A Building Block Approach to Intrusion Detection

Mark J. Crosbie
Hewlett-Packard Company
mark_crosbie@hp.com

Benjamin A. Kuperman
CERIAS, Purdue University
kuperman@cerias.purdue.edu

http://www.hp.com/products/security/ids/

Who are we?

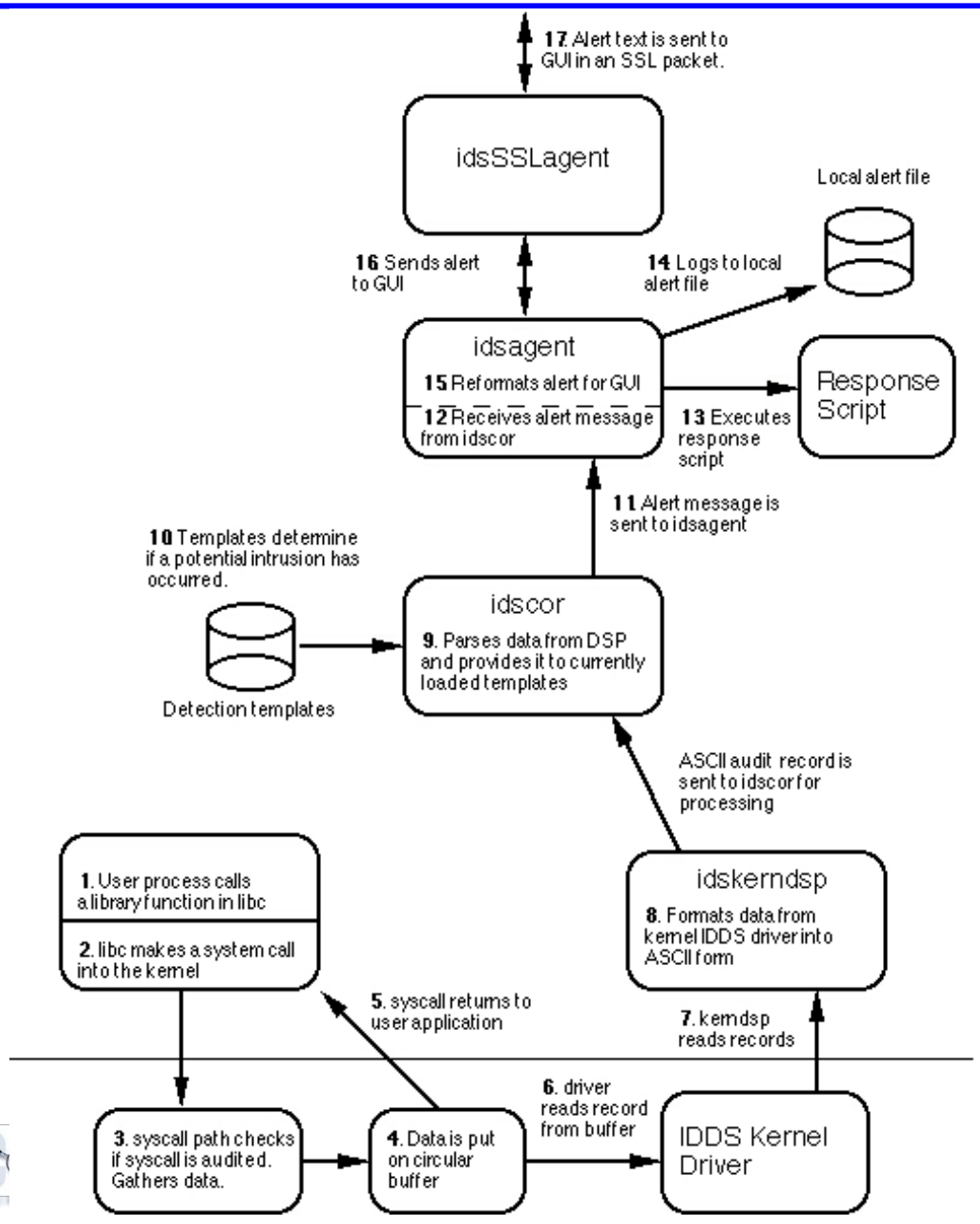
- Mark Crosbie
  - Security Architect at Hewlett-Packard
  - Designed and implemented IDS/9000
- Benjamin Kuperman
  - PhD student at CERIAS, Purdue
  - Worked with HP on IDS/9000
  - Designed and implemented detection templates

For more information...

- Product is available for zero-cost for HP-UX 11.0 and 11i systems.
- Download from http://software.hp.com
- Part number is J5083AA
- More information at http://www.hp.com/products/security/ids/
- Contact Mark Crosbie mark_crosbie@hp.com for further details.

What did we build?

- Kernel level audit source designed to support Intrusion Detection.
- An analysis engine that dynamically loads/unloads configurable detection template bytecode.
- Detection templates
  - Detect the *building blocks* of intrusions.
- Automated intrusion response mechanism.
- GUI, manual, product support, etc etc...


17. Alert text is sent to GUI in an SSL packet.
idsSSLagent
Local alert file
16 Sends alert to GUI
14 Logs to local alert file
idsagent
15 Reformats alert for GUI
12 Receives alert message from idscor
13 Executes response script
Response Script
11 Alert message is sent to idsagent
10 Templates determine if a potential intrusion has occurred.
idscor
9. Parses data from DSP and provides it to currently loaded templates
Detection templates
ASCII audit record is sent to idscor for processing
1. User process calls a library function in libc
2. libc makes a system call into the kernel
idskerndsp
8. Formats data from kernel IDDS driver into ASCII form
5. syscall returns to user application
7. kerndsp reads records
6. driver reads record from buffer
3. syscall path checks if syscall is audited. Gathers data.
4. Data is put on circular buffer
IDDS Kernel Driver

Why choose system call audit?

