Honeynets Applied to the CSIRT Scenario

Cristine Hoepers
NIC BR Security Office – NBSO
cristine@nic.br

Klaus Steding-Jessen
NIC BR Security Office – NBSO
jessen@nic.br

Antonio Montes
National Institute for Space Research – INPE
montes@lac.inpe.br

Abstract

A honeynet is a research tool consisting of a network specifically designed for the purpose of being compromised, with control mechanisms that prevent this network from being used as a base for launching attacks against other networks. Once compromised, the honeynet can be used to observe the intruders' activities, collect tools and determine new trends in network attacks. In this paper we discuss the implementation of a honeynet, based entirely on open source software, that meet the requirements listed above. We present its topology, the tools developed and the results achieved. We also discuss how valuable a honeynet can be to better understand the threats to the constituency of a Computer Security Incident Response Team (CSIRT).

1 Introduction

Due to the necessity of understanding both the attacks to Internet connected networks and the profile of the intruders, some research groups[1] have joined to develop, deploy and monitor honeynets.

Honeynets are research tools consisting of a network specifically designed for the purpose of being compromised [1,2]. Once compromised, the honeynet can be used to observe the intruders' activities and behavior. This allows a detailed analysis of the tools used by the intruders as well as the vulnerabilities used to compromise the honeypots.

To join the efforts of the honeynet community we have implemented a Honeynet in Brazil with the intention of helping us monitor the malicious activities from our country's point of view and share this data with people from other countries and honeynets. One of our main concerns was to implement a low cost honeynet and still have a high quality data control in place. With this in mind we started developing our data control architecture and tools all based on open source free software. The tools we developed are also freely available. After a test phase we put the Honeynet.BR in operation as a cooperative research work between NIC BR Security Office (NBSO) and the Brazilian National Institute for Space Research (INPE). The Honeynet.BR Project provides the means to observe ocurring attacks and intrusions, collect data and develop new tools to improve the honeynet technology.

In this paper we initially present the adopted architecture and methodologies applied to the Honeynet deployed. We also discuss the mechanisms implemented to contain the outgoing malicious traffic, capture data and generate alerts. The activities observed and the usefulness of a Honeynet to a CSIRT are discussed as well.

2 The Honeynet Components

The initial plans for the Honeynet.BR implementation were made in December 2001 and it started operations in March 2002. In the initial phase the main project decisions were taken regarding its topology, operating systems and tools to be used, and development of other tools to analyse and contain the traffic.

[1] http://www.honeynet.org/alliance/

The Honeynet.BR topology, shown in Fig. 1, is divided into two distinct parts: the Administrative Network and the Honeynet itself.

The Administrative Network has the main function of containing the outgoing malicious traffic and monitoring all the incoming and outgoing traffic. This network is completely transparent to both the Honeynet and the Internet, and it is comprised of:

- A Firewall that allows all incoming traffic to the Honeynet and blocks malicious outgoing traffic. This control is exerted in the layer 2 level with the Firewall operating as a bridge. The Firewall functionalities will be discussed in depth in section 2.2;
- A Hogwash machine configured to block the outgoing traffic that has well-known malicious content. This machine also operates as a bridge. More details in section 2.2.4;
- An IDS that captures and analyses all traffic related to the Honeynet, and sends alerts in case of a compromise. It also generates daily summaries about all the activities. Details about its implementation are available in sections 2.3 and 2.4;
- A file server (named Forensics) dedicated to the storage of artifacts and disk images of the Honeynet hosts. Details are further discussed in section 2.1.1.

The main mechanisms used to contain traffic and generate alerts were developed by the Honeynet.BR Team using OpenBSD as the operating system platform.

2.1 Honeypots

The Honeynet itself is composed of several hosts (the honeypots) running different operating systems and services. One of this hosts is the nameserver of the Honeynet and the central logserver. The honeypots installation details are discussed in the next section.

