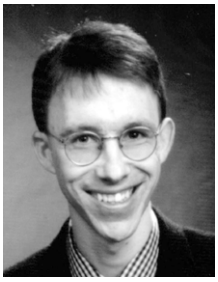
K.G. Saur Verlag, München, 2004

M. Dornseif, F.C. Gärtner, and T. Holz

Vulnerability Assessment using Honepots

Maximillian Dornseif has studied laws and computer science at the University of Bonn, Germany, where he wrote his PhD Thesis about the "Phenomenology of Cybercrime". He has been doing IT security consulting since the mid nineties. In 2004 he joined the Laboratory for Dependable Distributed Systems at RWTH Aachen University where he works in the area of detection and documentation of security incidents and vulnerabilities.



Felix Gärtner is a professor of computer science at RWTH Aachen University and head of the Laboratory of Dependable Distributed Systems. His PhD dissertation on formal foundations of fault-tolerant systems won the 2001 Dissertation Award of the German Computer Science Society (GI e.V.). From 2002 to 2003 he was Emmy Noether scholar on a grant by Deutsche Forschungsgemeinschaft at the Swiss Federal Institute of Technology in Lausanne, Switzerland. His research interests cover the whole range of theory and practice of dependable systems, especially the relations between fault-tolerant systems and secure systems.

Thorsten Holz is a research student at the Laboratory for Dependable Distributed Systems at RWTH Aachen University in current pursuit of his diploma thesis. His research interests include the practical aspects of secure systems as well as more theoretical considerations of dependable systems.

ABSTRACT

Honeypots are electronic bait, i.e. network resources (computers, routers, switches, etc.) deployed to be probed, attacked and compromised. Honeypots run special software which permanently collects data about the system and greatly aids in post-incident computer and network forensics. Several honeypots can be assembled into networks of honeypots called honeynets. Because of the wealth of data collected through them, honeynets are considered a useful tool to learn more about attack patterns and attacker behavior in real networks. This paper explains the motivation for using the honeynet methodology and describes experiences with a honeynet at RWTH Aachen University. In analyzing the data collected through our experiment, we discuss the value of honeynets for computer vulnerability assessment. The paper also gives an overview over ethical and legal aspects of honeypots and a look on possible directions for further research.

1 INTRODUCTION

The therm *honeypot* usually refers to an entity with certain features that make it especially attractive and can lure individuals (not only animals) into its vicinity. Recently, the concept of honeypots has been applied to the area of IT security. There, honeypots can be used as electronic bait to attract attackers and study their behavior. Spitzner defines a honeypot to be "a resource who's value is in being probed, attacked or compromised." [1]. Honeypots are equipped with special software (usually a patched operating system) that make them indistinguishable from "normal" network nodes from the outside but they permanently collect detailed data about network connections, user activity etc. In contrast to similar data collected on "normal" machines, the wealth of this data can be used to better study attack patterns and attacker behavior and greatly aids in post-incident computer and network forensics. For example, the specialized tools used by an attacker can easily be intercepted on a honeypot and would be hard to obtain on a normal desktop computer since they are usually removed by the attacker after a break-in. In contrast to previous work in this area [2, 3] which collected data in an ad-hoc, post-incident manner, honeypots offer a more systematic approach to studying attack patterns and general vulnerability assessment.

In the context of IT security, the use of honeypots promises to improve both the efficiency and the effectiveness of defensive countermeasures. On the one hand, since networks of honeypots (so-called *honeynets*) offer empirical data about attackers' behavior, administrators can concentrate on *relevant* attacks and thus can use their resources more efficiently. On the other hand, honeypots can also help discover new vulnerabilities and new types of attacks. Therefore, honeypots help to make countermeasures more effective.

To investigate the usefulness of honeynet technology, we have deployed a honeynet at RWTH Aachen University within the Laboratory for Dependable Distributed Systems. In this paper we report on our experiences. We firstly describe the background and goals of the honeynet approach in Section 2. In Section 3 we then present the architecture of the honeynet we deployed at RWTH Aachen University and present our empirical data in Section 4. In Section 5 we then discuss ethical and legal aspects. Section 6 contains an assessment of the quality

of data collected by honeypots and section 7 closes with an outlook on further research.

2 THE GLOBAL HONEYNET PROJECT

2.1 Motivation

Founded in 1999, the global *Honeynet Project* [4] is a non-profit research organization in which security professionals perform research in the area of computer security. The goal of this project is to learn more about the tools, tactics, and motives of the *blackhat* community and share the lessons learned. A *blackhat* is an attacker, who uses his skills for unethical or damaging reasons. The following points describe the focus of the project in more detail.