- Kernel has the only reliable view of system state.
- System calls do not have unintended side effects (compare to library audit).
- Trustworthiness of data.
- Ambiguity can be resolved at kernel level:
  - mapping inodes/file descriptors to pathnames.
  - symbolic/hard links, chroot.
- Reliable capture of before and after state.

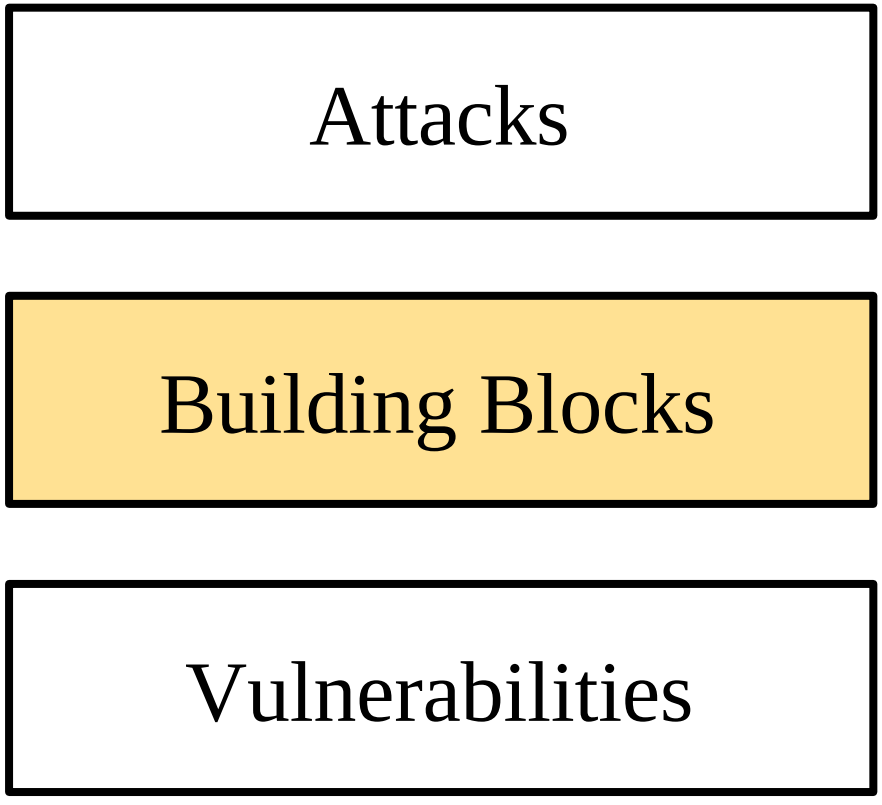
Requirement of audit system

- Audit record must capture as much state as possible.
  - Save before/after state of objects for modification actions (e.g. chmod, chown).
  - Do not query system while processing record.
- Resolve ambiguity in the kernel
  - symbolic links, hard links.
  - inodes mapped to pathnames.
  - chroot environments.
- Data format must be machine parseable.
- Data presented in timely and efficient manner.

What is a Building Block?

Definition:

An abstraction of attack activities undertaken to exploit a vulnerability.



Building Blocks Detected

- Login/logout activities, including su.
- Local and remote filesystem changes:
  - Directories and files.
  - Creation, deletion, attribute changes.
  - Changes to files owned by others.
  - Unusual Log file modifications.
- Creation of setuid “ backdoors” .
- Race condition (TOCTTOU) attacks.
- Unexpected change in privileges.

Some sample alerts

- Unexpected change in privilege levels with UID:100(GID:20) EUID:0(EGID:20) executing /usr/bin/ksh(1,42 246,"40000003") with arguments["/usr/bin/ksh", "-c", "foobar"] and system call kern_setuid as PID:19854

- UID:0 (EUID:0) Reference:/dev/emsagent_fifo currently kern_open:/dev/emsagent_fifo(8,1282,"40000003") was kern_mknod:/dev/emsagent_fifo(0,-1,"ffffffff") program running is /etc/opt/resmon/lbin/emsagent(1,1429,"40000003") with arguments ["/etc/opt/resmon/lbin/emsagent"] probable ATTACKER was UID:10

- User 0 opened for modification/truncation "/etc/opt/resmon/pipe/1652016795" executing /etc/opt/resmon/lbin/p_client(1,473,"40000004") with arguments ["/etc/opt/resmon/lbin/p_client"] as PID:2708

Automated Response

- Active response to a recent intrusion event.
- Examples of responses:
  - Locking a user account.
  - Collecting additional system state information.
- Recovering from configuration changes.
  - E.g. changes made under /etc
- Recovering from Trojan Horses:
  Replacing files changed in /sbin or /usr/bin.
- Recovering from a web server hack.
  - rsync web pages from remote source/CDROM.

Performance

- Difficult to measure - why?
  - What is a typical system load?
  - How is the IDS to be configured?
  - Rate of “ intrusive” activity? Bursty?
- How to measure performance impact?
  - System throughput, CPU time, response time.
  - IDS load, IDS throughput, IDS response time.
- See paper for more details on our performance tests.

What IDS/9000 does not do.

- Does not fix pre-existing vulnerabilities.
- Does not *prevent* activities from occurring.
- Does not detect changes made to local filesystems mounted on remote systems.
  - No file signature scanning.
- Not currently a network IDS.

Conclusions

- Possible to detect *Building Blocks* of intrusions.
- By building the IDS and audit system together:
  - More context for making decisions.
  - Reliable detection.
  - Data is tailored to detection.
- IDS can be used for more than security tasks:
  - monitor admins, misbehaving programs
- Performance measurement is environmentally sensitive.