SEVENTH FRAMEWORK PROGRAMME

Information & Communication Technologies
Trustworthy ICT

NETWORK OF EXCELLENCE

A European Network of Excellence in Managing Threats and Vulnerabilities in the Future Internet: *Europe for the World* [†]

Deliverable D5.3: Case Study: Malicious Activity in the Turkish Network

Abstract: This deliverable describes a case study on malicious activity in the TUBITAK (The Scientific and Technological Research Council of Turkey) Network, how the case study was performed and which results were collected.

Contractual Date of Delivery	November 2012
Actual Date of Delivery	February 2013
Deliverable Dissemination Level	Public
Editor	Necati Ersen Siseci, Bâkır Emre, Huseyin Tirli
Contributors	TUBITAK
Quality Assurance	Magnus Almgren, Martina Lindorfer

The *SysSec* consortium consists of:

FORTH-ICS	Coordinator	Greece
Politecnico Di Milano	Principal Contractor	Italy
Vrije Universiteit Amsterdam	Principal Contractor	The Netherlands
Institut Eurécom	Principal Contractor	France
IICT-BAS	Principal Contractor	Bulgaria
Technical University of Vienna	Principal Contractor	Austria
Chalmers University	Principal Contractor	Sweden
TUBITAK-BILGEM	Principal Contractor	Turkey

[†] The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 257007.

Contents

List of Figures

List of Tables

1 Introduction

The Internet has become an integral part of our lives. We can now handle our daily routines much more quickly and efficiently. With the Internet having such a prominent role in our lives, information security concepts were also transformed. Many attacks on personal privacy and on the integrity of critical infrastructures now originate from the Internet.

Security solutions such as firewalls, anti-virus (AV) scanners, IPS and IDS systems work well in detecting and stopping already known attacks. However, for the discovery and analysis of 0-day attacks, that are not well defined or known, decoy systems, often called honeypots are needed. These honeypots are deployed to understand the attacks used prior to the exploitation of productive systems in order to take preemptive countermeasures. Thus, the usage of honeypots and honeynets enables pro-active defense capabilities against cyber-attacks towards systems and institutions.

Honeypot systems create simulated environment that attract attacks by imitating a service, operating system or network. Honeypots are generally classified into three types as low, medium or high interaction honeypots according to their abilities. They can also be in client or server roles.

To be able to effectively analyze the threats against an organization and to take appropriate countermeasures, it is necessary to analyze the data collected from its honeypot systems, IDS/IPS alarms, IP traffic info, network flows and DNS queries. The information collected from the individual sources becomes more meaningful when correlated together. This ensures the proper identification of the target, content and the scope of an attack, which is necessary for the development of proper countermeasures.

The TGS has an architecture that can centrally analyze, detect and classify the threats and malware received at distributed networks. This deliverable introduces a case study on malicious activity in the TUBITAK network and provides the results that we collected with the TGS during a three-month observation period from September to November 2012.

The improvement in this report is in the way that we collect the malwares. The analysis is classical static analysis.

This report is organized as follows: In the first part we introduce the different types of honeypots and related work. In the second and subsequent parts we elaborate on our own approach and results.

2

Related Work

Honeypots are one of the architectures used to detect attacks on information systems. They are designed to attract hacker activity and malware attacks such as worms and viruses in the networks they are deployed in. Since honeypots use unannounced (non-routable) IP addresses, any traffic they receive is considered suspect. Furthermore, honeypots operate in an isolated fashion from their networks, their being compromised by attacks does not generally compromise the security of their institutions.

The main feature of a honeypot is to collect attack records and malware samples by imitating network services, real operating systems or networks. These services or operating systems contain specific vulnerabilities and thus the collected data helps in the determination of the attackers and their methods to exploit these specific vulnerabilities.

Depending on their abilities, honeypots are classified into three categories: low, medium and high interaction honeypots.

Low-interaction Honeypots

Low-interaction honeypots consist of a network service, operating system or software that emulates a whole network. The Honeyd [11] application is a good example of a low-interaction honeypot. It attracts the attackers by imitating the network services such as an SMTP Server, IIS and Apache. Its installation, configuration and maintenance are relatively straightforward. However, since the services offered are imitated and do not contain vulnerabilities, detailed information about the attacks on Honeyd cannot be obtained. The collected information is limited to statistical data such as the most targeted services, ports and IP addresses. Modules that produce different outputs (such as HP Procurve 2848 Switch, IIS 4.0 on Windows 2000 etc.) can be added to the system by using scripts.

Mid-interaction Honeypots

Mid-interaction honeypots either provide an isolated operating system (e.g. FreeBSD Jails [9]) or work like a low-interaction honeypot and also try to interact with malware (e.g. nepenthes [4]). The attacker interacts with the operating system in the jail rather than the main operating system on the honeypot. However, the main handicap of these systems is the exposure to attack of the main operating system if vulnerabilities exist within the jail system.

High-interaction Honeypots

High-interaction honeypots offer network services on real operating systems as real services that also include vulnerabilities. They thus offer more opportunities for in-depth attack analysis. Even though high-interaction honeypots allow for a detailed analysis, their setup usually is complex and their management, maintenance and reuse after malware clean-up is challenging. As an example, the architecture of the Honeynet Project [12], includes a Honeywall gateway for recording traffic arriving at the network, a high-interaction honeypot called Sebek as well as a a low-interaction honeypot such as Honeyd behind it. The honeypots themselves are allowed limited Internet access.

