Chapter 8

THE MITNICK CASE: HOW BAYES COULD HAVE HELPED

Thomas Duval, Bernard Jouga and Laurent Roger

Abstract Digital forensics seeks to explain how an attack occurred and who perpetrated the attack. The process relies primarily on the investigator's knowledge, skill and experience, and is not easily automated. This paper uses Bayesian networks to model the investigative process, with the goal of automating forensic investigations. The methodology engages digital evidence acquired from compromised systems, knowledge about their configurations and vulnerabilities, and the results of previous investigations. All this information is stored in a database that provides a context for an investigation. The utility of the methodology is illustrated by applying it to the well-known Kevin Mitnick case.

Keywords: Computer crime investigations, Bayesian networks

1. Introduction

Two important goals in digital forensic investigations are to explain definitively how a computer system or network was attacked and to identify the perpetrators [9, 15]. The investigative process has certain subjective elements, which draw on the investigator's knowledge, skill and experience [8], and are not easily automated.

This paper uses Bayesian networks to model the investigative process, with the goal of automating forensic investigations. The XMeta system described in this paper engages digital evidence from compromised systems, knowledge about their configurations and vulnerabilities, and the results of previous investigations. Given the facts of a case, XMeta reasons about the situation, providing information about likely attacks, additional actions performed by attackers, the most vulnerable software systems, and the investigation techniques that should be used.

The following sections discuss Bayesian networks, and the XMeta model and implementation. The methodology is illustrated by applying it to the well-known Kevin Mitnick case [12, 13].

2. Bayesian Networks

Bayesian networks are directed acyclic graphs whose nodes are variables and links are causal connections weighted with conditional probabilities [2]. Bayesian networks are useful for modeling complex situations for which information is incomplete and/or uncertain.

For example, suppose a web page on an Apache server A has been defaced. Assume that there are two possible causes: (i) the attacker used an exploit E, or (ii) the attacker stole the administrator's password S. A well-formed Bayesian network provides a probabilistic answer. Specifically, according to Bayes' Theorem, the probability that an exploit E is performed given the presence of web server A is:

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

where $P(E, A)$ is the probability of having exploit E and web server A, and $P(E)$ is the probability of having exploit E.

The construction of a Bayesian network involves three steps: (i) constructing a causal graph, (ii) constructing probability tables associated with nodes, and (iii) propagating probabilities. A Bayesian network is typically constructed by interviewing domain experts to obtain information about nodes, causality links and probability values. Alternatively, the network structure and probabilities may be learned from examples, e.g., from a cases database.

An example Bayesian network is presented in Figure 1. In this network, if a `DoS` attack has occurred in addition to an `Apache` server compromise and a `web defacement`, the probability values of the `Exploit` and `Usurp` nodes will change accordingly. Also, the probability values of the software nodes (e.g., `MS Office`) will change to reflect their vulnerability to the `DoS` attack. Thus, inferences in a Bayesian network proceed in the top-down and bottom-up directions.

Bayesian networks have been already widely used in expert systems, including several security and forensics applications [1, 3, 14]. Costa and co-workers [5] have used Bayesian networks to reason about communications between hosts. Our work deals with communications between systems as well as system events.

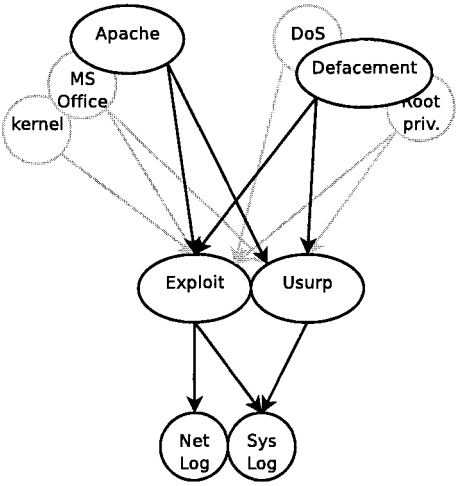

Figure 1. Bayesian network.

3. XMeta Model

XMeta uses a Bayesian network to model and reason about computer systems and networks that are compromised by attacks. A system compromised by a particular attack is modeled using an investigation plan. An investigation plan is a Bayesian network built on demand at the start of system analysis, which takes into account the system configuration.

