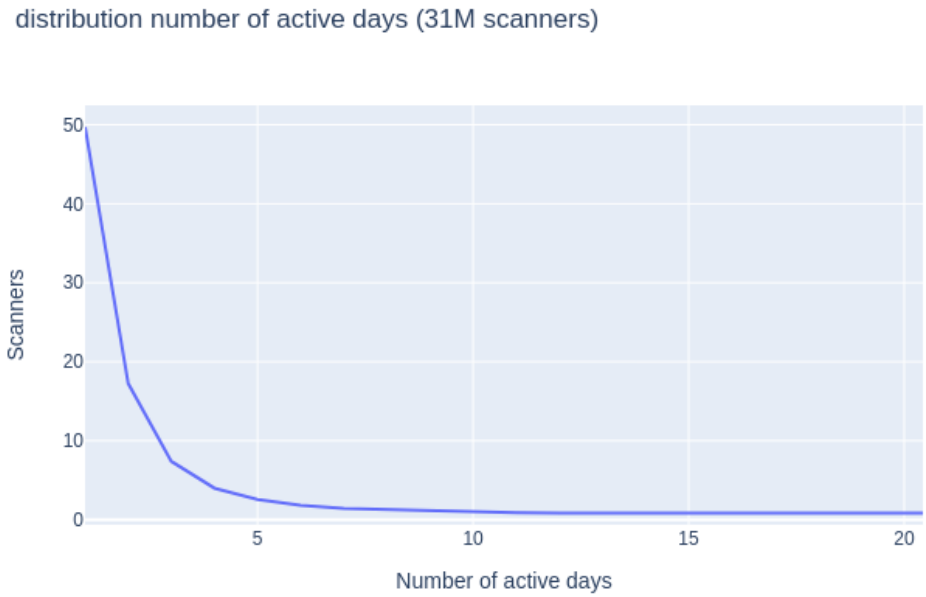

Figure 7.6: Probability density function of active days per scanner

180 active days. Even more problematic is that 50 percent of the scanners are only active for a single day in which case there are only 360 combinations. Such a small number of possibilities is nowhere near enough to uniquely identify groups within the pool of over 15 millions scanners without containing many false positives. Continuing on this thought, this work acknowledges that correlation based on activity bitstrings with a low-false positive rate can only be reliably performed for scanners exceeding a minimum number of active days. During post-analysis of the correlation results, the corresponding activity threshold can be set in accordance with the false-positive tolerance to query clusters meeting this criteria. A lower false-positive rate comes at the expensive of fewer clusters returned, but due to the large size of the database this should still yield ample results. In other words, settings such an activity threshold is an assumption made on the number of active days which exclude certain type of scanners.

Another interesting observation made is the relationship between active days and the number of distinct campaigns launched with the scanner. A campaign is defined as an uninterrupted sequence of active scanning days, inactivity of at least a full day marks the end of a campaign. In addition to knowing what fraction of the year a scanner is active, this also provides more insight to how long campaigns tend to last. In 7.7 a surface plot is provided that with one picture is able to roughly illustrate 360 cumulative distribution functions of how many campaigns are launched by the scanners categorized by active days. Fewer campaigns imply that most of the activity appear in consecutive days which is especially the case for less active scanners, partly due to the mere fact that fewer combinations are possible as previously discussed. Scanners approaching the 180 days of activity have more options and this is reflected in the same figure by a less steep rise of the curve. However, in general most of the active days are concentrated in a significant less number of campaigns which supports

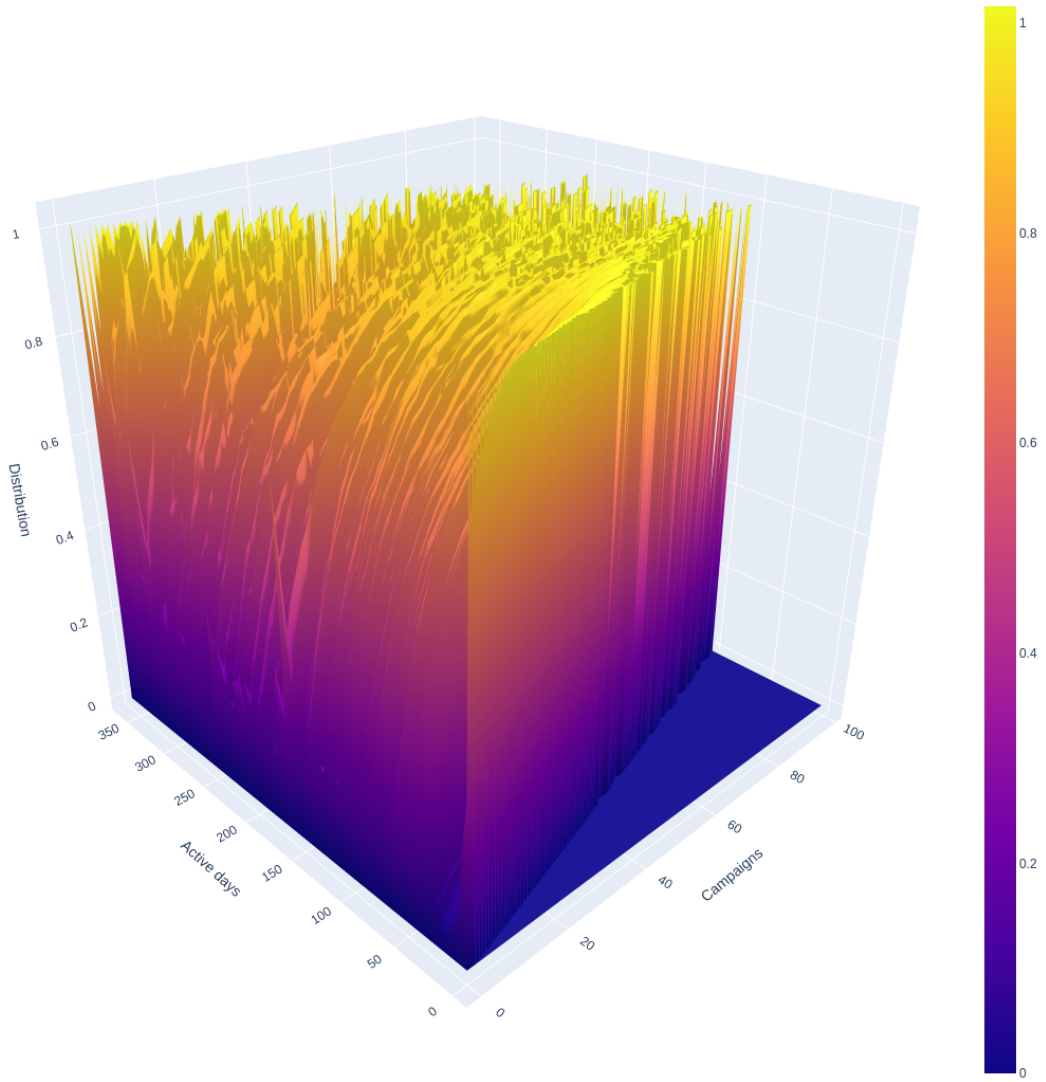

Figure 7.7: 360 Cumulative distribution functions of campaign count

the idea that once scanners arrive in an (in)active state they are more likely to continue to remain there in following day.

8 Methodology for correlating distributed scanners

8.1 Correlation features

Within the observation period of one year a vast amount of data has been collected from the university network telescope. During this time more than 31 million scanners have been sighted. Distributed scanners are considered a form of advanced scanning technique where multiple scanners are under the control of the same initiator. In this work an attempt is made to detect such a coordinated relationship based on the assumption that scanners part of the same group are likely to exhibit similarities in terms of their scanning behavior due to economies of scale. Similarities can express itself in many characteristics of scanner behavior, but two types are of particular focus of this work. These are a scanner's activity pattern and targeted services which are expected to be highly correlated within a group. More specifically, when the entire observation period is divided into roughly equal time intervals, then all scanners within the groups are likely to be simultaneously in the same operational state, either active or inactive depending whether or not at least one probe has been received. Additionally, in the active state also the set of targeted destination ports must match across the entire group. Due to the degree of uniqueness of such observed behavior, seemingly unrelated scanners in a very large candidate pool can be associated to a single group with low false-positive rate.

8.2 Dataset characteristics

The dataset contains an unknown number of clusters, because there is neither a lower nor upper limit (besides the number of available IPv4 addresses) of how many distributed scanners have been observed by the telescope within the timespan of 360 days. However, due to the lower barrier of entry and simplicity the majority of scanner are assumed to be operating solo as this approach is often sufficient for a typical network reconnaissance without obfuscation requirements. Therefore, the majority of datapoints are considered noise as solo scanners are not the subject of interest, only a small subset belongs to an unknown number of clusters. For a dataset of this scale containing roughly 31 million scanners, this should still yield a sufficient amount of results with reasonable confidence despite inevitably some distributed scanners remain undetected, because the aforementioned assumptions of similar activity patterns and target ports are a generalisation and do not apply to the whole class of distributed scanners.

8.3 Common clustering algorithms

To solve the clustering problem there are many popular algorithms which have been extensively used in other research areas. The choice of algorithm largely depends on the nature of the problem. For this reason, the main data characteristics discussed in 8.2 are taken into consideration during the selection process. If none of the options perfectly fit this specific case then this necessitates developing a tailored solution. Clustering algorithms are primarily classified based on the distribution of data. Each class makes assumptions about the pattern or structure in which the data points are arranged in. For example, the well known K-means algorithm tends to be effective in situations where the data points are arranged in a circular shape surrounding the cluster's center, also referred to as the centroid. Being one of the most simple and fast unsupervised learning techniques has now made it become ubiquitous in literature. The letter K in the name stands for the pre-defined number of clusters that will be generated. Centroids initially are placed at an arbitrary locations which are subjected to an iterative optimisation process. The end goal is to assign all data points to a centroid with minimal distance. Several reasons can be named why this algorithm is not a viable option to cluster the data in this work. The most important one relates to the core assumption of being able to specify the number of clusters in advance which is not possible in the context of distributed scanners present in real-world data. Additionally, there is no concept of noise and it is designed to work with numerical features as opposed to categorical features. Therefore, K-means can be quickly disregarded as a potential solution to the cluster problem.

Keeping the earlier criticism in mind which was used to judge the K-means algorithm, DBSCAN Ester [2017] is a better alternative. DBSCAN stands for Density-Based Spatial Clustering of Applications with Noise. It performs well in separating high density from low-density areas. Clusters can be formed of any arbitrary shape as long as the data points are considered to be in a high density area, the rest are deemed outliers. Depending on the context of the application, outliers can provide valuable information such as in intrusion detection where anomalies indicate deviations from normal behaviour. However, in this work the focus lies on the high-density areas that represent distributed scanners with similar behavior. Solo scanners share little to no commonalities with others and thus can be treated as unwanted noise which should be filtered out. Another major advantage of DBSCAN is that no a priori knowledge about the number of clusters is required. The algorithm is able to detect how many clusters are present though some trial and error is required during parameter configuration to achieve optimal results. The outcome can greatly vary depending on the choice of minPts and epsilon. Sensitivity to parameter configuration is a known drawback of this algorithm and it can be computationally expensive for large datasets due to its $O(n^2)$ runtime which gets even worse for high-dimensional data. Therefore, clustering on the full dataset will be computationally infeasible. Even if the process could be completed in acceptable runtime, the end result of cluster algorithms in general can be hard to interpret. Different parameter configurations can lead to completely different results which might pose some uncertainty to the validity of the labelled data points by the algorithm. Additional adjustments can be made after the evaluation of produced clusters, but this requires measuring the performance by comparing to the ground truth, which unfortunately is absent in real-world data. This problem can be partially mitigated by running the algorithm with artificially generated data, but these are often not representative of reality because its hard to correctly model the true distribution. As a consequence, the produced data is a simplification which incorporates the assumptions made by the analyst for which

the performance metrics can paint a too optimistic picture of its efficacy. Both the scale of the dataset and the difficulty of cluster interpretation were the main reason to be in favor of creating a custom algorithm especially suited for the task which is more fast and simple.
After reviewing these existing clustering algorithm we decided that none of them are suited for detection on large scale data. The algorithms are computationally expensive due to calculation of pairwise distances and the results are hard to interpret. Therefore, we opt to require scanners have exact matching patterns instead of being similar in order to drastically reduce the complexity. The following section will discuss the custom algorithm in more detail.