Another example, the NoAH Project [7] is comprised of three components: NoAH Core includes Honeyd (low-interaction honeypot) and Argos [6] (high-interaction honeypot) as well as servers that analyze attacks.

Honey@home [10] is designed to analyze attacks targeting home users. Users install the Honey@home client on their PCs. This client receives an IP address from DHCP and forwards the attack traffic to this IP address towards the honeypots on the NoAH core servers for analysis. To ensure the privacy of the users this traffic is forwarded to the NoAH core via TOR servers. A more sophisticated version of Honey@home redirects the traffic arriving at the unused IP ranges of organizations to the NoAH servers through tunneling/funneling.

Canto et al. [20] offers three main lessons that they have learned. First one is creating a represantative malware colletion. Second is false-negatives. It is an error that an antivirus used in the system does not recognize a malware recognized by other antiviruses. Third one is false-positives. Tuning scanners heuristic parameters may lead false-positives.

Jiang et al. [18] proposes a virtual machine-based architecture for network attacks. The idea behind the proposal is based on decentralized architecture composed of a large number of high interaction honeypots deployed in different networks. Collapser has three different components: the redirector, the front-end, and virtual honeypots. The traffic redirectors, located in different networks, redirects traffic via GRE tunnels to front-end of the Collapsar center. The second part is the front-end of the Collapsar center. And third part is virtual honeypots. Collapsar center contains lots of virtual honeypots which runs extended version of User-Mode Linux.

3 TGS (Threat Observation System) Architecture

The Threat Observation System Core (TGSM) we realized consists of two major components: The *Honeypot network* and the *Management Network* where the attacks are analyzed. The honeypot sensors that are distributed to different networks are controlled by a central entity that receives and analyzes the attacks observed by the honeypot sensors and produces a threat analysis report.

Overall TGSM consists of the following components:

- The honeypot sensors in different networks that forward attacker network traffic to the central processing core.
- Virtualization environment that contains the high-interaction honeypot sensors.
- IDS that generates alarms from the attack traffic received by the high-interaction honeypots.
- A web interface where the attacks can be visualized.
- A module for remote cleaning of malicious binary files received at high-interaction honeypots.
- A server for analyzing spam e-mails and virtual operating systems on the virtualization domain.
- A file server for storing potentially malicious binaries.
- A malicious software scanning system.

The sensors used to forward traffic towards the TGSM have a customizable architecture [5]. Their main objective is to forward the attack traffic received at their IP address to the honeypots implemented in the VirtualBox virtualization solution in the TGSM. Since each sensor has a transparent architecture it can represent the virtual system as if it were working on the network it is installed on. The operating system that is working virtually at the TGSM is responding to all the attacks, port scans and operating system scans received at the local IP address of the sensor. Detailed information about the operation of the sensors is given in the following sections.

Each sensor communicates with the TGSM through a secure channel due to the sensitivity of the data handled. For this purpose, OpenVPN, with its SSL/VPN capabilities, is deployed in the system. The general topology is shown in the Figure 3.1.

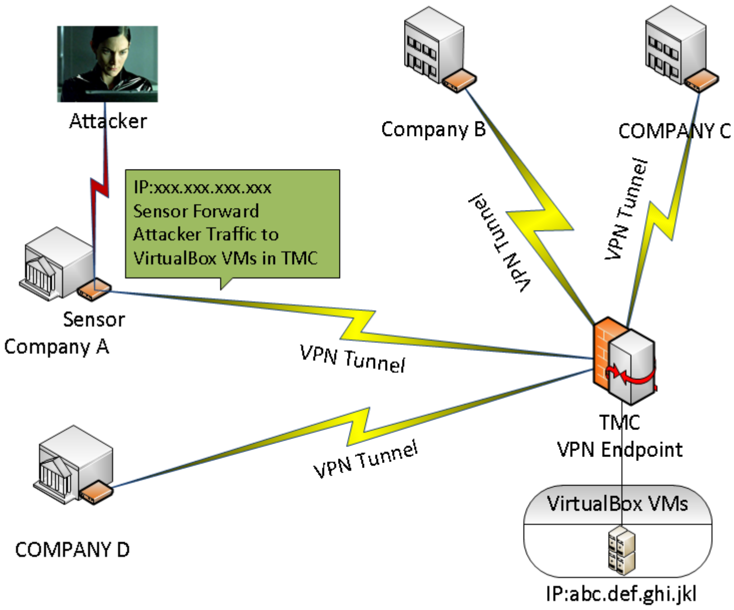

Figure 3.1: Topology of sensors forwarding attacker traffic to the honeypots on the VirtualBox in the TGSM.

The sensors operate in a plug-and-play fashion and can be deployed on a network without changing its topology. Depending on their intended purpose sensors can be placed before or after the firewall of an organization. It is sufficient to define a single real IP address for the sensor to operate properly. For the operating system of the sensors we chose FreeBSD. The hardware properties of the sensor are defined in the Table 3.1.

All honeypots are implemented in virtual machines making their configuration very flexible.

	
CPU	500 MHz AMD Geode LX800
DRAM	256 MB DDR DRAM
Storage	CompactFlash socket, 44 pin IDE header
Connectivity	3 Ethernet channels (Via VT6105M 10/100)
I/O	DB9 serial port, dual USB port
Board size	6 x 6” (152.4 x 152.4 mm)

Table 3.1: Hardware properties of sensors.

