Experiences and observations from the NoAH infrastructure

Georgios Kontaxis, Iasonas Polakis, Spiros Antonatos and Evangelos P. Markatos
Institute of Computer Science
Foundation for Research and Technology Hellas
{*kondax, polakis, antonat, markatos*}*@ics.forth.gr*

***Abstract*—Monitoring large chunks of unused IP address space yields interesting observations and useful results. However, the volume and diversity of the collected data makes the extraction of information a challenging task. Additionally, the maintenance of the monitoring infrastructure is another demanding and time-consuming effort. To overcome these problems, we present several visualization techniques that enable users to observe what happens in their unused address space over arbitrary time periods and provide the necessary tools for administrators to monitor their infrastructure. Our approach, which is based on open-source standard technologies, transforms the raw information at the network level and provides a customized and Web-accessible view. In this paper, we present the design, implementation and early experiences of the visualization techniques and tools deployed for the NoAH project, a large-scale honeypot-based infrastructure. Additionally, we provide a traffic analysis of data collected over a six month period of our infrastructure's operation. During the data collection period, we observed that the number of attackers continually increased as did the volume of traffic they generated. Furthermore, interesting patterns for specific types of traffic have been identified, such as the diurnal cycle of the traffic targeting TCP port 445 (Windows Directory Services), the port that receives the largest volume of attack traffic.**

I. Introduction

During the last few years, we have been witnessing an increasing number of cyberattacks on the Internet. Viruses, worms, trojans, spyware and other types of malicious programs are obstructing the effective use of the Internet and crippling the network infrastructure. We can provide endless examples of worms and viruses that are fast and can cause severe damage. The SQL Slammer worm was able to infect more than 70,000 victim computers in less than 15 minutes. During the summer of 2003, the Blaster worm managed to infect more than 400,000 computers. In 2001, more than 4,000 Denial-of-Service (DoS) attacks were launched on the Internet every week [1], often attracting the interest of popular media. Although the problem of Denial-of-Service attacks was already widely known, the magnitude of the problem had been largely underestimated. It is difficult to measure the damage caused by viruses and worms, however some estimates put the cost in the order of billions of dollars [2]. However, this damage may actually be small compared to what these attacks can potentially do, as illustrated in a study on so-called Warhol worms [3]. Such worms can cause massive damage of unprecedented effect causing severe disruption to the Internet infrastructure and services.

Although tools and systems that can help us protect our infrastructure from cyber-attacks exist, these tools are usually limited to protecting against known forms of attacks. Antivirus systems can protect users against known viruses, but are usually helpless when confronted with a new virus instance. Similarly, Intrusion Detection Systems can generate alerts for known forms of worms but are of little help when confronted with a previously unknown attack. Thus, we need a security infrastructure that is able to detect new forms of attacks, and has the critical mass to detect them as early as possible. This will allow scientists and engineers to study, analyze and rapidly develop the appropriate defenses.

The NoAH project is a scalable and extensible architecture based on the use of a distributed system of cooperating honeypots. Honeypots are non-production systems that monitor unused resources, such as unused IP address space. By default, honeypots should not receive any kind of activity and, thus, any communication from a host is flagged as suspicious and must be examined. The importance of honeypots lies in the fact that they can detect novel attacks without any false positives, unlike traditional defense mechanisms such as network intrusion and prevention systems. The NoAH infrastructure has been deployed for over a year and covers close to ten thousand unused IP addresses.

In this paper we *present the statistics gathered by the NoAH infrastructure and the visualization techniques used for the data representation.* We have implemented and deployed several web-based interfaces to display statistics of the collected traffic, perform queries over the data and monitor the status of deployed components. Furthermore, we *provide results from the traffic analysis performed on the data collected from one of our largest sensors*. We present a *case study for port 445*, the most attacked TCP port. While the aggregated traffic does not present a specific pattern, traffic on that port presents an interesting diurnal pattern.

The paper is structured as follows. Section II describes the related work while Section III presents the NoAH infrastructure. The deployed visualization techniques are described in Section IV. Traffic analysis over gathered data from one NoAH sensor is presented in Section V. In Section VI we explore future directions in extending our infrastructure and conclude in Section VII.

II. Related work

Collapsar [4] proposes a decentralized architecture composed of a large number of honeypots deployed in different network domains. This approach tries to address the problem that centralized honeypot farms have limited view of Internet activity. The core idea of Collapsar is to deploy traffic redirectors in multiple network domains and examine the redirected traffic in a centralized farm of honeypots.

