“I run a Linux server, so we’re secure”

András Veres-Szentkirályi

Silent Signal
vsza@silentsignal.hu

HACKTIVITY
18 September 2010

SILENT SIGNAL
VÉSZJELZÉS HELYETT...

Linux from a security viewpoint

- we're talking about the kernel, not GNU/Linux distributions

Linux from a security viewpoint

- we're talking about the kernel, not GNU/Linux distributions
- not inherently insecure (satisfying HUP trolls)
  - numerous security features had been added to the mainline kernel, such as ASLR and protection against null-pointer dereferencing
  - there is possibility for hardening using grsecurity and SELinux

Linux from a security viewpoint

- we're talking about the kernel, not GNU/Linux distributions
- not inherently insecure (satisfying HUP trolls)
  - numerous security features had been added to the mainline kernel, such as ASLR and protection against null-pointer dereferencing
  - there is possibility for hardening using grsecurity and SELinux
- nor inherently secure (the point of this talk)
  - mainline kernel maintainers not necessarily look at security as they should be (see also "Security people are leaches" [sic])
  - it's being used for *enterprisey* functionality (eg. running some bloated proprietary BLOB) more and more, which often conflicts with security countermeasures

chroot – basics

- FACT: There are lots of applications that need certain privileges (eg. root) to function properly.
- FACT: If an attacker can take over one of these applications (it'll happen sooner or later), she can access "everything".
- One common solution is using chroot from the UNIX world.
- "A chroot environment can be used to create and host a separate virtualized copy of the software system. [...] This also simplifies the common arrangement of running the potentially-vulnerable parts of a privileged program in a sandbox, in order to pre-emptively contain a security breach.

chroot – basics

- FACT: There are lots of applications that need certain privileges (eg. root) to function properly.
- FACT: If an attacker can take over one of these applications (it'll happen sooner or later), she can access "everything".
- One common solution is using chroot from the UNIX world.
- "A chroot environment can be used to create and host a separate virtualized copy of the software system. [...] This also simplifies the common arrangement of running the potentially-vulnerable parts of a privileged program in a sandbox, in order to pre-emptively contain a security breach. Note that chroot is not necessarily enough to contain a process with root privileges."

  Wikipedia

chroot – attack

- There are several techniques to escape from a chroot "jail".
  - DEMO: Do a chroot to a subdirectory, leaving the current directory outside the jail, then `chdir("..")` a few times.

chroot – attack

- There are several techniques to escape from a chroot "jail".
  - DEMO: Do a chroot to a subdirectory, leaving the current directory outside the jail, then `chdir("..")` a few times.
  - Use `mknod` to create a raw disk device, thereby doing pretty much anything you like to the system.
  - Use `mknod` to create `/dev/mem` and modify kernel memory
  - Find a carelessly-left hard link that leads outside the jail (though symbolic links don't escape jail, hard links do).
  - Use `ptrace` to trace a process living outside the jail. We may be able to modify this program to do our bad stuff on our behalf.
- Almost all jail breaking requires root privileges.

chroot – defense

- "chroot is not and never has been a security tool" – Alan Cox
- TRIVIAL: coders should use `setuid()` after `chroot()`
- grsecurity can restrict chroot in many ways
  - no double `chroot`
  - enforced `chdir("/")` after `chroot`
  - no `mknod`
  - no `ptrace` outside chroot
  - full list: `http://grsecurity.net/features.php`

NX bit – basics

- because of how von Neumann architecture works, there is no difference for the CPU between data and code (remember Smashing the Stack for Fun and Profit from 1996?)
- writable pages (heap, stack) should not be executable thus preventing foreign code execution
- it can be checked using software methods (PaX, ExecShield) but causes overhead
- Wikipedia: "The NX bit, which stands for No eXecute, is a technology used in CPUs to segregate areas of memory for use by either storage of processor instructions (or code) or for storage of data [...] for security reasons."
- support implemented since Linux 2.6.8 (14 August 2004)

NX bit – attack: return to libc

- Wikipedia: "A return-to-libc attack is a computer security attack usually starting with a buffer overflow in which the return address on the stack is replaced by the address of another instruction and an additional portion of the stack is overwritten to provide arguments to this function. This allows attackers to call preexisting functions without the need to inject malicious code into a program."

