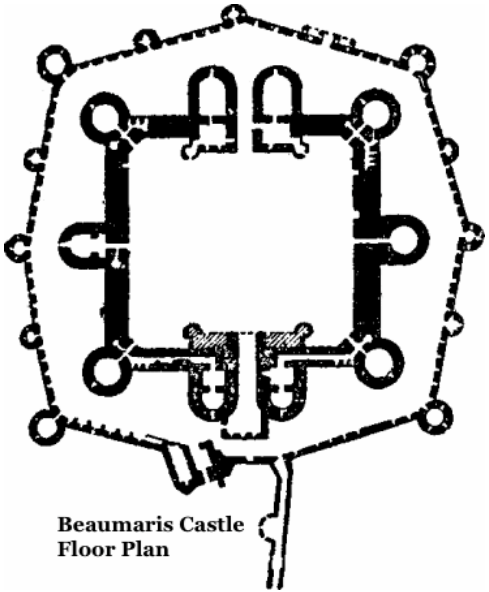
Squash Those IoT Security Bugs with a Hardened System Profile

Presentation Outline

Context: Operational Environment

- Organizational Ethics
  - Policies
  - Controls
  - Monitoring
- Insider Threats
  - Education and Training
- Untrusted Networks
  - Can you have a "trusted" system?
- Minimal Attack Surface
  - Complexity
- Default Deny
- *Defense-in-Depth*
- Reduce Exposure
  - Compartmentalization
  - Least Privilege
  - Insecure-Bootstrap Principle
- Input Validation

Defense-in-Depth



"The principle of defense-in-depth is that layered security mechanisms increase security of the system as a whole. If an attack causes one security mechanism to fail, other mechanisms may still provide the necessary security to protect the system."

Layered Architecture

- Authentication
- VLANs
- Firewalls
- Encryption
- Detection
- Hosts
    - Certificates
    - ACLs / CAPs
    - Mandatory / Role-based Access Controls
    - Discretionary Access Controls

Layered Architecture

- Authentication
- VLANs
- Firewalls
- Encryption
- Detection
- Hosts
  - Certificates / Keys
  - Mandatory / Role-based Access Controls (MAC / RBAC)
  - Access Control Lists / Linux Capabilities (ACLs / CAPs)
  - Discretionary Access Controls (DAC)
  - Kernel config / Toolchain
    - **PaX**
    - **PIE / SSP**

Example: Potential 0-day Kernel Exploit

Mishandled Object References in Kernel Keyring - A 0-day local privilege escalation vulnerability has been identified by the perception point research team. It has been reported that a vulnerability in the keyring facility possibly leads to a local privilege escalation.

- CVE 2016-0728
  - http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=2016-0728
- Original Report
  - http://tinyurl.com/2016-0728

Typical Vendor Responses:

- RedHat, MRG 2 and RHEL 7, Suse Enterprise 11 and below
  - https://bugzilla.redhat.com/show_bug.cgi?id=1297475
  - https://www.suse.com/security/cve/CVE-2016-0728.html
- Gentoo Linux (gentoo-sources, hardened-sources)
  - https://bugs.gentoo.org/show_bug.cgi?id=572384
  - https://bugs.gentoo.org/show_bug.cgi?id=572604

Hardened response - hardened-sources with default settings (in particular CONFIG_PAX_REFCOUNT) significantly reduces the effect of this issue to a local DoS rather than a privilege escalation.

What Does PaX Do?

PaX adds security enhancement to the area between the kernel and user land.

- Automatically enforces memory restrictions and address space randomization on all running processes
  - Can relax certain PaX restrictions on a per ELF object basis
  - Can also be configured to run in SOFTMODE (permit by default)
- Emulates trampolines (mainly for nested functions in C and some JIT compilers)
- Prevents the introduction of new executable pages into running processes
- Kernel land enforcement of *PAGEEXEC* and *MPROTECT* cannot be disabled while running

Old PaX used ELF program header markings, whereas new PaX prefers filesystem extended attributes marking (both are still available in PaX kernel config).