2.1.1 Raise Awareness

The project aims at raising public awareness of the threats and vulnerabilities that exist in the Internet today. This is done by demonstrating *real* systems that were compromised by *real* attackers.

2.1.2 Research on Old and New Attacking Techniques

Honeynets provide the technology and methods of gathering information about real attacks in the Internet. Empirical analysis of data collected about known attacks can help administrators defend their systems more efficiently, e.g., by investing resources into countermeasures that have the highest impact. Through a careful analysis of the attackers' behavior, it is also possible to gather additional information about their motives, how they communicate, when they attack systems and their actions after compromising a system. Therefore, it is also possible to learn about *new and unforeseen* ways of attacking a system. This is almost impossible in the usual *a posteriori* computer forensics approaches.

2.1.3 Active Network Defenses

Honeynets can also be a part of a defensive infrastructure in that they can be used to divert the adversary's attention from more valuable targets. In case an attack is detected, it is even possible to transparently divert the adversary's access from such a valuable system to a specially prepared honeypot for further study.

The German Honeynet Project was founded in June 2004 and is affiliated to the Laboratory for Dependable Distributed Systems at RWTH Aachen University [5]. One of the main focuses of the German Honeynet Project is bringing Honeynet research to a solid scientific foundation and assessing the value of Honeynet technology as a research tool.

2.2 Approach

A honeypot is usually a computer system with no conventional task in the network and no regularly active users. These assumptions aid in attack detection: Every interaction with the system is suspicious and could point to a possibly malicious action. Therefore all network traffic to and from the honeypot is logged. Usually, a honeynet consists of several honeypots of different type (different platforms and/or operating systems). This allows to simultaneously collect data about different types of systems.

The Honeynet Project is developing special tools that support honeypot operation. One of the main software tools is the monitoring system Sebek [6], which tries to capture all activities of an attacker on a honeypot. Sebek is a client/server system: The honeypot runs the Sebek client which closely monitors and logs all user activity in a manner meant to be undetectable: It replaces the read() system call with its own version and can thus record all data accessed via read(). It can for example log all SSH-sessions, recover files copied via secure copy (SCP) and record all passwords used by intruders. The logged information is sent over the network (while equally trying to be undetectable) to the Sebek server, usually a dedicated machine which stores the data and facilitates browsing and analysis. Sebek achieves (almost) undetectable communication by sending all logged data directly to the device driver and thus bypassing all logging mechanisms on a host. In order to hide its presence from an observer, Sebek uses a technique borrowed from the rootkit adore [7]. It unlinks itself from the list of installed modules and thus makes detection of its presence harder. This is not a completely robust method for hiding its presence and we found some ways to detect Sebek on a host [8]. Sebek in its primary version is a kernel module for Linux 2.4 kernels. It was ported to other operating systems like Solaris, OpenBSD and Windows. Currently, also a port to MacOS X is under development by Christian N. Klein from the German Honeynet Project and Thorsten Holz is porting Sebek to Linux 2.6. More detailed information on the workings of Sebek can be found in [6].

Another approach in honeypot-technology is more leightweight: Instead of deploying a physical computer system which acts as a honeypot one can also deploy one physical computer which hosts some virtual machines which act as honeypots. This leads to easier maintenance and less physical requirements. Usually VMware [9] or User-mode Linux [10] is used on order to set up such virtual honeypots.

If the operator of a honeynet is primarily interested in quantitative results, one can even go one step further and abandon the emulation of a whole computer system. This approach is called *low interaction honeypot* in contrast to the *high interaction honeypots* described above. One implementation of low interaction honeypot technology is called honeyd [11]. Honeyd is a small daemon which creates virtual hosts on a network. It simulates the TCP/IP-stack of different operating systems and can be configured to run arbitrary services. These services are generally small scripts that emulate real services like POP3 or SMTP. Honeyd enables a single computer system to claim multiple addresses by intercepting ARP requests and redirecting them to honeyd. It can simulate arbitrary network topologies including dedicated routes and routers and can be configured to feign latency and packet loss.

2.3 Experiences up to now

There have been some success stories about the usefulness of Honeypots in the literature, in which the respective authors claim that a lot of valuable information has been collected about the motives and techniques of their adversary. In January 2002 for example, the collected data of a honeypot allowed to identify an exploit which used a known vulnerability in the "dtspcd" soft-

ware (CDE Subprocess Control Service) to launch an attack [12].