4 Sensor Usage

As mentioned in the previous section, attacker network traffic is forwarded to the honeypots on the VirtualBox in the TGSM by the sensors. The local IP address of a sensor is transferred transparently to the honeypot in the virtualization domain. This transparency can be explained as follows: Let's assume that the sensor in organization A has an IP address `xxx.yyy.zzz.ttt` and the corresponding honeypot has a Windows XP operating system. The network gateway of the Windows XP honeypot is assigned as the IP address of the sensor in organization A. The attacker trying to connect to the sensor's IP address `xxx.yyy.zzz.ttt` will in fact connect to the honeypot system running the Windows XP OS in the Virtualization environment. If the attacker compromises Windows XP and would connect to another node on the Internet via the compromised Windows XP system, its IP address would be observed as `xxx.yyy.zzz.ttt`. Hence, the IP address of the honeypot would be hidden through this architecture.

For the network traffic forwarding towards the TGSM and for IP address transitions we use OpenVPN [15], OpenBSD [13] and Packet Filter [14]. Any communication between a virtual machine configured for one organization and other virtual machines, as well as other network elements such as IDS or firewall systems is prevented by the rules of the firewall policy. Figure 4.1 shows the physical connection between an attacker and the Windows XP system, Figure 4.2 shows the corresponding logical connection.

The attacker supposes the sensor machine as if the system is a Windows XP behind a firewall performing NAT.

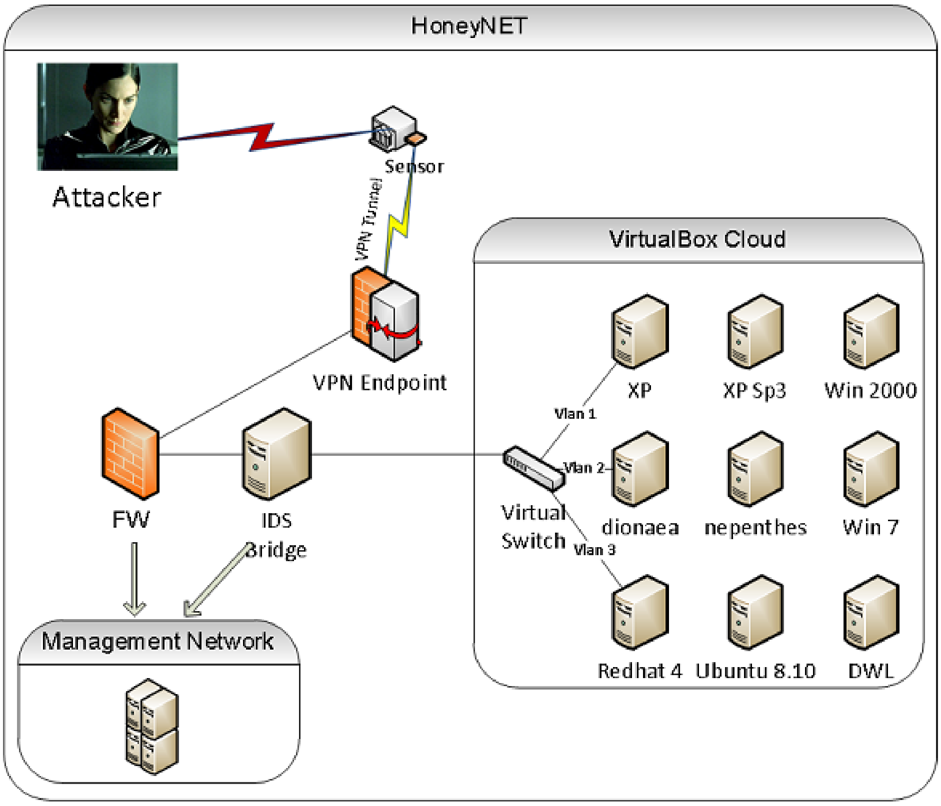

Figure 4.1: Honeypot network in virtualization environment and management network.

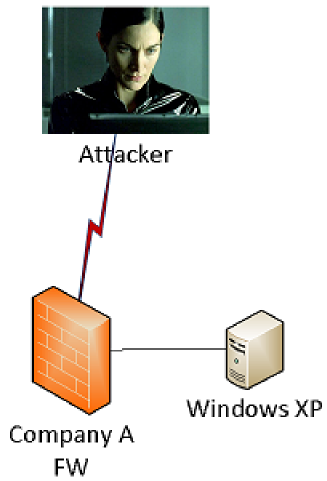

Figure 4.2: Logical topology of honeypot.

5

The Honeypot Network and Data Processing Servers

An Oracle VirtualBox virtualization system hosts the high-interaction honeypots of our architecture. The system consists of multiple high-interaction honeypots each residing on a separate VLAN. Each virtual OS on this network corresponds to a sensor in a different organization. Each VM is placed on a separate VLAN for isolation and for ease of monitoring of their traffic as well as to block direct communication among the VMs. The traffic that reaches the honeypots is also routed to the passive IDS via the firewall.

