Karlsruhe Institute of Technology

Incidents and Forensics

Tobias Dussa • GridKa School 2012

COMPUTER EMERGENCY RESPONSE TEAM

Incidents – Introduction and Overview

This part addresses the following questions:

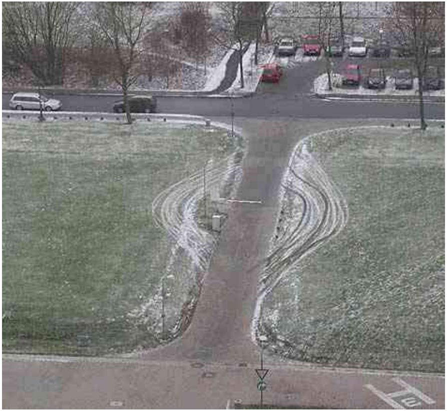
- What is an IT security incident?
- What to do if an incident occurs?
- What *not* to do?

It does *not* address:

- Operational aspects of an incident.
- Legal implications of an incident.

(Some yucky problems will be pointed out though.)

Incidents – Evidence



Source: Unknown (the vast spaces of the Internet).

Incidents – Definition

For the purposes of this talk, an IT security incident

- is an event
- involving an IT system
- that has either
  - direct or
  - indirect

  implications
- on the security of
  - the above or
  - any other

  IT system.

Incidents – Examples

Obvious incidents:

- User-level intrusion:
  - Compromised user account.
  - Compromised service; recent example: `exim`.
- Root-level intrusion:
  - Compromised root account.
  - Compromised root-level service; example: Linux kernel.

Not-so-obvious (potential) incidents:

- Web server defacement.
- Hard drive replacement due to failure.
- *(... append to suit your own paranoia.)*

Incidents – Detection

As a general rule of thumb: *Detection is not trivial.*

- Detection partially depends on the attacker's stupidity: The dumber the attacker, the easier is detection.
- Some traces are unavoidable though:
  - Host logs.
  - Firewall logs, NAT logs, router logs.
  - Netflow data.
- Centralized collection and analysis helps a lot.
- Outbound network traffic is equally important.
- (*Caveat: Beware of legal issues – data procection!*

Incidents – Handling (1)

"So you've got yourself an incident."

- There is an EGI Incident Response Procedure; available as a regular document and in condensed form as a print-out checklist.
- For EGI site folks: *It's not just a good idea, it's the law.* The EGI IR Procedure is *mandatory.*
- For all other folks: *It's not just a good idea, it's a really good idea.* Not mandatory, obviously, but still a very good procedure.

Incidents – Handling (2)

- Phase 1 – Discovery. *Spread the word.* If applicable, inform
  - your local site security officer,
  - your National Grid Infrastructure security officer,
  - your network uplink provider’s security officer,
  - the EGI CSIRT Duty Contact.
  - This does not mean “tell the whole world!”
- Phase 2 – Containment. *Isolate the affected host(s).*
  - *Do NOT* just pull the network cables!
  - Prevent users from logging into the machine.
  - Prevent new jobs to be run on the machine.

Incidents – Handling (3)

- Phase 3 – Confirmation. *Verify you're not seeing a glitch.*
  - Make sure you're seeing what you think you're seeing.
  - If possible: See to your security people for assistance.

- Phase 4 – Downtime Announcement. *Let your users know.*
  - People will want to know if there's a problem.
  - Informing them will keep them off your back.
  - This does not mean "tell them every detail!"

Incidents – Handling (4)

- Phase 5 – Investigation. *Go get them!*
  - Collect evidence as swiftly as possible; see the forensics part.
  - Talk to your security people!
  - If you are an EGI site: Follow up on EGI CSIRT requests immediately.
  - If necessary, reiterate earlier phases.

Incidents – Handling (5)

- Phase 6 – Debriefing. *Tell people what you have learnt!*
  - Write up a summary of what happened.
  - Help others by letting them know what went wrong.
  - For EGI sites: The debriefing must be done within one month.
- Phase 7 – Restoration. *All systems back to normal, Captain!*
  - Restore the affected systems.
  - Resume normal operation, BUT
  - fix the original problem(s), obviously.