8.4 Implementation

8.4.1 Overview

Figure 8.1 presents a high-level overview of the correlation method. Starting from the university telescope data, scanner related traffic captured within the observation period was aggregated into a more compact format which has been discussed in section refsection:telescope. Without the necessary reduction in data size, long-term analysis would not have been feasible. A total of 360 summaries, one for each 24 hour period, contain all source IP addresses that have contacted the telescope including the corresponding number of packets destined to each TCP port. The presence of a specific scanner in any particular day can be queried in $O(1)$. When performed over all 360 days in the observation, a scanner's activity can be represented as a string of bits where a 1 value indicates the active state and a 0 means that the scanner was absent.

8.4.2 Bitstring correlation

For all 31 million scanners in the dataset their corresponding bitstring are calculated. The initial clusters are produced by matching scanners with equal bitstrings. Scanners that are seen either active or inactive on exactly the same days for the whole year are likely to be cooperating. In the example output from figure 8.1, IP2 to IP5 have the same bitstring and therefore they are included in the same cluster. Here the strings are simplified to length 6 for illustration purposes. In reality there are 360 bits, one representing the state for each day. IP1 is the only scanner in its cluster and is therefore operating on its own. A lower limit for the number of scanners can be set in order to filter out small clusters. Correlation based on bitstrings alone can be sufficient depending on the tolerance for the number of false-positives in the results. Due to the large number of data samples, there is a non-negligible chance that two unrelated scanners have matching bitstrings by coincidence. A longer observation period divided into many intervals somewhat mitigates this issue due to increased entropy.

8.4.3 Target Port Set Correlation

In an effort to reduce the number of false-positives even further, this work employs an additional correlation step that also takes into account which TCP destination ports are targeted. A commonly observed strategy for coordinated scanners is that they target the

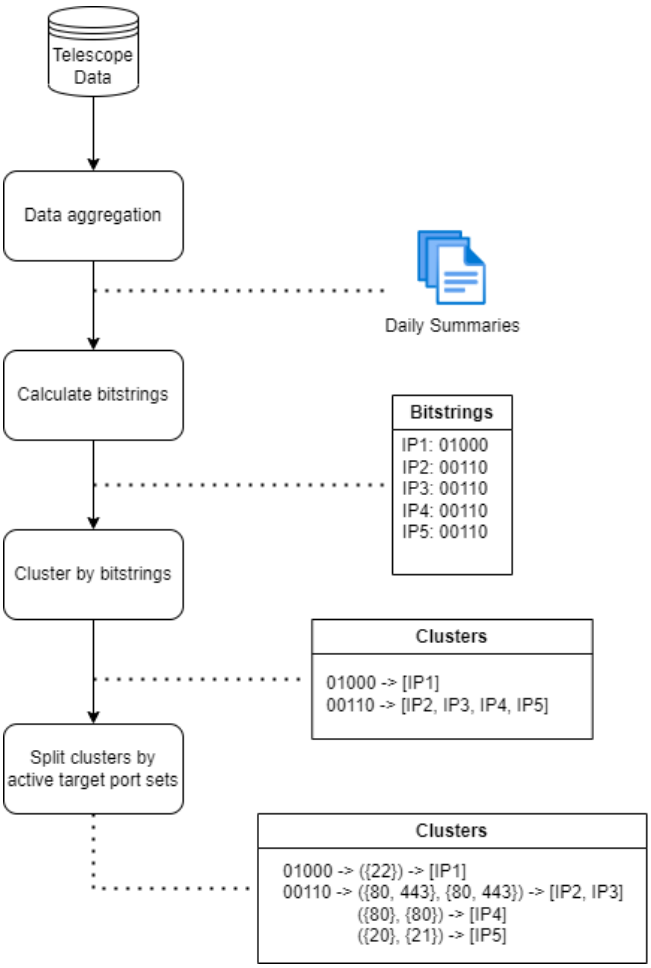

Figure 8.1: Methodology overview: Correlation steps and intermediate results

same destination ports. Therefore, the previously produced clusters are split according to the assumption of matching port sets. A port set includes all the destination ports a scanner has contacted within a day. The number of active days indicated by the bitstring is accompanied by an equal number of port sets. After all, being active implies scanning at least one port. The size of a set can vary anywhere between one and the theoretical upper limit where all ports are included, which is $!(2^{16} - 1)$. However, we typically see only a handful of ports being targeted. Continuing on the previous example in Fig 8.1, IP2 to IP5 which share the same bitstring do not all have the same sequence of target sets. These scanners with this particular bitstring will have two port sets, because that equals the number of times they have been observed in the active state. Only IP2 and IP3 have matching port sets across all days, thus IP4 and IP5 end up in a different cluster. IP1 had a unique activity pattern to begin with, no split can occur with just a single member in the cluster.

8.4.4 Storing results

Matching based on both bitstrings and port sets should occur fast with no false-positives, preferably in $O(1)$. Fast look-up times allow scanners to be quickly placed in their respective clusters. One common data structure that possesses such property is a dictionary. In a dictionary, data is stored as key-value pairs where the key is used as the input to a hashing algorithm. A good hashing algorithm transforms keys to distinct values which serve to point to the exact memory locations containing the corresponding value. Given that enough memory is allocated to accommodate the number of keys, instances where two keys produce the same hash (collisions) will be rare and look-up can complete in constant time. Scanners are first matched according to their bitstrings and subsequently to port sets. Consequently, this approach is reflected in how clusters are stored. A two-layer nested dictionary uses bitstrings as key in the first layer, the value will hold a second dictionary using port sets as key. Finally, the scanners matching both keys are added to the cluster. Unseen keys are added to the dictionary at first occurrence. A requirements for keys is that the data is immutable which a standard set data type to hold the ports is not. Therefore, an immutable version of the set called a frozenset is used. The ordering of frozensets matter as they indicate which ports are scanned on any active day. Lists preserve the order, but are also not immutable. Thus the key for the second-layer dictionary will have one or more frozensets contained in a tuple.

8.5 Algorithm analysis

In fig 8.2 the distinct steps of the algorithm will be described in psuedocode which can be understood by anyone with basic programming knowledge. Psuedocode is useful at explaning the thought process without also having to deal with language-specific syntax. It is more detailed than a flow-chart, but much easier to read than actual lines of code.

The input to this algorithm are all encountered scanners in the dataset represented by their source IP. Therefore, a run-time analysis is provided which is entirely dependent on the number of scanners. The first loop at line 2 cycles through every scanner and that is exactly the number of times the rest of the code is executed, from line 3 till the end. As a result, the algorithm run-time can be immediately identified to be at least $O(n)$. Within the loop, first the scanner specific bitstring and targets variable are initialised. Continuing on, the

Algorithm 1 Correlation algorithm

```
1: SET clusters to empty dictionary                          ▷ To hold all detected clusters
2: LOAD summaries
3: for each ip in the dataset do
4:     SET bitstring to empty bitstring
5:     SET targets to empty list
6:     for each summary do                    ▷ 360 daily summaries in the observation period
7:         if ip in summary then                              ▷ ip seen active in current day
8:             APPEND bitstring with 1                    ▷ a '1' indicates active state
9:             GET portset of ip from summary          ▷ ports the scanner has targeted
10:            APPEND targets with portset
11:        else
12:            APPEND bitstring with 0
13:        end if
14:    end for
15:    CONVERT bitstring and targets to immutable datatype
16:    if bitstring not in clusters then                          ▷ check bitstring existence
17:        SET clusters[bitstring] to empty dictionary
18:    end if
19:    if targets not in clusters[bitstring] then                 ▷ check targets existence
20:        SET clusters[bitstring][targets] to empty set
21:    end if
22:    INSERT ip to clusters[bitstring][targets]      ▷ add scanner to its corresponding cluster
23: end for
```

Figure 8.2: Psuedo code of the correlation algorithm

second for loop at line 5 is responsible for determining both which days the scanner is active and what ports it has scanned. The necessary information are all contained in the summary files which have been already loaded into memory. In this specific context there are a fixed number of summaries (360) and no dependency on the input size thus the for loop is considered to be executed in constant time. Upon completion the remaining instructions place the scanner in a set collection associated by its corresponding bitstring and sequence of targeted port sets. The dictionary data structure was specifically chosen to hold bitstrings since a lookup can be completed in $O(1)$. Although the statements contained in the initial for loop take non-negligble amount of time, they do not increase the run-time complexity beyond $O(n)$.

The quality of an algorithm besides execution time is also defined by its efficient usage of memory. Typically in computer science both time complexity and space complexity analysis is performed to evaluate an algorithm. The latter consists of the total memory of the input space and auxiliary space combined as function of the input. Auxiliary space is the extra space or temporary memory required during execution of the algorithm. For instance, to store variables and other constant which are not present in the final output. The correlation algorithm takes as input the observed scanners contained in the summary files which are loaded into memory. Hence, the total size of the summaries is directly correlated to the number of scanners. An increase in the number of scanners corresponding to a linear increase of the input space, because each scanner can only occur once in a summary regardless of how many other scanners are present. The correlation result is stored in the ”clusters” variable of the dictionary type. Memory allocated for dictionaries in python grow as more key-value pairs are added. They are implemented as hashtables and typically will have n

keys and n values which leads to $O(n)$ space consumption. In the worst case scenario, each scanner has a distinctive bitstring and will end up in a cluster of size 1. With n scanners the cluster dictionary has n key entries.

9 Results

The detection methodology discussed in the previous section was able to detect a significant number of clusters containing distributed scanners. Briefly said, there are a total of 54611 clusters and 1.7 million participating scanners. Correlation of coordinating scanners as proposed in this work is based on the assumption of similar behavioral activity (from the perspective of the defender's network) across the entire observation period. Fig 9.1 provides a simple example of the necessary condition in which a set of scanners is assigned to the same group. Each scanner's entire activity is defined as a sequence of daily observations of the targeted port numbers. Unique ports that have received at least one probe are recorded in a corresponding set for that particular day. In the complete absence of probes, the set remains empty and the scanner is considered inactive for that day. Those scanners possessing exactly the same sequence of targeted ports during the entire span of 360 days are considered to be collaborating due to the small likelihood that these happen to match by chance. In essence this is how correlation is implemented and this chapter will go into more detail how the end result is stored as a data file following the same hierarchical structure. Furthermore, guidance is provided in how to query the end result for future analysis. One such analysis is subsequently performed to grant insight into the size distribution of detected clusters.

9.1 Algorithm output

Implementation The implemented algorithm tasked was tasked with finding all distributed scanners within a one year observation period. Without the necessary pre-processing steps on the data, correlation as proposed in this research would not have been computationally feasible. Trimming and aggregating the massive amount of data into daily summaries has enabled the algorithm to effectively complete in less than a few hours.

Data structure of the result The result containing all detected clusters and participating scanners are stored in a single file. Fig 9.2 shows the hierarchical structure in which the data is stored. Additionally, an effort has been made to elaborate which values are valid for the various elements in set notation. The data storage follows the same approach as the proposed methodology which consists of two layers. First, the presence of activity in any of the 360 days recorded as boolean values. And second, the corresponding port numbers which have been scanned during active days. This means that any detected cluster can be unambiguously identified by these two parts since all participating scanners with have a matching activity sequence. As a consequence, a scanner can not be part of more than one cluster according to this definition. The top layer consists of an unordered set of boolean sequences of length 360, each corresponding to a single day. Depending on which ports have been targeted, the group of scanners matching the first part are further partitioned into smaller clusters when some have deviating targets. For this reason, scanners can be considered part of many different cluster even though they have been witnessed active during

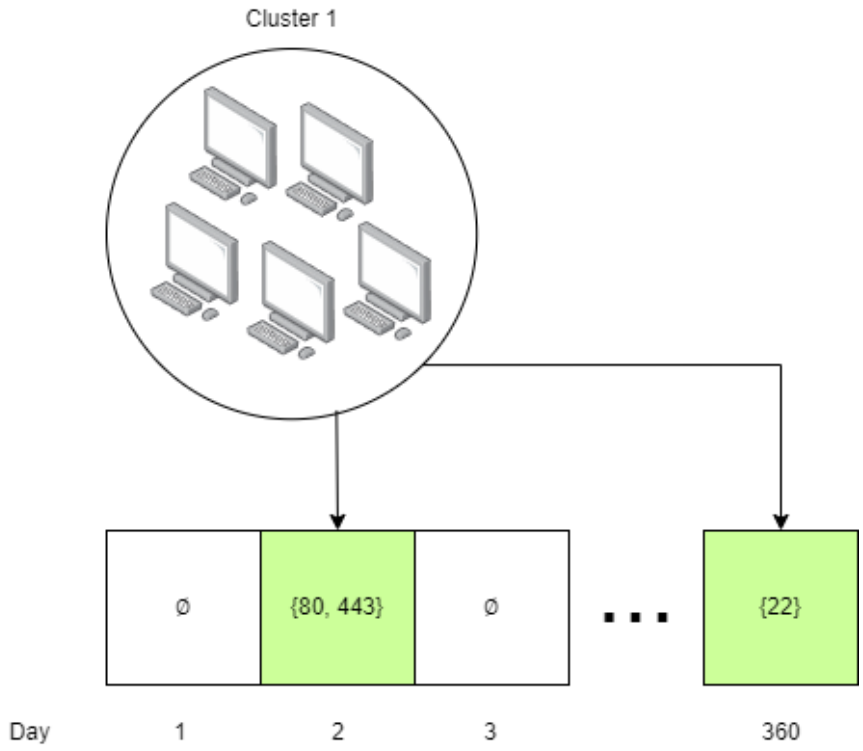

Figure 9.1: Simplified clusters correlation example based on targeted ports during active days

the same days. It can be argued that there is a reasonable chance that these smaller cluster are actually part of the same scanning effort, because there is some degree of similarity in their strategy. A stronger case can be made for activity sequences that involve multiple state transitions, but this discussion will be continued in the next chapter.