When malware is detected on one of the high-interaction honeypot VMs, the status of this VM is recorded and it is transferred to the file server in the malware analysis center. In addition, suspect network activity is stored in the file server as PCAP files and subjected to further analysis. Malware infecting the VMs can be observed and recorded both via the IDS and the VMs themselves. After the malware is stored in the file server, it is scanned via the *Malware Scanning System*.

Data gathered from the analysis and data processing servers can be visualized by the web application. As illustrated in the samples screen in Figure 5.1, the web application provides the classification of attacks and the output of statistical information. Furthermore, the attacker IP addresses and their location information can be viewed graphically as shown in Figure 5.2.

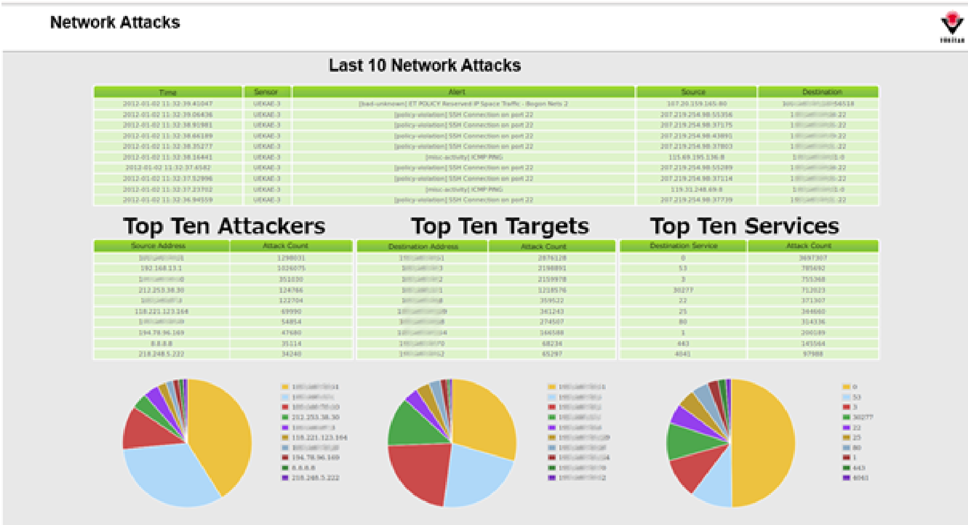

Figure 5.1: Network attack monitoring dashboard.

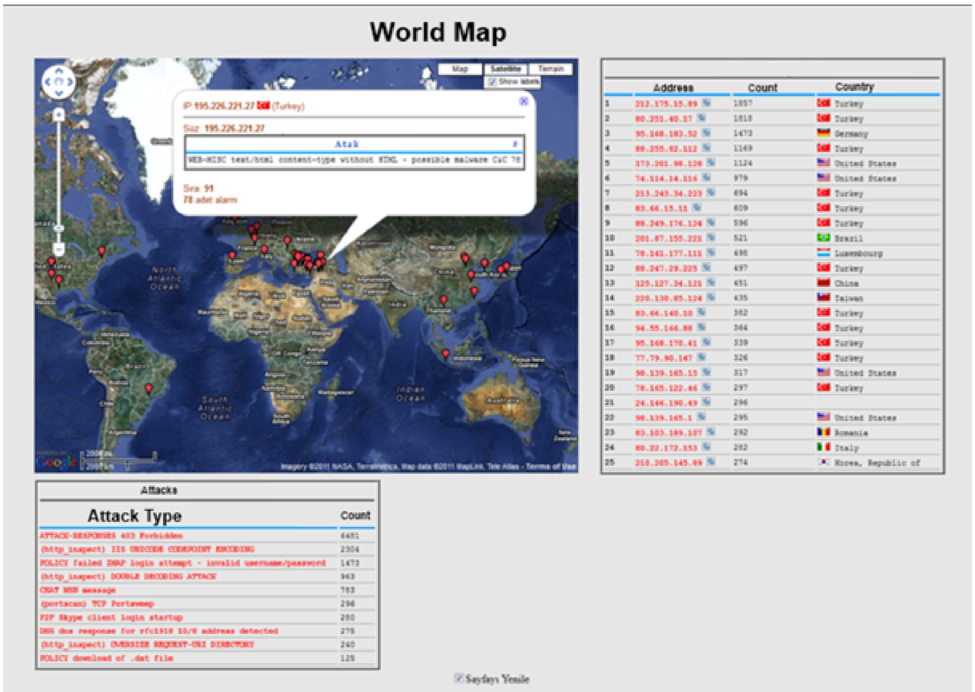

Figure 5.2: Statistics and geographical distribution of attacks.

6 Spam E-Mail Analysis

Some of the high-interaction honeypots use spam e-mails as the method to catch malware. Within the scope of spam e-mail analysis, e-mails received at selected domains (`tubitak.gov.tr` and `uekae.tubitak.gov.tr`) are classified and marked as spam e-mail by users or by a spam filter. The e-mails tagged as spam are then analyzed at the honeypot network data processing servers. This analysis involves two stages: the detection of all URLs contained in the spam e-mails and their recording in a database and the extraction of attachments like PDF, DOC and XLS documents as well as executables and their storage on a file server. Collected spam mail count and other statistics are detailed in section 7.

Spam URL Analysis

The URLs contained in the spam e-mails were visited using a sandbox environment comprising of Windows XP SP3 operating systems running on the VirtualBox domains that also host the HPs. During these visits, the web sites that included exploit kits or malware were allowed to infect the operating system. Since all activity including all web site visits and file downloads are recorded, any executable files that are downloaded by the OS are stored on the file server and subjected to further analysis by the Malware Scanning System.

