Bachelor thesis TU Delft

Classification of Distributed Strategies for Port Scan Reconnaissance

Stijn Pletinckx, Vincent Ghiëtte, and Christian Doerr
TU Delft
Cyber Security Group
2628CD Delft, The Netherlands
{s.r.g.pletinckx@student., v.d.h.ghiette@, c.doerr@}tudelft.nl

***Abstract*— Prior to exploiting a vulnerable service, adversaries perform a port scan to detect open ports on a target machine. If an adversary is aiming for multiple targets, multiple IP addresses need to be scanned for possible open ports. As sending all this probing traffic with one source IP address causes a lot of suspicion in an intrusion detection system, attackers have adopted towards a more distributed approach by using multiple source IP addresses to perform a port scan.**

In this paper, we describe various strategies on how a distributed port scan is performed by adversaries in the wild. The results in this paper are found by analyzing network packets that stem from a large network telescope. Concretely, we analyzed network traffic from one month received by 2 /16 networks. From this analysis, we conclude that many levels of coordination are exhibited by adversaries performing distributed port scans.

***Index Terms*— port scan, threat intelligence, network security**

I. Introduction

Before starting an attack on a victim's systems it is important for the adversaries to know where the vulnerabilities of that system are present. In the case of remote network attacks, the availability of open ports for protocols and services give insight into the ways a system can be penetrated and exploited. As such, it is often seen that criminals perform a reconnaissance phase before executing an actual attack in order to get as much information as possible. This allows attackers to adopt their attack accordingly, later in the cyber kill chain [1].

One way of getting information regarding a system's vulnerabilities is by scanning the machine for open ports, specifically ports listening to known protocols and services. If such a port turns out to be open, it allows the attacker to further proceed to the next step by exploiting potential vulnerabilities that are present in the service residing behind the open port. Depending on the attacker's intent, the exploit attempt can be aimed at a specific port, a specific host, or a general mix between the two. Here the attacker has three options: 1) perform a scan for multiple ports on one single IP address. This is called a vertical scan. 2) perform a scan on multiple IP addresses for one single port. This is called a horizontal scan. 3) perform a scan on multiple IP addresses for multiple ports. This is called box scanning.

For security experts, this method of probing is of great importance as it provides the information of possible threats. This threat intelligence is key for building a network's defense system as it provides already a lot of detail regarding the attacker's intent. For example, it can show whether your entire network is targeted for every port, or a small section of the network on one port. Additionally, it also provides insight into the amount of traffic the adversary needs to send prior to its attack. This information helps in detecting and identifying the attack, as well as providing insight into the attacker's resources and capabilities.

Gathering this information before an adversary is able to perform its attack allows for the ability to mitigate the threat during the reconnaissance phase. Adjusting the defense system of a network according to the actual threat received by that system will result in more efficient and effective mitigation measures, as well as a financial benefit since resources are allocated according to the needs of the system. Hence, harvesting as much data as possible during the reconnaissance phase of an attacker allows for getting an insight into what attackers are planning to do against your system.

Performing this reconnaissance phase, however, can already cause a lot of suspicion for the system's security infrastructure. This may lead to an early detection of the attacker's intent, allowing the target to already take the necessary measures for avoiding exploitation. As adversaries got aware of this plausible early detection, strategies and techniques emerged that allow for adequate port scan reconnaissance that is both stealthy and effective. This puts a gap into the machine's abilities for performing early detection of malicious probing.

In this paper, we provide a classification of distributed port scan reconnaissance strategies based on empirical data. Concretely, we analyzed network traffic received by two /16 networks. From this data, we are able to show different techniques and strategies in probing for potential targets. Specifically, we see source IP addresses, originating from the same address block, being used in two ways for target probing. One is very coordinated and contains no overlap among destinations scanned by the source IP addresses. The other is less coordinated and contains very much overlap in destinations scanned among the source IP addresses. Furthermore, we report on a strategy where the source IP addresses do not reside in the same address block but still perform coordinated bursts.

The remainder of this paper is structured as follows. Section

II presents an overview of the existing literature on port scan behavior. Section III explains the dataset out of which the results from this paper were obtained. Section IV gives a general overview of the IPv4 header and the TCP header. In section V we discuss the findings of our research. Section VI summarizes our findings.