Minimum cluster requirements and storage space The algorithm computes the activity sequence for every scanner in the dataset, but only those that have a minimum number of matching scanners will be recorded in the end result. This significantly reduces the required storage space when only clusters of non-trivial size are of interest. In the current implementation the minimum cluster size has been set to 5. Another requirement is the minimum number of active days. Scanners should be active for at least 3 out of 360 days in an attempt to reduce false positives. Recall that there are a total of 31 million scanners in the dataset. Combining that with the typically low number of active days seen per scanner necessitates such a filter requirement. Furthermore, the majority of scanners are assumed to be operating individually. Subsequent post-analysis would be hindered if such false-positive noise are included in the resulting collection of detected distributed scanners. The final disk size to store the result is 19.5 MB.

Result Queries Queries performed on the result is straightforward and fast especially when the cluster activity and target ports can provided as an input. Due to hierarchical structure of the dataset, accessing the top layers is easier as it requires less iteration than for example leaf nodes which are the scanner IP addresses. Both activity sequences and the ports set can be accessed in $O(1)$, because the hash of these values are computed for the purpose of storage and retrieval. Only when these values have to meet a certain condition is it necessary to iterate over all entries in the unordered dictionary in which these have been stored. For example, retrieving those clusters matching a specific 360 day sequence of activity will be performed instantaneously. In contrast, when filtering for specific activity on a subset for days requires iterating over all entries. The slowest query is looking for which cluster a specific scanner IP address belongs to. In the worst case both the activity sequence and the port sets have to be fully iterated which requires $O(n^2)$ runtime. However, due to the manageable number of detected clusters this does not pose a significant issue as of yet and such type of queries can still be completed within reasonable time. Should the need arise in the future to speed up certain type of queries for which the current structure is currently deemed inefficient, then it can be readily transformed into for example a binary tree. An alternative is to the reverse the layers to support looking for specific scanners.

9.2 Cluster Size

Why size matters At the start of this chapter it was mentioned that a total count of 44491 cluster have been detected which involved 1.6 million scanners. Not all clusters are equal and one of the main characteristics is its size. As mentioned before, the size of cluster is an indication of the adversary's sophistication. Assembling and coordinating a large number of scanners requires both many resources and advanced skills. This can involve developing custom software which are not readily available tools such as nmap. Deploying a large number of scanners in a single campaign enables them to complete reconnaissance faster while also avoiding detection. For these reasons, larger clusters are potentially more

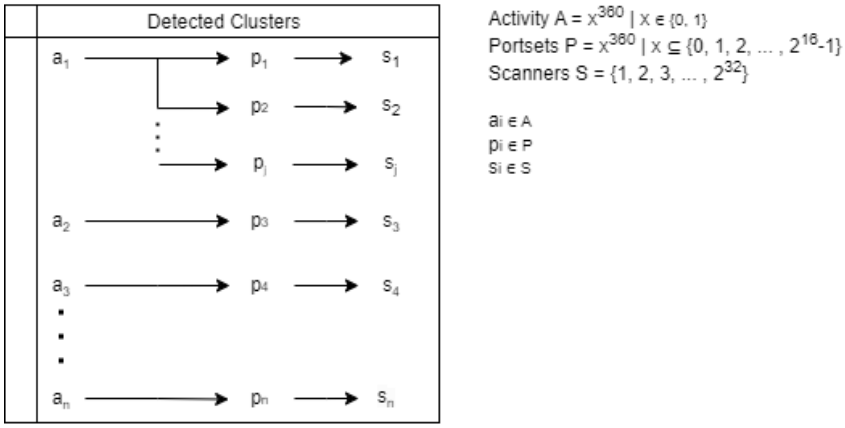

Figure 9.2: Hierarchical structure output file of detected clusters

dangerous. Some basic analysis has been performed on the initial results in order to provide insights into the distribution of the clusters in terms of size. Intuitively, fewer instances of clusters can be expected as they grow. This hypothesis is supported by Fig. 9.3 which shows a histogram of the cluster sizes. The number of clusters decreases exponentially as the size doubles in each bin. One cluster involved 9017 participating scanners and is by far the largest in the result.

A more detailed perspective of the cluster distribution is provided in table 9.4. However, instead of isolated bins now the counts are shown for clusters that exceed a particular size. This helps answer questions regarding the prevalence of threats that are above a threshold that's considered dangerous. A group of 40 scanners working together is in our opinion already pretty significant of which there are 8219 those present. Above size 160 are 2045 occurrences which likely includes entire \24 subnets. Further analysis will be performed on the largest clusters in order to understand how the size is being utilised. There might be a correlation between cluster size and scan rate of the individual scanner if the purpose is obfuscation. An interesting observation can be made by counting the total number of scanners across clusters above a certain size. There a many more smaller clusters, but the majority of scanners participate in those that are larger. For instance, 60 percent of the largest clusters account for 92 percent of the scanners. And also, the top 9% contain 58% of all scanners. In other words, the relatively few instances of large clusters does not necessarily mean a lower number of threats in terms of scanner count.

9.3 Case Study: Slow Korean Scanners

Manual investigation of all detected clusters is too time consuming. There are however some notable clusters that deserve increased attention. Since the detection methodology has been developed with the capability of detected extremely slow clusters, it would be interesting to dive deeper into such an example where the scanners deliberately send probes at a low rate which would have gone unnoticed by conventional detection methods.

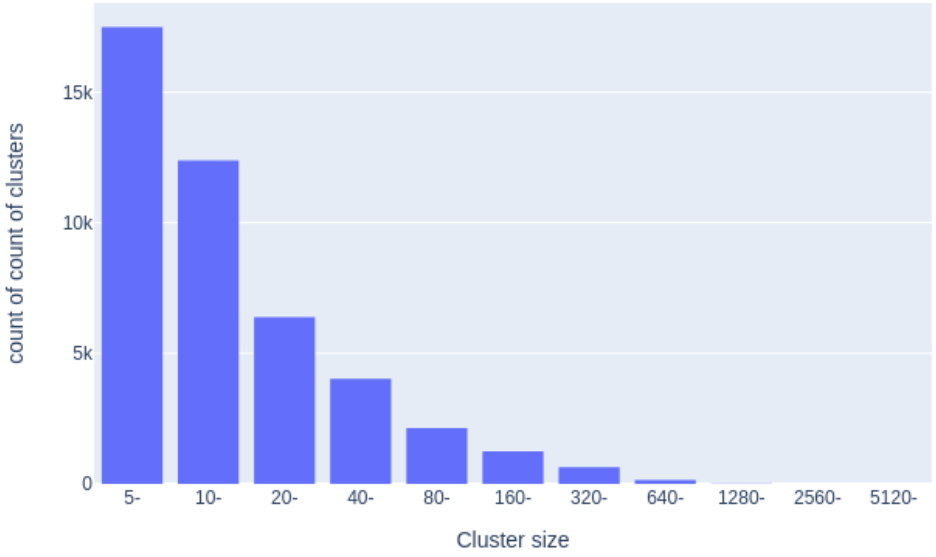

Figure 9.3: Histogram of cluster size distribution

Cluster size	Cluster count	Clusters %	Participating Scanners	Scanners %
≥ 5	44491	100%	1639304	100%
≥ 10	26997	60.7%	1514310	92.38%
≥ 20	14605	32.83%	1348818	82.28%
≥ 40	8219	18.47%	1174302	71.63%
≥ 80	4189	9.42%	951291	58.03%
≥ 160	2045	4.6%	715219	42.32%
≥ 320	800	1.8%	450242	26.64%
≥ 640	163	0.37%	165306	9.78%
≥ 1280	20	0.04%	48248	2.86%
≥ 2560	7	0.01%	26641	1.58%
≥ 5120	1	<0.01%	9017	0.53%

Figure 9.4: Table of cluster size distributions

Revealing the slowest Scanners The rate at which scanners operate is determined by the number of packets received within a specified time window. If scanners deliberately adjust their scan rate to avoid detection then we can expect them to be configured to never cross a certain output threshold. A scanner's scan rate may vary, but it should never exceed its configured threshold which is why measuring the maximum scan rate is of particular interest. The availability of timestamped packets from all the scanners present in the result allows us to move a sliding window along the entire observation period and count the maximum number of packets at any give moment during a scanner's operation. However, determining the window size is not trivial. Scanners operate in short bursts of activity of varying duration. Setting a window size that is too long will make scanners appear slower than they actually because inactive periods are included in calculating the average scan rate. On the hand, very short window sizes will likely be able to overlap a fully active period. However, the ability to distinguish very slow scanners becomes impossible, because the lower limit for any scan rate is 1 packet per window size. Therefore, there is no single fixed window size that will provide a correct estimation in all cases. In our attempt to highlight the slowest scanners, the window size is set to one hour. For every scanner the maximum number of packets received within any one-hour interval has been recorded. Selecting a windows size of one hour was already conservative, because that is well below what any conventional threshold-based algorithm is able to detect. Still, we discovered 12277 scanners that never exceeded 1 packet per hour. This means that they potentially are far slower which the current window size is unable to capture.

Single origin Quick inspection of the source IP addresses revealed that the majority belonged in the same 42.42.0.0/16 subnet. 12073 out of the intial 12277 of slow scanners come from the same Autonomous System (AS). This simplifies validation as in addition to the abnormally low scan rate it is highly probable that all scanners from the same AS are all part of the same group. The corresponding AS number is 9644 and it is registered by SKTELECOM in South Korea, a wireless telecommunications operator with more than 50% local market share. The prefix of this AS is 42.32.0.0/12 which is 4 times larger than a /16 prefix.

Activity Cluster activity is plotted in increasing detail. Fig 9.5 contains the full view of the telescope during its one-year observation period. The vertical axis contains all the scanners from the cluster and the horizontal axis is the timeline from march 2018 till march 2019. We can reduce the observation period to only include the dense regions which can be seen in fig 9.6. Apart from some minor noise, the obvious bursts of activity are concentrated in 4 periods: july 29th, august 21st till august 29th, september 14th till september 20th and finally october 31st till november 3th. The highest participation of scanners occurs in august. To better quantify the number of active scanners on any given day, the vertical axis has been replaced with scanner count. Figure 9.7 again shows four clear periods of activity where the number of participating scanners spikes with the most prominent one happening in august. Leveraging the summaries in PCAP allows us to zoom in the august campaign to get a more detailed perspective. Figure 9.8 reveals an interesting pattern of how the cluster operates during its most active period. Some form of scheduling can be observed that leads to oscillating cluster activity reaching its peak once a day.