Mail Attachment Analysis

All files extracted from the attachments of spam e-mails are also stored on the file server and are subsequently scanned by the Malware Scanning System.

7 Malware Scanning System

All files gathered from the individual components of our system during the September to November 2012 observation period were stored on the file server. These files include all malware targeting the HPs, all spam e-mail attachments and all files that were downloaded when visiting URLs contained in the spam e-mails. Table 7.1 lists the number of unique files observed from each source.

Sensor Type	Number
Extracted by Suricata IDS	36
Mid-Interaction Honeypots	63
PCAP Analysis	868
Spam Attachment	4,310
Spam URL	1,609
Total	**6,886**

Table 7.1: Number of unique files recorded.

Malware scanning software from different vendors has been installed on separate virtual machines to form a Malware Scanning System. All the files gathered on the server were appended to a queue and scanned by the different scanners. Table 7.2 lists the number of unique files that were scanned with the Malware Scanning System. Only files that have been scanned and identified during the last month have been included in the statistics to prevent discrepancies among the results of different anti-virus software that have been incorporated in the system at earlier stages.

Sensor Type	Number
Extracted by Suricata IDS	2
Manual Upload	190
Mid-Interaction Honeypots	57
PCAP Analysis	847
Spam Attachment	1,939
Spam URL	207
Total	**3,202**

Table 7.2: Number of unique files scanned.

From the 3202 samples we scanned, we detected a total number of 911 unique malware samples (listed in Table 7.3). Additionally to scanning them with anti-virus software, we uploaded more than 200 files that were determined as harmful to the Anubis dynamic malware analysis system [2].

Overall the majority of identified malware has been observed to spread through the file sharing service of Windows. Another popular infection vector we observed was the spam e-mails.

Sensor Type	Number
Extracted by Suricata IDS	2
Manual Upload	18
Mid-Interaction Honeypots	36
PCAP Analysis	842
Spam Attachment	3
Spam URL	10
Total	**911**

Table 7.3: Number of unique malware detected.

Spam E-Mail Statistics

One of the most common ways of spreading malware is through spam e-mails. We extracted a total number of 167,410 spam e-mails from the `tubitak.gov.tr` and `uekae.tubitak.gov.tr` domains during the three-month observation period. However, the mail gateways positioned before the mail servers used virus scanners and eliminated the majority of malware containing e-mails beforehand. Therefore, the amount of malware collected through this method is smaller than expected. E-mails that are not detected as malware by the gateway and those containing malware as attachments are included in our study.

Overall we extracted a total of 2,636,675 URLs from spam e-mails. We whitelisted common, well known URLs (such as shopping sites, social networks) and did not subject them to further analysis (see Table 7.4 for whitelisted URLs and keywords). The remaining 1,197,759 URLs not matched by keywords in our whitelist were visited and their screenshots were taken.

.jpg	www.directmarketingturkey.com
.jpeg	mail.ameriprise.com
www.w3.org	www.yemeksepeti.com
.gif	www.linkedin.com
.png	help.linkedin.com
mailto:	gmailsndr.com
@	.gittigidiyor.com
grupanya.com	.akbank.com
http://www.sehirfirsati.com	urunleritakipet.com
grupfoni.com	.sehirfirsati.com
.morhipo.com	www.pandora.com.tr
.trendyol.com	yakala.co
.markofoni.com	www.facebook.com
.markafoni.com	.subscribe.
www.gruppal.com	bultengonderi.com
s.gruppal.com	www.hayatimizfirsat.com
www.sanalmarketim.com	link.guncelfirsat.com
www.tubitak.gov.tr	www.sndr-server.com
www.ume.tubitak.gov.tr	www.promoskop.com
http://odeon	kacirmayiz.com
http://www.ekstrafiyat.com	.netvarium.com
http://www.bultenonline.com	.1v1y.com
http://www.tnksender.com	bulten.1v1y.com
http://mobile.twitter.com	crm.ikea.com.tr
http://twitter.com	www.altincicadde.com
http://www.youtube.com	bultenaltincicadde.com
www.railwdr.com	thejns.org
http://www.w3c.org	www.zt-server.com
www.vipdukkan.com	mailing.evim.net
info.vipdukkan.com	.perabulvari.com

Table 7.4: Whitelisted URLs and keywords.

The summary of statistics collected from the spam e-mails is listed in Table 7.5. From the total number of 13,552 unique files extracted from the spam e-mails, 246 files were detected as executable and 48 of these files were marked as malware. In addition, 1 file was detected by the TGSM as malware which was not identified as such by the mail gateways.

Captured screenshots	1,155,691
Executables downloaded from the URLs	246
Malware detected from e-mail attachment	1
Malware detected from the URLs	48
Number of URLs extracted from the e-mails	2,636,675
Total number of spam e-mail collected	167,410
Unique files extracted from e-mail attachments	13,552
URLs visited	1,197,759

Table 7.5: Statistics of spam analysis.