II. Related Work

Internet traffic is highly diverse, making it very challenging to categorize its nature in a way that makes every packet's intent clear from its context. As such, network traffic, specifically port scan traffic, has been an interesting research topic for many years.

One of the earliest port scan strategy that has been studied is that of a single source scan. Here, the attacker uses one IP address to send all its port scan traffic from. This type of scan has been studied by Jung et al. [2] and Leckie et al. [3]. Both works aim at detecting single source port scans by relying on statistical and probabilistic tests performed on the network packets.

From here on, we see an evolution towards more distributed port scan attacks. In this strategy, attackers use multiple IP addresses to send its port probes from. This produces less noise coming from one IP address, allowing the attacker to perform the port scan in a more stealthy way. Robertson et al. [4] report on methods to detect distributed port scans based on source IPs residing closely together. However, this leaves out the possibility of revealing strategies that spread their sources across multiple networks, far apart from each other.

Gates [5] was able to develop an algorithm that, given a set of IP addresses, could detect coordinated probing. However, this algorithms works under the assumption that the attacker synchronizes its resources to minimize target overlap. In this paper we show that in reality strategies occur that do not minimize the overlap among source IPs.

One specific port that is of high interest when it comes to compromising a machine is port 22, the default port for the Secure Shell (SSH) protocol. In the work of Javed et al. [6], a study has been performed on attempted SSH attacks on the authors' network. The authors were able to find evidence for coordinated distributed SSH attacks. The attacks were aimed at brute-forcing SSH logins at a low rate, in order to avoid detection.

While port 22 contains a prevalent service to probe for vulnerabilities, Heo and Shin [7] concluded that shifts emerge throughout the years of ports that receive the most scans. In their study, the authors report on a high increase in Telnet port scans compared to previous literature. Furthermore, the authors list the top targeted services as well as the top countries and AS's where port scans originate from.

Also botnets make use of network probing before they try to compromise a system. In [8], Dainotti et al. performed empirical analysis of network telescope traffic. From this traffic, a botnet could be identified which gave the authors more insight into the botnet's modus operandi. Concretely, the authors discovered both coordinated as well as uncoordinated aspects in the port scan activity of the botnet.

Most academic work on port scan behavior was performed on artificial or short network traffic data. However, in 2007, Allman et al. [9] were the first to perform a longitudinal study of scanning behavior perceived at one site. They were able to illustrate the evolution in terms of number of scanners, targets, and patterns. The authors concluded that the overall volume of scans has experienced a radical increase over the years. Furthermore, the authors report on two types of scanning strategies: (1) sending high amount of network packets in a short time, and (2) sending very little amount of network packets over a relatively long period. Especially the latter category is considered hard to distinguish from regular network traffic

This so called slow scan behavior described by Allman et al. has been further studied by Dabbagh et al. [10] and Shao et al. [11]. Both works report on a detection for slow port scan behavior.

We also see that more and more (open source) tools become available to probe for potential targets. In the work of Ghiëtte et al. [12] it is shown that tool chains used by adversaries to probe potential targets can already be detected via port scan traffic. Through this, the authors were able to depict differences in tool adoption across the world as well as show that discrepancies exists in the use of tools depending on the type of port scanning.

Concretely, we see that the current literature on port scanning already covers a lot of dimensions. However, we also notice a gap on studies aimed at discovering and describing patterns and coordination strategies for probing. In this paper, we try to fill this gap by reporting on distributed strategies for port scan reconnaissance based on analysis of traffic retrieved by a network telescope. In the next section, we elaborate more on the dataset used for this research

III. Data Collection & Sanitation

The findings in this work stem from the analysis of a 3x /16 network telescope. The dataset contained only traffic to unused IP addresses, meaning that no machine was associated with them. Since these addresses were unused, the only traffic received by the network were port scans and backscatter. In total, the telescope was able to collect data for a period of 3 years.

As the total dataset comprised 15 TB of information, it was impossible to perform a packet per packet analysis. As such, the findings of this paper are based on a snapshot of the data. The snapshot contains data from the month April of 2015.