Target In terms of target ports the entire cluster solely scans port 5555. This port is used by the Android Debug Bridge which is a feature that is usually turned off by default. If

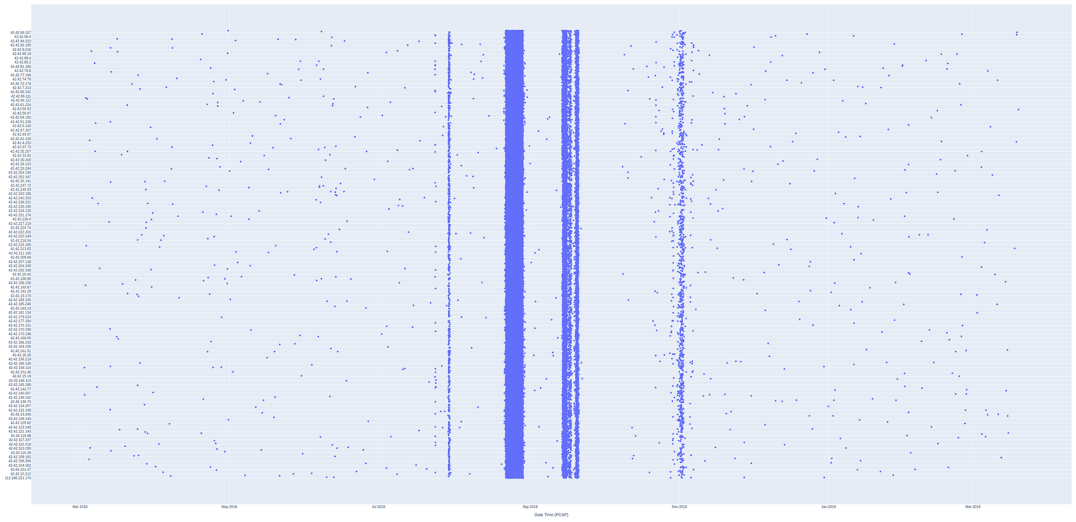

Figure 9.5: Scatter plot of individual slow Korean scanners presence from march 2018 till march 2019

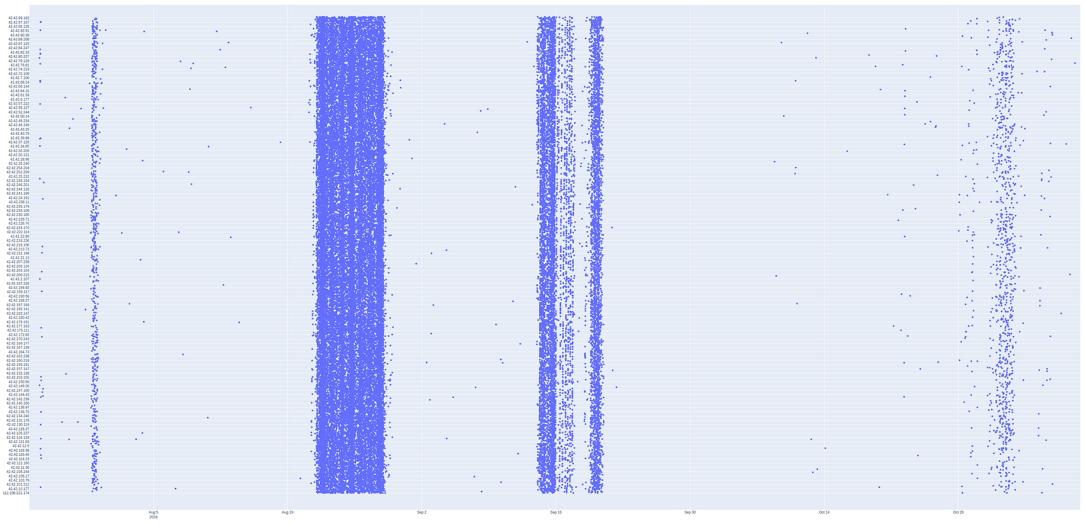

Figure 9.6: Scatter plot of individual slow Korean scanners presence from august 2018 till november 2018

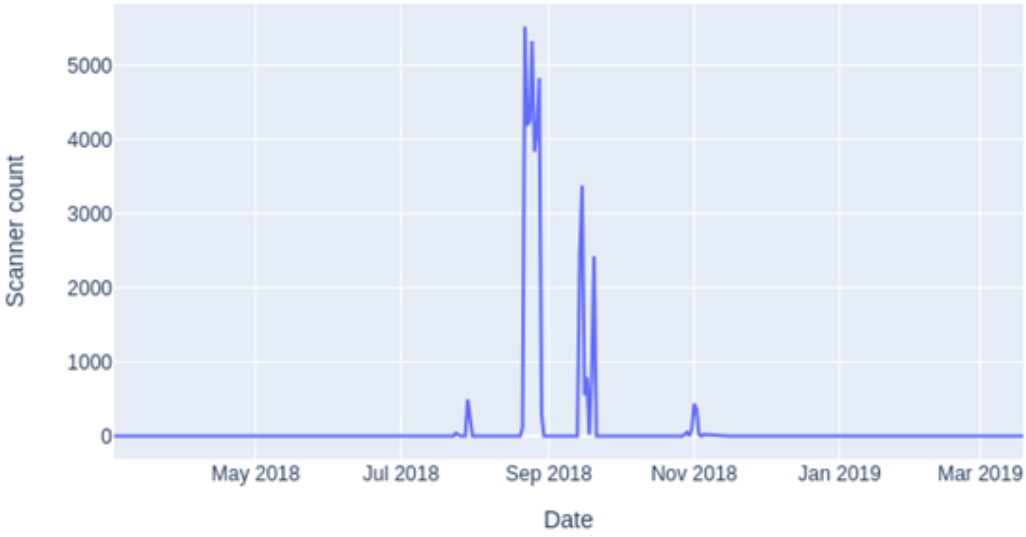

Figure 9.7: Counting scanner participation during the full observation period from slow

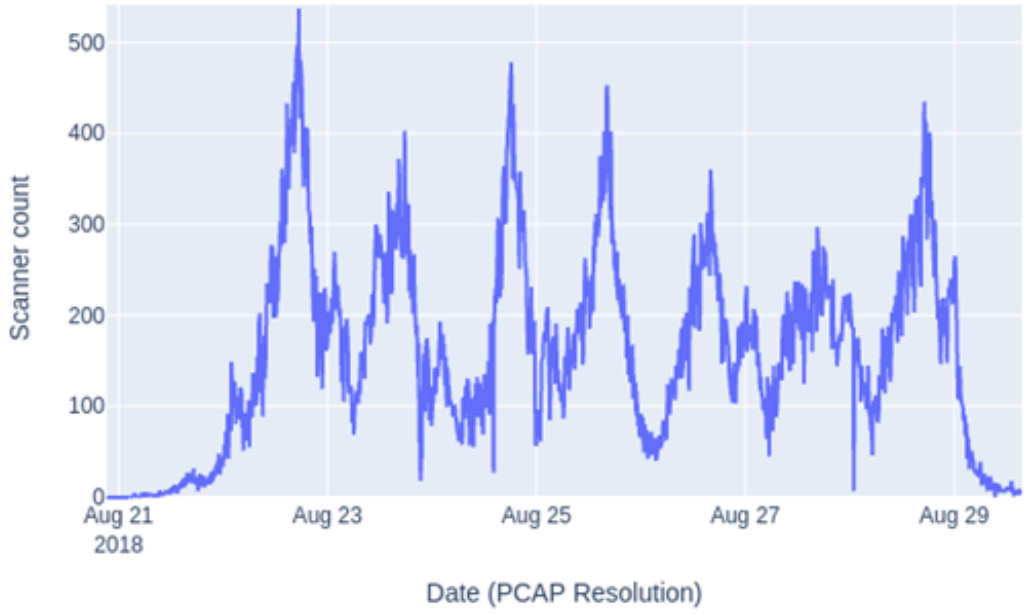

Figure 9.8: Counting scanner participation from august 21st 2018 till august 29th 2018 from slow korean scanners in PCAP resolution

however the user decides to allow external control over the device which is common for jailbraking or application testing then port is opened. Devices capable of running ADB can range from smartphones, mediaplayer, TVs and more. Additionally, sometimes and Chinese products in particular are shipped with ADB turned on. A port that is heavily targeted in internet-wide scans often indicates an exploitable vulnerability of the service. In this case however, whenever ADB is remotely reachable from the internet is an immense security risk by itself. Any malicious who is able to connect to an ADB enabled devices will have full control. The service is working as intended, but should always be aware of the security risk when turning ADB on and at the very least only make it only locally accessible. We suspect only a small percentage of all Android devices have ADB facing the internet, but the sheer number of devices still make it a worthwhile target. Apart from the port we can also look at the destination IP addresses and observe any patterns in how the cluster traverses the addresses of the telescope. The same heatmap as in Fig 6.1 and 6.2 got slowly populated over time in what appears like randomly generated target IP addresses. This is in line of our expectation how most scanners traverse the IPv4 space.

Scan rate The scan rate of individual scanners and the cluster as a whole is the reason for performing this case study. This cluster stood out, because the scanners never exceeded 1 packet per hour during the entire observation period. Even more interesting, the majority of them originate from a common AS. A window size of one hour was insufficient to accurately measure the true scan rate. Therefore, another approach was taken to estimate the scan rate specifically for the analysis of this cluster. Instead of running a sliding window, we can calculate the expected time between consecutive packets from the same scanner. This is also known as the interarrival time of packets. It is still important not to include any large periods of inactivity in the measurement. For this reason, currently the focus lies on the most active period from august 21st till august 29th. Figure 9.9 shows the expected interarrival times, due to the very slow scan rate it is measured in hours. Some scanners are slower than others and it is not uncommon to only see a single packet per day. And in extreme case down to one packet every 6 days. To call these scanners slow would be a huge understatement. However, the telescope is not able to observe the entire internet thus the perceived scan rate which has been estimated needs extrapolation. Previously the true telescope resolution was estimated at approximately 52000 IP addresses. Excluding reserved special IPv4 address blocks, the total number of IPv4 address are larger by around magnitude 77100. So for every packet that reaches the telescope, another 77100 is expected to have been sent by the scanners. This seems like a lot, but considering that for many of the scanners in this case study we witnessed less than 1 packet per day. After extrapolation that amounts to less than 1 packet per minute for the true scan rate and in some cases 1 packet every 5 minute. This is by a large distance the slowest detected distributed group that has been encountered.

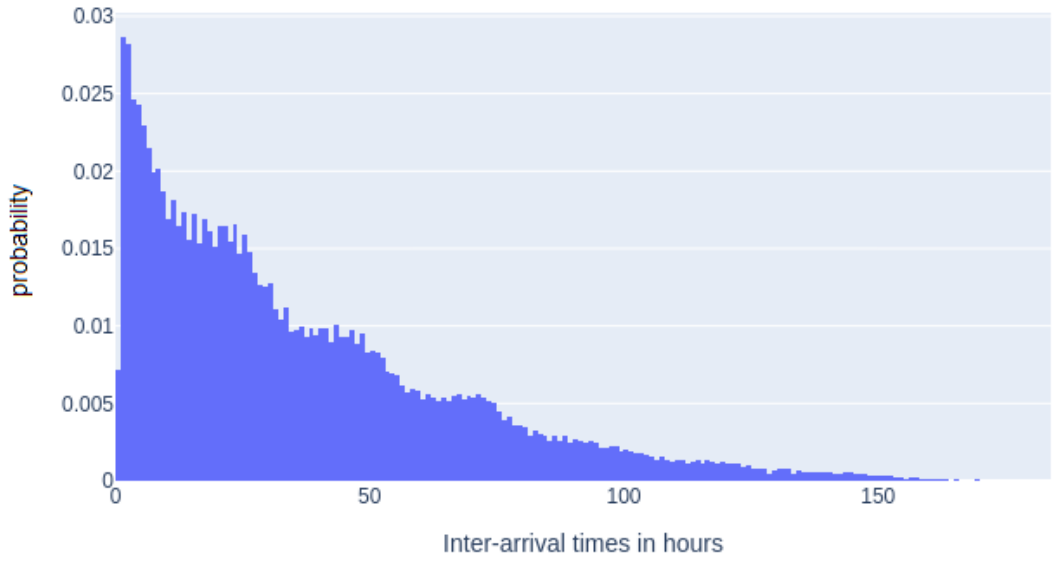

Figure 9.9: Histogram of packet interarrival times and their probabilities from august 21 till august 30

10 Evaluation

Detection challenges One of the biggest challenges in distributed scan detection research is validation of the results. In the absence of ground truth, in most cases it is impossible to know for certain whether a group of scanners are collaborating. The issue lies in the limited perspective of the defender. A scanner being part of a distributed scheme is not an attribute that is announced by the attacker from the stream of packets arriving at the destination. Therefore, defender's will have to resort to a best-effort attempt into establishing such a relationship between scanners. However, A moderately sized network could receive malicious probes from a million unique scanners each day. To distinguish multiple separate clusters from the noise is a daunting task if not infeasible. Despite that, some research has been performed on this topic and even reported having great results. However, much improvement can still be made in particular to the verification of the developed detection methodology which ultimately provides the necessary credibility to the research.

