THE MITNICK CASE: HOW BAYES COULD HAVE HELP

Thomas DUVAL (DGA/CELAR, Supélec)
Bernard JOUGA (Supélec)
Laurent ROGER (DGA/CELAR)
Supélec: Avenue de la boulaie, BP 81127 F-35511 CESSON SEVIGNE FRANCE
DGA / CELAR: route de Laillé, F-35170 BRUZ FRANCE
thomas.duval@supelec.fr
bernard.jouga@supelec.fr
laurent.roger@dga.defense.gouv.fr

Abstract *Computer forensics provides systems administrators and investigators with tools and methods that help them to explain attacks, to clean up corrupted systems, and to unmask the offenders. This analysis is mainly based on investigators knowledge and experience, and poorly automated.*

We propose in this paper to use Bayesian networks to model the investigation process, and automate -under some conditions- computer investigations. Our model, XMeta, works with: 1) the knowledge of the networks and systems configurations and vulnerabilities, 2) evidence gathered on the crime scene and 3) previous investigations results. This information is stored in a database that is updated with the already solved cases, and used to give some assumptions about the current investigation.

The interest of our scheme is illustrated by processing a part of the well-known Kevin Mitnick case.

Keywords: Computer forensics, digital evidence, computer crime, Bayesian networks.

Introduction

”Is the attacker still active ?”, ”Where did he go ?”, ”How did he break in ?”, etc. Even if information systems are more and more safe, computer criminality increases [1], and the need for forensic analysis increases too. Computer Forensics can be defined as ”*the process of unearthing data of probative value from computers and information systems*” [2]. Computer Forensics helps to elucidate a case and unmask offenders.

When arriving on the crime scene, the investigators first job is to collect the maximum of data about the attacked organization and the compromised systems. The objectives are to learn about the organization activity, the technical architecture, the reported damages, the involved persons. Investigators have to collect and preserve technical materials: one of the first essential tasks is to record non-persistent system variables (like active logins, processes, opened files, etc.) and to make images of (non-)removable media. After the first findings, the investigators come back to the lab with the collected devices that need deeper analysis. Those analysis help to find important elements like:

- presence of specific data or softwares (like Trojan horse),
- commands history,
- configuration files misuse,
- hidden network channels on hosts, communication and security equipments,
- hidden or deleted files,
- etc.

Computers being connected with networks in an open environment, the investigators must then jump from one compromised host to another, in order to trace the attack back to its source. Besides these technical investigations, other information such as the context of the accident and testimonies are also important.

Endly, investigators must write a report with all the gathered observations and evidence. This report must be clear enough to be understood by lawyers (they may not understand some technical details), but also complete in order to accurately record the steps of the investigation.

Those records must include:

- all devices involved,
- softwares configurations,
- dates and times of attacks,
- evidence files,
- investigators identities,
- investigation description,
- etc.

In this paper, we propose a system which helps investigators in their inquiries by giving them some assumptions about the crime. It is based on an expert system which uses Bayesian networks for its inference engine. The Bayesian approach has been chosen for three reasons:

- it can model complex situations,
- it offers a non-binary (probabilistic) logic,
- it is resistant when data are unknown or unsure.

The first section summarizes the use of Bayesian networks in security related applications. The second section presents some related works, then in the third section we present our scheme. The fourth section shows how investigators can use the XMeta system to investigate a true criminal case. Finally, we conclude in the last section.

1. Bayesian networks

Bayesian networks (BN) have been already widely used in expert systems (SpamAssassin [3]) and for some forensics or IDS (Intrusion Detection System) applications [4–5]. BN are directed acyclic graphs where nodes are variables and links are causal connections weighted with conditional probabilities [6]. A Bayesian Network can model situations of which the perception is partial. For example, an administrator notices that his Apache Web Server A has been defaced. There are only two causes (in this example) for this situation:

- the attacker used a exploit E,
- the attacker stole the administrator's password S.

A well-formed Bayesian network will propose an answer in a statistical format: There is X chance in a hundred to perform the exploit E on a Web Server A, the probability value been given by the Bayes Theorem:

$$P(E \mid A) = \frac{P(E, A)}{P(A)}$$

with:

- $P(E \mid A)$ the probability to perform the exploit E knowing that we have a Web Server A,
- $P(E, A)$ the probability to have an exploit E and a Web server A,
- $P(E)$ the probability to have an exploit E.