Honeynets also have been used to monitor conversations over the Internet Relay Chat (IRC), a communication technique often used by blackhats on compromised systems. Using honeynet technologies, the Honeynet Research Project at Azusa Pacific University recently discovered a so-called "botnet", i.e., a network of compromised machines which could be "remote controlled" using IRC. This botnet consisted of more than 15,000 computers [13]. Botnets are a great threat towards the availability of Internet hosts since they easily allow to launch powerful denial-of-service attacks. Other discoveries using honeypots include special IRC channels for organized credit card fraud [14]. As a public service, the data collected by members of the Honeynet Project is published and can be used as the basis for vulnerability assessment [15].

3 HONEYNET ARCHITECTURE AT RWTH AACHEN UNIVERSITY

The Laboratory for Dependable Distributed Systems at RWTH Aachen University operates a honeynet since January 2004. In this section, we firstly describe the initial setup of our system which is running more or less unchanged since the beginning of the project. Following that we explain additions and refinements we have been testing. Further information on the general software- and hardware-architecture of honeynets can be found in the article by Reiser and Volker in this issue.

3.1 Initial Setup

The initial setup of our honeynet is shown in Fig. 1. The connection to the Internet is guarded by a honeywall. Originally the honeywall was running on a Pentium PC (233 MHz, 256 MB RAM, 20 GB hard disc). Due to a hardware failure in early summer we replaced the machine with another Pentium PC (450 MHz, 256 MB RAM, 20 GB hard disc). The honeywall is based on a Debian/GNU Linux 3.0r2 operating system on which means of data capture and data control are implemented.

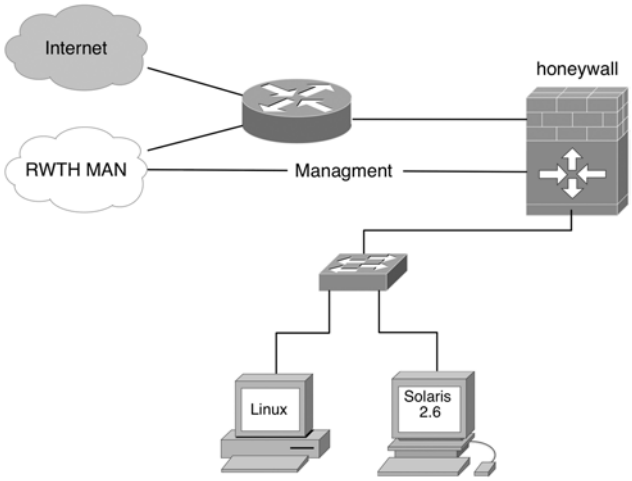


Fig. 1 Initial Honeynet at RWTH-Aachen University

Using the netfilter/iptables subsystem of the Linux kernel, data control is implemented. For example, outgoing TCP traffic is limited to 25 connections per day and similarly, only 50 outgoing ICMP packets are allowed. These numbers are currently used as best-practice values: These values should make sure that an attacker is not able to generate lots of harm due to a denial-of-service attack against another host.

The packetfilter capabilities of netfilter/iptables are augmented by snort_inline. snort_inline is based on the popular snort intrusion detection system but has been extended to allow rules which modify or drop packets passing snort_inline. Via the netfilter/iptables-functionality all outgoing traffic is passed through snort_inline. While snort_inline might be called an intrusion prevention system, we deploy it as an outtrusion prevention system. Since aim of data control is to reduce the risk of intruders using the honeypots to successfully mount outgoing attacks on other systems we use snort_inline to examine outgoing traffic and block outgoing attacks. This is achieved by rewriting outgoing traffic with known attack payload in a way that the payload will fail.

Fig. 2 is an example of a rule which modifies packets containing shellcode for the x86 architecture in a way that would result in the attacked process safely crashing instead of compromising the system. This sterilization of attacks gives us the ability to allow an intruder to attack other systems because we can assume that all of the attacks will fail. Given the difficulties of making exploits work in the wild and limited sophistication of many intruders there is a high probability that intruders for some time would not detect the presence of the honeywall and therefore continue to try different forms of attacks allowing us to observe them to a greater extent.

```
alert ip $HONEYNET any -> $EXTERNAL_NET any
     (msg:"SHELLCODE x86 stealth NOOP"; rev:6; sid:651;
      content:"|EB 02 EB 02 EB 02|";
      replace:"|24 00 99 DE 6C 3E|";)
```

Fig. 2 Example of a rule for snort_inline

We added two honeypots to the initial honeynet: The Linux-honeypot runs SuSE Linux 8.0 Professional and offers HTTP (Apache 1.3.23 including PHP 4.1.0), FTP (vsFTPd 1.0.1) and SSH (OpenSSH 3.0.2p1) as services. We also installed PHP-Nuke in Version 5.0 and MySQL 3.23.53, in order to observer attacks on web applications. Compared to a normal installation, only a few changes were made: The important modification was the installation of the Sebek client. Furthermore, we deposited some honeytokens on the system. These are different kinds of data (e.g. mails, spreadsheets or encrypted data), which stimulate the interests of an attacker. Honeytokens should help to analyse the data flow after a successfull compromise [16]. In addition, the system was configured to appear like a "normal" system with three users.