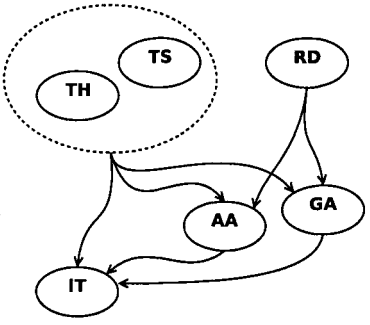

Figure 2. Investigation plan.

Figure 2 shows the structure of an investigation plan. It has six types of nodes: targeted hardware (TH), targeted software (TS), reported damage (RD), generic attacks (GA), additional actions (AA), and investigation techniques (IT).

Table 1. Attack nodes.

Variable	Description
listing	List a DNS entry (example)
net_listen	Listen on the network to get a password (example)
decrypt	Use a dictionary or a brute force attack to obtain passwords
exploit	Use an exploit to enter a system (e.g., buffer overflow)
bypass	Bypass a security element
broadcast	Broadcast packets (e.g., ping)
chaff	Use a fake server to steal information
embezzlement	Man-in-the-middle (example)
listen	Listen for host events
parasite	Transform software functionality
degrade	Alter a network/host-based service (e.g., web defacement)
diversion	Use a diversion
intercept	Intercept data
usurp	Use someone else's identity
bounce	Log into multiple hosts before attacking
trojan	Use a Trojan horse to install software
repeat	Scanning sweeping (example)
blocking	Block a network service
overrun	DoS, DDoS (examples)
brute_force	Use a brute force attack
control	Intercept and block a host

Table 2. Action nodes.

Variable	Description
msg	Send a message that signifies an attack
attribute	Escalate privileges
scan_use	Find host services by scanning a host
encrypt	Encrypt data
hidden_channel	Use a protocol weakness to send data
infection	Add information in a file (e.g., steganography)
illic_cnx	Connect to a host without approval
trap	Use a trap door
invert_trap	Use an inverted trap door
inhib_detect	Inhibit detection (e.g., IP spoofing)
del	Delete data
login_inst	Install a new login

A typical investigation plan has 40 to 50 nodes. The French Ministry of Defense (DGA) has identified 21 possible generic attacks (Table 1). Also, DGA has listed a set of 12 additional actions (Table 2).

Table 3. Investigation techniques.

Variable	Description
`image`	Make a forensic copy of media
`syst_check`	Check system log files
`net_check`	Check network log files (e.g., firewall logs)
`syst_var`	Check system variables (e.g., logins, processes)
`retrieve`	Retrieve hidden or deleted files
`net_log`	Use a sniffer to listen to attackers' actions
`int_topo`	Check the compromised network topology
`ext_topo`	Check the compromised network interconnections
`comm`	Analyze communications (e.g., IRC, mail logs)
`physic`	Analyze physical access to the computer

A new investigation plan is created by entering the host configuration (TH and TS) and the observed damage (RD). Much of this information can be obtained from the ICAT vulnerability database [11], which contains data about more than 7,000 software systems. A database of previous cases is used to set causality links and probability values (via the K2 learning algorithm [4, 10]). The Bayesian network uses the likelihood weighting approximate inference technique [6] to reason about attacks (GA) in Table 1 and actions (AA) in Table 2. Attacks (GA) are mandatory to compromise a host. On the other hand, actions (AA) are not mandatory, although their presence can assist investigations (e.g., an "I 0wn y0u!" message was sent, or a new login was created). Based on the data provided, XMeta proposes the investigation techniques (IT) that may be used. The list of investigation techniques is presented in Table 3.

When an investigator checks a particular attack or action, this fact is entered in the system, which correspondingly adjusts the values in the Bayesian network. When a host has been checked completely, i.e., the source of the attack has been identified, the following logical process is followed.

- If the source attack address is local, and
  - If the attacker had legitimate access, then the investigation is complete.
  - If the attacker gained access before launching the attack, then the investigation must continue and a new investigation plan is created using the same software configuration.

- If the source attack address is not local (i.e., internal or external), then the next step in the investigation depends on whether or not the next host is accessible for investigation.