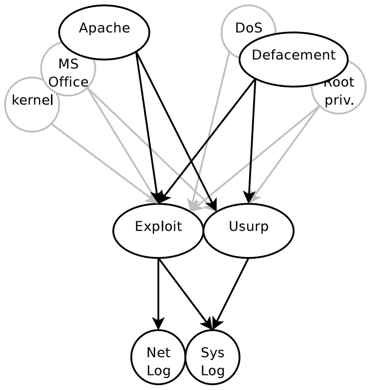

Figure 1. An example of a Bayesian Network

The construction of a Bayesian Network follows three steps:

a) construction of the causal graph,

b) construction of probability tables associated to nodes,

c) propagation of probabilities in the network.

Two methods can be used to build a network: the first one consists in interviewing experts of the domain, who will set causality links and probability values. In the second method, the network structure and probabilities are learned from a cases database. Inference can be not only top-down but also bottom-up. For example in figure 1, if we set that a DoS (Denial of Service) has occurred, the values of Exploit and Usurp nodes will change in accordance with the DoS observation, and the values of softwares nodes (at the upper left side) will also change, reflecting the most vulnerable softwares for a DoS attack.

2. Related works on computer forensics and Bayesian networks

Several projects make use of Bayesian networks for security issues. In [7], the authors analyze systems to find which relevant data can be gathered in order to reduce data amount and infer (with a Bayesian network) communications between several hosts. This project only deals with communications between hosts. In our proposal we not only look for events between systems but also for events inside hosts.

In [8], Levitt and Laskey present a research on non-computer crimes. Their model of the crime scene is made of 4 elements: involved persons, entities,

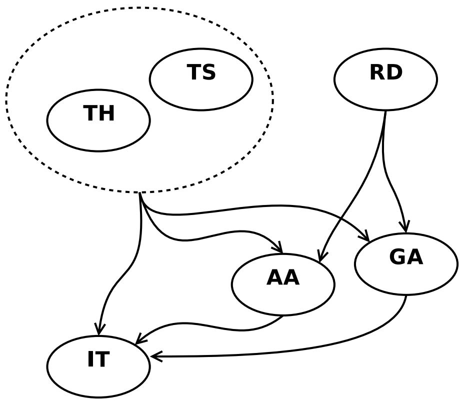

Figure 2. Investigation Plan structure

events and context. It is able to propose assumptions about guilt and the scheme of the crime. We have the same final goal, unmask offenders, and our proposal is designed to help investigators in their research on:

- attack steps and chronology,
- evidence underscoring,
- attacker(s) profile.

3. Our proposal

3.1 The XMeta system

A system compromised by a particular attack is modeled with an Investigation Plan (IP). It is a Bayesian network which is built on demand at the beginning of the system analysis, taking into account the system configuration. Figure 2 shows the IP structure. A typical IP is composed of 40-50 nodes classified in 6 types (a taxonomy of attacks and actions is given in annex):

- the Targeted Hardware: **TH**,
- the Targeted Softwares: **TS**,
- the Reported Damages: **RD**
- the Generic Attacks: **GA** (21 possible attacks, as listed by the DGA[1]),
- the Additional Actions: **AA** (12 possible actions listed by the DGA),

Variable name	Description
image	make a forensic copy of media
syst_check	check system log files
net_check	check network log files (like firewall logs)
syst_var	check system variables (like logins, processes, etc.)
retrieve	retrieve hidden or deleted files
net_log	use a sniffer to listen to the network for attacker's next actions
int_topo	check the compromised network topology
ext_topo	check the compromised network interconnections
comm	analyze communications, like IRC and mail logs
physic	analyze physical access to a computer or a building.

Table 1. Investigation techniques

- the Source Address: (the address can be local: the same host, internal: the same network or external),

- Investigation Techniques to use: **IT**.