The data from the telescope originates from traffic sent to the TU Delft network. Particularly, the data stems from three IPv4 address blocks that were assigned to the university:

- 130.161.0.0/16, private network
- 130.180.0.0/16, private network
- 145.94.0.0/16, public network

Analyzing the full dataset, we noticed a significant difference in packets received by the private networks as compared to packets received by the public network. A hypothesis for

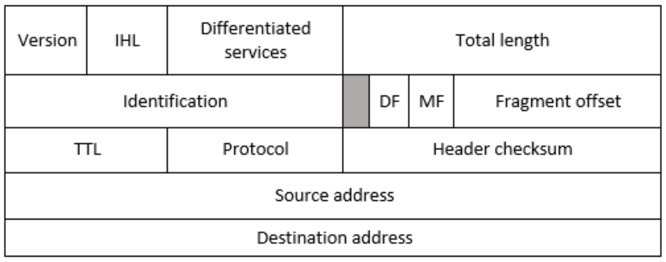

Fig. 1. The IPv4 header.

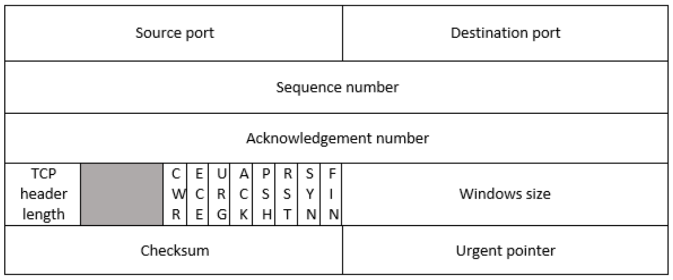

Fig. 2. The TCP header.

this difference could be the wireless nature of the public network. As this part of the network is mostly used by students and visitors, it is very common that connections get abruptly closed i.e. by turning off a cell phone or closing a laptop. If the service was using a connection based on the UDP protocol, it will keep sending packets after the connection was abruptly closed. TCP has a somewhat similar behavior, however, TCP will stop sending packets once the windows size is full.

The type of traffic sent by UDP or TCP after a connection was closed may be mistakenly labeled as port scan behavior during the research. As such, we decided to filter out all data from the public network.

IV. IPv4 and TCP header

The results in this paper are based on port scan packet analysis. To motivate some of the decisions made in the research process, we first give a general overview of the IPv4 header and the TCP header, and how these would be constructed in regular, non probing, network traffic. Later in the paper we will show manipulations of these headers used by attackers for port scanning purposes. All information in this chapter is retrieved from the book *Computer Networks* by Andrew Tanenbaum [13].

A. IPv4 header

The IPv4 datagram is constructed in two parts: a header part and a payload part. In the header, a block of 20 bytes contains the most fundamental information to send the packet around. Figure 1 depicts the format of the IPv4 header. The *Version* field, of 4 bits, shows the version of the IP protocol, of which version 4 is the most dominant on the Internet today. Next is the *Internet Header Length* (IHL) that tells how long the header is, expressed in 32 bit words. Following the IHL is the *Differentiated service* which has changed meaning over the years. Initially, the field was meant to articulate the type of service used. However, since no one knew what to do with this field, the meaning has changed to provide congestion notification information. Further we have the *Total length* field which shows the length of the entire datagram, so both header and payload.

The next 32 bits of the IPv4 header start with an 8 bit *Identification* field (IPID). Since IP packets can be fragmented into smaller chunks, the IPID is used to group these fragments together such that the destination is able to construct all into one entity. For each entity the IPID is random number between 2 and 2^{16} Next are 3 bits of which the first remains unused and the other two are flags used for fragmentation. The *Don't Fragment* (*DF*) flag orders the router not to fragment the packet. The *More Fragments* (*MF*) flag indicates that more fragments are coming, with a 1, or that this is the last fragment, with a 0. The *Fragment offset* then follows to depict where in the packet this fragment belongs.

The third row of 32 bits starts with a *Time To Live* (TTL) field which is a counter that gets decremented at every hop on the packets route. This circumvents the possibility that network packets travel around forever on the Internet. Next, a *Protocol* gets specified in 8 bits such that the network layer knows what to do with the packet. Examples can be TCP (00000110) or UDP (00010001). Because Internet transmission is quite volatile, a checksum in the *Header checksum* field gets added to the header which enables routers to check whether no modification to the data has happened in transit. To end the IPv4 header, two 32 bit IP addresses are added: the *Source IP* and the *Destination IP*, respectively.