Investigators can create as many investigation plans as needed. These plans may be linked to reflect the attack progression (i.e., multiple links are allowed).

4. XMeta Testbed

An XMeta testbed was developed using a Bayesian toolkit [7] for inference and a Python/GTK-based GUI. A newer version of the XMeta testbed, currently under development, is only based on Python/GTK.

Ideally, a Bayesian inference system should be initialized using the results of previous investigations. However, in the absence of real data, the ICAT vulnerability database [11] was used. The database provided information about software vulnerabilities and losses, as well as attacks and actions. Next, the K2 learning algorithm [4, 10] was used to set causality links and probability values. The only facts that could not be extracted from ICAT pertained to investigation techniques.

The initial version of the XMeta testbed only considered the most vulnerable software systems (based on the number of vulnerabilities and observed losses), and possible attacks and actions of attackers. In the following, a fictitious case is used to demonstrate the types of results obtained with the initial testbed.

Consider a situation where confidential information was stolen from a workstation. The compromised host ran a Linux Debian (without patches). The investigation plan was initialized with Debian software (kernel, libc, Windowmaker, OpenSSH) and a confidentiality loss. The XMeta system indicated that the source of the attack was probably local to the machine, which was true. XMeta also indicated the most vulnerable software systems were XFree86, Linux libc and the kernel (the kernel was actually compromised). Finally, XMeta identified the actual attack (the attacker used an exploit to become root and copy the stolen file) as the seventh most likely of the 21 possible attacks (see Table 1). Clearly, all 21 attacks should checked in detail in a real investigation. Depending on the context, some attacks can be eliminated, e.g, a DoS attack is not relevant to a data theft investigation.

Since the initial implementation, the XMeta database has been augmented significantly, including adding information about investigation techniques. The results of the current XMeta testbed are much better than those of the initial version.

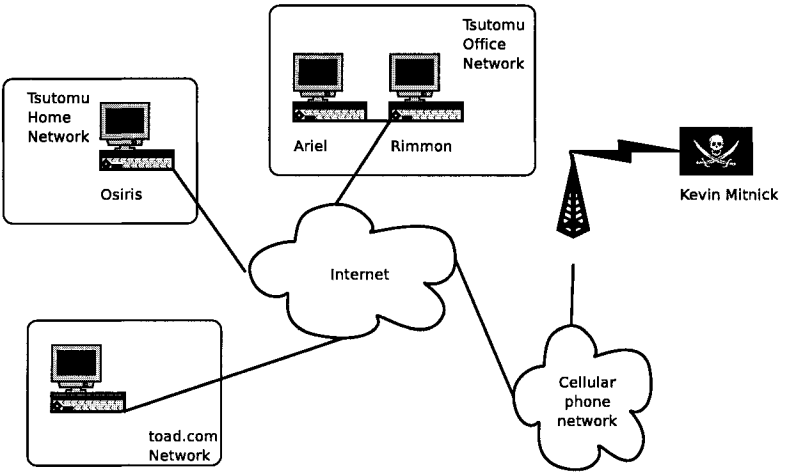

Figure 3. Computer system involved in the Kevin Mitnick case.

5. The Kevin Mitnick Case

In 1994, an unknown attacker hacked into computers at the San Diego Supercomputer Center. After seven weeks of intensive investigations, Tsutomu Shimomura, who worked at the center, tracked the perpetrator, Kevin Mitnick, to an apartment in Raleigh, North Carolina. Mitnick was arrested on February, 14 1995. He was convicted of breaking into some of the United States' most secure computer systems [12, 13].

This section describes the application of XMeta to the Mitnick case. The case is interesting because of its complexity and the number of systems involved. It provides an excellent context for comparing XMeta's results with those from Shimomura's original investigation.

5.1 The Mitnick Investigation

Upon examining compromised systems at the San Diego Supercomputer Center, Shimomura noticed that numerous network scans had been conducted the previous night. One of the first clues was found on a computer named `Ariel`. A large file (`oki.tar.Z`) had been created. This file was transferred to an unknown address, and then deleted from `Ariel`. It was later discovered that `oki.tar.Z` contained confidential data about cell phone firmware.