An investigator creates a new investigation plan by entering the host configuration (**TH** and **TS**) and the observed damages (**RD**). The proposed values are extracted from the ICAT vulnerability database [9], including data about more than 7,000 softwares. A database of previous cases is used to set causality links and probabilities values (*k2* learning algorithm [10–11]). Inference on this network (*Likelihood Weighting* approximative inference [12]) produces some clues about the compromise system by giving **GA** and **AA**. **GA** are mandatory to compromise a host, **AA** are not necessarily found but can help investigators (a "I 0wn y0u!" message was sent, a new login was created, etc.). Finally, XMeta can propose the investigation techniques to use for in depth analysis: **IT**. Our definition for these techniques is given in Table 1.

When the investigator checks a particular attack or action, he can set it into the system, which changes its values thanks to this new fact.

When a host has been entirely checked (the source of **this** attack has been found):

1 the source attack address is local,

   a) the attacker had a legitimate access to the computer, then the technical investigation is complete,

   b) the attacker gained an access before launching his attack, the investigation continues, a new investigation plan is built with the same software configuration.

2 the source attack address is not local (internal or external) and then the follow-up depends on the next host accessibility for analysis.

Investigators can create as much investigation plans as needed, and link them to reflect the attack progression (multiple links are allowed).

3.2 Testbed

A testbed, XMeta, has been developed, using ”Bayesian Tools in Java” [13]for inference and a Python/GTK based GUI (a new version under development is only based on Python and GTK). The initialization of the inference system should be done with previous investigations results, but at this time we miss real data. In a first step, we used the ICAT vulnerability database [9]. This database gives us relations between softwares and losses. It is also parsed to extract attacks and actions. Then, we apply a bayesian learning algorithm to the extracted data. We choose a *K2* algorithm because it gives better results in terms of speed and accuracy. The only facts that cannot be extracted from ICAT are investigation techniques. The first version of our testbed only gives the most vulnerable softwares (in fact, we obtain the softwares which have the greatest number of vulnerabilities, depending on observed losses) and possible attacks and actions of the attacker(s).

For example here is a (fictitious) case where confidential information has been stolen from a workstation. The compromised host ran a Linux Debian (with no patch), the Investigation Plan has been initialized with Debian softwares (kernel, libc, Windowmaker, OpenSSH, etc.) and a confidentiality loss. XMeta answers that the source of the attack was probably local to the machine (it was true) and the most vulnerable softwares were XFree86, the Linux libc and the kernel (the real compromised software was the kernel). Endly, the attacker used an exploit to become root and to copy the stolen file. This attack was in the 7th position in the answer (over 21 possible attacks). In the case of a real investigation, all the attacks should have been checked. Depending on the context, some of them could be quickly dismissed (for example, a DoS attack has no sense for a data theft).

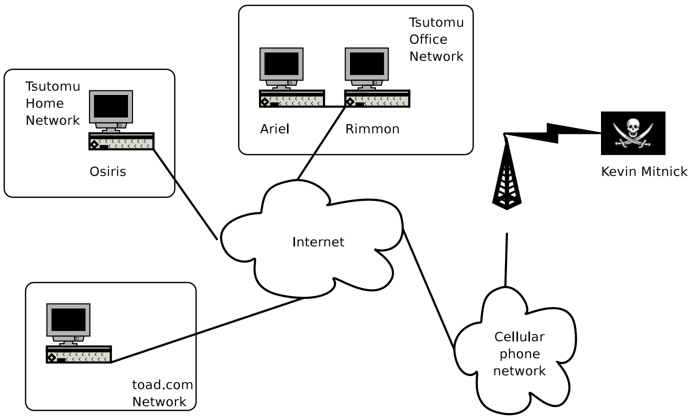

Figure 3. Networks and hosts involved in the Mitnick case

After some of these trivial case studies, the database of our inference system has been updated with new records (including investigation techniques) and XMeta now gives better results for similar cases.

4. The Kevin Mitnick case

4.1 Introduction

We propose to revisit with XMeta some parts of the Kevin Mitnick case. This case is one of the best-known computer crime. It has been widely described by Tsutomu Shimomura [14], the owner of the compromised systems, who helped investigators to arrest Mitnick. The case is interesting because of its complexity and the number of involved systems. Tsutomu Shimomura is a Senior Fellow at the San Diego Supercomputer Center. He has been working on various research topics, including computer security. In 1994, an unknown attacker hacked Tsutomu's computers. After 7 weeks of intensive tracking, Tsutomu located Mitnick in Raleigh, North Carolina. Kevin Mitnick was arrested on Februar, 14th of 1995. He was charged with breaking into some of the United States' most secure computer systems.