2.1.1 Procedures Applied to Honeypots

During the honeypots deployment process a set of procedures is followed to maintain a record of the system

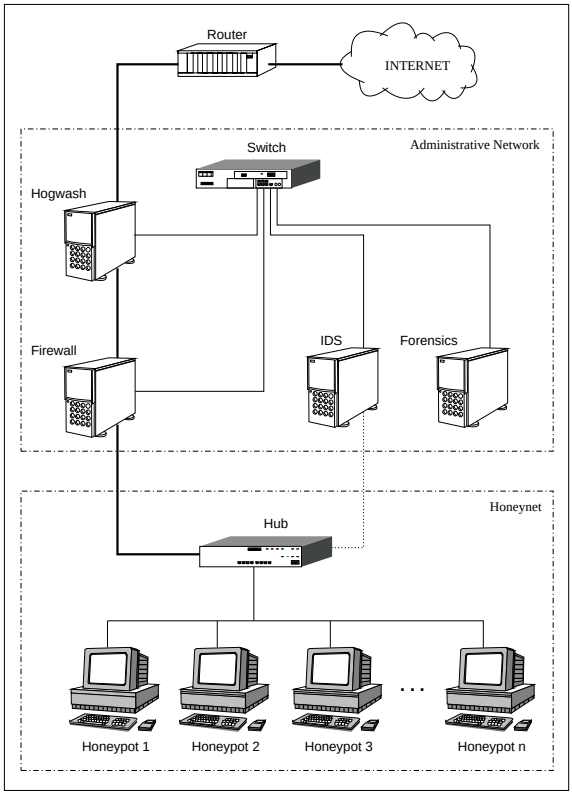

Figure 1: Honeynet.BR Topology

and services installed in each honeypot, as well as to prevent data related to previous intrusions to remain in the disk.

The basic steps taken during the honeypots deployment process are:

1. The previous disk data is erased by writing zeroes over the entire disk. This procedure also has the advantage of providing a better compression rate of the disk image [3];
2. The honeypot operating system is installed and its services configured;
3. A compressed disk image of the newly installed honeypot is stored in the Forensics machine;
4. This entire process is recorded in a logbook;
5. The honeypot is connected into the Honeynet.

The honeypot is closely monitored after its deployment. Once compromised, all the malicious activities related to the honeypot and the artifacts col-

lected are recorded and analyzed. In general the monitoring process of the honeypot consists of:

1. Recording in another logbook all activities observed while the intrusions last;
2. Storing all tools used by the intruders in the Forensics machine;
3. Disconnecting, when appropriate, the honeypot from the Honeynet, making an image from the compromised disk and storing it in the Forensics machine;
4. Restarting the deployment process.

2.2 Data Control

One of the most important requirements of a Honeynet is to contain the malicious outgoing traffic. This guarantees we can observe the intrusion and not let the intruder use the Honeynet as a base to launch new attacks.

In the following sections we describe the methods developed to contain outgoing traffic in the Honeynet.BR.

2.2.1 Firewall Rules

The Firewall is configured as a bridge, this means it does not have an IP address and does not decrement the TTL (*Time to Live*) of the IP packets that pass through it, reducing the chances of the Firewall being noticed.

The rules are implemented using the OpenBSD Stateful Packet Filter [4] (`pf`). They allow any incoming traffic and drop all the potentially malicious outgoing traffic. Some examples of the potentially malicious traffic it drops are:

- spoofed IPs;
- some ICMP packets;
- UDP packets depending on its source and destination;
- TCP traffic directed to well-known vulnerable services or with unusual characteristics.[2]

[2]For example when the source port is the same as the destination.

2.2.2 Outgoing Traffic Normalization

There are some attacks that overlap IP fragments to confuse intrusion detection systems and firewalls, as well as some scanning tools that use invalid TCP flag combinations to achieve the same goal. These invalid flags are also used for remote TCP/IP fingerprinting.

To contain this kind of malicious packets the `pf` normalization mechanism is used to reassemble and discard invalid packets.

In addition to the usual normalization made by `pf`, we have also modified the `pf` source code to discard packets with both `SYN` and `FIN` flags on, feature that is frequently used by scanning tools.

2.2.3 `sessionlimit` Implementation

The OpenBSD `pf` is a stateful packet filter and is configured to create sessions for the outgoing packets that are not dropped by the rules mentioned before. By inspecting the `pf` state table it is possible to obtain some information about each created session, for example:

- direction (in or out);
- protocol;
- source and destination IPs and ports;
- total number of bytes and packets;
- creation and expiration time;
- status (`SYN_SENT`, `ESTABLISHED`, etc);

Based on these characteristics we have built an open source tool called `sessionlimit`. This tool continuously monitors the state table entries and interacts with `pf` inserting and removing rules as necessary.