Leurre.com [5] is a distributed honeypot environment that operates a broad network of honeypots covering around 30 countries. Honeypots run a modified version of honeyd and emulate three different operating systems; two from the Windows family (98 and NT server) and Redhat 7.3. Traffic and security logs are retrieved daily and stored into a centralized database. Apart from logs, raw traffic is also analyzed, mainly to derive information about attackers and specifically IP geographical location, DNS names, OS fingerprinting and TCP stream analysis. As honeyd emulates three operating systems, each platform needs 3 dark IP addresses to listen to. These IP addresses are consecutive and each emulated OS is assigned to listen to one of them. The reason for listening to consecutive IP addresses is to identify attackers that scan subnets.

A honeynet is an architecture proposal for deploying honeypots. Deployed honeypots can be both low and high-interaction honeypots but honeynet architecture discusses mainly about high-interaction ones. According to the Honeynet Project [6] architecture, honeypots live in a private subnet and have no direct connectivity with the rest of the Internet. Their communication is controlled by a centralized component, actually the core of the architecture, called honeywall. Honeywall performs three operations: data capture, data collection and data analysis. Data capture mechanism monitors all traffic to and from the honeypots. The challenge here is that a large portion of the traffic is over encrypted channels (SSH, SSL, etc.). To overcome this problem, Sebek [7] was introduced to the honeynet architecture. Sebek is a hidden kernel module that captures all host activity and sends this activity to the honeywall. As Sebek runs in the host level, it can capture traffic after being decrypted. The data control mechanism tries to mitigate the risk of infected honeypots. As honeypots will eventually be compromised, they can be used for attacking other non-honeypot systems. Data control can be performed in many ways; limiting the number of outbound connections, removing attack vectors from outgoing traffic or limiting the bandwidth.

Several tools have been implemented to visualize security-related data. Pkviz [8] is a packet animation visualizer. It accepts tcpdump files and replays the trace by plotting, using two-dimensional space, each byte in a packet from left-to-right (x axis) from the first byte to the last. The value of the y axis depends on the actual value of the byte (0-255). The rapid successive representation of packets in a trace produces an animation describing the network traffic. Circos [9] is designed to visualize two-dimensional tabular data. A circle, split in two arches, is used to depict the attributes of the dataset e.g. honeypot sensor and attacking country. Color ribbons between two arches relate to the strength of the relationship (here: a lot of attacks from one country to a specific honeypot). The thicker the ribbon, the more prominent the relationship. CAIDA's Cuttlefish [10] produces animated images that reveal the correlation between the diurnal and geographical patterns of displayed data. Besides drawing an absolute value for the data (e.g. volume) on the map, it illustrates the sun's movement over the globe, therefore defining any given moment using time, place and data value. This is suitable for pointing out patterns between local time and the visualized events.

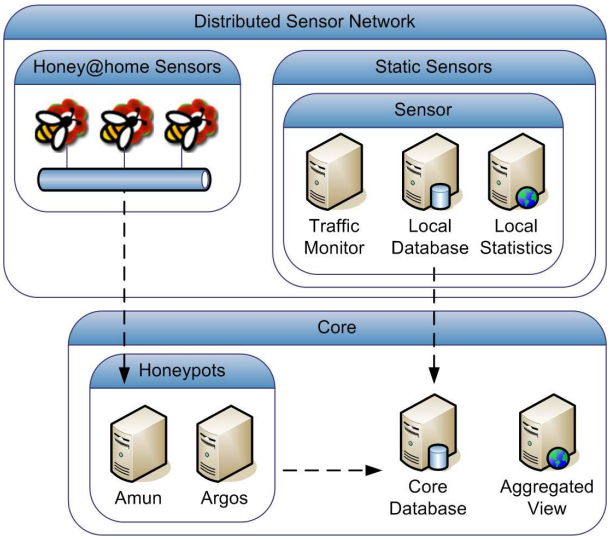

Figure 1: The design of the NoAH infrastructure. Note that Honey@home and static sensors may be co-located on the same physical machine.

III. The NoAH infrastructure

In this section we outline the three basic components of our infrastructure; *a distributed network of sensors, a combination of high and low-interaction honeypots and a web-based front-end*. Figure 1 presents this design.

Distributed Sensor Network. The first basic component are the sensors. They are the "eyes and ears" of the architecture as they own the unused IP address space and forward traffic to the honeypots for further analysis and attack detection. In the NoAH architecture two types of sensors exist; *static* and *Honey@home (dynamic)* sensors.

Static Sensors. They are physical machines which monitor large portions of unused IP address space (usually more than 256 addresses) and run specialized software to capture and store traffic going to that space. Static sensors are usually hosted by organizations and institutions. They present a low

demand for maintenance as they only have to be monitored for their uptime; as they do not run any other service except packet capturing and forwarding, their risk for infection is minimal. Each sensor runs its own web interface for displaying the statistics it has gathered. We will refer to these statistics in more detail in the next Section.