Cluster validation in related work A common approach that has been taken to verify the chosen methodology is through testing the system with synthetic data. The idea here is to generate traffic resembling that of distributed scanners according to a mathematical model, and mixing it with another live dataset. However, this model is a simplification of the entire class of distributed scanners. It only produces samples which adhere to the assumptions made about the real entity. Since the detection method is designed to be effective at recognising a specific type of pattern, it wouldn't be surprising to see a high detection accuracy. What such tests do achieve is showing that the algorithm is working as intended, but the resulting clusters are left unvalidated. This raises the question whether the assumptions made actually apply to distributed scanners in live data. In **??** the authors performed confirmation through manual inspection, but they did not elaborate on the criteria used to judge correct detection of a potential cluster. This leaves the reader unable to assess the quality of their work. Furthermore, such a labor intensive approach is only feasible in small-scale experiments.

Evaluation approach In order to address the concerns in related work, this work aims to extensively evaluate the proposed method through validation of detected clusters. The dataset gathered from the network telescope provides a unique opportunity to view the scanners from different perspectives which can contribute to increased confidence in the results. The motivation for this approach is to independently validate detected clusters through alternative characteristics inherent to typical distributed groups. The goal of cluster validation is to establish certainty beyond reasonable doubt. If the majority of the clusters can be confirmed then the methodology as proposed in this work is considered effective. However, satisfying said requirement is challenging and that may have partly been the reason why there is limited research done in this area despite the threat of distributed scanners being as old as the internet and potentially even more prevalent in the current day. Additionally, validation can

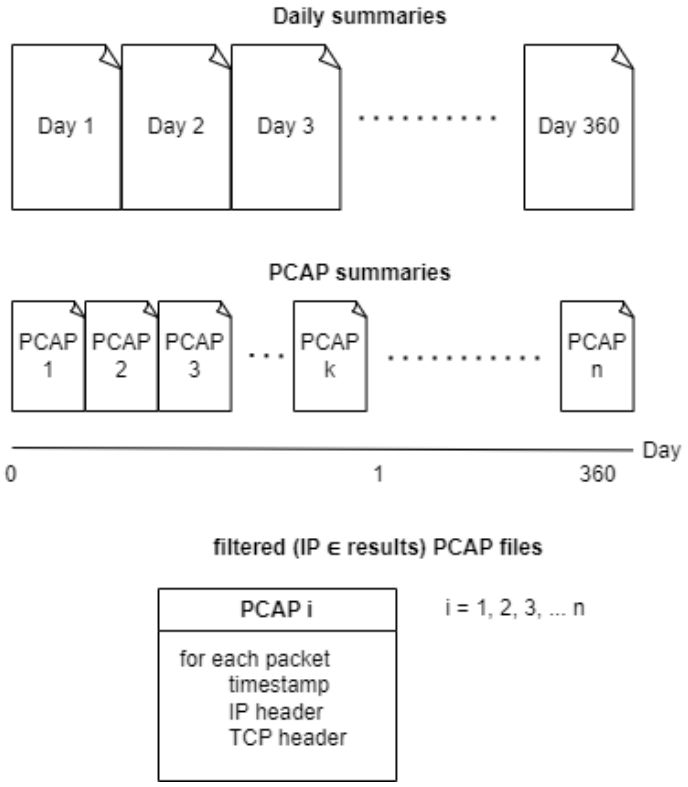

Figure 10.1: Dataset in varying levels of granularity

also serve to eliminate noise from the results in favor of a high true-positive rate at the risk of excluding good clusters.

Data perspectives Fig 10.1 provides a brief overview of the available perspectives that have been created to analyse the scanner's in varying resolutions. At the most course level there are 360 daily summaries where each contain all the scanners that have been observed including the aggregated number of packets sent to each distinct port number. This allows quick determination of a scanner's activity on a daily level as been utilised in the detection method. Additionally, the same type of summaries are also present in PCAP resolution. However, these do not span time windows of fixed length since they follow the original raw recording files. What they do offer is more detailed analysis into the activity patterns that take place within a specific day by splitting it into approximately 100 segments. To complement analysis at the finest level, a copy was made of every PCAP file but only including packets that belong to scanners present in the detection results. Valuable information such as exact timestamp of packet arrival and TCP/IP protocol header values are preserved. Keeping the available data perspectives in mind, several patterns that indicate a coordinated relationship between distributed scanners can be checked. The presence of any one of these patterns provide sufficient evidence for accurate detection.

Validation techniques First, the current methodology is evaluated from a statistical point of view. Scanners are assumed to have a similar strategy during the observation period due to economies of scale. A low-false positive rate is achieved, because the probability of unrelated scanners having matching behavior is suspected to be unlikely for a moderately long observation period. In the next subsection the aim is to better support this hypothesis by attempting to quantify occurrences of collisions. For this purpose, scanners are modelled as a Markov chain in order to assess the uniqueness of each scanner's observed strategy. Secondly, the cluster's response to their C&C is exposed. Every scanners from a particular cluster will at one point relatively simultaneous receive a command to deviate from the current operation. Notably during a transition from one state to another as defined in the Markov chain. For example, the moment a cluster is seen active after a period of inactivity. The degree of synchronisation is determined by measuring the duration in which every scanner in the group has sent its first probe. High and/or consistent synchronisation throughout multiple operational transitions validates the cluster. Third, cluster members are check whether their IP addresses lie in the same subnet. Scanners from the same /16 or even /24 subnet can be considered to belong to the same organisation due to the way IP addresses are issued by the Internet Corporation for Assigned Names and Numbers (ICANN).

10.1 Modelling behavioral complexity as a Markov Chain

Entropy The detection method assigned scanners with the same activity sequence to the same cluster. The probability in which two unrelated scanners will have a matching sequence was instinctively assumed low due to the many possible combinations that could occur for a large observation period. However, not sufficient evidence has yet been provided to support this claim. A scanner's activity is a sequence 360 daily observation of which destination ports have been targeted. Each day a scanner can be completely inactive or have sent at least one probe to any of 2^{16} port numbers. So in theory, if any of the port numbers can either receive more than 1 probe or none at all then are a total of $2^{2^{16}}$ different combinations of port sets possible. In practise however, certain ports are more often targeted than others and the number of unique ports are limited. Therefore, the effective entropy is greatly reduced from its theoretical maximum which can only follow from a uniform distribution over the values. Some activity sequences are definitely less frequently occurring than others which an analyst will intuitively be able to confirm that scanners with that kind of behavior are likely to be cooperating. To help avoid manual confirmation of each cluster and to better quantify uniqueness for the sake of correctness, consistency and transparency, a markov chain is created from the available data.

Building an initial Markov Chain The idea of modelling scanner behavior using a Markov Chain follows from the observation that how a scanner acts on the next day is largely determined on its action on the current day. In general, a scanner is much more likely to continue scanning once it has started. Conversely, after being switched off there is an even smaller chance of expecting it to come online again. These rules for a behavioral model are able to describe the majority of scanners which are seen to be active in bursts of several days before becoming dormant again. Initially we can define at least two states, a simplified scanner is either inactive or active. The probability for transitioning between any of the states is probabilistic rather than deterministic and can be calculated from the available data. For this

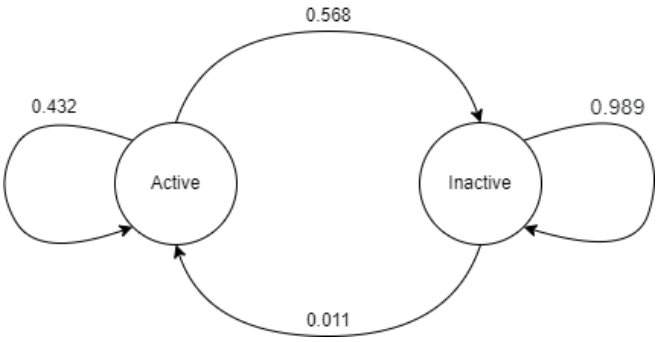

Figure 10.2: Markov chain for a simplified scanner to describe daily activity

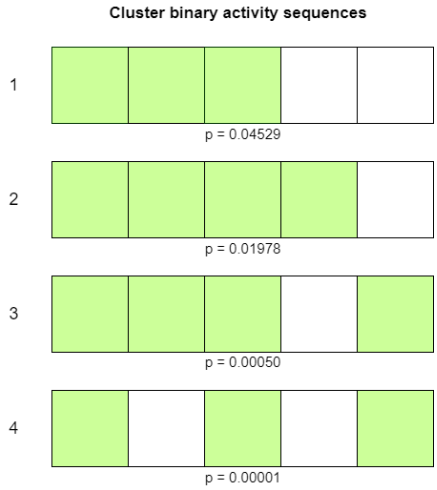

Figure 10.3: Decreasing probability of cluster activity sequences calculated with Markov chain

reason, a Markov chain is well suited since its purpose is to describe a sequence of events where the probably for each event only depends on the current state.

In Fig 10.2 the two states and corresponding transition probabilities are shown. Probabilities are calculated from a large sample size of scanner activity sequences to account for variance. With just two states, the behavior is very easy to understand. Almost halve of the time, an active scanner stays active. However, once it does transition to an inactive state there is just a 1 percent chance for it to resume scanning again. Such a model is helpful in determining what type of observed sequence is a rarity. And if it is rare, then it is unlikely to be a coincidence when multiple scanners share this behavior. Therefore, there is enough supporting evidence to confirm a coordinated relationship. Fig 10.3 provides several example cluster sequences with decreasing probability that have been calculated with the markov chain model.

Accounting for destination ports There are approximately 50000 ports that are used by the TCP protocol for identifying and application or service. Certain ports are more common,

because these have become the default port for a popular service like port 22 for SSH. Malicious actors show more interest in services with a large user base due to the higher chance of finding a potential victim. This is also reflected in the dataset where the most highly scanned ports are associated with HTTP, SSH, FTP, Telnet, SMTP, MySQL and so on. While it is possible for the user to deviate from the default port and therefore achieve security by obscurity, not all of them do for reasons such as convenience in favor of the connecting clients. The discrepancy between targeted port numbers might be leveraged to further increase confidence in clusters that have scanned ports that are relatively uncommon. However, the frequency in which a port is scanned is also heavily influenced by the recent discovery of a new vulnerability. Spikes of specific port activity can be observed in internet-wide scans for services during the period a critical vulnerability remains unpatched and is being actively exploited. The implication of this time-sensitive phenomena leads to reduced effectiveness of generalised port distribution statistics for the purpose of evaluation. The distributions are calculated over a large time period. A relatively rare port number can receive a sudden increase in interest by a large number of unrelated scanners even though the corresponding service is ignored most of the time. For this reason, the choice was made to not make a distinction between common and uncommon ports.
One observation made regarding the targeted ports is that scanner typically do not deviate from their initial configured behavior once started. It would not be surprising to witness an adversary lose interest in a particular service and goes on to direct all scanners within the cluster to another set of ports. However, in practise this does not happen frequently as is illustrated in table 10.4. The number of transitions have been calculated from all the scanners in detected clusters. A port set transition is defined as the occurrence of scanning a different set of ports on the subsequent active day. Two subsequent sets are equal if the sets have the same cardinality and contain the same port numbers. The aim is to notice a change in behaviour in which all scanners within a cluster seem to follow. However, less than 6% of the scanners have one or more transitions in their targets. So relatively few scanners change targets, but those that do are very likely to be indeed part of the same cluster. Ideally this probabilistic behaviour would be incorporated in the same Markov chain which described the online/offline activity of 10.3. Unfortunately, it does not make sense to translate a property such as dissimilarity of targets to independent states in the Markov chain. Regardless of the target, the scanner is in principle performing the same type of function. Both the Markov chain and the port set distribution table can be used as seperate guidelines to judge the level of complexity in the adversaries long-term strategy.

