# Introduction to InfoSec – Recitation 13

Nir Krakowski (nirkrako at post.tau.ac.il)
Itamar Gilad (itamargi at post.tau.ac.il)

# Today

- Rootkits
  - Rootkits 101
  - Motivations & basic methods
- Forensics
  - Forensics 101
  - Threat types
  - Gathering information
  - Research methods
  - Planning for a future incident

# What is a rootkit ?

- The name ‘rootkit’ originally came from UNIX/linux utilities that were used by hackers after gaining root on a target machine
- The goal of the rootkit is to allow a hacker to roam free about the system, while still maintaining root
- The rootkit hides the hacker and allows him to evade detection by the system admin

# What can/should a rootkit do ?

- Hide the hacker's files (a hacker would often have a "working directory" and/or files hidden at various locations)
- Hide the hacker-owned processes:
  - Eg. Any process starting with the words: "w00t" will not be visible.
- Hide open ports, hide sniffing

- Let the hacker back in without using an exploit
  - Using the exploit to re-enter can makes too much noise
  - No need to cleanup after re-entry

# Application based rootkits

- The first rootkits seen in the 90s were replacements for the set of system utilities in /bin/
- For example hackers used a modified version of /bin/ls
- In open-source systems such as Linux this is very easy
  - Download original code, modify, compile, install on target
- In closed-source systems such as Windows / proprietry UNIX
  - Binary patch the relevant files

# Application Layer Dilemmas

- If you patch one program, you never know if you covered all your bases.
  e.g.:
  - patch ‘ps’ but forget to patch ‘top’
  - Patch ‘ls’ but forget to patch ‘mc’ (midnight commander)

- What happens when the software gets upgraded?

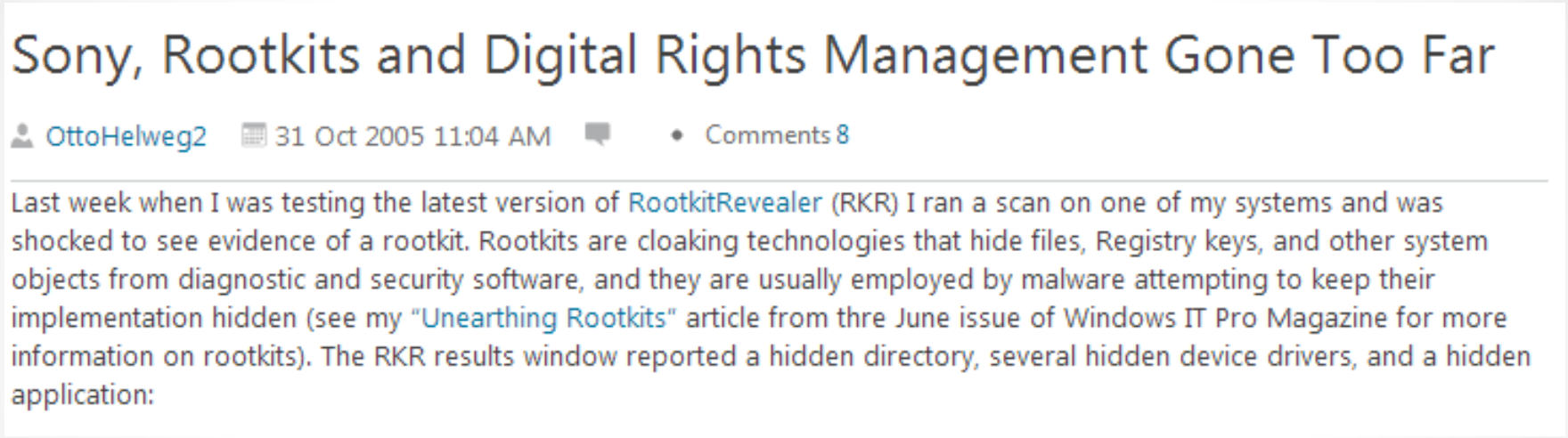
# Better solutions

- Patch system libraries to control the API
- Patch system-calls
- Patch Kernel structures

# Rootkit Detection

- Whitelist based –
  - Integrity checking of binaries (compare md5 of files to a list of 'known good' signatures).
  - e.g.: tripwire
- Blacklist based –
  - Find signatures in files and memory known to be 'evil'
  - This is the technique most anti-viruses use
- Difference based –
  - Find differences between views that should be identical
    - API vs. kernel memory
    - memory vs. on disk
  - **Most effective**