Honey@home Sensors. We implement dynamic sensors with the use of Honey@home. Honey@home is designed for monitoring unused address space from users that are not familiar with honeypot technologies or are not able to maintain a fully-fledged sensor. It runs in the background of any personal computer for both Unix-like and Windows operating systems. By default, it creates a pseudo-interface in the user's machine and requests an extra IP address from the local DHCP server. All the traffic going to that extra IP address is forwarded to the central farm of honeypots. In the absence of a DHCP server or in the case where the user is behind NAT, the tool reverts to unused port monitoring. That is, it captures and forwards traffic going to specific ports not bound by the user, for example, ports usually bound by a database server or a Web server. All clients are connected to a front-end server that authenticates them, monitors their availability and stores statistics about the forwarded traffic. As Honey@home is run by home users, it is expected that each client will present low availability. What is more important is to monitor the availability and normal operation of the front-end server. In regards to the users' privacy, no user traffic is at any moment monitored, captured or stored by Honey@home. Furthermore, there is the option to communicate with the core over Tor [11] so that users forwarding traffic to the farm of honeypots cannot be identified and possibly attacked by malicious parties.

Honeypots. We employ the Amun [12] low-interaction honeypot to simulate the most common service vulnerabilities. Traffic, forwarded from the sensors, is directed to the appropriate service and, if it contains malicious data, the respective vulnerability will be triggered and the attack will be recorded for further analysis. Amun simulates a finite set of known vulnerabilities. To expand its capabilities, more vulnerability modules have to be written. To detect previously unknown attacks, we employ the Argos [13] high-interaction honeypot and its memory tainting mechanism. However, as high-interaction honeypots are heavily instrumented machines, their processing power and memory overhead is high. The availability of honeypots is a critical issue; their uptime must be 100% so as not to miss any attacks.

Web-based Front-end. The last basic component is the central website that displays aggregated statistics from all static sensors and Honey@home clients. Additionally, it monitors and visualizes the availability and normal operation of both honeypots. It is basically the single point where NoAH administrators can see if the infrastructure is working as intended, which sensors are working or not, which

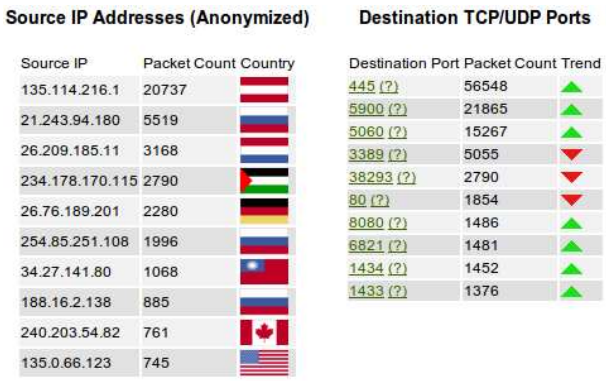

Figure 2: The top 10 source IP addresses and destination ports as monitored by a NoAH sensor for one day

Honey@home clients are connected and the statistics about the traffic monitored by the sensors at various timescales.

In the next Section, we will present all the deployed techniques for representing the attack traffic received by the NoAH sensors as well as the monitoring tools that check the availability and normal operation of the infrastructure.

IV. Statistics and Visualization

The NoAH infrastructure so far includes ten static sensors and dozens of Honey@home clients, monitoring more than nine thousand unused IP addresses. The static sensors are geographically distributed and monitor unused addresses from diverse environments; from universities and institutions to ISPs and medical centers. On average, the high-interaction honeypots process around half a million packets per day. Manual inspection of such a large volume of traffic is practically impossible and, thus, automatic mechanisms that display statistics and trends about received traffic are needed. We present the types of statistics gathered by each sensor and how they are visualized.

A. Statistics

Each sensor runs three software components. The first one is a minimal daemon based on the pcap library [14] that listens to an interface and captures packets going to a given unused IP address space. Specific pieces of information of the captured packets are stored in a local Postgres database (the second software component). This information includes the packet protocol, source and destination IP addresses, source and destination ports, flags in the case of a TCP packet and finally the timestamp of when the packet was captured. The last component is a set of PHP files that retrieve and render statistics on the collected attack traffic:

Top source IP addresses. By default the top 10 source IP addresses that sent the most packets during the last 2 hours is displayed. For each IP address the number of packets it sent and its geographic location are also displayed.