B. TCP header

To make use of the TCP protocol, some additional information needs to be provided to the payload of the IPv4 datagram. Concretely, a TCP header and payload are necessary to communicate via the TCP protocol. Figure 2 shows an overview of the TCP header fields.

The TCP header starts of with 32 bits allocated for the *Source port* and the *Destination port*. Together with the source and destination IP addresses from the IPv4 header, they form the two end points between which a TCP connection needs to be made.

Then follows a 32 bit *Sequence number* and a 32 bit *Acknowledgement number*. These numbers are used to setup and maintain the TCP connection during the three-way handshake phase. We will elaborate more on these numbers and the three-way handshake phase in the next subsection.

The following 32 bits start with the *TCP header length* which indicates, in 32 bit words, the length of the TCP header. Next come 4 unused bits, after which 8 bits of flags follow. The first 2 flags (*CWR* and *ECE*) are used for congestion control. The *URG* flag is set to 1 when the *Urgent pointer* of the TCP field is being used. The *ACK* flag is set to one when the *Acknowledgement number* is valid. Next follows the *PSH*

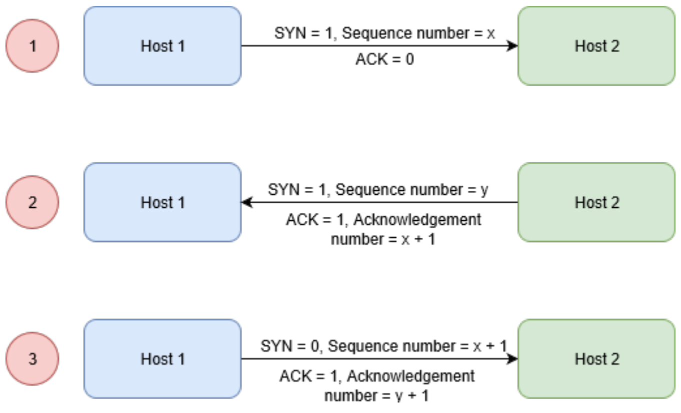
1
Host 1
SYN = 1, Sequence number = x
ACK = 0
Host 2
2
Host 1
SYN = 1, Sequence number = y
ACK = 1, Acknowledgement number = x + 1
Host 2
3
Host 1
SYN = 0, Sequence number = x + 1
ACK = 1, Acknowledgement number = y + 1
Host 2

Fig. 3. The TCP Handshake.

flag to indicate that the data sent should not be buffered by the receiver. The *RST* flag is used when a connection needs to be reset due to a crash from a host or some other reason. Furthermore, the *RST* bit also gets used to reject a connection. Next follows the *SYN* flag which is essential in establishing the TCP connection via the three-way handshake. In short, the *SYN* bit gets set to 1 upon a connection request or when a connection gets accepted. Finally, there is the *FIN* flag, which indicates that the sender has no more data to send and that the connection can be closed. Next follows the 16 bit *Window size*. TCP uses a sliding window protocol to control the flow.

Just as with the IPv4 header, the TCP header contains a *Checksum* field to check for modification that may have happened to the data while in transit. Finally, we have the *Urgent pointer* which indicates where urgent data can be find, inside the payload, by specifying a byte offset.

C. *Three-way handshake*

To establish a TCP connection, the protocol makes use of what is called a 'three-way handshake'. This is depicted in Figure 3. The handshake starts with host 1 sending a SYN packet to host 2. This packet has the SYN flag set to 1 and contains a sequence number. At the start of a new connection, this sequence number is a random value between 0 and 2^{32}. If the port at the IP address receiving the packet is listening, and hence open, the destination sends back a SYN+ACK packet. This has both the SYN and ACK flag set to 1 and sends a different (random) sequence number back to host 1. Additionally, an acknowledgement number is send back which is the sequence number first sent by host 1, incremented by one. To finish the connection setup, host 1 sends back an ACK packet to host 2. The initial sequence number of host 1 gets incremented by one and is send back to host 2. The acknowledgement number from step two is also incremented by one and send back to host 2. A TCP connection is now established.

In this next chapter we will show how these header values have been manipulated by adversaries in their coordination of distributed port scans.