4.2 Investigation

When Tsutomu's computers were hacked, first clues were found on *Ariel*, one of his computers. A lot of network scans have been done on Ariel's network in the previous night. A large file (*oki.tar.Z*) had been created, transfered from *Ariel* to an unknown address and then deleted. It was later discovered that this file contained confidential data about cellphone firmware.
This information and the software configuration has been entered in XMeta

```
14:09:32 toad.com# finger -l @target
14:10:21 toad.com# finger -l @server
14:10:50 toad.com# finger -l root@server
14:11:07 toad.com# finger -l @x-terminal
14:11:38 toad.com# showmount -e x-terminal
14:11:49 toad.com# rpcinfo -p x-terminal
14:12:05 toad.com# finger -l root@x-terminal
```

Figure 4. first logs from *toad.com*; target is *Ariel*, server is *Rimmon* and x-terminal is *Osiris*

(as we have not a lot of precision about installed softwares, the list may be uncomplete or inexact):

> **Software:** SunOS, GNU Tar, GNU Ghostscript, fingerd, ruserd and FTP
> **Losses:** LT_Confidentiality

XMeta answers:

> **ARIEL**:
> The most probable **attacks** are: security bypass (65%), diversion (56%) and brut_force (56%).
> The most probable **additional actions** are: file infection (83%), detection inhibition (81%) and login account creation (71%).
> The highlighted softwares are: GNU Tar (73%), Finger Service (73%) and FTP (27%).
> Finally, no investigation techniques are proposed (no similar case in our dedicated database at this time).

Those answers imply to check the suspicious applications log files. Given the three listed attacks, investigators can infer that the attack comes from outside, this assumption being strengthened by the observed massive scans. The attacker has probably bypassed the security between *Ariel* and another computer, or made a diversion to upload the datafile (the brute force attack can be dismissed because an attacker cannot enter a system with such an attack). The presence of the brute force attack can be easily explained: our database (built with the ICAT database plus dedicated data) contains at this time mainly records extracted from ICAT. Given those softwares and loss, one of the three most frequent attacks in ICAT is the brut force attack. We have the same problem with the vulnerable softwares, as explained in section 1.3.2. This *malfunction* should disappear with the improvement of our database.

Tsutomu found that a lot of network scans came from a host in the Internet domain *toad.com* (Figure 4 from [15]). He also found that his computer *Osiris* had a strange behavior: there were blank windows on the top of the screen. The facts are:

- *Ariel* and *Osiris* had strong relationships,
- *Osiris* stands in Tsutomu's home (and has no direct connection with *toad.com*),
- a lot of network traffic had also been observed towards *Osiris* (figure 5).

```
14:18:25.906002 apollo.it.luc.edu.1000 > x-terminal.shell: S 1382726990:1382726990(0) win 4096
14:18:26.094731 x-terminal.shell > apollo.it.luc.edu.1000: S 2021824000:2021824000(0) ack 1382726991 win 4096
14:18:26.172394 apollo.it.luc.edu.1000 > x-terminal.shell: R 1382726991:1382726991(0) win 0
14:18:26.507560 apollo.it.luc.edu.999 > x-terminal.shell: S 1382726991:1382726991(0) win 4096
14:18:26.694691 x-terminal.shell > apollo.it.luc.edu.999: S 2021952000:2021952000(0) ack 1382726992 win 4096
14:18:26.775037 apollo.it.luc.edu.999 > x-terminal.shell: R 1382726992:1382726992(0) win 0
...
```

Figure 5. logs from *Osiris*

```
14:18:22.516699 130.92.6.97.600 > server.login: S 1382726960:1382726960(0) win 4096
14:18:22.566069 130.92.6.97.601 > server.login: S 1382726961:1382726961(0) win 4096
14:18:22.744477 130.92.6.97.602 > server.login: S 1382726962:1382726962(0) win 4096
14:18:22.830111 130.92.6.97.603 > server.login: S 1382726963:1382726963(0) win 4096
14:18:22.886128 130.92.6.97.604 > server.login: S 1382726964:1382726964(0) win 4096
...
```

Figure 6. logs from *Rimmon*

This implies that the investigation must continue on *Osiris* and not (for the moment) on *toad.com*. *Osiris* may be the source of the attack (or at least an intermediate system).