`Sessionlimit` can block the traffic related to a specific host based on one of the following criteria:

1. when the number of states related to a source IP is increasing too fast;
2. when the source IP has reached a predefined limit of outgoing connections (the default is 20 connections).
3. when the number of bytes associated with an ICMP state has reached a predefined limit.

If any of these conditions is satisfied a rule blocking all outgoing traffic from this host is inserted in the current `pf` rules, and all outgoing sessions for this IP are removed.

It is important to note that the blocking rule inserted by `sessionlimit` affects the outgoing traffic only. All incoming sessions already established to this host are not affected. This typically includes the intruder's interactive session.

`Sessionlimit` removes a blocking rule from the list of active rules in the Firewall after a predefined timeout.

All actions taken by `sessionlimit` are logged via `syslog`, as shown in Fig. 2.

```
Jul 15 04:42:49 fw sessionlimit[29832]: starting

[...]

Jul 23 19:58:29 fw sessionlimit[29832]: \
    ICMP: blocking xxx.xxx.xxx.xxx (65016 bytes)
Jul 23 19:58:29 fw sessionlimit[29832]: \
    11 state(s) killed from xxx.xxx.xxx.xxx
Jul 23 20:28:29 fw sessionlimit[29832]: \
    expiring xxx.xxx.xxx.xxx after 1800 seconds

[...]

Oct  2 13:04:27 fw sessionlimit[29832]: \
     blocking xxx.xxx.xxx.xxx (20 states)
Oct  2 13:04:27 fw sessionlimit[29832]: \
     20 state(s) killed from xxx.xxx.xxx.xxx
Oct  2 13:34:27 fw sessionlimit[29832]: \
     expiring xxx.xxx.xxx.xxx after 1800 seconds
```

Figure 2: `sessionlimit` log excerpts. Some lines are broken to facilitate reading. The real IPs were removed.

2.2.4 Outgoing Content Filters

Besides the methods described above, that are implemented by the Firewall, malicious outgoing packets can be dropped by the Hogwash machine based on its contents.

We use the open source tool `hogwash`[3] to do this analysis. The packets are dropped by this tool whenever the contents match a well-known attack signature.

`Hogwash` uses the same rules as the `snort`[4] Intrusion Detection tool. In our Honeynet `hogwash` is executed in a machine assigned for this purpose, as shown in Fig. 1.

One of the advantages of using `hogwash` is the facility to update its signatures. They can be obtained from the security community or can be created based on attacks previously observed in the Honeynet.

2.2.5 Bandwidth Limitation

As an additional measure to contain malicious traffic we have decided to restrict the available outgoing bandwidth by using `ALTQ` (*Alternate Queueing*)[5].

The goal is to limit the intensity of a Denial of Service attack originated in the Honeynet, in case the other data control mechanisms fail.

2.3 Data Capture

All incoming and outgoing traffic, as well as the traffic inside the Honeynet, is captured and stored. The data is captured in two places:

1. Firewall

   The Firewall stores all incoming and outgoing traffic in `tcpdump` binary format[6]. This is done through the `pf` logging mechanism. The use of this standard format facilitates the data manipulation, sice it allows the use of well-known tools like `tcpdump`, `ethereal`, `ngrep`, etc.

2. IDS

   The IDS has a network interface, with no IP address assigned to it, which captures all data circulating between the honeypots and the incoming and outgoing data.

   The data capture is made by a script that uses `tcpdump` to read the data and store it in files named by year, date and time of the beginning of the capture.

[3] `http://hogwash.sourceforge.net/`

[4] `http://www.snort.org/`

[5] `www.csl.sony.co.jp/person/kjc/software.html`

[6] With the exception of the spoofed traffic, typically used in Denial of Service attacks, because of the huge volume of logs it generates

The data captured by the IDS is used by the alerting mechanisms and also used to generate the daily summaries. This is described in detail in section 2.4.

2.3.1 Data Rotation and Compression

All the data captured by the Firewall and by the IDS are rotated and compressed every 24 hours. The name of the generated files follows our convention, which is year, month and day of the generation. After a period of 30 days each file is moved to an off-line storage media.