The second honeypot was a Solaris-based honeypot running Solaris 8 as operating system on a Sun Ultra 1. This system also offers the services HTTP (Apache 1.3.12), FTP (wuftpd-2.6.1) and SSH (OpenSSH 3.02p1). The Sebek-client also ensures that all activities of an attacker can be monitored. Except for the deactivation of some services of inetd (e.g. telnet, echo or talk), no further modifications were made. Due to hardware failures we had to disconnect the Solaris honeypot after a few weeks.

3.2 Variations

After the first two months, we changed the setup of the honeynet and added some other hosts to the network. We added

three virtual machines running inside a VMware, a system which enables multiple operating systems and their applications to run concurrently on a single physical machine. Thus, we added only one physical machine to the honeynet and this resulted in three additional honeypots, see Fig. 3. These three honeypots are close to default installations and follow the same design principles as outlined in section 3.1 and the article Reiser and Volker in this issue. Furthermore, we used honeyd to simulate additional 16 machines.

The mechanisms used for alarming and analysis are similiar to those described in the article by Reiser and Volker in this issue. We therefore refer the reader to this article for further information.

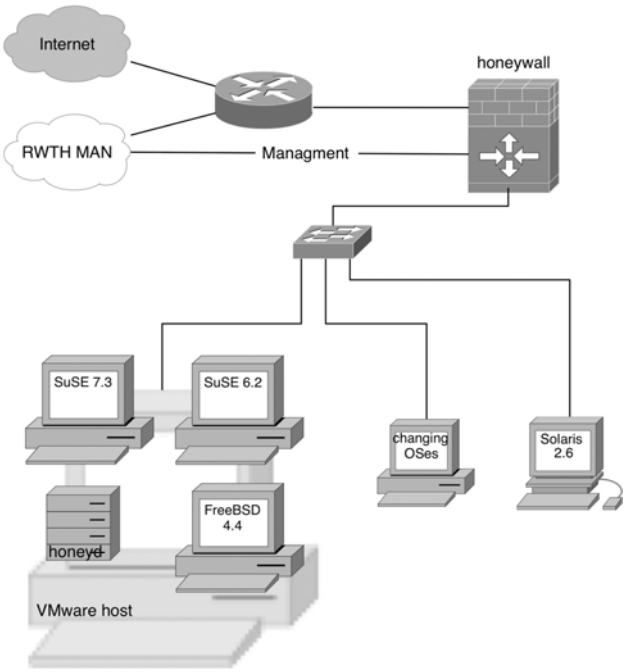


Fig. 3 Honeynet using virtual honeypots at RWTH Aachen University

4 STATUS

The initial deployment of the whole Honeynet was very smooth. Only the automatic loading of the Sebek-Client module during system start-up and subsequent hiding of this module caused some problems. This is a known problem and will be addressed in one of the future versions of the Sebek-Client. After all, the setup of the Honeynet depicted in Fig. 1 took less than one week. During testing of the invisibility of the Sebek-Client module an easy way to detect this module was found: An intruder would be able to detect the presence of Sebek even without any higher privileges by just using some simple commands. We investigated this problem further and found several ways to detect, circumvent and disable Sebek. This leads to a different view of the data quality obtained by Honeynets (see section 6).

The staff members of the Center for Computing and Communication at Aachen University who are responsible for network security naturally formed a bigger obstacle. They take care of the complete incoming and outgoing traffic and control this traffic at a central point. Understandably, there was initial scepticism on their side: After all, honeynets exhibit some kind of risk for the entire network infrastructure and other computers within the network. After a successfull compromise of a honeypot, an intruder can try to attack further hosts inside the network. In general, hosts within the same subnet are more likely to be an easy target because there seldomly are firewalls that filter the traffic within subnets. Snort_inline only helps to block known attacks, but this is no guaranteed protection.

In a constructive dialog a solution satisfying the needs of both parties was found: A dedicated subnet with 64 IP-adresses was allocated to the honeynet. A dedicated line from the Center for Computing and Communication to the honeynet allows on the one hand an unobstructed operation. On the other hand, this setup ensures that no malicious activities can pose a threat to other hosts within the same network. After all, no direct physical connection between the honeynet and the other hosts exists.

In the middle of February 2004 the Honeynet changed from testing- to production mode. Since this time, the honeynet collects data on the network traffic and activities of attackers. So far, no compromise of one of the honeypots occurred. But the honeynet collected some valuable quantitative data. This data allows us to draw conclusions of blackhat activities in our network.