A new Investigation Plan is created for *Osiris*. Tsutomu discovered that this computer seems to be disconnected from his office network and especially from *Ariel*.

> **OSIRIS**:
> **Software**: SunOS, GNU Tar, GNU Ghostscript, fingerd, ruserd and FTP
> **Losses**: LT_Availability

The XMeta System answers:

> The most probable **attacks** are: repeat (for example: scanning sweeping, 100%), overrun (DOS, DDOS, smurf, fraggle, etc., 89%) and bypass (68%).
> The most probable **additional actions** are: infection (73%), trap (backdoor, 62%) and data deletion (45%).
> The highlighted softwares are: FTP (73%), GNU Tar (38%) and GNU Ghostscript (38%).

This implies that *Osiris* may only be an intermediate system, because an attacker cannot penetrate into a host using scanning sweeping or overrun. Those results encouraged us to search for an other computer.

Osiris is a X-Window terminal connected to *Rimmon*. Maybe the terminal server was also attacked. This was confirmed by the logs found by Tsutomu (figure 6).

As we miss information on *Rimmon*, we suppose that it is configured like *Osiris* and *Ariel*. Tsutomu discovered that an unauthorized user succeeded to install on *Rimmon* a kernel module named *Tap 2.01*, this implying having root privilege (figure 7).

> **RIMMON**:
> **Software**: SunOS, GNU Tar, GNU Ghostscript, fingerd, ruserd and FTP
> **Losses**: LT_Obtain_all_priv and LT_Availability

XMeta answers:

> The most probable **attacks** are: Trojan (93%), bypass (78%) and brut_force (58%).

```
x-terminal% modstat
Id  Type  Loadaddr      Size   B-major  C-major  Sysnum   Mod Name
 1  Pdrv  ff050000      1000              59.             tap/tap-2.01 alpha

x-terminal% ls -l /dev/tap
crwxrwxrwx  1 root      37,  59 Dec 25 14:40 /dev/tap
```

Figure 7. Some system variables of *Rimmon*

> The most probable **additional actions** are: login installation (58%), infection (51%) and trap (46%).
> The highlighted softwares are: FTP (59%) and GNU Tar (41%).

From those results, we can infer that if we don't find a Trojan in *Rimmon*, the computer was used as an intermediate system like *Osiris*. Since Tsutomu did not find a Trojan but a flooding attack (known as overrun in XMeta), we can strongly think that *Rimmon* was also an intermediate system used to gain access to *Osiris* and *Ariel*. The overrun attack is in the 10^{th} position[2] over 21. But Tsutomu also found that the attacker installed a kernel module, so he had a root access to *Rimmon*.

4.3 Results of XMeta

We found the following elements:

- a file *oki.tar.Z* has been transfered from *Ariel* to an unknown address, using a bypass attack or a diversion,

- a host in *toad.com* has been used to scan Tsutomu's networks,

- the attacker used the repeat attack on *Osiris* to obtain information, or bypassed security to enter this host,

- the attacker exploited the strong trusted relationship between *Osiris* and *Rimmon* to access to *Osiris*,

- the attacker was able to install a kernel module and this module was certainly used to access to *Ariel*.

Those elements can be saved, giving to future investigators or observers the ability to replay the attack or the investigation, in a trial for example. We have defined a XML-based data format, CFXR, Computer Forensics XML Report, that can save system configuration, hacker's attacks and additional actions, investigation techniques, attack and investigation progression.

The next steps in this investigation would be to precise how the attacker gained a root access on *Osiris* and on the host in *toad.com*, and what was the destination of the file *oki.tar.Z*.