2.4 Alerts and Summaries

2.4.1 Alerts

The generation of alerts follows the principle that any traffic observed in the Honeynet is malicious. Outgoing traffic from the Honeynet is a clear indication of a compromise.

The alerts can be generated as follows:

1. Outgoing traffic

   A script using `tcpdump`, running in the IDS machine, filters the captured data. Any outgoing packet originating from the Honeynet, that is not in response to an incoming packet, generates an alert. All alerts are grouped and sent by email periodically.

2. Shell commands

   The Honeynet Unix machines have a modified shell that sends the commands history to the log server via the syslog service. A script running in the IDS machine monitors the traffic and produces an alert if the logs generated by the shell are detected.

An example of an alert generated by a shell command is shown in Fig. 3.

One single alert can contain any of the types described above. A copy of all alerts is maintained in the IDS machine for future reference.

The generation of alerts can be easily configured to have different levels of sensibility. This is important to reduce the number of false positives.

```
2002/09/22 02:41:01 host:514 -> loghost:514
HISTORY: UID=48 rm -rf /tmp/.unlock.uu
/tmp/.unlock.c /tmp/.update.c /tmp/httpd
/tmp/update /tmp/.unlock;

2002/09/22 02:41:01 host:514 -> loghost:514
HISTORY: UID=48 cat > /tmp/.unlock.uu <<
__eof__;

2002/09/22 02:41:10 host:514 -> loghost:514
HISTORY: UID=48 uudecode -o /tmp/.unlock
/tmp/.unlock.uu; tar xzf /tmp/.unlock -C /tmp/;
gcc -o /tmp/httpd /tmp/unlock.c -lcrypto; gcc -o
/tmp/update /tmp/.update.c;
```

Figure 3: Example of an alert generated because of the capture of a shell activity by the IDS. Some lines were edited because of legibility. The real IPs were removed.

In addition to email, the alerts can be sent by pager or mobile phone.

2.4.2 Summaries

A summary is issued daily containing the activity observed in the Honeynet the day before. This summary is sent by email and also stored in the IDS machine. The input data for each summary is the IDS compressed data captured during the period of one day. The output contains:

1. Statistics

   - total number of packets captured;
   - number of packets by protocol and corresponding percentage;
   - hosts that originated most TCP, UDP and ICMP traffic;
   - TCP and UDP ports with more incoming traffic.

2. `snort` alerts

   The `snort` program is used to read the captured data and to generate alerts, which are listed in the summary.

3. Incoming traffic

   We include a condensed `tcpdump` output of the incoming traffic only. This is particularly useful to verify if a certain behavior, like a scan, happened only against one host or in the whole honeynet.

3 Activities Observed

During the observation period several malicious activities were detected. This allowed us to collect tools and monitor both the vulnerabilities explored and the exchange of information between the intruders. In this section we discuss the various types of attacks and trends we have observed so far.

3.1 Top Scanned Services

The scans in their majority were targeted at the `http` (80/TCP) and `ftp` (21/TCP) services. These were, by far, the most scanned services in the Honeynet. The other scans were directed mostly to the following ports: 22/TCP, 23/TCP, 111/TCP, 515/TCP and 6112/TCP.

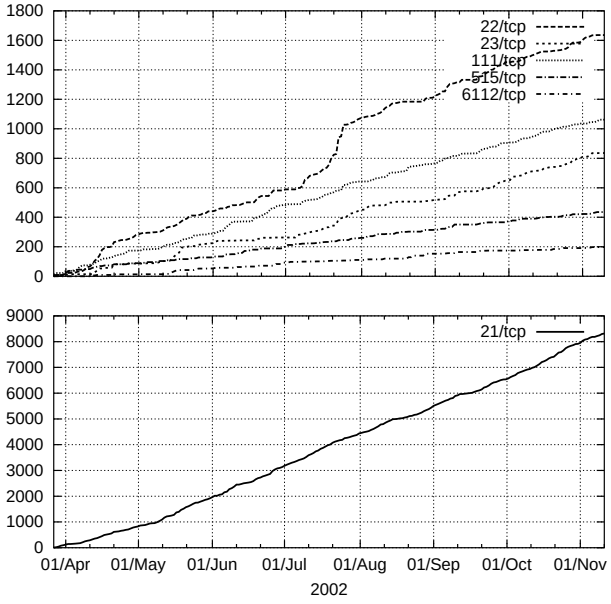

Figure 4: Scanned ports – cumulative

In Fig. 4 we see the cumulative number of scans, with the exception of port 80/TCP, directed to these ports. It is interesting to notice that the scans for `ssh` port show a marked growth in mid July, nearly a month after the release of the CERT® Advisory CA-2002-18[7].