In the following part, a summarization of the collected data will be given:

- During the first two months, our honeynet consisted mainly of two honeypots as described in section 3.1. During this time, about 61 MB of network traffic for the Linux honeypot was captured. This accounts to approximately 425,000 data packets. Altogether more than 9500 different IP adresses could be identified that sent at least once data to the honeypot. Probably many of these packets were spoofed, e.g. they were sent from another host with a counterfeited sender address. Overall we conclude that even the passive existence of a computer system in the Internet, i.e. a connection to the Internet without any outgoing network traffic, leads to attacks.
- The first attempt to attack one of the honeypots was noticed about ten minutes after the whole honeynet was attached to the Internet. The system was systematically searched for weaknesses (port scan) and the attacker tried to exploit a known vulnerability in the Internet Information Server (IIS). After this short period of time, an unpatched version of this server would have been compromised.
- Most traffic that we have seen (97%) is TCP. ICMP and UDP represent only about 1,5% of the whole traffic. The ports 445, 135, 137 and 139 – all belonging to Netbios, the protocol favored by the Microsoft Operating System family – see by far the most traffic. Furthermore, port scans against port 80 (HTTP) could be observed quite often.
- The other offered services from the honeypot, e.g. FTP, SSH and HTTP serving PHP-Nuke, were not contacted very often. Only a few scans for world-writeable FTP-servers could be noticed. Connection attempts for the other two services occured also only seldomly. We weren't able to register more advanced attacks like SQL injection or cross site scripting.

After these first two months, we added some other machines to the honeynet as described in section 3.2. Particularly, we added some honeyds and a few virtual honeypots. In the following months we captured much more traffic, on average more than 100 MB of network traffic per week. Again, a short summarization of the obtained results will be given:

- With the help of the different honeyd-hosts we were able to collect lots of data. For example, more than 2.4 million con-

nection attempts could be monitored during a two week period between 07-21 and 08-04. Table 1 summarizes the quantity of connection attempts. During this time, almost 48,000 unique IP-adresses could be oberved. Fig. 3 gives an overview of the connection attempts in relation to the time of day.

Protocol	Number of packets
Total	2,404,766
TCP	2,010,921
UDP	363,230
ICMP	30,615

Fig. 4 Number of packets per protocol

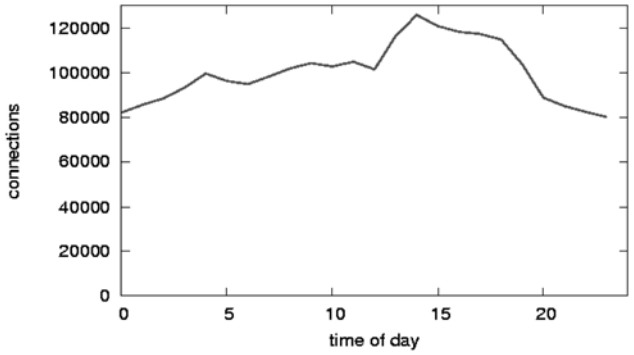


Fig. 5 Number of connections per hour

– The top 10 accessed ressources show clearly that Netbios dominates the traffic in our honeynet: By far the most traffic is generated by these ports. HTTP-traffic and ICMP echo-requests follow on the next positions. Port 1025/TCP is commonly used by Microsoft Windows as Remote Procedure Call (RPC) service. We also observed a lot of worm and malware traffic: Port 2745/TCP is commonly used by a number of variants of the Bagle worm as a backdoor port. Similarly, port 6129/TCP is used by the Dameware remote administration software, for which an exploit exists, and port 3127/TCP is used by the myDoom/Novarg worm as a backdoor port.

Rank	Resource	Number of connections
1	445/TCP	965,723
2	139/TCP	800,960
3	137/UDP	357,453
4	135/UDP	93,275
5	80/TCP	33,463
6	8/ICMP	28,694
7	1025/TCP	23,871
8	2745/TCP	23,566
9	6129/TCP	17,042
10	3127/TCP	16,178

Fig. 6 Number of connections per resource

– The top 10 source hosts show that some hosts generate lots of traffic and consistently send pakets to some hosts in our honeynet. A closer inspection of these hosts show that most of seem seem to be connected to the Internet via DSL, so they probably belong to dial-up users which are all infected with some kind of worm which tries to spread and thus generates lots of traffic.