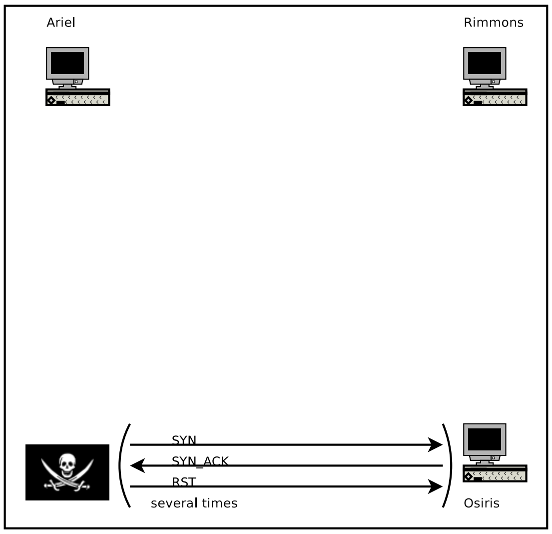

Figure 8. Kevin Mitnick attack: first step

Results of Tsutomu's investigations

The attack, as described by Tsutomu Shimomura, was divided into three parts. In a first step, the attacker tried to guess initial TCP sequence numbers for *Osiris* incoming connections, by sending a lot of SYN packets immediately followed by a connection reset (figure 8).

In a second step, he was able to spoof *Rimmon* for opening a connection with *Osiris*, and then issue with *rsh* the command ”`echo ++ >>/.rhosts`”, gaining root privilege. This attack is known as *security bypass* in XMeta. We have seen that it was in third position with a percentage of 68%. Flooding (known as *overrun* in XMeta) was used to gag *Rimmon* during the three-way handshake when establishing the TCP connection with *Osiris* (figure 9).

In the third step, the attacker installed the software ”*Tap*” and used it to hack the connection between *Osiris* and *Ariel* (figure 10). He gained an access to *Ariel*, and created and downloaded the *oki.tar.Z* file.

Conclusion and future works

We propose an expert system that can infer on computer and network attacks by giving some clues about attacks and additional actions performed by the attacker, the most vulnerable softwares and some investigation techniques to use. The Kevin Mitnick case shows how or scheme can apply on a non trivial case. Efforts must still be done to improve our database, in order to have more accurate results (in particular for investigation techniques). XMeta can

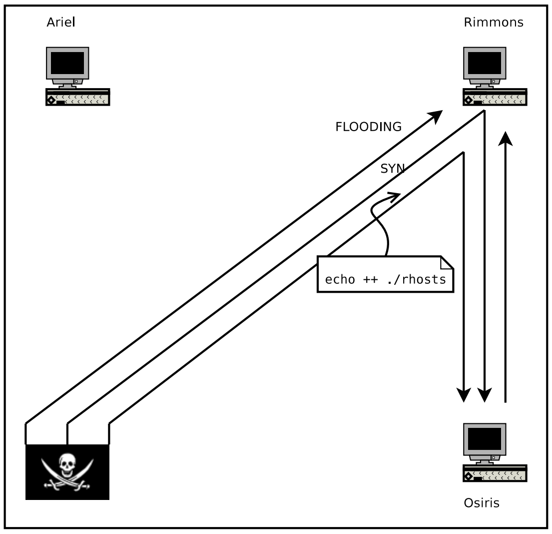

Figure 9. Kevin Mitnick attack: second step

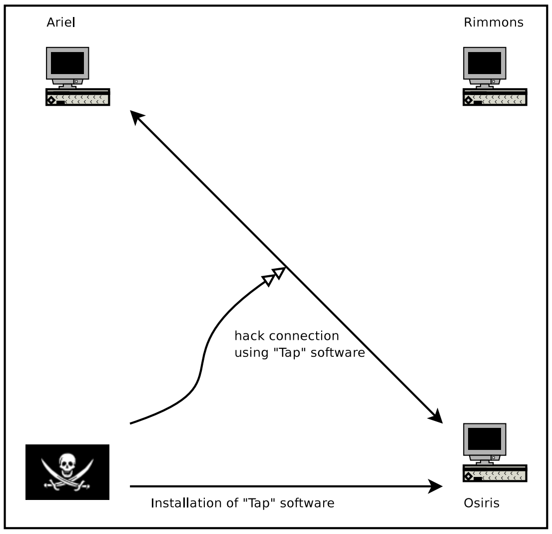

Figure 10. Kevin Mitnick attack: third step

help investigators, and it is also able to save essential data about the attack progression and the inquiry.