Determining the minimum threshold One could set a minimum threshold and only retrieve those clusters that exhibit patterns with a low probability of occurring. These tend to be more complex behaviours which imply a coordinated effort from a single entity in order to have all participating scanners have the same target on each day of the year. It is up to the analyst to determine the threshold depending on the desired false-positive rate. A high threshold will more likely retain only true distributed scanners, but with the possibility of being too strict and therefore unintentionally exclude certain groups. From both the Markov chain in 10.3 and port set distribution in table **??** it is clear that there is a huge drop in probability in two cases. The first case is an active scanner which stops scanning, after which it resumes again between one or multiple days of inactivity. And secondly, a scanner that has at least one transition from one port set to another. One suggestion would be to demand a certain number of days a cluster must be active and set the threshold accordingly for which the distribution is show in Fig 10.5. A steep decline in number of applicable clusters can be observed from 8 to 9 days. While the probability still decreases exponentially

Port set transitions	Scanners %	probability
0	19042061	0.937
1	1201696	0.059
2	65865	0.003
3	8706	0.001
4	1950	0.0002
5	400	0.0001
6	83	<0.0001
7	27	<0.0001

Figure 10.4: Table of port set transition probability distribution

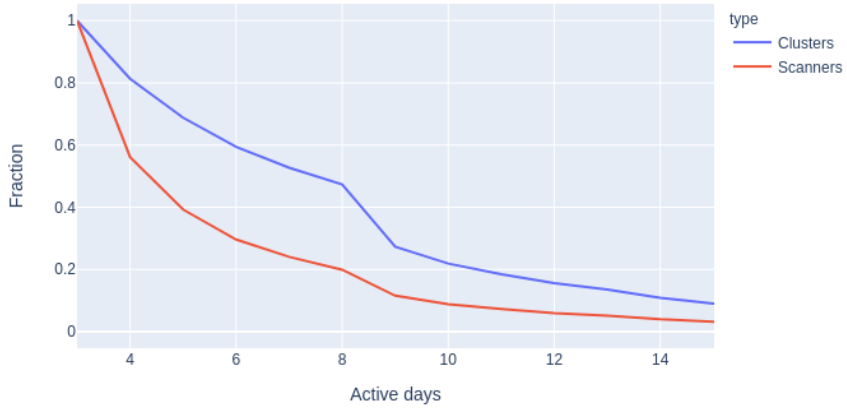

Figure 10.5: Effect of probability threshold expressed in number of consecutive active days on fraction of validated clusters

with each additional active day, the threshold is easier to gradually adjust than the steep decline associated with state transitions.

10.1.1 Synchronisation during behavioural changes

Cluster manual command and control Scanners are software programs which are maintained and configured by the controlling entity. Basic parameters such as scan rate, range of destination IP addresses and TCP ports are determined during initialisation. The degree of autonomy can vary, but typically a scanner continues with its pre-configured settings until an explicit command is issued to deviate from the current operation. A change could be the stop scanning or to switch to a different target. Due to the scale of large clusters, one would expect that controlling multiple scanners is done by issuing commands which are applicable to the cluster as a whole. The speed in which commands can be communicated to all

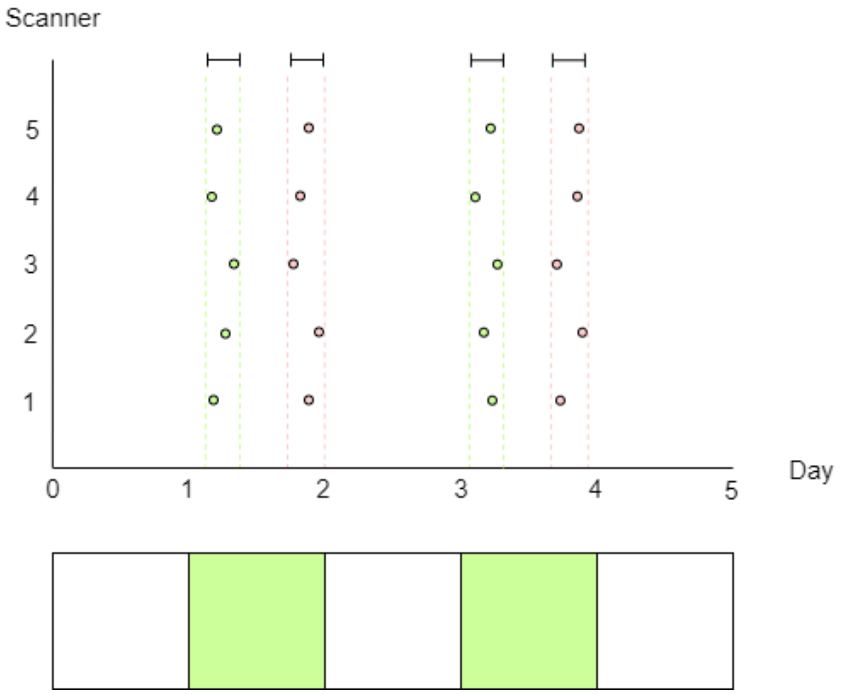

Figure 10.6: Arrival times of packets near a state transition of synchronised scanners

nodes heavily depends on the chosen topology. Botnets for example, which can be utilised as a scanning cluster, have been encountered with a centralised, hierarchical or peer-to-peer architecture. Each come with their own characteristics such as speed, resiliency, scalability and complexity. The most important part is that the botmaster is able to maintain a communication channel between the command and control (C&C) and the bots. This allows the botmaster to update malicious software or perform large scale disruptive attacks. In the context of distributed scanners, it is suspected that individual scanners will respond within a reasonable time frame as the command propagates throughout the entire cluster. Therefore, one way to confirm a potential cluster is to expose their degree of synchronisation. In other words, scanners are more likely to be cooperating when the entire cluster switches from one state to the next within a relatively short time window. Such a relationship has already been established on a long-term scale due to the nature of the detection method. The detected clusters are confirmed to target the same ports on a daily basis. This validation method takes this approach one step further by zooming in on the moments in which the cluster as a whole transitions between the active and inactive state.

Visualising transitions Considering a simple cluster of just 5 members in the results. That implies that the detection method has confirmed scanners having a matching activity pattern throughout the entire observation period of one year on a daily resolution. For illustration purposes, the observation period is reduced to 5 days. The cluster is active on 2 out of 5 days, with a day of inactivity in between as depicited in 10.6. So on two occasions, the cluster transitions from the inactive to active state, subsequently referred to as a start transition. Conversely, a transition from active to inactive is defined as a pause which in this

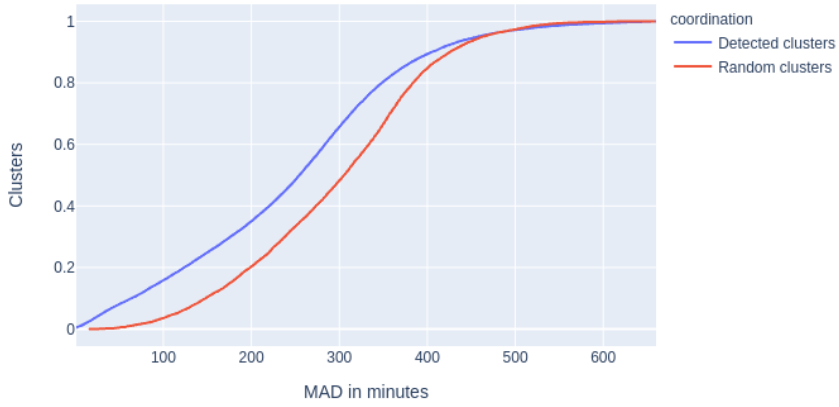

Figure 10.7: CDF of cluster start transition packets dispersion as MAD

example also happens twice. In order to validate the cluster, additional evidence can be provided by inspecting the similarity in arrival times of packets during those start and pause operations. Since it is already known in which days the transitions occur, the timestamp of every scanner's first appearance within that particular day is recorded in the case of a start transition. These packets are highlighted as a green dot. Red dots indicate the last packets before at least a full day of inactivity. The deviation between the packet timestamps shows the reactivity of the cluster as a whole. The frequency and amount of packets received in between the first and last packets are of no interest for this particular analysis and thus ignored. Only those packets on the borderline of a behavioural changes might indicate collaboration on the finest scale. The hypothesis is that distributed scanners tend to generally be more synchronised during state transitions compared to unrelated scanners in a false-positive cluster. Additionally, they'll have consistent response times throughout the multiple transitions which indicates a common topology for the propagation speed of C&C commands.

Measure degree of synchronisation In order to determine the degree of synchronisation it necessary to quantify dispersion of incoming packets during a state transition. Both the Standard Deviation (STD) and the Median Absolute Deviation are commonly used in statistics to measure variability of univariate dataset. The MAD seems the more obvious choice due to its robustness to outliers. Outliers would have been more heavily weighted in the calculation of STD, because the distances from the mean are squared. Especially larger scan clusters might contain a few false-positives, but that result should not be disregarded if the majority of scanners are actually part of the same scheme. Additionally, the STD assumes a normal distribution of the data samples which might not be applicable. Calculation of the MAD is achieved by taking the median of the absolute deviations from the data's median. From a set of timestamps $\{1532000001, 1532000002, 1532000003, 1532000005, 1532000007\}$, the median value is 1532000003. Therefore, the absolute deviation of the samples from this median is $\{2, 1, 0, 2, 4\}$. Taking once again the median will lead to the MAD which in this case is 2. An important discussion is what to consider as a low MAD value which provides sufficient confidence for a correctly classified cluster. This discussion will continue shortly after reviewing the MAD of all detected clusters.

Calculation of MAD MAD calculation is fast and straightforward, but retrieving the packet timestamps is very resource intensive. The daily summaries can point to the exact day in which a transition occurs and the corresponding pcap summaries can instantly confirm the presence of a scanner. In the last step the filtered pcap file is being iterated linearly packet by packet to find the first or last occurrence of a scanner depending on the transition type. Performing this search operation for the many clusters in the results was computationally feasible due to hierarchical approach of being able to quickly locate the exact pcap capture file. For clusters with multiple instances of a start transition, the lowest MAD value is currently used to provide a general overview of synchronised transitions in Fig 10.7. Stop transitions have been excluded in this graph to speed up computation and because they typically have higher MAD values for which evidence will be provided shortly. The blue line corresponds to the MAD of clusters in the results expressed in minutes instead of seconds in order to avoid large numbers. Deviation in minute precision is sufficient to confirm coordinated behaviour since a day in which any transition happens consists of 1440 minutes. Therefore, the maximum MAD is 720 in the worst case where the values are furthest apart from eachother. Clearly not all clusters exhibit synchronised transitions, but a significant portion of the result does have a low MAD. Almost 10 percent of the clusters have starting packets that deviate within 60 minutes on a transition.

Low MAD as an indicator of collaboration Close to 5 percent of the clusters have scanners that collectively transition with a 30 minute MAD. In order to put the calculated MAD values into perspective, these are compared to artificially generated clusters which should resemble unrelated scanners as a false-positive result. After a full day of inactivity, the start transition of a single scanner can take place at any moment within a particular day. Therefore, a group of multiple scanners having absolutely no coordinated relationship among eachother is expected to show a uniform distribution of their first arriving packet. The MAD will be much higher for these random clusters that have timestamps evenly spread throughout the day. Therefore, this extreme worst case scenario of having only false clusters can be used to highlight the significance of the observed synchronisation characteristics during a transition. For this purpose, 40000 artificial clusters, approximately the same number has the result size, have been generated. The size of clusters follow the same distribution as the real detected clusters recorded in Table 9.4. The scanners have only one active day with one probe inserted at random. The MAD for every random cluster is calculated in the same way as the detected clusters and plotted in the same figure 10.7. From this figure a significantly lower MAD can be observed for the group of detected clusters which are suspected to be controlled by a single entity. Compared to the random clusters, the difference is overall not enormous but obvious nonetheless. Only when taking a closer inspection on the lower spectrum less than 60 minutes do we see a larger contrast. Fig 10.8 is an enlarged view with a maximum MAD of 60 minutes. Such a low degree of dispersion is very unlikely given the uniform distribution of unrelated scanners. There are 10 times as many instances from the detected clusters at this level which does provide compelling evidence for synchronised behaviour. The difference in likelihood of both sets only increases as the MAD decreases. Therefore, any group of scanners with a high degree of synchronisation during their start transitions is more likely to be collaborating. This degree of synchronisation can be inferred from a low MAD, the definition of low and the corresponding threshold value is determined by making a trade-off between false-positive and false-negative rate.

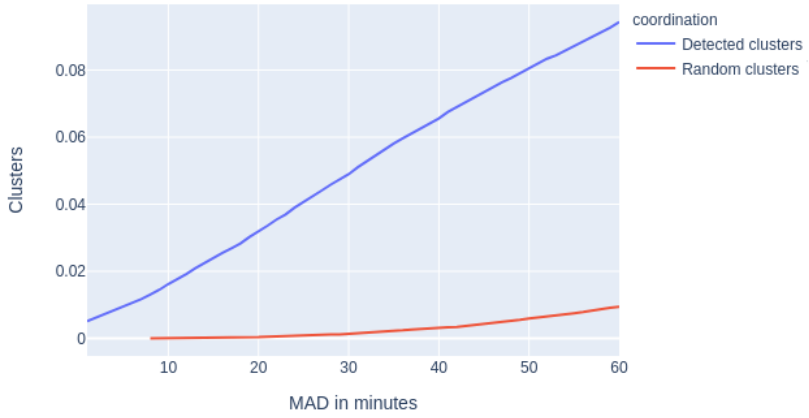

Figure 10.8: CDF of cluster start transition packets MAD max 60 minutes

Synchronisation during resume and terminate state Preliminary analysis on random samples revealed that scanners responded within a smaller time window during a resume transition than on a pause transition. This was the reason why only the former was used to infer the degree of synchronisation. For some reason, the last packet of every scanner in the cluster before going into inacitivty is more spreadout throughout the day. It is worth investigating if this also applies to a larger sample size of tightly coordinated scanners. Therefore, all clusters for which a very low MAD value was observed during their start transition were used in a more extensive comparison. In total there are 2160 cluster with a MAD lower than 30 minutes which can be considered strong indication for collaboration. The expectation was to observe similar and consistent deviations in packet arrival times during transitional phases with such a strictly controlled group of scanners. Fig 10.9 illustrates both distributions as a CDF. There is clearly much more deviation in the time required for a cluster's scanners to collectively stop their actions. The difference with start transitions is evident which strengthens the initial suspicion that clusters are more synchronised when switching from the inactive to active state. Therefore, opting to disregard pause transitions is understandable considering the objective is to validate clusters showing coordinated behaviour through alternative statistics.