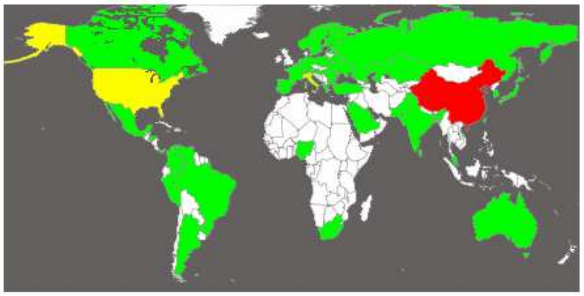

Figure 3: The geographical distribution of attackers as monitored by a NoAH sensor for one day

The geographic location is retrieved by a local MaxMind[1] database. Additionally, each IP address is clickable. By clicking it, users are redirected to a webpage that displays all packets sent by that IP address for a configurable time period. Users are able to select the time period which varies from two hours up to the last month.

Top destination ports. The top destination ports targeted by attackers are displayed. For each port the number of packets and a trend indication are also shown. The trend indication represents whether the sensor received more or less packets at that port in comparison to the previous time period. Again, the user can configure the time period up to the last month. By clicking a port, a webpage containing all traffic sent to that port is presented. A screenshot that shows the top ten source IP addresses and destination ports for one day can be seen in Figure 2.

Attack map. This page includes two global maps. The first map displays the geographic distribution of distinct source IP addresses. Each country is colored based on how many IP addresses are hosted in that country. Countries that host no attackers are colored as white, low activity countries are colored as green while countries that host lots of attacking IP addresses are red. The second map looks like the first one but is based on the number of packets sent by each country. The scale of both maps is calculated dynamically based on the traffic volume. Maps are generated on demand. A map with the number of attackers as seen by a NoAH sensor during one day is shown in Figure 3.

Attack graphs. This page includes three graphs. The first one is a breakdown of the TCP ports while the second one is a breakdown of UDP ports for the last two hours. The third one is a breakdown of traffic in terms of how much TCP, UDP and ICMP traffic was received during the last day.

Backscatter traffic. Each sensor receives unsolicited backscatter traffic, i.e. traffic sent from a host in response to attacks that had a spoofed source IP address. It is trivial to identify such traffic by inspecting the TCP flags of each packet. A plot for the number of backscatter packets received during the last week is displayed on a separate page.

[1]http://www.maxmind.com/app/geolitecity

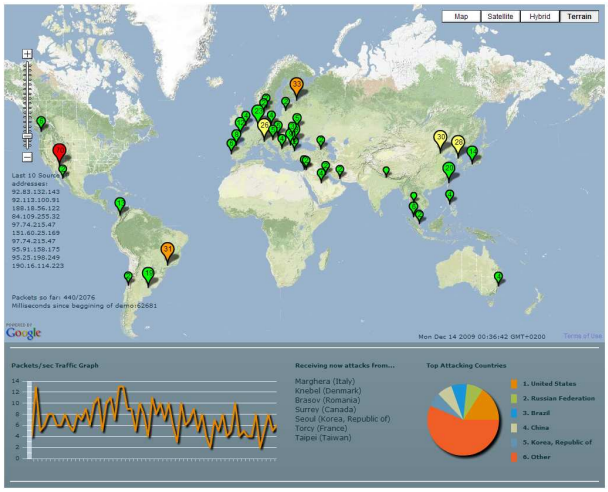

Figure 4: Screenshot from TrGeo, a Flash application that displays the geographic origin of attackers

B. Attack Traffic Visualization

TrGeo. In an effort to present an overview of what traffic our honeypots capture daily, we have implemented TrGeo. TrGeo is a platform for geographic visualization of packets captured by the NoAH infrastructure. The basic concept behind TrGeo is to track locations of the attackers and display the traffic volume they send on the global map. For the purposes of this work, we have implemented TrGeo as an Adobe Flash application that renders desired data on top of Google maps. On each source location a balloon is drawn that represents how much traffic the location sent in terms of packets. As time passes, the radius of the balloons changes according to the volume of traffic received by the location they represent. In fact, TrGeo implements the visualization of a sliding time window. For example, if the sensor has not observed packets from a location for a long time period, the balloon for that location will have its counter and size decreased. The information about packet volume and geographical origin is extracted from queries submitted to our database. Aggregation is done at the level of countries. A screenshot of TrGeo is shown in Figure 4.