V. Classification

The main goal of threat intelligence is to get as much information as possible about possible attacks or strategies that can be performed against your system. By knowing how these threats work, what their procedures are, and what their intent is for attacking your system, better defense mechanisms can be built that are tailored towards the capabilities of the threats.

In this research, we performed threat intelligence on the first phase of the cyber kill chain, namely the reconnaissance phase [1]. With the insights of this paper, we show that it is possible to already get valuable information about a possible threat long before an attack has even taken place. This allows for the development of security measures that can already mitigate attacks during this reconnaissance phase. In what follows, we present some theories on possible port scan strategies, followed by empirical evidence found in our telescope data.

A. *Distributed Port Scans*

As was already shown by the literature, scanning with one single source IP address can cause suspicious network traffic that is relatively easy to detect. As such, we see that attackers have managed to coordinate multiple IP addresses to work together with the goal of probing potential targets in a more stealthy way.

Such coordination can happen in many different ways. Assume the attacker has a block of source IP addresses that are located close to each other. Additionally, the attacker has a set of targets, which consists of different destination IP addresses and destination ports. A first strategy for probing could be to naively let every source IP address in the block scan for each destination. This requires very little amount of coordination effort for the attacker and the individual traffic for each source IP address gets reduced compared to using only a single source IP address to perform the same action. The more source IP addresses available to the adversary, the less traffic produced by a single IP address in the block. However, this strategy would create a lot of overlap in the destinations scanned by each source IP address. This reduces the efficiency of the attacker as a trade-off for less coordination effort.

Another strategy is to assign each source IP address in the block a designated list of destinations. This strategy would still enable the attacker to produce less individual network traffic for each source IP address while having the efficiency of not scanning the same destination twice. However, compared to the previous strategy, this method requires more coordination from the attacker.

Both strategies already reduce the chance of being detected as they lower their individual network traffic by grouping source IP addresses together. In the end, they achieve the same result as a single source scan but in a more stealthy way. Nevertheless, since these source IP addresses belong to the same block, their traffic could still be accumulated and evaluated by an intrusion detection system which allows it to still detect the probing endeavors.

A solution for this is to get a pool of source IP addresses that do not belong to the same block. Instead, the attacker

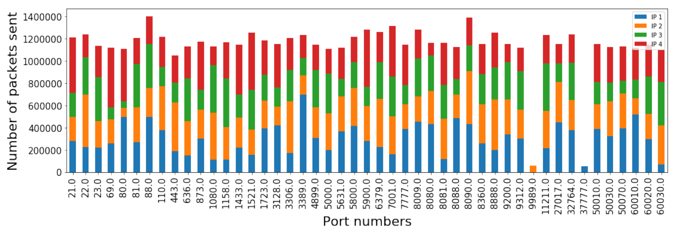

Fig. 4. Accumulated port distribution. Each color represents an IP address from the block.

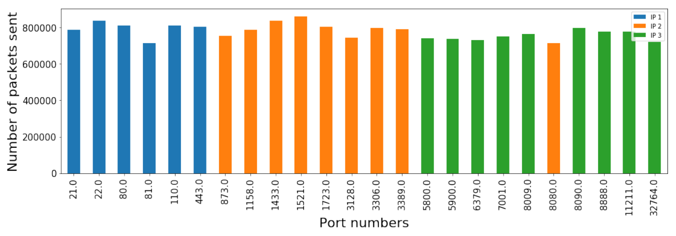

Fig. 5. Accumulated port distribution. Each color represents an IP address from the block.

would spread his source addresses across multiple IP subnets such that the IP addresses are not closely related. This makes it harder for detection systems to group them as one entity and hence, makes the probing a lot stealthier. Albeit these benefits, this strategy requires more coordination work compared to the previous strategies as the complexity increases.

B. Findings in Telescope Data

Looking for port scan strategies in our telescope data, we found that the most interesting values of the network packets were: epoch time, source IP address, source port, destination IP address, destination port, sequence number, and IPID. As explained in section II, both IP addresses and ports form the end points of the TCP connection. These fields allow to group potential coordinated probes. This is interesting to analyze since coordinated port scans might contain overlap in the destination IP addresses they want to scan. Another possibility is that there exists an incremental/sequential pattern for scanning destination IP addresses. Furthermore, we also explained that the sequence number and IPID are random values that are necessary for setting up the connection or for fragmentation, respectively. If an attacker chooses a certain tool, it might be the case that these fields get static values, or that their values are based on a distinct pattern instead of being random [12]. Also the attacker himself might decide to implement his own values before executing the port scan. Hence, if we would see static values or patterns in these numbers, the likelihood of coordinated probing increases.