10.2 Scanner origin

Autonomous systems and IP address allocation Scanners are associated with their source IP address which is a value of 32 bits in the IPv4 protocol. An IP address can either be used as the source or destination for effectively routing packets in two-way communications across the internet. In order to understand the origin of an observed scanner, a brief introduction is provided about addressing in the internet. The internet is actually not just one network, but a collection of many large subnetworks called autonomous systems (AS), each controlled by a single administrative entity. Organisations that heavily depend on networking capabilities or Internet Service Providers (ISP) enabling internet access to businesses and consumers typically operate one or multiple autonomous systems. The relationship

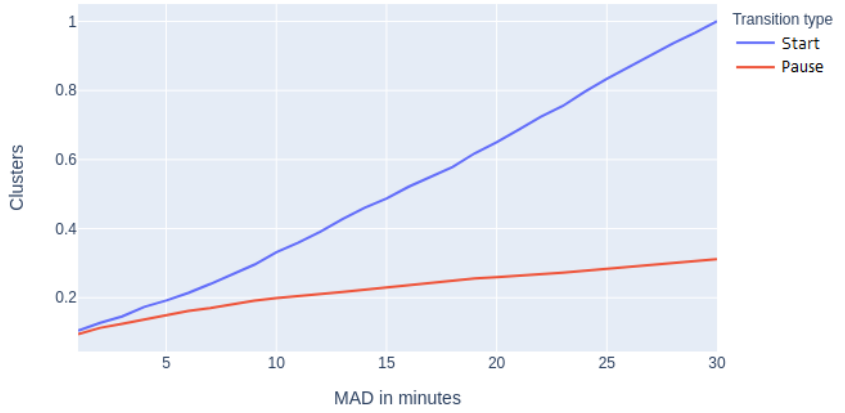

Figure 10.9: Comparison of start and pause transition synchronisation of 2160 clusters with a start transition lower than 30-minute MAD

and connectivity between the many autonomous systems is what defines the internet. Border Gateway Protocol (BGP) routing finds the most efficient path from the available peers of one AS to the next. For this purpose, each AS is assigned a unique Autonomous System Number (ASN) and a collection of routing prefixes. ICDR notation is used to specify the size of the routing prefixes in human readable form such as 123.123.123.1/24. The prefix denotes the network part of the address and the suffix behind the slash symbol specifies the number of significant bits. Prefixes can regarded as blocks of usable IP address which are assigned and managed on an international level by the Internet Assigned Numbers Authority (IANA). In turn it is the responsibility of five regional Regional Internet Registries (RIR) to assign both the ASN and IP prefixes to local organisations. ASN initially were 16-bit numbers wtih a maximum of 65536 assignment, but the rapid growth of the internet necessitated expansion to 32-bit before all ASN were depleted. IPv4 faces a similar problem with its addressing which is why there is a strong push towards IPv6 allowing plenty of addresses for the forseeable future. So scarcity, cooperative nature of the internet and efficient routing mandates the careful management of IPv4 addresses and ASN. Thus, from the IP address of the scanner it can be quickly determined under which prefix and AS it falls to reveal information about its origin within the internet.

Validation based on origin One of the earliest methods of determining whether a group of scanners belong to the same entity is by their IP address proximity. Yegneswaran et al. [2003] would even go as far as to define distributed scanners as a group of at least five scanners in the same /24 subnet active in the same hour. Since the IPv4 space is partitioned into many smaller blocks, it seems understandable to imply a relationship between scanners within the same subnet, because the scanners have their origin in the same part of the internet. More so if the subnet is relatively small as is the case with a /24 suffix of only 255 address space. Without extensive data analysis, there is visible evidence of coordination from just the source IP address of individual scanners operating within a specified time period. However, this type of indicator becomes less convincing for subnets with a larger prefix length such as a /16 subnet. Furthermore, from the IP address alone it is not possible without additional queries to know whether it is an actual subnet operated by a single administrative entity

and acknowledged by the respective RIR. Autonomous systems can have prefixes of varying length, but its safe to assume that they are at least the size of /24 subnet. To complicate the matter, an AS can have a collection of prefixes. Seemingly unrelated IP addresses in human readable form might still fall under the same administration. Regardless of the subnet size, the important potentially common characteristic is whether the source IP addresses in scan clusters originate from the same administrative domain. It is possible for a cluster to utilise the entire range of IP addresses belonging to the corresponding AS during network reconnaissance. In such a case, the defender might imply the involvement of the AS operator in the malicious activity. Establishing such a relationship is difficult to prove and outside of the scope of this paper. Similar to an ISP who is not always held accountable for the digital actions of its end users. All things considered, IP address origins in terms of their respective network portion associated with an AS provides valuable information which can be used in conjunction with this work detection method for the purpose of cluster validation. A common scanner origin is only applicable to certain type of clusters where likely the hosts are under direct control and ownership of the adversary. This as supposed to a collection of infected hosts organised as a botnet which may involve all parts of the internet.

IP to ASN mapping Team Cymru is a security orientated organisation that provides a free service to map IP addresses to the corresponding ASN. Other AS related information such as country code, BGP prefix, allocation date, regional registry can also be retrieved. For the purpose of cluster validation through origin detection we are only interested in the ASN. A single query to their WHOIS server allowed the mapping of all 1.6 million scanners involved in the detected clusters to their respective ASN in bulk. The next step is to iterate over the cluster results and discover how many of them actually originate from the same AS. A minimum similarity threshold of 95 percent of the scanners is set to accommodate potential outliers for very large clusters. It turns out 4661 out of the 44491 total detected clusters can be traced back to a single AS which is a little over 10 percent. More interestingly, there are only 113 unique ASN which implies that the detection method has fragmented larger clusters into smaller ones where each has a slight variation in its activity and targets. Therefore, while the total number of involved scanners might be correct there may be an overestimation in the number of clusters due to fragmentation. Depending on the chosen definition it is arguable whether clusters with an entirely strategy different should be viewed as distinct clusters.

10.3 Validation of results

This work has discussed three methods for validating clusters. The basis for each validation method is an assumption made about observable patterns from the perspective of the network telescope. Such patterns are an expression of the implement strategy of the adversary controlling the clusters. Encountered strategies can vary considerably in both temporal and spatial complexity with many adjustable variables. The purpose of every scanner is to perform reconnaissance on the target network of interest, but the actual implementation and execution is entirely determined by the initiator. Therefore, there is no single data pattern which can capture the entire class of distributed scanners. Nevertheless, an attempt is made to validate clusters exploiting three different characteristics that intuitively should apply for a reasonable number of clusters due to similarity in certain aspects of their operation.

Markov validation From Fig 10.5 a sensible choice would be to set the threshold at a probability equal to 7 or 8 consecutive number of days. From 9 days there is steep decline in number of applicable clusters. So the aim is significantly reduce the likelihood of false-positive clusters without eliminating too many results. Therefore, the markov validation technique is applied with a probability equal to 8 consecutive days. This however does not mean that every cluster needs to be active for at least 8 days in order to pass this validation. The reason being that the markov chain describes a much lower probability for clusters that show at least a full day of inactivity between active sessions. Thus a clusters with only 2 active days separated by a one day pause would still pass the validation with the current threshold. For simplicity and time constraints the decision was made to currently omit inclusion of port set transitions in the calculation of probabilities. Application of this validation techniques revealed that a total of 21049 cluster containing 325869 scanners met the criteria which is 47 percent of the clusters.

Synchronised Transition validation In situations where scanners within a clusters are tightly synchronised we expect a small deviation in arrival times during a transition from the inactive to active state. Referring to Fig 10.7 a threshold of 240 minutes would retain almost 50 percent of the clusters. Depending on the needs of subsequent analysis, reduction of this threshold is beneficial to the reduction of false-positive results. This thesis will opt for more strict validation, because the large result pool allows for more aggressive filtering while retaining sufficient number of clusters to analyse. Therefore, a 60-minute MAD threshold was set which typically means that a cluster completes a transition with a two-hour window. For 4163 clusters involving 129946 scanners such synchronised transition were observed, less than 10 percent of the total clusters.

Origin Validation Cluster validation based on origin compared to the previous two techniques much more straightforward. Origin is defined as the IP prefix belonging to the autonomous system. The corresponding ASN can be looked up from the source IP address of a scanner. If the majority scanners from a cluster originate from the same AS then they are assumed to be collaborating. In this case at least 95 percent of the scanners must share the same origin to account for noise data points. This was the case for 4661 clusters, approximately 10 percent of the total clusters in the results.

Combined validation Every cluster in the results is subjected to all three validation methods. The outcome can be concisely summarised in 10.5 through definition of three sets, one for each method. Clusters that pass the validation method are member of the corresponding set. One additional set indicates the fraction of clusters left unvalidated through current means.

The markov chain is able to validate a large portion of the results. In addition, both synchronised transitions and the common origins method provide the necessary confidence to approximately 10 percent of the clusters. However, a cluster can pass more than one method which is why the union of these three sets will inform what portion of the results can safely be said that they are actually collaborating. 51.5 percent of the results can be validated by at least one of the described methods.

Set	Clusters %	Cluster count	Scanners
M (Markov Chain)	47.3	21049	325869
S (Synchronised Transitions)	9.4	4163	129946
O (Common Origin)	10.5	4661	70786
$M \cap S$	5.4	2388	56016
$M \cap O$	7.3	3233	36781
$S \cap O$	0.3	137	3271
$M \cap S \cap O$	1.4	606	9108
$M \cup S \cup O$	51.5	22903	412317
$Unvalidated$	48.5	21588	1226987

Figure 10.10: Sets of various validation methods

Unvalidated clusters 48.5 percent of the clusters in the unvalidated set are not provided with additional evidence. This neither confirms nor denies a collaborative relationship between the scanners, only that we can not confirm them through the limited three validation methods which do not cover all types of scanners. Relaxing the thresholds is a trade-off between the number of unvalidated clusters and the risk of false-positives. However, the detection methodology already yields only clusters for which the scanners exhibit some degree of similarity. Namely, their temporal and spatial activity matches exactly during a one-year observation period. There is a minimum of 3 active days with cluster sizes larger than 6. Thus there is already some statistical evidence by design. Depending on the requirements additional validation might not be necessary.

11 Discussion

From data analysis obtained from the network telescope we noticed similarities in long-term activity between scanners from the same group. A scanner's daily activity can be described by the set of ports it has contacted. In this manner, we created sequences of 360 days long as a simple representation of a scanner's operational behaviour over the course of one year. This eliminates much of the dependence on the original dataset which is far too large to work with, especially to answer simple queries such a whether scanner A is active in any day of the year. Much of the original data is redundant for long-term analysis as scanner typically scan the same target port many times within a few minutes, let alone a full day. Activity sequences served as a powerful fingerprint to identify unique clusters. We detected many clusters of which its scanners shared exactly the same behavioral pattern. Due to the high entropy of an activity sequence, and also by additional verification steps we can say with high confidence that many of the groups contain scanners that are actually under the control of a single entity. The resulting dataset can be shared with other researchers so they can study other properties of distributed scanners or evaluate new detection methods. It is the best alternative for a labeled dataset which currently does not exist. Techniques used in this thesis can also be applied on data from other network telescopes and/or different observation periods.

11.1 Limitations

Overall the results indicate a decent number of detected clusters of which many are successfully validated later on. We therefore are satisfied with that the initial hypothesis of similar activity patterns is applicable to many clusters present in the dataset. Moreover, this also provided confidence in the correct implementation of the algorithm. However, the developed detection methodology is not without limitations. A perfect solution would be able to detect all distributed scanners in the dataset with 100% accuracy, sadly that is not case. Therefore, a discussion is provided where the approach of scanner correlation is critically assessed. We highlight its shortcomings and associated impact on the obtained results.