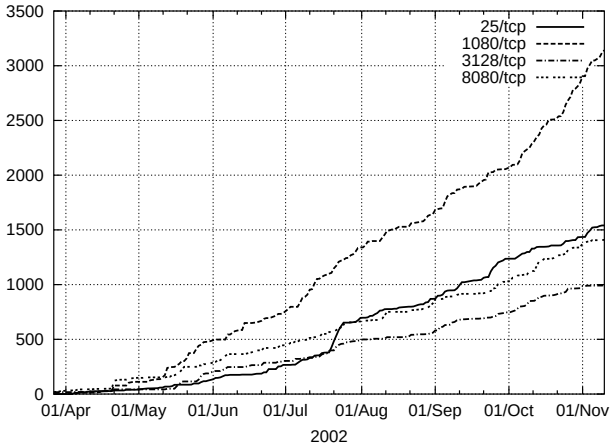

Figure 5: Scan for open proxies and relays – cumulative

We also detected that searches for open proxies and poorly configured email servers that permit email relay have been constant, as shown in Fig 5. Sometimes the activity is more intense than the search for vulnerable services.

3.2 DoS and DDoS Attacks

We observed the intruders trying to use several different Denial of Service tools and techniques. This varied from the trivial "`ping -f`" command, to tools using UDP fragmented packets.

We have also seen coordinated distributed attacks launched via IRC. They have used the tool `kaiten`[8], that is an IRC based DDoS client. This client connects to a specific server and receives commands via an IRC channel.

3.3 Worms

We have detected a high number of attacks made by worms. This attacks, in general, were targeted against web servers.

[7] `http://www.cert.org/advisories/CA-2002-18.html`
[8] `http://packetstormsecurity.nl/irc/kaiten.c`

In Fig. 6 we can see incoming traffic to ports associated with worm activity. It is possible to observe that scans for port 80/TCP (Nimda, CodeRed, and others) have shown a constant rate since the beginning. The scans for port 1433/TCP (SQL Worm) started in May, around the same time CERT®/CC released its "Incident Note IN-2002-04"[9]. And the scans for port 443/TCP (Slapper and variants) had a small increase in activity starting in the first half of September, when CERT®/CC released its "CERT® Advisory CA-2002-27"[10], but the major increase occurs after the release of the worm variants about September 20th.

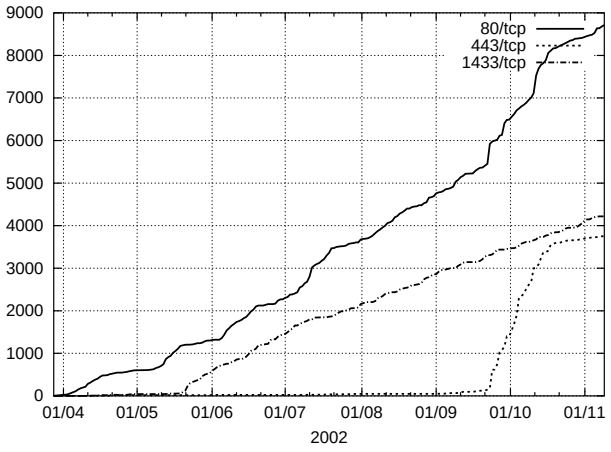

Figure 6: Worm related activity – cumulative

3.4 Intruders Profile

The intruder's profiles were very similar. Almost all of them, after obtaining privileged access, installed scan tools, exploits, massrooters, rootkits and IRC related programs. In some cases Denial of Service tools were also installed, as discussed above.

All the backdoors installed in the Honeynet made use of some cryptographic mechanism, preventing the capture of the traffic related to these sessions. Because of this, the observation of the intruders' sessions remained restricted to the data obtained through the modified shell.