Anti-Virus Scan Statistics

All files stored on the file server during the three-month period were analyzed by the Malware Scanning System. The anti-virus softwares used in the system have been updating their signature databases daily but not all anti-virus softwares were deployed at the same time. Thus, there are some major discrepancies between the results of the anti-virus softwares used. Consequently, we only include files that have been scanned and identified during the last month in our statistics in order to prevent discrepancies among the results of different anti-virus softwares that have been incorporated in the system at earlier stages. The daily distribution of the files scanned and the number of files detected as malware by at least one anti-virus software is illustrated in Figure 7.1.

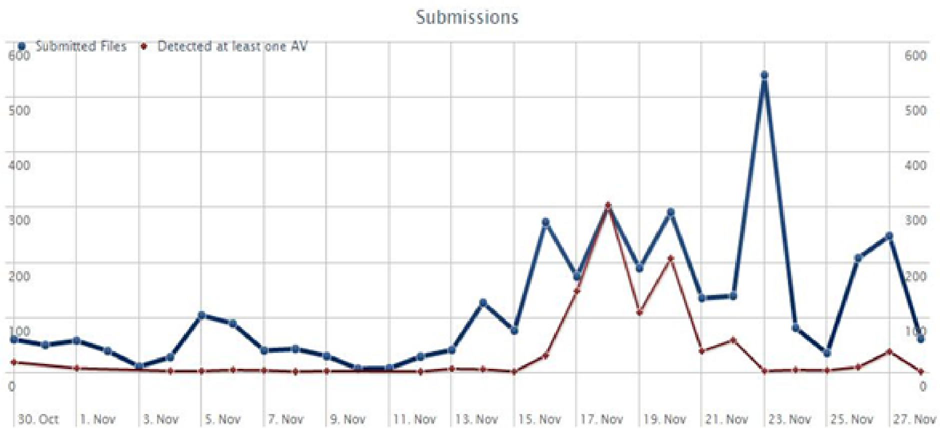

Figure 7.1: Distribution of submitted files.

Table 7.6 lists the number and percentage of malware detected by the various anti-virus software in November 2012.

AV	Number & Percentage
Vendor1	814 ~89%
Vendor2	798 ~87%
Vendor3	721 ~79%
Vendor4	656 ~72%
Vendor5	655 ~72%
Vendor6	612 ~67%
Vendor7	598 ~65%

Table 7.6: AV detection ratios

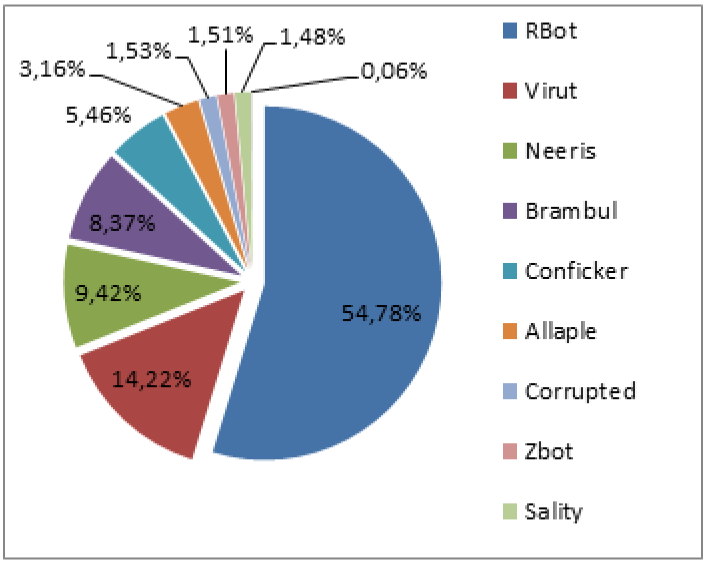

Figure 7.2: Distribution of malware families.

The distribution of the individual malware families identified by the anti-virus scanners are shown in Figure 7.2. The most detected malware families were:

- **RBot ~54%**
- **Virut ~14%**
- **Neeris ~9%**

Other popular malware families detected were Conficker, Brambul, Allaple, Sality, Zbot (Zeus), Koblu and Symbian YXE. Some of the files recorded via the IDS system or received via spam e-mail have been seen to be corrupt. Overall, the percentage of files that have not been scannable by the anti-virus software was around 1,5%. The total list of the malware families identified above and their detection numbers are given in Table 7.7.

Family	**Number**
RBot	2546
Virut	661
Neeris	438
Brambul	389
Conficker	254
Allaple	147
Corrupted	71
Zbot	70
Sality	69
AdWare	59
Trojan Dropper	34
PDF Exploit	6
Trojan Downloader	5
Symbian YXE	3
Koblu	3
Trojan Clicker	2
Total	**4757**

Table 7.7: Number of samples detected for each malware family.

The following Tables 7.8 to 7.13 list the number of different variants and their detection numbers for samples from the rBot, Virut, Conficker, Allaple, Neeris and Brambul malware families.

rBOT		
Vendor3	Backdoor:Win32/Rbot	402
Vendor2	Win32:Rbot-GKN [Trj]	306
Vendor7	Win32/Rbot trojan	299
Vendor4	Trojan horse Generic_r.QP	252
Vendor5	Trojan.Mybot-5073	217
Vendor6	Backdoor.Win32.Rbot.bqj	168
Vendor2	Win32:Neptunia-ACS [Trj]	147
Vendor6	Net-Worm.Win32.Kolab.aefe	131
Vendor1	Worm/Rbot.246784.1	130
Vendor1	BDS/Rbot.A.366	117
Vendor5	Trojan.Mybot-10186	116
Vendor1	Worm/Rbot.246784.17	100
Vendor2	Worm/Rbot.268288.3	61
Vendor2	Win32:Rbot-DQS [Trj]	38
Vendor6	Backdoor.Win32.Rbot.adqd	31
Vendor6	Backdoor.Win32.Rbot.bni	30
Vendor1	Worm/Rbot.50176.5	1
Total		**2,546**