What Exactly is PaX?

PaX was originally a patch to the Linux kernel that provided additional hardening in three important ways:

1. Judicious enforcement of non-executable memory

   Prevents a common form of attack where executable code is inserted into the address space of a process by an attacker and then triggered. PaX preemptively protects against this class of attacks.

2. Address Space Layout Randomization (ASLR)

   Randomization of the memory map of a process (makes it harder for an attacker to find the exploitable code within that space). If application is built as a Position Independent Executable (PIE), even the base address is randomized.

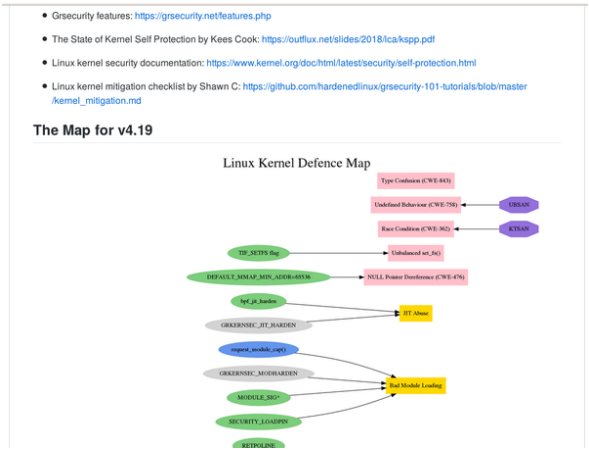
3. Miscellaneous hardening on stack- and memory handling

   Additional hardening features include erasing the stack frame when returning from a system call, refusing to dereference user-land pointers in some contexts, detecting overflows of certain reference counters, correcting overflows of some integer counters, enforcing the size on copies between kernel and user land, and providing extra entropy.

PaX In The Mainline Kernel (as of 4.17+)

Various bits of the last "open source" version of the PaX kernel patch have been making it into the mainline kernel over the last few years (as each specific subset goes through the normal kernel review process). Many of the core options have already been merged, and more are currently under review.

In the following slides, the specific PaX options listed show (yes)* where there is a roughly corresponding mainline feature. Alex Popov has prepared a diagram (available on github) for a better and more visual representation. The full graphic is too high a resolution for one slide, so only a screen shot is shown below:



PaX Kernel Options - NOEXEC Features

PAX_NOEXEC	This option enables the protection of allocated pages of memory as non-executable if they are not part of the text segment of the running process. It is needed for PAGEEXEC, SEGMEXEC, and KERNEXEC.
PAGEEXEC (yes)*	The kernel will protect non-executable pages based on the paging feature of the CPU. This is sometimes called "marking pages with the NX bit" in other OSes.
SEGMEXEC	This is like PAGEEXEC, but based on the segmentation feature of the CPU and it is controlled by the PaX -S and -s flags (only on x86).
EMUTRAMP	The kernel will emulate trampolines (snippets of executable code written on the fly) for processes that need them, e.g. nested functions in C and some JIT compilers.
MPROTECT (yes)*	The kernel will prevent the introduction of new executable pages into the running process by various techniques.
KERNEXEC (yes)*	This is the kernel land equivalent of PAGEEXEC and MPROTECT. It cannot be disabled while the kernel is running.

PaX Kernel Options - ASLR Features

PAX_ASLR (yes)*	The kernel will expand the number of randomized bits for the various section of the address space. This option is needed for RANDMMAP, RANDKSTACK, and RANDUSTACK.
RANDMMAP	The kernel will use a randomized base address for mmap() requests that do not specify one via the MAP_FIXED variable. It is controlled by the PaX -R and -r flags.
RANDKSTACK (yes)*	The kernel will randomize every task's kernel stack on all system calls. It cannot be disable while the kernel is running.
RANDUSTACK	The kernel will randomize every task's userland stack. This feature can be controlled on a per ELF binary basis by the PaX -R and -r flags.