We now present empirical evidence of the strategies described in the previous subsection. To find coordinated blocks, we start with sorting all source IP addresses based on the amount of network packets they sent in the course of one month. This immediately revealed potential coordinated scans as several source IP addresses residing in the same block had similar amounts of network traffic. One of those blocks contained 4 IP addresses and uniformly scanned a total of 48 destination ports. However, the distribution of destination ports scanned by each IP address in the block showed no uniform trend and was rather random. This shows that the IP addresses appear to have random behavior, but in reality, they are coordinated to perform a uniform scan on a set of destination ports. Figure 4 shows a stacked bar graph of the ports scanned by all IP addresses in the block. In the figure, it can be seen that the individual distributions are not uniform but rather random.

If we now also look at the destination IP addresses scanned by the group, we notice that almost 90% overlap exists in the destination scanned between each tuple of source IP address. What we can conclude from this is that very little coordination effort is done among the source IP addresses in this group. Due to the large amounts of overlap, the attacker reaches limited efficiency while probing for potential targets. This is contradictory to the assumption, present in some of the literature, that distributed port scans have small overlap in the destination addresses scanned.

Looking at when these port scans happen, we discover another pattern in the modus operandi of this block. Figure 7 depicts when each port got scanned over the period of one month. The time intervals between a scan of the same port is constant. However, the source IP address actually performing the scan is every time random. This shows that coordination effort has been made in timing when a scan should happen, but no coordination seems to happen regarding which IP address should perform the actual scan.

We also found an example of a more coordinated block of IP addresses. The block contains 3 addresses that, again, have more or less the same amount of network packets over the period of one month. The block scans a total of 24 ports with no overlap in destinations scanned. Looking at the accumulated destination port distribution, we see a completely different pattern compared to the previous block. Figure 5 depicts the distribution and we see a clear division of destination ports among the IP addresses. Here, the individual distributions are also uniform. Both the no-overlap and the division of destination ports show that the adversary behind this block has performed more coordination efforts compared to the previous block. This results in more efficiency as no destinations are scanned twice by a different source IP address. Another difference compared to the previous group is that each source IP address in the block uses the same IPID, 54321, for every packet sent.

If we then plot again these port scans over time, another pattern emerges that uses consistent intervals at which a port gets scanned. Figure 8 shows a depiction of these ports scans over the time.

Figure 7 and 8 adequately illustrate the difference in coordination efforts between the two blocks discovered in our telescope data. Where one block assigns to each scan a random IP address, the other shows a clear division for

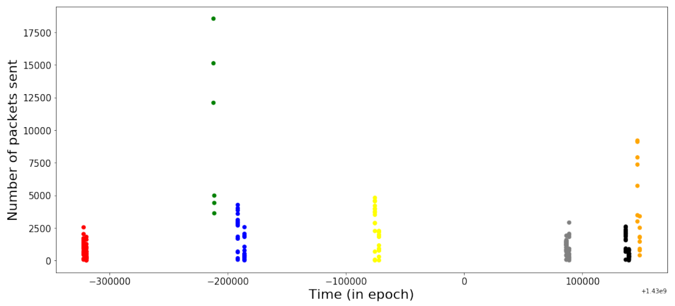

Fig. 6. Packets sent over time. Each color represents an IP address from the pool.

which port needs to be scanned by which IP address. Both graphs illustrate that the two blocks have a consistent interval at which the ports get scanned. We also want to make clear to the reader that not every block of IP addresses that we found with a similar amount of network traffic belongs to one entity performing a port scan. We encountered many examples where such a traffic was merely a coincidence as their destination overlap and interval times showed no correlation.

As already mentioned, using multiple source IP addresses from the same block is stealthier compared to using only a single IP address, but still relatively simple to detect as their activity can easily be grouped together. A way of circumventing this is by using source IP addresses that do not reside in the same address block. This strategy was also visible in our telescope data.