Rank	Source IP address	Number of connections	DNS PTR record for source IP address
1	66.94.77.121	129,528	*unknown*
2	66.94.77.121	63,471	adsl-68-73-254-233.dsl.chcgil.ameritech.net
3	69.156.110.172	53,693	*unknown*
4	64.229.170.202	47,737	HSE-MTL-ppp64773.qc.sympatico.ca
5	64.144.104.227	40,296	64-144-104-227.client.dsl.net
6	68.254.25.41	27,993	*unknown*
7	64.228.68.148	27,856	HSE-Toronto-ppp130807.sympatico.ca
8	66.134.211.10	26,971	h-66-134-211-10.lsanca54.covad.net
9	64.229.170.226	24,963	HSE-MTL-ppp64797.qc.sympatico.ca
10	63.196.246.88	24,424	adsl-63-196-246-88.dsl.lsan03.pacbell.net

Fig. 7 Number of connections per Source IP

– When analysing the source addresses of packets arriving at the honeynet via reverse DNS lookups, we got some results which differ from those by Reiser and Volker. We did a reverse DNS lookup for about 48,000 hosts and about 42% of them failed, resulting in either NXDOMAIN or SERVFAIL. We could only observe a small percentage (about 4% of all systems) of hosts originating from t-dialin.net or t-ipconnect.de. Altogether, we found a more or less uniform distribution of source addresses: ne.jp (7.5%), verizon.net (4%) and hinet.net (2.5%) were the most common sources.
– With the help of honeyd and the tool p0f, an educated guess on the attacker's operating system can be undertaken: About 70% of all connection attempts could be associated with an operating system. More than 90% of these connection attempts were caused by a machine running Windows, whereas only about 3% could be identified as originating from Linux machines.

5 ETHICAL, LEGAL AND ECONOMIC ASPECTS

Traditionally the honeynet community discusses besides the purely technical aspects of honeynet technology to some extent the legal aspects of it. [17] We believe that a successful analysis of the legal issues arising from the use of honeypot technology can be only accomplished with prior evaluation of the ethical issues emerging from honeypot technology. While it might seem that the legal view on problems nowadays is completely detached from the ethic view, for subjects which lay outside the traditional realm of legal discussion it is usually a good way to first evaluate if a course of action is ethical before assessing the legal status of this action. Possibly some prospective honeynet operators are even more concerned to act unethically than to act illegally.

We will also discuss the Micro-economic effects of honeynets which will shed some light on the question to which extent the operator of a honeynet can expect some gain from the operation of the honeynet.

5.1 Ethical Aspects

When examining the ethical aspects of honeynet operation the relationship between honeynet deployment and the Internet as

a whole and the relationship between honeynet operation and the (unsolicited) users of the honeynet have to be considered.

Installing a honeynet usually means that the operator adds systems to the Internet which are not secured in a state-of-the art fashion. It could be argued that by doing so the total security of the Internet is being reduced. This means that the operator of a honeynet has some special responsibility to persons which systems are attacked by blackhats using the honeynet as a stepping stone. But considering the overall state of security on the Internet, even only loosely secured honeypot systems probably have a security well over the average security of an Internet system thus actually increasing the overall security of the Internet.

Furthermore one has to consider that (1) by deploying a honeypot the population of prospective victims is enlaged and thus the chance of getting attacked is reduced for each single system and (2) by strictly controlling outgoing traffic the percentage of systems which could be easily misused as a stepping stone for further attacks is reduced. On this basis it could be argued that by deploying a honeynet a third party's risk of becoming the victim of an attack is slightly reduced.

Also a honeynet is deployed in the hope that research results gained by deploying the system on the long run will help to make the Internet more secure. So ethically the deployment of a honeynet can be considered justifiable in relation to the Internet Community as a whole.

Evaluation of the ethical aspects of monitoring a non consenting party (the blackhat) without its knowledge is more difficult. Especially if one takes into account that a voyeuristic thrill might be some or even the principal motivation to some operators for deploying a honeynet. Generally experiments involving non consenting humans have to be rejected. Even if a subject is dealing ill this does not automatically allows us to make that person part of an experiment. Especially if we consider the huge international differences of attitudes and laws regarding appropriate behavior on the Internet. One might argue that for mostly technical actions of an attacker like installing tools and committing further attacks on other systems the immediate risk of this actions and the possible direct gain in security by observing the actions and readjusting defences accordingly might justify the spying on the non consenting individuals. But this evaluation is bound to change if the attacker is using the system for social interaction like IRC conversation with its peers: Observation of these usually would only result in very little direct gain in security. Meanwhile it can be argued that spying on communication between humans is a much deeper intrusion in privacy, than the observation of actions like installing tools. So there is very little argument that intrusion of deemed private conversation of an attacker by the honeynet operator can be ethically justified as long as you do not subscribe to the school of thought that an attacker has lost all rights by intruding in a system

5.2 Legal Aspects

The discussion of legal aspects of honeynet technology is dominated by the US-american view on justice and considers mostly wiretapping and related areas [17]. We are more concerned about criminal and civil liability. While the legal issues can't be discussed to their full extent in this article, we try to give an overview about the most pressing issues concerning German laws. While a blind transfer of our results to other jurisdictions is not possible, the reader might consider that many legal systems in South America and Asia are modeled after the German one – especially with respect to criminal law.