PaX Kernel Options - Misc Features

STACKLEAK (yes)*	The kernel will erase its stack before it returns from a system call. This feature cannot be disabled while the kernel is running.
UDEREF	The kernel will not de-reference userland pointers in contexts where it expects only kernel pointers. This feature cannot be disabled while the kernel is running.
REFCOUNT (yes)*	The kernel will detect and prevent overflowing various (but not all) kinds of object reference counters.
USERCOPY (yes)*	The kernel will enforce the size of heap objects when they are copied in either direction between the kernel and userland.
SIZE_OVERFLOW	The kernel recomputes expressions of function arguments marked by a size_overflow attribute with double integer precision.
LATENT_ENTROPY (yes)*	The kernel will use early boot code to generate extra entropy, which is especially useful on embedded systems.

Manipulating PaX Flags

There are five PaX protections that can be enforced (in SOFTMODE) or relaxed (in non-SOFTMODE) on a per ELF object basis: PAGEEXEC, EMULTRAP, MPROTECT, RANDMMAP and SEGMEXEC.

paxctl - This is the traditional upstream package for setting PaX flags. It is limited only in that it sets PT_PAX only, not XATTR_PAX. It is provided by emerging sys-apps/paxctl.

getfattr / setfattr - These are not PaX specific utilities but are general utilities to set a file's extended attributes. On Gentoo, they are provided by emerging sys-apps/attr. Can be used to set XATTR_PAX via the user.* namespace.

> ***Warning***
>
> setfattr and getfattr know nothing about PaX, so they will not perform any sanity checking of field contents. You've been warned...

paxctl-ng - paxctl-ng is the new swiss army knife for working with both PT_PAX and XATTR_PAX markings. It can be built with support for just one or the other or both types of markings.

Hardened Toolchain

The Gentoo Hardened project introduces a number of changes to the default behavior of the toolchain (gcc, binutils, glibc/uclibc) intended to improve security. It supports other initiatives taken by the hardened project; most directly PaX and Grsecurity, but can also be applied to SELinux and RSBAC.

- Default addition of the Stack Smashing Protector (SSP)

  The stack smashing protector arranges the code so that a stack overflow is very likely to be detected by the application, which then aborts.

- Automatic generation of Position Independent Executables (PIEs)

  Allows the application to be loaded at a random address; most effective when running a PaX kernel with Address Space Layout Randomisation (ASLR).

- Default Mark Read-Only Appropriate Sections (RELRO)

  Causes the linker to include an extra header informing the loader which sections can be marked read-only after the loader has finished with them.

- Default full binding at load-time (BIND_NOW)

  Increases the effectiveness of setting RELRO, making attacks that involve overwriting data in the Global Offset Table (GOT) fail.

Potential Toolchain Issues and Caveats

The SSP implementation in gcc-3.x is not perfect, and SSP implementation in gcc-4.x/5.x is completely different (switches are also different, but is in general much better). The standard (non-hardened) toolchain is now enabling SSP (strong), RELRO, and FORTIFY (default PIE and SSP are in git for gcc 6.0).

Where an application builds libraries without -fPIC, it is necessary to modify the build process to avoid -fPIE being added by the compiler (or patch to build with -fPIC).

Some applications have been reported to segfault when built as PIEs (mostly older versions of gcc).

No issues found so far with switching on RELRO by default. It can make the executable image a little bit bigger (on average by half a page i.e. 2K bytes) which may be of interest for targets with extremely limited memory.

Some packages may still have issues with BIND_NOW, and it has to be relaxed somewhat for them:

- Xorg - some drivers consist of several libraries which are co-dependent, and the modules frequently have references to modules that they load.
- transcode - relies on lazy binding to be able to load its modules; the issues are similar to the X issues.