The following information pertaining to the Mitnick investigation was provided to XMeta. Note that the information was obtained from public sources [12, 13], and is incomplete and/or inexact.

Software: SunOS, GNU tar, GNU ghostscript, fingerd, ruserd, ftp
Losses: LT_Confidentiality

The following results were provided by XMeta:

Ariel:
The probable **attacks** are: `bypass` (65%), `diversion` (56%), `brute_force` (56%).
The probable **additional actions** are: `infection` (83%), `inhib_detect` (81%), `login_inst` (71%).
The highlighted **software systems** are: GNU tar (73%), finger service (73%), ftp (27%).
The proposed **investigation techniques** are: none.

Note that investigation techniques were not proposed by XMeta because a similar case did not exist in its database at the time.

Xmeta's answers indicate that the log files should be checked for suspicious applications. Based on the three attacks that are listed, one might infer that the attack came from outside (this assumption is strengthened by the network scans that were observed). The attacker probably bypassed the security between `Ariel` and another computer, or made a diversion to upload the data file. The brute force attack can be dismissed because it is not possible to enter a system using such an attack. Note that XMeta indicated a brute force attack was possible because its database was populated mainly with ICAT data; for the specified software systems and loss, a brute force attack is one of the three most common attacks in ICAT.

```
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. `toad.com` logs.

Shimomura observed that the network scans originated from a host in domain `toad.com` (see Figure 4 [12]). In Figure 4, target refers to `Ariel`, server to `Rimmon`, and x-terminal to `Osiris`. Shimomura also observed that his computer (`Osiris`) exhibited strange behavior – blank windows on the top of the screen. The facts are:

- `Ariel` and `Osiris` had strong relationships
- `Osiris` was located at Shimomura's home and had no direct connection with `toad.com`

```
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. `Osiris` logs.

- Considerable network traffic was directed at `Osiris` (Figure 5).

These facts imply that the investigation should continue with `Osiris` and not (for the moment) with `toad.com`. Furthermore, `Osiris` may be the source of the attack or an intermediate system in the attack.

Consequently, a new investigation plan is created for `Osiris`. In fact, Shimomura discovered that `Osiris` was disconnected from his office network and especially from `Ariel`.

The following facts were provided to XMeta:

> **Osiris**:
> **Software**: SunOS, GNU tar, GNU ghostscript, fingerd, ruserd, ftp
> **Losses**: LT_Availability

The following results were provided by XMeta:

> The probable **attacks** are: `repeat` (e.g., scanning sweeping) 100%, `overrun` (e.g., DoS, DDoS, smurf, fraggle) 89%, `bypass` (68%).
> The probable **additional actions** are: `infection` (73%), `trap` (backdoor) 62%, `del` (data deletion) (45%).
> The highlighted **software systems** are: ftp (73%), GNU tar (38%), GNU ghostscript (38%).

The results indicate that `Osiris` was an intermediate system because an attacker cannot penetrate a host using scanning sweeping or overrun. Therefore, it is necessary to search for another computer.

`Osiris` was a X-Window terminal connected to `Rimmon`; possibly, it was also attacked. This is confirmed by Shimomura's logs (Figure 6).

Since information is not available about `Rimmon`, it is reasonable to assume that it had the same configuration as `Osiris` and `Ariel`. Shimomura discovered that an unauthorized user succeeded in installing a kernel module named `Tap 2.01` on `Rimmon` (Figure 7). This implies that the unauthorized user had root privileges.

```
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. Rimmon logs.

```
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. Rimmon system variables.

The following facts were provided to XMeta:

Rimmon:
Software: SunOS, GNU tar, GNU ghostscript, fingerd, ruserd, ftp
Losses: LT_Obtain_all_priv, LT_Availability

The following results were provided by XMeta:

The probable **attacks** are: `trojan` (93%), `bypass` (78%), `brute_force` (58%).
The probable **additional actions** are: `login_inst` (58%), `infection` (51%) and `trap` (46%).
The highlighted **software systems** are: ftp (59%), GNU tar (41%).