Parallel-coordinate Graphs. During the design and implementation of the maps and graphs from above, we constantly found ourselves limited by the two-dimension barrier. Therefore, we seeked new ways of visualization and decided to employ a tool capable of plotting data in parallel coordinates, called Picviz [15]. To depict the n-dimensional space of network traffic (source/destination IP address/port, time, payload size, etc), “n” parallel equally-spaced vertical lines are drawn and a point in that space i.e. an event, such as a network packet arriving, is represented

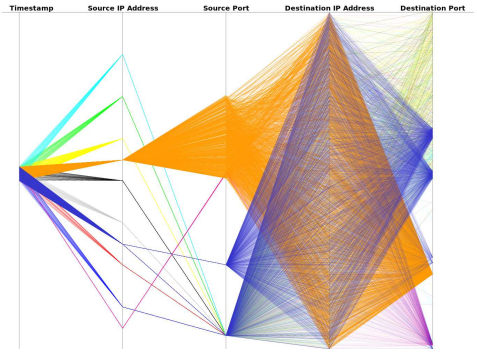

Figure 5: Screenshot from PicViz, visualizing the activity of the top 10 attackers for one hour in one of the NoAH sensors

Figure 6: Location and monitoring of NoAH sensors

as a polyline with vertices on the parallel axes where the position on the i-th axis corresponds to the i-th coordinate of the point. Traditional techniques search for abnormal patterns using signatures or behavioral analysis. The Picviz approach is excellent for analyzing multivariate data of logs and traces since anomalies simply stand-out. Picviz uses an intermediate language to define the different axes and types of data represented. Our experience with Picviz involved feeding it with network traffic datasets coming from our honeypots. Figure 5 presents an example where we used the tool to reveal the pattern of activity for the top 10 attackers during a one hour period for one of our sensors. The first axis is the timestamp of the activities. The second axis represents the source IP addresses. These hosts were attacked from several source ports (third axis) that targeted the full range of destination addresses (fourth axis) and a wide range of destination ports (last axis).

C. Infrastructure Monitoring

Static sensor monitoring. To monitor the availability of NoAH sensors, a web page that shows the sensors drawn on a Google map was constructed. A screenshot can be seen in Figure 6. Each NoAH logo represents a sensor. On the left of the map, a list of the sensors is displayed. Next to the name of each sensor there is a status icon. A green tick means that the sensor is up and displays statistics. A red x mark icon means that the server is either down or not accessible. The status of each sensor is inspected in the background by an external script. Furthermore, the script is linked to an alerting module that automatically sends a notification e-mail when a sensor is down for more than 2 hours. All sensor monitoring pages are password protected and accessible by few IP addresses over SSL. This security measure ensures confidentiality for sensor owners.

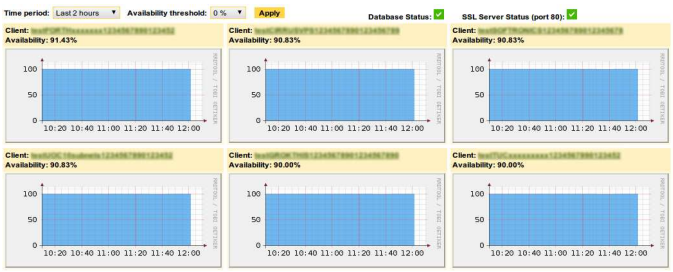

Figure 7: Uptime graphs for Honey@home clients

Honey@home monitoring. A deployment challenge is the efficient and scalable monitoring of Honey@home clients. Honey@home clients are designed to send a heartbeat pulse to the NoAH core every minute as an indication of their uptime status. Using the heartbeat information, we know which clients are currently online and how long they have been. We reconstruct the heartbeat information as SNMP-like graphs which are published in a private web page. Figure 7 is a screenshot of the uptime graphs. We have two parameters to customize the displayed graphs. The first one is the monitoring period. Changing this period, we can see which clients were active for at least one minute in a time period ranging up to one year. The second one is the availability threshold. By default this threshold is 0%, meaning that clients that send at least one heartbeat are displayed. Setting this threshold to 100% only the clients that were up for the whole monitoring period are displayed. The graphs are comprised of two parts. The first one is the client's key. As the monitoring page is available only to NoAH administrators, there is no need to anonymize the client keys. Clicking on the client key the user is redirected to a page that displays traffic statistics for the specific client. Additionally to these statistics, the attack conversations and malware instances captured by the client are displayed. The second part is the SNMP-like graph that displays the availability of the client as defined by the two parameters described before. The graph is also clickable, redirecting the user to a page containing the availability graphs for all

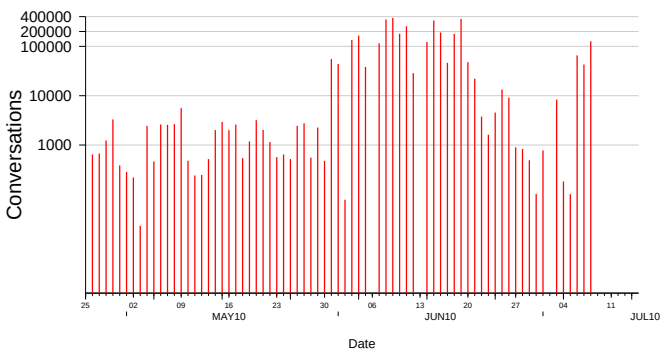

Figure 8: Conversations with attackers.