Hardened Issues and The State of PaX

If you're still running **hardened-sources** or similar:

- PT_PAX flags are still valid (and the default) but are being phased out.
  - Current version of binutils/bfd linker have been patched, but that patch will go away
  - The gold linker (required for LTO plugin) does not support PT_PAX
- XT_PAX migrate script should be used as soon as possible (and disable PT_PAX support).
  - Default PT flags will migrate to empty XT flags (since kernel falls back to default)
  - Only binaries with non-default flags will have XT flags marked
  - Libs needing less PaX enforcement will need their flags "back-ported" to the binaries that use it

For running **gentoo-sources** or mainline:

- The recent mainline kernel (4.17.x and higher) is where you should go for "PaX" hardening features in Gentoo now. Some patches are still under review for later kernel releases, but many useful patches have already been ported to mainline and released.
- A nice graphviz diagram showing the relationships between various kernel and hardware security features, as well as potential vulnerabilities and exploit techniques, is available on github.

Where Can I Get Some PaX?

Gentoo's hardened-sources are no longer maintained, as the PaX patches are no longer publicly available. The remaining (Gentoo) hardening patches for the kernel are in the gentoo-sources package (and toolchain). Be sure and get the latest version available:

```
# emerge --ask sys-kernel/gentoo-sources
```

Gentoo Linux

- Select the desired hardened profile, including a MAC framework (eg, SELinux, Tomoyo, SMACK, etc) and rebuild your kernel, then your toolchain. See the Hardened Project SELinux Guide, the RSBAC Guide, or the Grsecurity Quickstart for more information.

Other Linux

- If only the PaX patches are desired they can be obtained in isolation from one of the Grsecurity maintainers. Install your favorite kernel sources and download/apply the PaX patchset and look into your toolchain config.

Those interested in learning more about Grsecurity hardening in general should read the Grsecurity Quickstart or the grsecurity features page.

Some Hardened Resources

Gentoo Hardened Project

- https://wiki.gentoo.org/wiki/Hardened/Introduction_to_Hardened_Gentoo
- https://wiki.gentoo.org/wiki/Hardened/PaX_Quickstart
- https://wiki.gentoo.org/wiki/Hardened/Toolchain
- https://wiki.gentoo.org/wiki/Hardened/PaX_Utilities
- https://wiki.gentoo.org/wiki/Hardened/Overview_of_POSIX_capabilities

Gentoo Hardened Subproject Starters

- https://wiki.gentoo.org/wiki/Project:RSBAC
- https://wiki.gentoo.org/wiki/Project:SELinux
- https://wiki.gentoo.org/wiki/Project:Integrity

Other Resources

- https://github.com/a13xp0p0v/linux-kernel-defence-map
- http://pax.grsecurity.net/
- http://en.wikipedia.org/wiki/NX_bit
- http://people.redhat.com/drepper/dsohowto.pdf

General References and Specifications

Engineering Principles for Information Technology Security (EP-ITS), by Gary Stoneburner, Clark Hayden, and Alexis Feringa, NIST Special Publication (SP) 800-27 (PDF)

Secure Design Principles from "Foundations of Security: What Every Programmer Needs To Know" by Neil Daswani, Christoph Kern, and Anita Kesavan (ISBN 1590597842)

High-Assurance Design by Cliff Berg, 2005, Addison-Wesley. Foreword by Peter G. Neumann. Design principles and patterns for secure and reliable design.

DoDI 8500.01, "Cybersecurity" Information Assurance (IA) guidance. http://www.dtic.mil/whs/directives/corres/pdf/850001_2014.pdf

DoDI 8510.01, "Risk Management Framework (RMF) for DoD IT". http://www.dtic.mil/whs/directives/corres/pdf/851001_2014.pdf

License and Thanks!

Author:	Stephen L Arnold
FOSS Hat:	Gentoo Linux Developer
Contact:	answers@vctlabs.com
Revision:	0.3
Date:	2021-01-23, 22:50 PST8PDT
License:	CC-Attribution-ShareAlike
Copyright:	2016 VCT Labs, Inc. and 2018 Orchard Systems, Inc.