Most of the intruders have launched the exploits from machines located outside Brazil, even in the cases where we identified the intruders as Brazilians.

Observing the IRC traffic it was possible to identify that some intruders were Brazilians, although the great majority was formed by Romanians.

Regarding the motives, we have determined that their intention was mostly the use of the machines as IRC bouncers, lauch points to other attacks and stepping-stones. In one case we also identified that the intruders were involved in credit card fraud.

In Fig. 7 we put together some statistics correlating countries to the sources of scans, exploits and backdoor access. The scans statistics were made considering only the ports associated with services that had received any exploit attempt.

To define to which country an IP belongs we considered to which country the IP block is assigned, not taking into account DNS lookups.

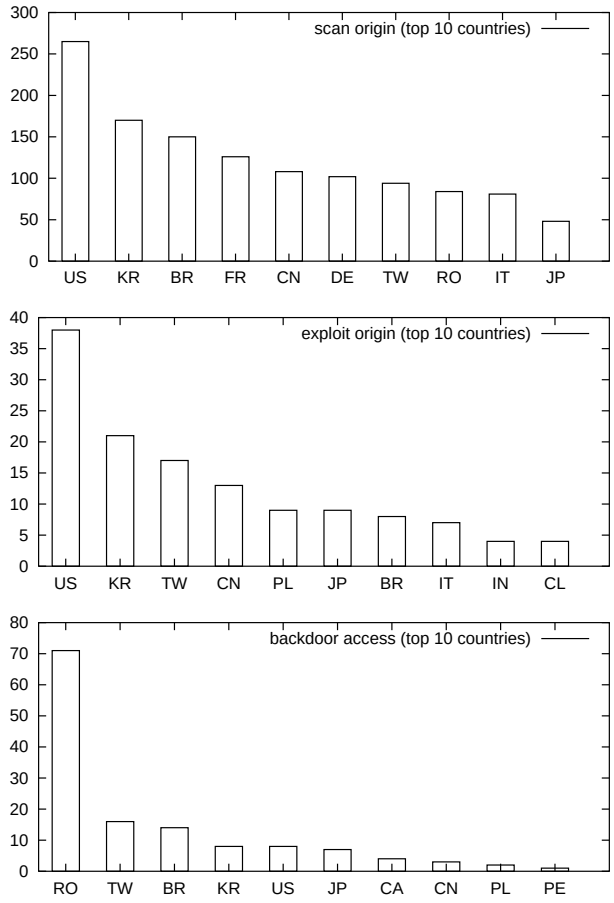

Figure 7: Intruders origin – top 10 countries

[9] http://www.cert.org/incident_notes/IN-2002-04.html

[10] http://www.cert.org/advisories/CA-2002-27.html

3.5 Defacements

The Honeynet has no Web server and was not our intention to attract defacers, but even so they came. They compromised a honeypot, which did not have Web services running, via an OpenSSH vulnerability. Once they gained privileged access they started the `httpd` daemon and made the "defacement" only to announce it to a mirror Web site.

4 Usefulness to CSIRTs

During the observation period of the Honeynet it has proved itself a valuable tool to a CSIRT. Some of the advantages of having such technology in place are discussed next.

4.1 Detection of Attacks

The NBSO constituency is the Brazilian Internet connected networks. At first we thought that having only one Honeynet, placed in a small part of the Brazilian address space, would show only limited data about what is really happening. But we were positively surprised when we realized that the data captured in the Honeynet reflected the activity we were seeing through incident reports coming from various parts of the country.

We have followed some attacks in the Honeynet that were very specific and with some caractheristics that made clear they were peculiar to the Brazilian address space. In some cases this data was correlated to incident reports in order to have a better understanding of the overall picture.

4.2 Source of Training Material

Once the Honeynet is up and running, and the methodologies for deploying the honeypots and preserving evidences is in place, it becomes a great source of material to learn how to perform artifact analysis and forensics.

This is very interesting as a way to maintain the team in close contact with material coming from intrusions, and ready to act in case of a real compromise inside the constituency.

It is also a great tool to train new incident handlers to deal with log analysis and intrusions, and to get them experienced with the process of preserving evidence.