We found a group of IP addresses that contained the same amount of network traffic produced over one month for each source IP address. In contrast to the previous group, the source IP addresses do not reside together in the same block. Additionally, network packets sent by the source IP addresses in the group all used source port 6000 and contained the same static IPID value, 256.

Looking deeper into what this group scans for, we notice that only a handful of destination ports are being targeted. The destination IP addresses is a fixed list, and each source IP address from the group goes incrementally through this list. All these features lean towards the assumption that the source IP addresses in the group perform a coordinated port scan. When we plot the amount of scans over time, it can be observed that all IP addresses from the pool scan in high density burst. This is depicted in Figure 6.

VI. Conclusion

The reconnaissance phase is the starting point of the cyber kill chain. It allows adversaries to gain information about potential targets prior to commencing the attack. Port scanning is one of such reconnaissance tactics that looks for open ports on a target machine. These open ports can allow the adversary to exploit possible vulnerabilities that are present in the service behind the open port.

There are many ways to approach port scan reconnaissance and in this paper we have reported on distributed strategies in probing for potential targets. The findings in this paper stem from a large network telescope that allowed us to study realistic network behavior of intrusion attempts. We studied network activity from one month containing data received by 2 /16 networks. By analyzing the network packets, we were able to classify three different distributed strategies based on their coordination and values in the IPv4 and TCP header fields.

Contradicting to what have been assumed in some of the literature, we have shown that distributed port scans are not necessarily coordinated on every possible aspect. Concretely, we have found distributed scans containing much overlap among the destinations scanned by their source IP addresses, effectively making them less efficient. Additionally, we have shown that distributed scans can be performed, both, by source IP addresses residing closely to each other and by source IP addresses that are far apart.

References

[1] T. Yadav and A. M. Rao, “Technical aspects of cyber kill chain,” in *Security in Computing and Communications* (J. H. Abawajy, S. Mukherjea, S. M. Thampi, and A. Ruiz-Martínez, eds.), (Cham), pp. 438–452, Springer International Publishing, 2015.

[2] J. Jung, V. Paxson, A. Berger, and H. Balakrishnan, “Fast portscan detection using sequential hypothesis testing,” *IEEE Symposium on Security and Privacy, 2004. Proceedings. 2004*, May 2004.

[3] C. Leckie and R. Kotagiri, “A probabilistic approach to detecting network scans,” *NOMS 2002. IEEE/IFIP Network Operations and Management Symposium. Management Solutions for the New Communications World(Cat. No.02CH37327)*, 2002.

[4] S. Robertson, E. Siegel, M. Miller, and S. Stolfo, “Surveillance detection in high bandwidth environments,” *Proceedings DARPA Information Survivability Conference and Exposition*, Apr 2003.

[5] C. Gates, “Coordinated scan detection,” *16th Annual Network and Distributed System Security Symposium*, 2009.

[6] M. Javed and V. Paxson, “Detecting stealthy, distributed ssh bruteforcing,” *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security - CCS 13*, 2013.

[7] H. Heo and S. Shin, “Who is knocking on the telnet port: A large-scale empirical study of network scanning,” in *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*, ASIACCS ’18, (New York, NY, USA), pp. 625–636, ACM, 2018.

[8] A. Dainotti, A. King, K. Claffy, F. Papale, and A. Pescap, “Analysis of a ”/0” stealth scan from a botnet,” *Proceedings of the 2012 ACM conference on Internet measurement conference - IMC 12*, 2012.

[9] M. Allman, V. Paxson, and J. Terrell, “A brief history of scanning,” *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement - IMC 07*, 2007.

[10] M. Dabbagh, A. J. Ghandour, K. Fawaz, W. E. Hajj, and H. Hajj, “Slow port scanning detection,” *2011 7th International Conference on Information Assurance and Security (IAS)*, 2011.

[11] G.-L. Shao, X.-S. Chen, X.-Y. Yin, and X.-M. Ye, “A fuzzy detection approach toward different speed port scan attacks based on dempster-shafer evidence theory,” *Security and Communication Networks*, vol. 9, no. 15, p. 26272640, 2016.

[12] V. Ghiette, N. Blenn, and C. Doerr, “Remote identification of port scan toolchains,” *2016 8th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*, 2016.