From these results, one can infer that if a Trojan horse was not found in `Rimmon`, the computer was used as an intermediate system like `Osiris`. Since Shimomura did not find a Trojan horse but a flooding attack (known as `overrun` in XMeta), it appears that `Rimmon` was used as an intermediate system to gain access to `Osiris` and `Ariel`. In fact, XMeta indicated that the overrun attack was the tenth most likely of the 21 possible attacks. (According to XMeta, the ten most likely attacks were: `trojan`, `bypass`, `brute_force`, `broadcast`, `chaff`, `repeat`, `intercept`, `net_listen`, `bounce` and `overrun`. However, Shimomura also discovered that the attacker had installed a kernel module in `Rimmon` and, therefore, had root access.

5.2 XMeta's Results

XMeta discovered the following elements in the Mitnick case.

- A file `oki.tar.Z` was transferred from `Ariel` to an unknown address using a bypass attack or a diversion attack.
- A host in the `toad.com` domain was used to scan the network.

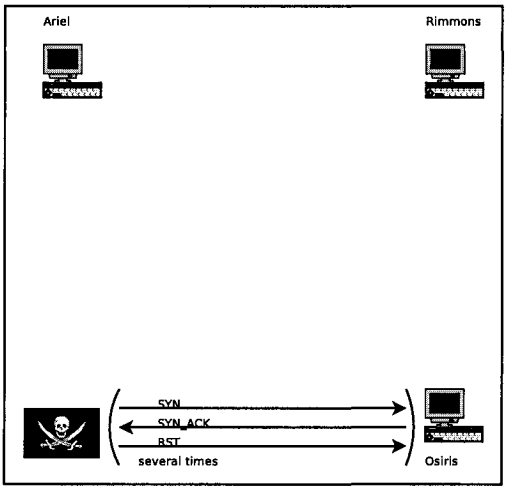

Figure 8. Mitnick attack (first step).

- The attacker either used the repeat attack on `Osiris` to obtain information or bypassed security to enter `Osiris`.
- The attacker exploited the trust relationship between `Osiris` and `Rimmon` to access `Osiris`.
- The attacker installed a kernel module and used it to access `Ariel`.

These elements can be saved, giving future users of the system the ability to replay the attack or the entire investigation (e.g., for a trial). To support this goal, we have defined the Computer Forensics XML Report (CFXR) System, which uses an XML-based format to save system configurations, attacks, additional actions, investigation techniques, as well as the progressions of attacks and investigations.

The next steps in the Mitnick investigation are to determine how the attacker gained root access to `Osiris` and the `toad.com` host, and the destination of the `oki.tar.Z` file.

5.3 Shimomura's Results

Shimomura [12, 13] broke down the attack into three steps. In the first step, the attacker tried to guess the initial TCP sequence numbers for incoming connections to `Osiris`. This was accomplished by sending SYN packets followed by a connection reset (Figure 8).

In the second step, the attacker spoofed `Rimmon` to open a connection to `Osiris`. Next, the attacker used `rsh` to issue the command `echo ++ >>/.rhosts` and gain root privileges. This attack (`bypass`)

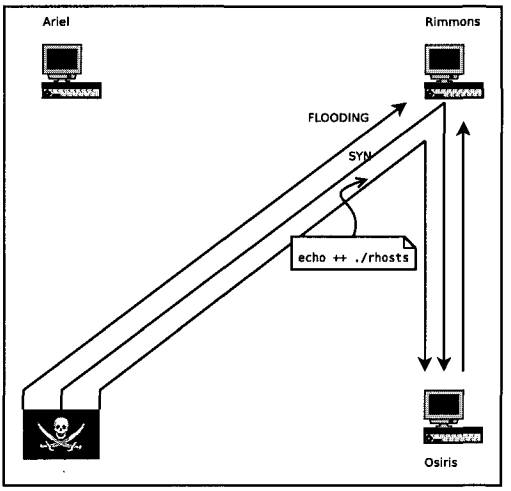

Figure 9. Mitnick attack (second step).

was identified by XMeta as the third most likely attack (68%). Flooding (`overrun` in XMeta) was used to gag `Rimmon` during the three-way handshake when establishing the TCP connection with `Osiris` (Figure 9).