4.3 Helping the Community

Now and then it is common to see a scan, or even a compromise, coming from an IP that looks like a compromised machine. Every time we identify any malicious traffic as coming from a compromised machine we send an email to its Whois contact advising about the problem.

This email is sent by NBSO exactly in the same way we send emails advising about activities observed in networks that do not want their names to be disclosed. It contains sanitized logs and some advice on how to check if the machine is compromised and how to recover from an intrusion.

It is very common to receive positive followups from these emails. In most cases the administrators were not aware of the intrusion and our email gave them the first warning to look at the network and find out about the problem.

Some new rootkits were collected and provided to the authors of the open source tool `chkrootkit`[11] to update the tool. In this way people that make use of the tool can benefit from our findings.

5 Future Work

The use of encrypted sessions is largely disseminated through the blackhat community. This makes very important the development of new techniques to monitor their activities. We plan to work in the improvement of new tools to do keylogging at kernel level, or using system libraries.

We also plan to improve `sessionlimit` in order to achieve better performance in high bandwidth networks.

6 Conclusions

Implementing a Honeynet is a challenge, but all the work of developing data control mechanisms and data analysis tools is rewarded by the valuable information it provides.

[11] `http://www.chkrootkit.org/`

To a CSIRT it is very important to be in contact with new tools and techniques used by intruders. In some cases people report an incident but they do not have all the information about the attack, or they do not know how to get that information. Correlate incident notifications with data captured in the Honeynet can clarify some attacks or add more information to them.

The correlation between data from the Honeynet and from Incident Reports can also help to distinguish between real threats and false positives.

Acknowledgments

This project is sponsored by both the NIC BR Security Office (NBSO) and the National Institute for Space Research (INPE).

Several other people and institutions have helped to set up the project. We would like to give our special thanks to the Honeynet.BR Team[12] for their support and ideias, and to the Ministry of Science and Technology and The State of São Paulo Research Foundation (FAPESP) for their support and donations.

We also would like to thank Monica Rubly for revising this paper.

References

[1] The Honeynet Project, *Know Your Enemy: Revealing the Security Tools, Tactics, and Motives of the Blackhat Community*. Addison-Wesley, 1st ed., August 2001. ISBN 0-201-74613-1.

[2] L. Spitzner, "Learning the Tools and the Tactics of the Enemy with Honeynets," in *Proceedings of the 12th Annual Computer Security Incident Handling Conference*, (Chicago, Illinois, USA), June 2000.

[3] D. Farmer and W. Venema, "Being Prepared for Intrusion," *Dr. Dobb's Journal*, vol. 26, April 2001.

[4] D. Hartmeier, "Design and Performance of the OpenBSD Stateful Packet Filter (pf)," in *Proceedings of the FREENIX Track: 2002 USENIX Annual Technical Conference (FREENIX '02)*, (Monterey, California, USA), June 2002.

[5] A. B. Filho, A. S. M. S. Amaral, A. Montes, C. Hoepers, K. Steding-Jessen, L. H. Franco, and M. H. P. C. Chaves, "Honeynet.BR: Desenvolvimento e Implantação de um Sistema para Avaliação de Atividades Hostis na Internet Brasileira," in *Anais do IV Simpósio sobre Segurança em Informática (SSI'2002)*, (São José dos Campos, SP), pp. 19–25, Novembro 2002. `http://www.lac.inpe.br/security/honeynet/papers/hnbr-ssi2002.pdf`.

[6] W. R. Cheswick, "An Evening with Berferd in Which a Cracker is Lured, Endured, and Studied," in *Proceedings of the Winter 1992 USENIX Conference*, (San Francisco, California, USA), pp. 163–174, 1992.

[7] S. M. Bellovin, "There Be Dragons," in *Proceedings of the Third Usenix Security Symposium*, 1992.

[8] C. Stoll, *The Cuckoo's Egg: Tracking a Spy Through the Maze of Computer Espionage*. Garden City, NY: Doubleday, 1989. ISBN 0-385-24946-2.

[9] C. Stoll, "Stalking the Wily Hacker," *Communications of the ACM*, vol. 31, pp. 484–497, May 1988.

[12] `http://www.lac.inpe.br/security/honeynet/`