[13] A. S. Tanenbaum and D. Wetherall, *Computer networks*. Pearson India Education Services Pvt, Limited, 2014.

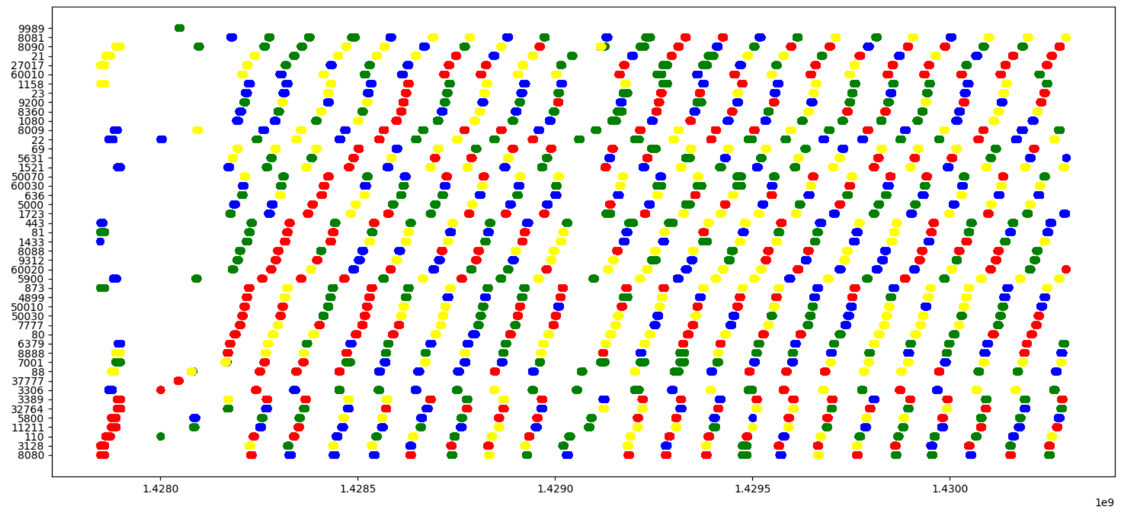

Fig. 7. Ports scanned over time. Each color represents an IP address from the block.

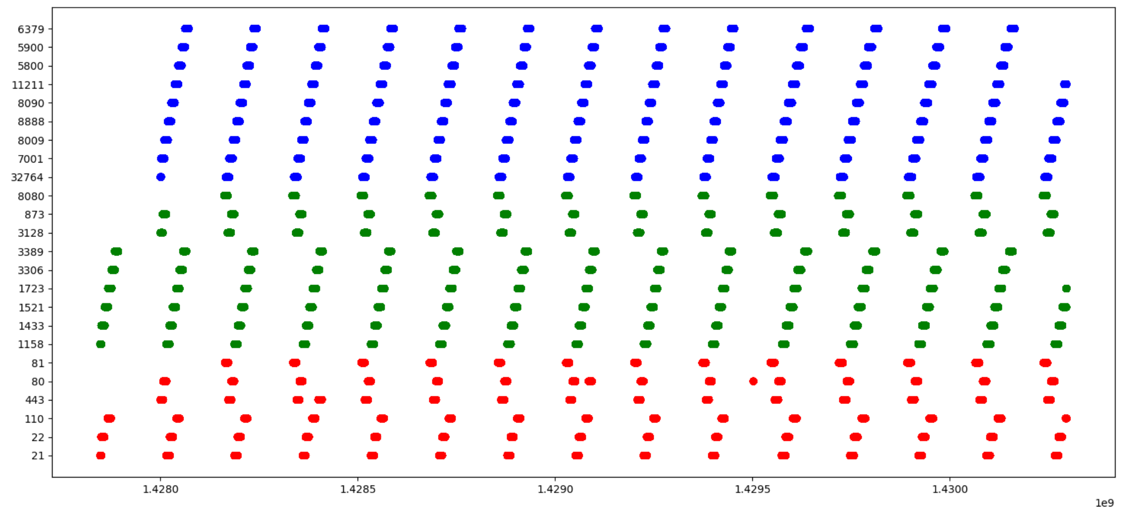

Fig. 8. Ports scanned over time. Each color represents an IP address from the block.