Incidents – Handling (6)

To wrap up:

- It is VERY helpful to have a plan to work along.
- Communicate! Let others know what's happening.
- Talk to security people whenever you are unsure.
- *CAVEAT: As soon as it looks like your boss would like to see the attacker in jail: STOP what you are doing immediately and call the police.*

Incidents – Information Flow

So *why* does everybody need to know I screwed up?

- This is *NOT* about pointing fingers.
- Informing others:
  - Helps them avoiding your mistakes.
  - Helps people to get the bigger picture quicker.
  - Enables faster recognition of a problem or a trend.
  - Likely will get them to share their lessons learnt with you, too.

Incidents – Legal Aspects

I Am Not A Lawyer. However:

- Be aware that depending on your jurisdiction, serious problems might arise as soon as things turn legal.
- You might break the “chain of custody” or otherwise render pieces of evidence non-usable in a court of law.
- You might even get into personal trouble by acquiring illegal software or data.
- *As always, if you are unsure, go see someone who should know better (in this case, a lawyer).*

Case Study – Incident

Actual incident:

- Large linux cluster was compromised at root level.
- Several hundred users affected.
- Detection was very quick: less than an hour after the break-in.
- KIT-CERT was alerted immediately.

Case Study – Detection

- All host log files were collected centrally.
- On the central server, the logs are analyzed for anomalies.
- If such anomalies are found, notification is sent to the admin crew.
- We got a bit lucky with the timing, but the general idea works very well.

Case Study – Impact

- Large machine with many users.
- In particular, lots of users external to KIT.
- Downtime is a VERY policital issue.

Case Study – Handling (1)

- Handling was almost done by the textbook.
- Phase 1 – Detection: Atypical log line seen.
- Phase 2 – Confirmation: KIT-CERT was called in, the break-in was confirmed.
- Phase 3 – Containment: Users were locked out of the affected systems.
- Phase 4 – Downtime announcement: Users were notified of "ongoing security operations."
- So far, all within four five hours of the breakin.

Case Study – Handling (2)

- Phase 4a – Normal operation restoration: After evidence collection, the admin crew restored normal operation.
- Phase 5 – Analysis: Admin crew and KIT-CERT worked together and put together a timeline of what has happened.
- Phase 6 – Debriefing: KIT-CERT prepared a report and handed over the facts to the legal department.

Case Study – Handling (3)

- During analysis, it was discovered that the attacker came from a different institute.
- The institute in question was contacted by KIT-CERT.
- The affected machines were also taken into custody and analyzed.
- The entire incident handling used up quite a bit of resources.

Forensics – Introduction and Overview

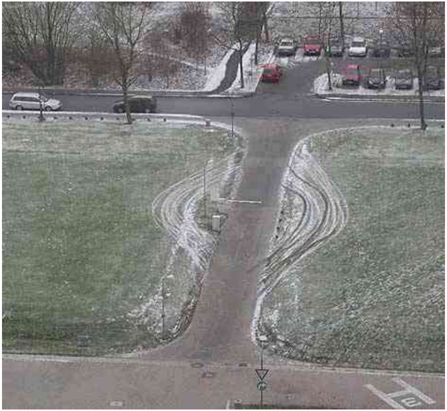
This part addresses the following questions:

- Why forensic analysis?
- Where and how to gather evidence?
- How to analyze evidence data?

It does *not* address:

- How to contain damage?
- What to communicate when to whom?
- How to recover from an incident?

Forensics – Evidence



Source: Still unknown.

What Is Forensic Analysis Good For?

To assess and answer several important questions about an incident:

- Where did the attacker come from?
- How was access gained?
- What damage was done?
- What other machines were affected?
- . . . and many more related questions.