Country	# Conversations
Russia	1,459,733
Taiwan	351,527
USA	336,098
Spain	191,313
Japan	89,838
Puerto Rico	85,980
Norway	77,949
China	66,533
France	55,551
Turkey	51,780

Table I: Top 10 source countries of attackers that targeted NoAH

monitoring periods. In that way, we can observe the behavior of the client for different time periods.

V. Traffic Analysis

Here we provide an overview of the statistics collected by the NoAH infrastructure during a three month period.

A. Attack statistics

The distribution of the conversations handled by the NoAH honeypots is shown in Figure 8. With the term *conversations*, we refer to connections that were established with the honeypots and sent application data that could be characterized as exploitation attempts. During a 3 month deployment period, from the middle of April to the beginning of July 2010, the infrastructure handled a total of 3,385,518 conversations with attackers that targeted the NoAH sensors. The maximum number of conversations handled in one day was 377,071 which occurred on the 9th of July.

Table I presents the top 10 source countries of attackers that initiated conversations with the NoAH sensors. The aggregated conversations from these 10 countries amount to 81.7% of the total conversations handled by our honeypots. Results show that Russia is the country that initiated the largest number of attacks against the NoAH sensors reaching almost 1.5 million attacks. Taiwan and USA follow with over 300 hundred thousand attacks. China ranks 8th which is much lower than expected and quite surprising.

IP Address	# Conversations	Country	Days
xx.xx.222.18	103,692	Russia	2
xx.xx.10.241	85,564	Russia	1
xx.xx.95.27	78,084	Taiwan	1
xx.xx.35.24	76,506	Norway	5
xx.xx.63.50	62,962	Japan	1
xx.xx.209.3	61,178	USA	4
xx.xx.136.78	55,353	France	2
xx.xx.153.35	53,601	Puerto Rico	1
xx.xx.58.93	51,969	Russia	2
xx.xx.86.28	50,169	Russia	2

Table II: Top 10 attackers that targeted the NoAH sensors.

In Table II we can see the statistics of the top attackers for the whole duration of the three month deployment period. It is interesting to note that all top attackers were active for only a few days, and almost always those days were consecutive. Port 445 was the most targeted port receiving over 3 million conversations initiated by attackers. The second most popular port is port 139 which attracted over 60 thousand conversations. We can identify another interesting port. Our infrastructure collected over 40 thousand attacks targeting port 2967 and exploiting a vulnerability in the Symantec anti-virus client.

B. A case study for TCP port 445: Diurnal Patterns in the Attack Traffic

Over the last year, we noticed an increase of almost two orders of magnitude in the suspicious inbound traffic aimed at TCP port 445 and captured by our honeypots. This port is reserved by the Microsoft Windows file sharing service.

As we began to further investigate the characteristics of the attack, we discovered that it followed a very distinctive diurnal pattern. Traffic volume increased during the morning, peaked at noon and adopted a descending rate towards the afternoon. This pattern osculated with human behavior and business hours. A straightforward explanation for this is that people turn on their computers in the morning, when they go to work, and turn them off late in the afternoon, when they go home, and some a little later at night, when they go to sleep. Therefore, if those computers are infected with some form of malware trying to spread or launch attacks as part of a botnet, they have a very specific power cycle during which they have to perform their evil deeds. Especially in business environments where the network and computer configuration, both hardware and software is characterized by homogeneity, a malware-hosting vulnerability usually means a very high infection rate among the population. As a result, there are groups of infected computers all working the same hours and exhibiting the same behavior.

Figure 9 depicts the incoming traffic volume for port TCP 445 during the timeframe of a week. It makes a separation between the volume of incoming packets and incoming unique source IP addresses. It is clearly shown that there is a direct correlation between the two. To further clarify

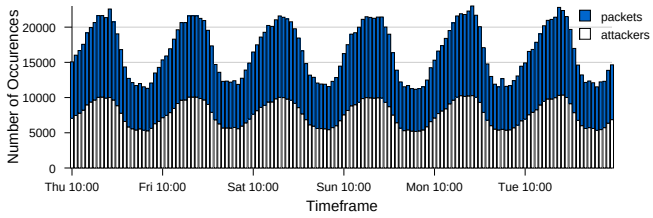

Figure 9: Incoming traffic to TCP port 445 in terms of packets and unique source IP addresses. Both metrics follow a diurnal cycle.