Table 7.8: rBot variants

Virut		
Vendor1	W32/Virut.Gen	87
Vendor1	W32/Virut.AX	62
Vendor4	Win32/Virut	62
Vendor2	Win32:Virtob	53
Vendor7	Win32/Virut.AV virus	50
Vendor5	W32.Virut.ci	31
Vendor7	Win32/Virut.NBP virus	31
Vendor3	Virus:Win32/Virut.AK	31
Vendor7	Win32/Virut.E virus	31
Vendor2	Win32:Virut	31
Vendor6	Virus.Win32.Virut.av	26
Vendor5	W32.Virut-17	26
Vendor3	Virus:Win32/Virut.BN	24
Vendor6	Virus.Win32.Virut.ce	24
Vendor3	Virus:Win32/Virut.AC	24
Vendor2	Win32:Vitro	24
Vendor4	Win32/Virut.dropper	24
Vendor1	W32/Virut.CEE	6
Vendor6	Virus.Win32.Virut.n	5
Vendor5	W32.Virut-54	5
Vendor7	Win32/Virut.AT virus	1
Vendor6	Virus.Win32.Virut.at	1
Vendor3	Virus:Win32/Virut.AA	1
Vendor1	W32/Virut.AT	1
Total		**661**

Table 7.9: Virut variants

Conficker		
Vendor4	Virusidentified Worm/Downadup	43
Vendor1	Worm/Conficker.gen	33
Vendor5	Worm.Kido-223	29
Vendor2	Win32:Confi [Wrm]	29
Vendor7	Win32/Conficker.AA worm	27
Vendor1	Win32/AutoRun.IRCBot.DI	27
Vendor6	Net-Worm.Win32.Kido.ih	27
Vendor3	Worm:Win32/Conficker.B	27
Vendor7	Win32/Conficker.Gen	2
Vendor6	Trojan-Downloader.Win32.Kido.bj	2
Vendor3	Worm:Win32/Conficker.gen!B	2
Vendor3	Worm:Win32/Conficker.D	1
Vendor5	Worm.Downadup-424	1
Vendor1	Worm/Conficker.B.5	1
Vendor7	Win32/Conficker.X worm	1
Vendor1	Worm/Conficker.D.2	1
Vendor6	Trojan-Downloader.Win32.Kido.a	1
Total		**254**

Table 7.10: Conficker variants

Allaple		
Vendor7	Win32/Allaple worm	35
Vendor5	Worm.Allaple-306	33
Vendor5	Worm.Allaple-2	33
Vendor3	Worm:Win32/Allaple.L	33
Vendor3	Worm:Win32/Allaple.A	2
Vendor6	Net-Worm.Win32.Allaple.d	2
Vendor2	Win32:Allaple [Wrm]	2
Vendor6	Net-Worm.Win32.Allaple.e	2
Vendor1	WORM/Allaple.Gen	2
Vendor2	Win32:Allaple-YF [Wrm]	1
Vendor5	Worm.Allaple-45	1
Vendor5	Worm.Allaple-199	1
Total		**147**

Table 7.11: Allaple variants

Neeris		
Vendor5	Trojan.IRCBot-3550	82
Vendor6	Backdoor.Win32.IRCBot.gxj	80
Vendor2	Win32:IRCBot-DMB [Trj]	72
Vendor1	BDS/Bot.94407.91	56
Vendor3	Worm:Win32/Neeris.gen!C	54
Vendor3	Worm:Win32/Neeris.AN	31
Vendor4	Worm/AutoRun.IN	31
Vendor7	Win32/AutoRun.IRCBot.FC	27
Vendor2	Win32:Neeris-B [Wrm]	5
Total		**438**

Table 7.12: Neeris variants

Brambul		
Vendor3	Trojan:Win32/Brambul.A	72
Vendor2	Win32:Agent-AOKX [Trj]	69
Vendor1	TR/Agent.mtv	66
Vendor6	Trojan-Spy.Win32.Agent.bmxb	63
Vendor4	Trojan horse PSW.Agent.AHCN	62
Vendor7	Win32/Pepex.E worm	32
Vendor7	Win32/Pepex.F worm	25
Total		**438**

Table 7.13: Brambul variants

8 Conclusion

In this deliverable we presented a case study on malicious activity in the TUBITAK Network that was performed using our Threat Observation System (TGS) during a three-month observation period from September to November 2012.

The honeypot sensors in the different networks correspond to the high-interaction honeypots on the virtual machines in the TGS core (TGSM). The attacks received at those honeypots were analyzed, the files obtained were stored and scanned by a malware scanning system. The system classified the identified malware into families and issued threat reports for each attack.

As our case study shows, the TGSM architecture has been shown to be successful in identifying new malware or new variants of existing malware families which have not been identified by existing anti-virus software and which exploit system vulnerabilities such as MS08-067. For example Figures 8.1 shows the analysis results from the VirusTotal [17] web site for a malware captured and detected by our system on 20/11/2012, that was not detected by any of the 43 anti-virus scanners used by VirusTotal.