11.1.1 Coverage

There is no standard way in how scanners operate since they are software implementations which satisfy the goals of the attacker. Open-source tools such as nmap or zmap can be more commonly in use due to their low-barrier for deployment. With no coding knowledge required, these tools provide moderately skilled cybercriminals with the ability of having a scanner running within a few minutes. As the sophistication level increases, so does the complexity of strategies and custom tools. The underlying host on which the scanners run may be self-owned or an infected device part of a botnet. All these factors contribute to a wide variety strategies for which no single observed pattern is applicable to the whole set of

distributed scanners. Despite the many clusters detected in the results, it is also important to discuss the method's weakness. Or in other words, how many clusters out there in the wild we have potentially missed. Once again there is no labeled dataset to provide a straightforward answer. Nevertheless, analysis of the correlation algorithm can tell what type of scanners will remain undetected. We approach this by considering the characteristics of certain classes of distributed scanners such as botnets and even deliberately trying to circumvent our own correlation algorithm by describing hypothetical scanners.

Detection circumvention The current methodology assumes an exact matching of activity sequences between scanners from the same cluster. Such a sequence is comprised of 360 daily consecutive observations. For each day a scanners is considered actively scanning a set of ports when at least one probe has been received for every unique port number in the set. In favor of a low false-positive rate the minimum number of active days is set at 3 with the minimum cluster size containing at least 6 scanners. So in order to evade detection, the attacker must distribute its scanners in such a way that less than 6 of its scanners can have a matching pattern. The simplest way to do that is by introducing small deviations in the targeted ports. Considering a cluster of size N. We can make each scanners appear slightly different by sending one additional probe to a unique port not present in the other $N - 1$ scanners which effectively misleads the detection algorithm. Another way to achieve obfuscation is to create variations in which days the scanners are active, but given the long daily intervals that requires more complex orchestration. In addition, scanners might operate on time constraints in order to quickly find the services of interest once a vulnerability has been disclosed to the public. In short, small variations in either active days or targeted ports over the entire sequence will prevent correlation to the same group. If 6 or more scanners from a larger cluster follow a particular variation then they will be treated as a separate cluster. This does not entirely evade detection as the subclusters are still included in the results, but the whole cluster will appear fragmented into smaller ones.

Botnets Botnets are a special type of distributed scanners in the sense that they are comprised of infected devices. The owners of these devices are unaware that these are being controlled by an external entity in the engagement of malicious activities. Botnets exploit a known vulnerability in order to amass as many zombie hosts as possible. Thus a botnet starts with several host and quickly grows overtime as it scans the internet for potential victims. Typically those infected also participate in reconnaissance to speed up the search. Therefore, the size of a botnet is very dynamic. It can grow very quickly, but also shrink when hosts become unreachable when turned off by their legitimate owner. The latter reason is also why botnet operators can't expect any of its zombie host to be available when requested, because those are not physically under his control. However, due to the sheer size of modern botnets ample firepower for performing scans or DDoS attacks remains. We highlight the dynamic nature of botnets, because these are more difficult to detect using the correlation algorithm in this thesis. This is due to the assumption of a coordinated relationship present during the entire observation period. Detection of the whole clusters only works for a static group of scanners, which zombie hosts in a botnet obviously are not. For a moderately sized botnet, subclusters are still detected if at least 6 scanners share an activity sequence. Thus the many zombie hosts still appear in the results but only as fragmented clusters since any of the subclusters will show similar but slightly deviating behaviour.

11.1.2 IP Churn

In research typically the source destination IP addresses are associated with a unique host. Thus its common to see IP addresses being used as a proxy metric to quantify the size of a botnet or group of scanners. This one-to-one relationship is a simplification which does not always hold true. IP addresses can be dynamically assigned by network operators to their clients upon request with no guarantee of maintaining the same address over time. Scarcity of IPv4 addresses forces network operators to maximise the utility of their available addresses. This means that through dynamic assignment a block of IP addresses can serve a greater number of hosts assuming they are not all active simultaneously. An IP addresses is leased for an unspecified amount time. After expiration of the lease the hosts receives a new address. This phenomena of hosts changing IP addresses is called IP churn and the rate of change varies from network to network. The implication of IP churn to this study is the potential overestimation of cluster sizes, because we identify scanners based on their IP address. Thus simply counting the number of distinct IP address in a cluster does not always produce an accurate estimate of size. In Griffioen and Doerr [2020b] the authors presented a novel method to quantify the prevalence of IP churn using large botnet traffic. They were able to measure the presence of IP churn for all Autonomous Systems (AS) in the dataset. IP churns varies between countries and even between netblocks within a single AS. Overall they observed an overestimation of 20% for the Mirai botnet. Additionally, The effect of both IP churn and Network Address Translation (NAT) were compared. The latter enables multiple host to operate behind a single IP address which leads to underestimation. They witness NATs having a larger effect on estimation than IP churns. Taking this into account, IP churns also hinder describing a scanner's activity using the 360-day sequence as done in this thesis. From the perspective of the network telescope a scanner may seem to have stopped scanning while in reality it continues using another IP address. Thus the activity sequence can only capture behaviour for a certain period of a scanner's operation until it switches to the next IP address. This may explain why in 7.5 almost 50% of the scanners appear to be active on only a single day during the whole year. So unfortunately distributed scanners that are affected by IP churn where the rate is less than 3 days can not accurately be detected. The correlation algorithm currently expects 3 active days as a minimum requirement in order to reduce false-positives. In Griffioen and Doerr [2020b] they observed 1 out of 6 infections of a botnet changed their IP address due to IP churn. In 90% of those cases, change of address occurred within 3 days. Therefore, clusters that are detected in the results of this thesis are unlikely to exhibit IP churn.

11.2 Future Work

Mitigating cluster fragmentation A limitation of the current detection method is that a single cluster in reality might appear as multiple smaller clusters in the results. Exact matching of activity patterns means that two scanners from the same cluster but with slight deviation in their activity will be treated as if they were from a separate group. This effect can be mitigated by loosening the similarity requirement which can accommodate such deviations. In other words, scanners from one group might not behave exactly the same but they should be very similar relative to the other unrelated data points. If the group is large enough then the majority of scanners will still be present in the result. The requirement is a cluster size of 6 scanners for any combination of activity sequence. Even moderately sized botnets can be captured in this way. Thus the problem becomes fusing separate clusters into larger

ones which are likely to be part of the same scheme. The search space for this is much smaller than the original telescope dataset with overwhelmingly more scanners. A solution can potentially be found through the use of state-of-the-art cluster algorithms.

Correlation between scanrate and cluster size One common assumption is that large distributed scanners are able to evade current IDS by lowering the individual scanrate below a certain threshold. It would be interesting to verify this hypothesis with the available data used in this thesis. Scanners that are present in confirmed clusters will have their scanrate compared to the unfiltered set encountered by the network telescope in order to observe any statistical difference. In addition, there might be a correlation between scanrate and cluster size. Some clusters are larger than others. How large clusters are utilise their size is worth investigating. For instance, these are able to afford a lower scanrate while achieving the same overall throughput.

Efficacy of current intrusion detection systems The current threat of distributed scanners can be assessed by estimating their prevalence in the overall landscape. Not only the quantity, but also how many of these forms of malicious network reconnaissance are able to evade state-of-the-art IDS. One way of benchmarking these systems is by replaying the packets from detected clusters and observe what portion is able to successfully evade detection. Some IDS might perform better than others. Through gained insights from this thesis and the benchmark some recommendations can be given to improve the overall detection accuracy. However, IDSs need to take into account the strict near real-time processing requirements.

12 Conclusion

This research is the first to study scanners for a prolonged period of approximately one year long. To make this computationally feasible, the dataset acquired from a network telescope was transformed into compact and efficient summaries. These enabled fast queries about any active scanner observed in daily intervals. A more detailed perspective is provided at smaller intervals of the length of a raw PCAP file ranging between 5 and 20 minutes. This effectively achieved a 18000 time size reduction of the data and much faster processing times while retaining all the relevant information for studying scanners. Initial analysis provided supporting evidence that scanners from the same distributed group are likely to be simultaneously active during the same days and target the same set of ports. By leveraging this pattern of group behaviour we were able to identify a large number of clusters for which these conditions hold true in acceptable run-time. More than 1.6M scanners distributed across 44k groups where the size of each group ranges from 6 up to 9K. However the assumption of matching activity pattern leads to cluster fragmentation which means that in reality the number of unique groups is likely lower, but bigger in size. The results were subjected to additional cluster validation techniques to confirm that the correlation methodology was able to accurately detect distributed scanners. A scanner's activity is modeled using a Markov chain to judge the rarity of an activity sequence. Additionally, we measure the cluster's response during state transitions. Finally, a cluster is check whether the majority of scanners originate from the same AS. The first two methods have adjustable thresholds that can influence the performance metrics in accordance with the analyst's requirements. This study prefers a low-false positive rate at the risk of missing some clusters. Due to the absence of ground truth, we aim to establish sufficient confidence beyond reasonable doubt. There is enough evidence to show that 50% of the resulting clusters have been correctly identified. We can provide this dataset of highly probable coordinated scanners to other researchers who want to conduct research on the topic. The current lack of data is a tough barrier for the study of distributed scanners. The correlation methodology is able to detect both very large and very slow scanning clusters. Stretching the observation period to a full year enabled the detection of distributed scanners that normally would be able to blend in with the noise in small-scale observations. A case study was provided to highlight a group with an extremely low scanrate. A single packet in every few days reaching the telescope from individual scanners is sufficient to identify the corresponding group. In addition, the generated perspectives in various resolutions of the original telescope data proved to be beneficial for analysis. It becomes trivial to reveal both long-term and short-term activity patterns and which services are being targeted.

Bibliography

Bhuyan, M. H., Bhattacharyya, D. K., and Kalita, J. K. (2012). AOCD : An Adaptive Outlier Based Coordinated Scan Detection Approach. 14(6):339–351.

Dabbagh, M., Ghandour, A. J., Fawaz, K., Hajj, W. E., and Hajj, H. (2011). Slow Port Scanning Detection. *2011 7th International Conference on Information Assurance and Security (IAS)*, pages 228–233.

Durumeric, Z., Halderman, J. A., Bailey, A. M., and Paxson, V. (2017). Fast Internet-Wide Scanning : A New Security Perspective.

Ester, M. (2017). DBSCAN Revisited , Revisited : Why and How You Should (Still) Use DBSCAN. 42(3).

Feng, Y. (2013). A Behavior-based Method for Detecting Distributed Scan Attacks in Darknets. 21(3):527–538.

Gates, C. (2006). Co-ordinated Port Scans: A Model, A Detector and An Evaluation Methodology. *Methodology*.

Griffioen, H. and Doerr, C. (2020a). Discovering collaboration: Unveiling slow, distributed scanners based on common header field patterns. Cited by: 14.

Griffioen, H. and Doerr, C. (2020b). Quantifying Autonomous System IP Churn using Attack Traffic of Botnets. In *International Conference on Availability, Reliability and Security (ARES)*.

Haas, S., Wilkens, F., and Fischer, M. (2020). Scan Correlation – Revealing distributed scan campaigns. *arXiv*.

Jing Yang, Liming Wang, Zhen Xu, Jigang Wang, T. T. (2019). Coordinated Web Scan Detection Based on Hierarchical Correlation. In *International Conference on Security and Privacy in New Computing Environments*, pages 388–400.

Jung, J., Paxson, V., Berger, A. W., and Balakrishnan, H. (2004). Fast portscan detection using sequential hypothesis testing. *Proceedings - IEEE Symposium on Security and Privacy*, 2004:211–225.

Lv, Y., Li, Y., Tu, S., Xiang, S., and Xia, C. (2014). Coordinated scan detection algorithm based on the global characteristics of time sequence. *Proceedings - 2014 9th International Conference on Broadband and Wireless Computing, Communication and Applications, BWCCA 2014*, pages 199–206.

Robertson, S., Siegel, E. V., Miller, M., and Stolfo, S. J. (2003). Surveillance detection in high bandwidth environments. *Proceedings - DARPA Information Survivability Conference and Exposition, DISCEX 2003*, 1:130–138.

Yao, S., Lv, Y., Li, Y., Xia, C., Ding, S., and Yuan, Z. (2013). Coordinated scan detection based on similarities of scan behaviors. *Journal of Computational Information Systems*, 9(16):6629–6641.

Yegneswaran, V., Barford, P., and Ullrich, J. (2003). Internet Intrusions: Global Characteristics and Prevalence. (May 2003):138.