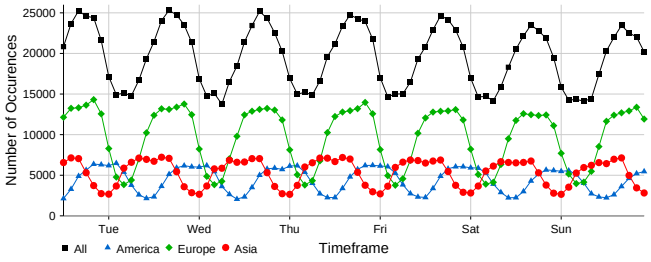

Figure 10: Attackers on port 445 divided into three groups based on the continent they reside. Each group follows a diurnal pattern.

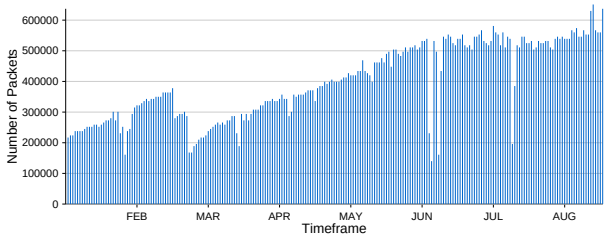

Figure 11: Incoming traffic to TCP port 445 for six months time.

1 day		1 week		1 month	
Port	Count	Port	Count	Port	Count
445	922.393	445	7.546.987	445	21.016.775
1433	7.320	1434	78.355	1434	321.850
1434	7.270	1433	46.110	1433	141.090
80	5.425	80	36.059	135	96.215
135	3.904	135	35.818	80	90.799
1025	2.307	139	24.994	3072	71.569
1024	2.245	1024	24.853	1024	70.340
3072	2.179	3072	24.792	139	53.706
1080	1.968	53	17.529	5900	49.502
8080	1.879	137	16.631	22	44.412

Table III: Top 10 destination ports for a day, week & month period.

that, we subsequently tried to discover the subnet or group of subnets that were flooding us with the largest volume of traffic. It turned out that there was a near even distribution of incoming packets among all attacking subnets. Additionally, we looked closer at the top 10 /8 attacking subnets in terms of traffic volume and discovered that each one followed an attack cycle with a 24 hour period, while maintaining a difference in phase compared to the others.

Having concluded that there was no single subnet responsible for this diurnal phenomenon, but rather that it was a combined effort, we tried to further analyze the characteristics of the traffic and attribute incoming packets to their respective time zones. To do that, we once again employed MaxMind's GeoIP tool which provided a mapping between IP addresses and their corresponding countries. That way, instead of plotting the entire volume against a single, specific time zone, we produced three different graphs, each one aligned with the dominant time zone of America, Europe and Asia respectively. Using that method, we indeed confirmed that attackers from different timezones followed their own diurnal pattern instead of a uniform one. Nevertheless, the three sine-like curves when merged compiled the single plot we initially produced. The results are shown in Figure 10.

Another interesting aspect that we revealed was that while the number of attacking sources followed a steady diurnal cycle across a week, so did the number of newcomers; source IP addresses we had never seen before. This means that the attacking population decreases towards nightfall, because people turn off their computers, but the next morning, a stable portion of the attackers that wake up has never attacked us before. We grouped attackers in /24 subnets and thereby eliminated more than 50% of the newcomers. This action confirmed our hypothesis that malicious computers powering on each day have their network addresses updated via DHCP and for that matter appear as different sources. However, a population of entirely new subnets, appearing each day, remained.

C. Traffic Volume & Distribution Trends

An interesting fact is that there has been a steady 5% increase in attack traffic since we started noticing the activity on port TCP 445 and for a period of 6 months. Up until this very moment there is no evidence that the increasing trend has ceased. Figure 11 visualizes this trend. Let there be noted that this trend also holds for the number of IP addresses and subnets observed as well. Furthermore, there exists no single source responsible for this increase in volume.

Regarding the distribution of traffic, it is evenly distributed among different subnets. A very interesting fact is that 99.8% of the traffic we receive targets hosts that belong to the first half of each subnets' IP address range. Furthermore, traffic is also evenly distributed amongst these IP addresses.

Additionally, the majority of ports under attack resides in the 0 - 1023 range. Such ports are used to officially host well-know services such as telnet, SSH, HTTP and a series of Windows services. Of course there is some meaning to this, considering it an optimization tactic to avoid attacking the unmapped upper ranges of the port spectrum. Table III displays the top 10 ports over a day, a week and a month period. The most prevalent port is 445, followed by ports 1434, 1433, 80, 135, 3072 and 1024. This means that