Data Sources for Forensic Investigations

- Broad classes of data sources:
  1. Highly volatile (e. g., CPU registers),
  2. Volatile (e. g., RAM),
  3. Static (e. g., hard drives), and
  4. Highly static (e. g., archive tapes).
- More volatile evidence must be gathered and preserved first, if possible.
- Obviously, not all classes available or applicable in every instance.

Find the Right Machine

Usually, this is the first thing to do.

1. Collect all relevant network-related data:
   - NAT data,
   - proxy data,
   - netflow data,
   - and so on.

   No problem if there are log files, interesting if not (live NAT tables etc.).
2. Correlate data to find your suspect host, if any.

So We Have a Suspect . . .

. . . or at least a suspect machine. Now what to do?

1. Gather general information and evidence:
   - Running processes,
   - open network connections,
   - who is logged on,
   - who was logged on,
   - mounted devices
   - and their mountpoints,
   - etc.
2. Look if there is anything suspect.

Freeze!

What to do with your suspect (process):

1. Stop the process (do *not* terminate it!).
2. Collect and secure:
   - the binary being executed,
   - its core memory,
   - its shared memory regions, if any,
   - its file handles,
   - other volatile data.

Now That We Have More Time

Finally, collect less volatile stuff:

- If possible and sensible, power off the machine and grab the hard drive.
- If not possible or sensible, at the very least collect the following stuff:
  - All related log files,
  - any files involved in the incident,
  - actually, if possible, the entire file system.

After Compiling, Interpretation!

Take a close look at the collected data. Some pointers:

- Inspect suspect executables (with `strings`, `hexdump`, `gdb`, `rec`, IDAPro, . . .).
- Look at core dumps (using `gdb`).
- Grep through log files and the like.
- Check files' MD5 sums against the known-good list.
- Perform further filesystem analysis, for instance with `autopsy` or `rkhunter`.
- If necessary, iterate.

Case Study – Gathering (1)

- Main sources of insight:
  - Compromised hosts' hard drives.
  - (Centrally-collected) host logs.
  - Process-accounting data (huge!).
  - Router netflow data.
- Attacker had already left, so no dynamic data to be gathered.

Case Study – Gathering (2)

- But we also made mistakes:
  - We failed to collect some dynamic evidence.
  - We were unable to rip images of some hard drives (hardware RAID controllers).
  - One of the first things done during confirmation of the incident was something like

    ```
    find / -mtime -2
    ```

Case Study – Timelining

- All evidence needs to be put in the proper sequence.
- This sounds easier than it is: chances are different logs use a vastly different timestamp format.
- We wrote a script for normalization, but nowadays there is better software, for instance `log2timeline`.
- Careful with timestamps: time zone woes, jitter!

Case Study – Results (1)

- Attacker compromised a user account on a different machine.
- Said user had access to the cluster.
- Attacker could log in, impersonating the user.
- Exploiting an unpatched kernel vulnerability, the attacker was able to get root access. This already triggered suspicious kernel messages that were not picked up!

Case Study – Results (2)

- Attacker then installed a backdoor by giving a system user a password.
- This triggered another log line, which WAS picked up by the admin crew. Kudos!
- Attacker then snooped around for a bit, decided that this machine was not interesting to him apparently, and left.
- Twenty minutes later, the break-in was detected.
- Attacker tried to log in to his backdoor on this and another (neighboring) cluster later on.

Case Study – Lessons Learnt

- Collect your logs centrally and independently.
- Understand your logs and its timestamps.
- Be prepared to normalize and analyze logs.
- Be prepared to dump and analyze disk images.
- Have a checklist prepared so that you don't
  - forget stuff or
  - do stuff in the wrong order.

Summary

- Incidents are a lot of work:
  - detection,
  - handling,
  - alleviating.
- Forensics is very time-consuming and difficult to get right.
- The better the preparation, the easier the work.
- Share your information and ask for help!

That's It!

Questions?

Answers!

PGP-Key `0x236636AE`; fingerprint:

```
0D29 63BE DB07 1264 DD1C
EFE0 34E7 F72A 2366 36AE
```