5.1.1 Criminal Law

If a honeynet is utilized to attack a third party the operator of the honeynet might be punishable for aid in that attack. But § 27 StGB (German criminal code) states that any punishable act of giving aid to a crime has to be intentional. Since the operator of a honeynet goes through great lengths so avoid beeing used as a stepping stone for further attacks by deploying connection limiting and an outtrusion prevention system we can't assume intent at the operators side.

Furthermore the operation of a system with the security level of a honeypot is socially accepted and is not an invitation to an attacker to misuse it. Many systems on the Internet are less secure than the typical honeypot.

The operator of a honeynet has to be aware that there is a certain chance that the honeypots are used by attackers to store information for which a form of information prohibition exists. Which types of information are banned varies from state to state but nowadays pictures displaying or resembling children in sexual context are one of the few near globally accepted forms of prohibited information. The operator of a honeynet has to carefully determine under which circumstances criminal liability from such prohibited information placed by others on the honeypots can result in criminal liability. The legal situation is extremely blurry and in some instances it might be better not to look into the content of data placed by attackers on the honeypots than to do otherwise.

5.1.2 Tort Law

The operator of a honeynet might be liable for damages occurring when attackers use the honeypot to harm the systems of third parties. The operator could be liable for neglecting to secure the honepot properly. The total lack of ligation regarding insecurely operating systems, applications, computers and the damages resulting from them indicates that it is socially acceptable to build and deploy systems even lacking the most basic security standards. In contrast to that the honeynet operator has various measures like outtrusion prevention and traffic limiting in place to avoid damages to third parties. Therefore the operation of a typical honeynet can't be considered neglect and and thus can't be the basis for civil liability. See [18] for further reference.

5.1.3 Privacy Law

The privacy aspects of honeynets are discussed to a great extent in the US american literature [17]. We did not focus very much on privacy matters since we assume that attackers generally are aware that machines are monitored in regular operation and forensically examined after an incident, so by conducting an attack they willfully accept that fact.

5.3 Economic Aspects

It is unclear to which extent there is economic gain in operating a honeynet. Recent research by one of the authors indicates that honeynets are unlikely to result in significant gain for average organisations. [19]. This might be different for organizations involved in security research, where honeynets might be parts of experimental setups. Still there is only a relatively low chance that honeynets will see highly skilled attackers and thus produce substantial qualitative gain in information.

5.4 Conclusion

Deploying honeypot technology can be generally ethically and legally justified although it seems that there is little economic justification on non-security research oriented organizations. There seem to be only very minor risks of criminal or civil liability to the operator of a honeynet.

Still monitoring of private conversations taking place on the honeynet raises serious ethical issues. Intruding into others peoples private conversations can most seldomly be considered ethical. Also such monitoring probably will be the first area where legal problems arise.

6 ASSESSMENT OF THE VALUE AND QUALITY OF DATA COLLECTED

To assess the quality of data collected by a honeynet, we have to assess the resolution and reliability of our tools used to monitor the honeynet. The honeywall should be able to log all network traffic to the honeypots with very high reliability. We assume that the honeywall fails safe, meaning that no further communication with the honeypots is possible in case of a failure of the honeywall. This allows very complete data collection capabilities as long as it is possible to decode the traversing network traffic. If the traffic can't be decoded – which is usually the case for encrypted payloads – data collected on the honeywall can be used only for traffic analysis.

Actions of attackers involving encrypted network traffic have to be logged on the honeynet itself. The dominant way to achieve this is the use of the Sebek rootkit as outlined above. Recent research indicates that Sebek is only of limited reliability and can be circumvented and deactivated by an attacker [8].

The overall results of our honeynet differ to a great extent from the results published elsewhere. So we were able to operate a machine running SuSE 6.2 for which dozens of known vulnerabilities exist for months without any compromise, while others have observed a time to compromise ranging from a few hours to a few days. Furthermore, we are not able to confirm some of the results by Reiser and Volker: Our results show that a smaller percentage of all attacks were originated from the IP-range of Deutsche Telekom. And we also have not seen any packets on port 57669, so the Stumbler/"Mysterium 55808" seems to be a temporary occurence. We are not aware of any reasons for this behavior and will investigate this topic in further research.