# Sony DRM: Famous Rootkit Case



Sony, Rootkits and Digital Rights Management Gone Too Far

OttoHelweg2 31 Oct 2005 11:04 AM • Comments 8

Last week when I was testing the latest version of RootkitRevealer (RKR) I ran a scan on one of my systems and was shocked to see evidence of a rootkit. Rootkits are cloaking technologies that hide files, Registry keys, and other system objects from diagnostic and security software, and they are usually employed by malware attempting to keep their implementation hidden (see my "Unearthing Rootkits" article from thre June issue of Windows IT Pro Magazine for more information on rootkits). The RKR results window reported a hidden directory, several hidden device drivers, and a hidden application:

- https://blogs.technet.com/b/markrussinovich/archive/2005/10/31/sony-rootkits-and-digital-rights-management-gone-too-far.aspx?Redirected=true

# Forensics 101

- We have a suspected machine / network installation
- You know little to nothing about the specific threat, and even less about how it got there
- You want to know everything!
  - How they got there
  - Find and fix any damage they've done
  - Find out if they took any sensitive information
  - Who they are, what do they want?
  - Finally – figure out how to prevent the next incident

# Threat Types

- Non targeted attack – script-kiddies, botnets, drive-by downloads, toolbars, scam sites, etc.
- Targeted attack, a.k.a. APT (Advanced Persistant Threats) –
  - They know who you are
  - They'll invest lots of resources to get what they want
  - Very hard to defend against
  - But if you do your work well – you'll know what they did

# Basic Data Sources

- Running process list, loaded Kernel module list
- Complete memory image – RAM + Swap
- Anything that's changed in the suspected time frame (time since last major system change is a good start)
- Checking file signatures against a whitelist
- Contents of config files – users, lowered hardening, anything an attack might want to change
- **LOG FILES**
- File / directory creation, modification and access times
- Network analysis – which machines download/upload more than they should? Which machines are talking to machines that they shouldn't?

# Gathering Information

- You could work in the client's production environment
- But then you could make mistakes that will destroy valuable 'bread crumbs' and/or reveal information to the adversary
- You want a perfect memory snapshot, and a perfect disk image to take to the lab

# Getting a snapshot of the system

- Getting the contents of the memory by asking the computer to hibernate / reading memory via FireWire
- Getting the contents of the disks by pulling the power immediately, and taking the disks to the lab
  - Extra: use advanced disk-recovery techniques to access deleted / overwritten data

# Disks or Memory – choose one!

- If the attacker was smart – her tools will hide better in some scenarios
  - She's put a hook on the hibernate function, to make the memory snapshot "clean", and maybe even clean her rootkit from the disk
  - She might scrub her files off the disk after loading, only writing them back on a regular shutdown, or not at all…
- You may have a better tool (Liquid NO2 + magic) – but you'll still have to choose one over the other

# Malware Analysis

- First – a quick check against any known signatures
- Then, lots of looking for potential malware
- Once good candidates surface, lots of reverse engineering
- The goal is to spend a little time initially to classify every finding as “interesting”, “maybe”, or “junk”
- Finally, start diving into the “interesting” and “maybe” bins
- You may find hints that will make you go back on the field and collect more information
-

# Expanding the Search

- You've identified a threat
- Next step is to build a detector, and spread it as far as you can
- Gather more information from new infections you found
- Continue to learn more about the attacker
- …
- Repeat

# Planning for Forensics

- Instead of reacting – we can plan the system / network to facilitate forensics, and make it much harder for the attacker

- So, what should we do?

# How to prepare

- Logging should be local AND network based, in multiple locations, and logging servers should be extremely secure
- Logging should be as deep as possible (forever is a good depth)
- Log anything important, especially anything touching the core secrets of the company
- Keep ‘good’ system images for important machines (and again – depth is your friend)
- Keep an accurate and central log for any maintenance event, to help quickly filter these events later on

# Monitoring

- Find a SOC (Security Operations Center) solution that suits you, and USE it!
- Build rules to filter out the noise
- Build rules to highlight important events
- Central logging will permit high-order anomaly detection, data clustering and machine learning based filtering to help you analyze all that data
- If possible – make this system report to the system administrators in real-time!
- The key is to actively look for the threats, not just install-and-forget…

# Questions?