We also work on an other way to make assumptions more accurate, with the creation of a criminal profile which can help to find the preferred targets and attacks of an attacker.

Notes

1. French Ministry of Defense

2. from 1 to 10: Trojan, bypass, brute_force, broadcast, chaff, repeat, intercept, net_listen, bounce and overrun

References

[1] CCIPS. Computer Crime and Intellectual Property Section of the Criminal Division of the US Department of Justice. http://cybercrime.gov.

[2] Kevin Mandia and Chris Prosise. *Incident Response: Investigating Computer Crime*. McGraw-Hill Osborne Media, June 2001. ISBN: 0072131829.

[3] Spamassassin. http://spamassassin.apache.org. An extensible email filter used to identify spam.

[4] D. J. Burroughs, L. F. Wilson, and G. V. Cybenko. Analysis of distributed intrusion detection systems using bayesian methods. In *IPCCC 2002*, April 2002.

[5] Alan M. Christie. The incident detection, analysis, and response (idar) project. Technical report, CERT Coordination Center, July 2002. http://www.cert.org/idar.

[6] E. Charniak. Bayesian networks without tears. *The American Association for Artificial Intelligence*, 1991.

[7] Paulo Costa, Jim Jones, Billy Liao, and Vijay Malgari. A system for collection, storage, and analysis of multi-platform computer system data. Technical report, George Mason University, 2003.

[8] Tod S. Levitt and Kathryn Blackmond Laskey. Computational inference for evidential reasoning in support of judicial proof. In *Symposium - Artificial Intelligence and Judiciary Proofs*, 2000.

[9] ICAT vulnerability database . http://icat.nist.gov/icat.cfm.

[10] G. Cooper and E. Herskovits. A bayesian method for the induction of probabilistic networks from data. *Machine Learning, (9) pp. 309-347*, 1992.

[11] Patrick Naïm, Pierre-Henri Wuillemin, Philippe Leray, Olivier Pourret, and Anna Becker. *Réseaux bayésiens*. Eyrolles, 2004.

[12] Fung R. and Chang K.C. Weighting and integrating evidence for stochastic simulation in bayesian network. In *UAI, volume 5, pp. 209-219*. Elsevier science publishing company, Inc, 1989.

[13] William H. Hsu. BNJ - Bayesian Network tools in Java. http://bndev.sourceforge.net.

[14] Tsutomu Shimomura with John Markov. *Takedown*. New York: Hyperion Press, 1996.

[15] Tsutomu Shimomura. Technical details of the attack described by markoff in nyt. Article: 14059 of comp.security.misc. 25 Jan 1995 04:36:37 -0800.

Annex: Attacks and actions nodes taxonomy

Variable name	Description
listing	list a DNS entry for example
net_listen	listen the network to get password, etc.
decrypt	use a dictonnary or a brut force attack to find passwords
exploit	use an exploit to enter a system (Buffer Overflow)
bypass	bypass a security element
broadcast	find computers with broadcast packets (example: ping)
chaff	use a fake server to steal information
embezzlement	example: Man In the Middle
listen	listen host events
parasit	transform software functionnality to restrain them
degrade	alter a network-based or host-based service (like web defacement)
diversion	use a diversion
intercept	intercept data which was destinated to someone else
usurp	use the identity of somebody without his/her agreement
bounce	log into multiple hosts before attacking
trojan	use a trojan horse to install a software
repeat	for example: scanning sweeping
blocking	block the functionnality of a network service
overrun	for example: DOS, DDOS
brut_force	use force brute attack
control	intercept and block an host

Table 2. Attack nodes

Variable name	Description
msg	send a message to sign an attack
attribute	privileges escaladation
scan_use	find host services by scanning this host
encrypt	encrypt data
hidden_channel	use a weakness in a protocol to send data
infection	add information in a file (example: steganography)
illic_cnx	connect to an host without agreement
trap	use a trap door
invert_trap	use an inverted trap door
inhib_detect	inhibit detection (example: IP Spoofing)
del	delete data
login_inst	install a new login

Table 3. Action nodes