7 CONCLUSION AND FURTHER RESEARCH

Honeynets are generally considered a valuable tool for vulnerability assessment: They aid in collecting detailed information about attackers' behavior and help in analyzing their tools, techniques and motives. Drawing from our own experience at RWTH Aachen University, our honeynet indeed was able to collect a wealth of data on attack patterns which currently prevail in the Internet. However, our research has identified some questions which need further investigation.

A major focus of our further research is to get an understanding why the results of our honeynet differ that much from the results published by the operators of other honeynets. We will try to operate honeynets at different address ranges in the academic and commercial Internet. To do so successfully, we have to develop tools for easy and fast deployment, configuration management, logging aggregation and data mining. We will try to place honeynets near "interesting" targets in the hope to see more sophisticated attack on these honeynets.

To better represent the population of Internet hosts, computer systems with Microsoft Windows as operating systems in a typical client settings have to be deployed in the honeynet. This leads to several challenges whereas the biggest is how to simulate client like behavior and how to differentiate normal traffic from attacks. Also repeated attacks of autonomous malware have to be contained to keep the honeypots in an operational state.

Another big area needing further research are techniques for monitoring and forensics. One avenue of new possibilities for monitoring is the use of hardware based monitoring systems [20].

8 ACKNOWLEDGMENTS

We wish to thank the responsible staff at the Center for Computing and Communication at RWTH Aachen University, Jens Hektor and Christian Bischof, for actively supporting our project. Thanks also go to Lexi Pimenidis for his help in setting up the honeynet. Work by Thorsten Holz was supported by the Deutsche Forschungsgemeinschaft (DFG) as part of the Graduiertenkolleg "Software for mobile communication systems".

1. The Honeynet Project, Know Your Enemy: Defining Virtual Honeynets. 2003. Internet: http://www.honeynet.org/papers/virtual/
2. Stoll, C.: Stalking the wily hacker. CACM, 1988. 31 (5): p. 484-497. 1988.
3. Cheswick, W.: An Evening with Berferd in which a cracker is Lured, Endured, and Studied. USENIX proceedings, Jan 20, 1990.
4. The Honeynet Project: The Honeynet Project. 2004. Internet: http://www.honeynet.org/
5. The German Honeynet Project. 2004. Internet: http://www-i4.informatik.rwth-aachen.de/lufg/honeynet
6. The Honeynet Project, Know your Enemy: Sebek. 2003. Internet: http://www.honeynet.org/papers/sebek.pdf
7. Zovi, D.D.: Kernel Rootkits. 2001.
8. Dornseif, M.; Holz, T.; Klein, C.N.: NoSEBrEaK – Attacking Honeynets. In: Proceedings of Information Assurance Workshop. 2004. Westpoint, NJ: IEEE Press.
9. VMware Homepage. 2004. Internet: http://www.vmware.com/
10. User-Mode Linux. 2004. Internet: http://user-mode-linux.sourceforge.net/
11. Provos, N.: A Virtual Honeypot Framework. In: CITI Technical Report. 2003, Center for Information Technology Integration, University of Michigan: Ann Arbor.

12. CERT/CC: CERT Advisory CA-2002-01: Exploitation of Vulnerability in CDE Subprocess Control Service. 2002. Internet: http://www.cert.org/advisories/CA-2002-01.html
13. McCarty, B.: Botnets: Big and Bigger. IEEE Security & Privacy, 2003. 1 (4): p. 87-90.
14. McCarty, B.: Automated Identity Theft. IEEE Security & Privacy, 2003. 1 (5): p. 89-92.
15. The Honeynet Project: The Honeynet Project: Statistics. 2004. Internet: http://www.honeynet.org/papers/stats/
16. Spitzner, L.: Honeytokens: The Other Honeypot. 2003. Internet: http://www.securityfocus.com/infocus/1713
17. The Honeynet Project: Know Your Enemy: Learning about Security Threats. 2nd ed. 2004, Boston: Addison-Wesley. 2004, 768 p.
18. Dornseif, M.; Gärtner, F.; Holz, T.: Ermittlung von Verwundbarkeiten mit elektronischen Ködern. In: Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA 2004), GI SIG SIDAR Workshop, 2004. Dortmund: Lecture Notes in Informatics, volume 46.
19. Dornseif, M.; May, S.: Modelling the costs and benefits of Honeynets. In: The Third Annual Workshop on Economics and Information Security (WEIS '04). 2004. Minneapolis.
20. Carrier, B.; Grand, J.: A Hardware-Based Memory Acquisition Procedure for Digital Investigations. Digital Investigation Journal, 2004. 1 (1).