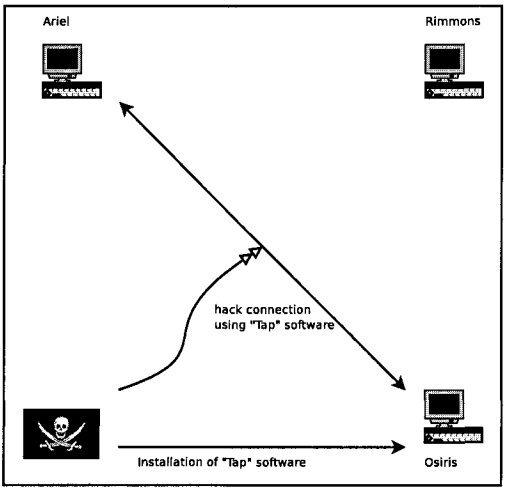

Figure 10. Mitnick attack (third step).

In the third step, the attacker installed `Tap` software and used it to hack the connection between `Osiris` and `Ariel` (Figure 10). The attacker thus gained access to `Ariel`, and created and downloaded the `oki.tar.Z` file.

6. Conclusions

The XMeta system uses a Bayesian network to reason about attacks on computer systems and networks. In particular, it provides information about likely attacks, additional actions performed by attackers, the most vulnerable software systems, and the investigation techniques that should be used. The application of XMeta to the investigation of the Kevin Mitnick case demonstrates its utility as a digital forensic expert system. A supporting Computer Forensics XML Report (CFXR) System uses an XML-based format to save system configurations, attacks, additional actions, investigation techniques, and the progressions of attacks and investigations.

Additional research is needed to enable XMeta to support forensic investigations. The database of cases must be enhanced to obtain better results, especially the suggested investigation techniques. It is also necessary to incorporate criminal profiles that will help refine the assumptions, resulting in more accurate information about targets and attacks.

References

[1] D. Burroughs, L. Wilson and G. Cybenko, Analysis of distributed intrusion detection systems using Bayesian methods, *Proceedings of the Twenty-First IEEE International Performance, Computing and Communications Conference*, 2002.

[2] E. Charniak, Bayesian networks without tears, *AI Magazine*, vol. 12(4), pp. 50-63, 1991.

[3] A. Christie, The Incident Detection, Analysis and Response (IDAR) Project, Technical Report, CERT Coordination Center, Carnegie Mellon University, Pittsburgh, Pensylvania, 2002.

[4] G. Cooper and E. Herskovits, A Bayesian method for the induction of probabilistic networks from data, *Machine Learning*, vol. 9(4), pp. 309-347, 1992.

[5] P. Costa, J. Jones, B. Liao and V. Malgari, A system for collection, storage and analysis of multi-platform computer system data, Technical Report, George Mason University, Fairfax, Virginia, 2003.

[6] R. Fung and K. Chang, Weighing and integrating evidence for stochastic simulation in Bayesian networks, *Proceedings of the Fifth Annual Conference on Uncertainty in Artificial Intelligence*, pp. 209-219, 1989.

[7] W. Hsu, Bayesian Network Tools in Java (bndev.sourceforge.net).

[8] T. Levitt and K. Laskey, Computational inference for evidential reasoning in support of judicial proof, *Cardozo Law Review*, vol. 22(5), pp. 1691-1732, 2001.

[9] K. Mandia and C. Prosise, *Incident Response: Investigating Computer Crime*, McGraw-Hill/Osborne, Emeryville, California, 2001.

[10] P. Naim, P. Wuillemin, P. Leray, O. Pourret and A. Becker, *Reseaux Bayesiens*, Eyrolles, Paris, France, 2004.

[11] NIST, National Vulnerability (formerly ICAT) Database (nvd.nist.gov).

[12] T. Shimomura, Technical details of the attack described by Markoff in NYT (blinky-lights.org/shimomura-25jan95.html), 1995.

[13] T. Shimomura and J. Markov, *Takedown*, Hyperion Press, New York, 1996.

[14] SpamAssassin, The Apache SpamAssassin Project (spamassassin.apache.org).

[15] U.S. Department of Justice, Computer Crime and Intellectual Property Section (www.cybercrime.gov).