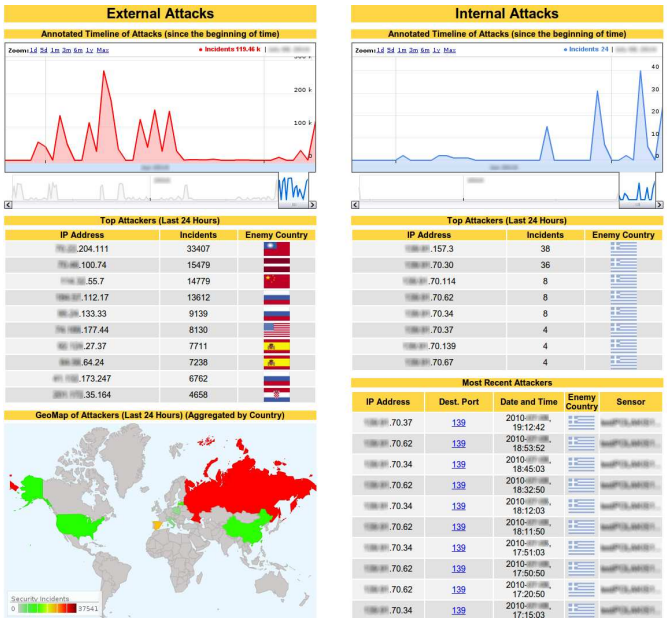

Figure 12: The new NoAH dashboard

70% of the top ports remain unchanged even in different timeframes.

VI. Future Work

We are currently extending the NoAH infrastructure to monitor the IPv6 address space. Additionally, our goal is to continue installing more sensors, so as to have a broader view of the attack landscape. Moreover, we are investigating novel ways of data visualization. Finally, we are looking into the use of a personalized dashboard for various networks and organizations. That dashboard provides an aggregated view of the security incidents inside and outside an organization's network. We aim to be able to answer questions such as: "Which attacks are coming from inside a network?", "Which attackers are simultaneously attacking various parts of a network or a group of networks?". Figure 12 provides a conceptual view of the dashboard.

VII. Summary and Concluding Remarks

NoAH is a distributed architecture that enables farms of honeypots to cooperate. Because of the large volume of traffic received by the NoAH infrastructure, there is need for aggregating and visualizing the most interesting statistics and trends. In this paper we presented several visualization techniques that enable users to observe what happens in their unused address space over arbitrary time periods and permits them to find answers to questions like "What is the most attacked port", "Which attacker is the most aggressive?", "Which part of the monitored space is attacked the most?". All statistics are displayed through a dynamic web site that utilizes well known packages for data visualization. Additionally, we provide the necessary tools to administrators for monitoring the status of a honeypot infrastructure. We also presented a preliminary traffic analysis for data collected over a six month period. 70% of the top attacked ports remain the same independently of the timescale chosen, while port 445 is by far the most attacked port. Specifically for port 445, we observed a diurnal pattern. Our geolocation analysis revealed that attackers from different continents all present a diurnal pattern and the composition of their patterns still forms a diurnal cycle.

Acknowledgments

This work was supported in part by the projects NoAH and SysSec, funded in part by the European Commission, under contract number 011923 and Grant Agreement Number 257007 respectively. We thank the anonymous reviewers for their valuable comments. Georgios Kontaxis, Iasonas Polakis and Evangelos P. Markatos are also with the University of Crete.

References

[1] D. Moore, G. Voelker, and S. Savage, "Inferring Internet Denial-of-Service Activity," in *Proceedings of the* 10^{th} *USENIX Security Symposium*, August 2001, pp. 9–22.

[2] C. Taylor, "Attack of the world wide worms," *TIME*, 2003.

[3] N. Weaver, "Warhol worms: The potential for very fast internet plagues," *Technical Report*, 2002.

[4] X. Jiang and D. Xu, "Collapsar: A VM-Based Architecture for Network Attack Detention Center," in *Proceedings of the* 13^{th} *USENIX Security Symposium*, August 2004, pp. 15–28.

[5] "Leurre.com honeypot project," *http://www.leurrecom.org/* .

[6] "The Honeynet Project," 2003, http://www.honeynet.org/.

[7] "Sebek homepage," *http://www.honeynet.org/tools/sebek/*.

[8] "Pkviz," http://sintixerr.wordpress.com/pkviz-packet-visualizer-and-animator.

[9] "circos," http://mkweb.bcgsc.ca/circos/.

[10] "Cuttlefish," http://www.caida.org/tools/visualization/cuttlefish/.

[11] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The Second-Generation Onion Router," in *Proceedings of the* 13^{th} *Usenix Security Symposium*, Aug. 2004.

[12] "Amun honeypot," http://amunhoney.sourceforge.net/.

[13] G. Portokalidis, A. Slowinska, and H. Bos, "Argos: an Emulator for Fingerprinting Zero-Day Attacks," in *Proceedings of ACM SIGOPS Eurosys 2006*, April 2006.

[14] S. McCanne, C. Leres, and V. Jacobson, "Libpcap," 2006, http://www.tcpdump.org/.

[15] "Picviz," http://wallinfire.net/picviz/.