Furthermore, our case study showed that the collected malware does not only target conventional PCs. Through the analysis of spam e-mail, malware targeting mobile devices such as Symbian-YXE has also been detected and collected by our system. We collected viruses like Koblu, Symbian-YXE etc. little known to us

Even if malware can be detected by anti-virus software, we often encountered malware that is updated up to 3 times a day. When we consider the time and effort it takes to detect and analyze such malware and subsequently produce signatures for anti-virus software and distribute it worldwide, the importance of more sophisticated malware detection techniques becomes apparent.

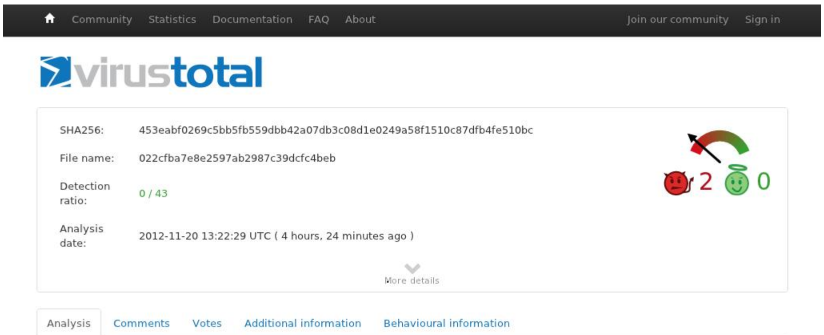

Figure 8.1: Malware that was detected by the TGSM but was not caught by the 43 anti-virus scanners (part I).

Antivirus	Result	Update
Agnitum	-	20121118
AhnLab-V3	-	20121118
AntiVir	-	20121119
Antiy-AVL	-	20121118
Avast	-	20121119
AVG	-	20121119
BitDefender	-	20121119
ByteHero	-	20121116
CAT-QuickHeal	-	20121119
ClamAV	-	20121119
Commtouch	-	20121119
Comodo	-	20121119
DrWeb	-	20121119
Emsisoft	-	20121119
eSafe	-	20121115
ESET-NOD32	-	20121119
F-Prot	-	20121119
F-Secure	-	20121119
Fortinet	-	20121119
GData	-	20121119
Ikarus	-	20121119
Jiangmin	-	20121119
K7AntiVirus	-	20121116
Kaspersky	-	20121119
Kingsoft	-	20121112
McAfee	-	20121119
McAfee-GW-Edition	-	20121119
Microsoft	-	20121119
MicroWorld-eScan	-	20121119
Norman	-	20121119
nProtect	-	20121119
Panda	-	20121119
Rising	-	20121119
Sophos	-	20121119
SUPERAntiSpyware	-	20121119
Symantec	-	20121119
TheHacker	-	20121118
TotalDefense	-	20121118
TrendMicro	-	20121119
TrendMicro-HouseCall	-	20121119
VBA32	-	20121119
VIPRE	-	20121119
ViRobot	-	20121119

Figure 8.2: Malware that was detected by the TGSM but was not caught by the 43 anti-virus scanners (part II).

Bibliography

[1] S. Antonatos, E. P. Markatos, and K. G. Anagnostakis. Honey@home: A new approach to largescale threat monitoring worm07. 2007.

[2] http://anubis.iseclab.org. Anubis: Analyzing unknown binaries.

[3] http://en.wikipedia.org/wiki/Netflow. Netflow is a network protocol developed by cisco systems for collecting ip traffic information.

[4] http://nepenthes.carnivore.it. Nepenthes is a versatile tool to collect malware.

[5] http://pcengines.ch/alix.htm. Pc engines.

[6] http://www.few.vu.nl/argos/. An emulator for capturing zero-day attacks.

[7] http://www.fp6 noah.org. The project aims to gather and analyse information about the nature of internet cyberattacks.

[8] http://www.freebsd.org. Freebsd is an advanced operating system for modern server, desktop, and embedded computer platforms. based on bsd unix.

[9] http://www.freebsd.org/handbook/jails.html. Mechanism is an implementation of operating system-level virtualization.

[10] http://www.honeyathome.org. Honey at home is the ”@home” implementation of the noah project, aiming to facilitate the gathering of information on cyber-attacks.

[11] http://www.honeyd.org. Honeyd virtual honeypot.

[12] http://www.honeynet.org. The honeynet project is a leading international security research organization, dedicated to investigating the latest attacks and developing open source security tools to improve internet security.

[13] http://www.openbsd.org. Openbsd.

[14] http://www.openbsd.org/faq/pf/. Openbsd packet filter.

[15] http://www.openvpn.net. Open source vpn solution.

[16] http://www.virtualbox.org. Powerful x86 and amd64/intel64 virtualization product.

[17] http://www.virustotal.com. Virustotal free online virus, malware and url scanner.

[18] X. Jiang and D. Xu. Collapsar: A vm-based architecture for network attack detention center. August 2004.

[19] C. Leita, V. H. Pham, O. Thonnard, E. Ramirez-Silva, F. Pouget, E. Kirda, and M. Dacier. The leurre.com project: Collecting internet threats information using a worldwide distributed honeynet. August 2008.

[20] H. Sistemas, M. Dacier, E. Kirda, and C. Leita. Large scale malware collection: lessons learned. 2008.