NX bit – defense

- Stack smashing protection (SSP): detect stack corruption and abort process
  - PaX
  - ProPolice
  - StackGuard
  - StackShield
- Address space layout randomization (ASLR): see later

ASLR

- Wikipedia: "Address space layout randomization (ASLR) is a computer security technique which involves randomly arranging the positions of key data areas, usually including the base of the executable and position of libraries, heap, and stack, in a process's address space."

ASLR

- Wikipedia: "Address space layout randomization (ASLR) is a computer security technique which involves randomly arranging the positions of key data areas, usually including the base of the executable and position of libraries, heap, and stack, in a process's address space."
- implemented in a weak form since Linux 2.6.12 (17 June 2005)
- better implementation in PaX (2001) and ExecShield

ASLR

- Wikipedia: "Address space layout randomization (ASLR) is a computer security technique which involves randomly arranging the positions of key data areas, usually including the base of the executable and position of libraries, heap, and stack, in a process's address space."
- implemented in a weak form since Linux 2.6.12 (17 June 2005)
- better implementation in PaX (2001) and ExecShield
- 8 or 13 bits makes using brute force feasible
- `fork(2)` keeps randomization
- NOP slide, heap spraying
- Return to libc was possible to `linux-gate.so.1` till 2.6.20

Null-pointer dereference

- OWASP: “A null-pointer dereference takes place when a pointer with a value of NULL is used as though it pointed to a valid memory area.”

Null-pointer dereference

- OWASP: “A null-pointer dereference takes place when a pointer with a value of NULL is used as though it pointed to a valid memory area.”
- CWE: “In very rare circumstances and environments, code execution is possible.”

Null-pointer dereference

- OWASP: "A null-pointer dereference takes place when a pointer with a value of NULL is used as though it pointed to a valid memory area."
- CWE: "In very rare circumstances and environments, code execution is possible."
- NULL page can be mapped with mmap
- If the kernel calls the function pointer located at the NULL page, the attacker can execute her code with kernel privileges.
- Fix: `mmap_min_addr` in Linux 2.6.23

Null-pointer dereference

- OWASP: "A null-pointer dereference takes place when a pointer with a value of NULL is used as though it pointed to a valid memory area."
- CWE: "In very rare circumstances and environments, code execution is possible."
- NULL page can be mapped with mmap
- If the kernel calls the function pointer located at the NULL page, the attacker can execute her code with kernel privileges.
- Fix: `mmap_min_addr` in Linux 2.6.23 – workaround has been published in June 2009 by Tavis and cr0 (SUID PA FTW)
- PaX / grsecurity protects against attack like this with KERNEXEC on x86
- Tavis Ormandy found a bug in August 2009 that affects all Linux kernels since 2001 and is exploitable using NPD
- the `mmap_min_addr` issue was fixed in 2.6.30.2

grsecurity

- Wikipedia: "grsecurity is a set of patches for the Linux kernel with an emphasis on enhancing security. Its typical application is in computer systems that accept remote connections from untrusted locations, such as web servers and systems offering shell access to its users."
- PaX is bundled with it, flags data memory non-executable and program memory non-writable
- RBAC, chroot and other (dmesg, logging, etc.) restrictions
- solves a few problems, but hard to configure properly (cf. proprietary BLOBs)

grsecurity

- Wikipedia: "grsecurity is a set of patches for the Linux kernel with an emphasis on enhancing security. Its typical application is in computer systems that accept remote connections from untrusted locations, such as web servers and systems offering shell access to its users."
- PaX is bundled with it, flags data memory non-executable and program memory non-writable
- RBAC, chroot and other (dmesg, logging, etc.) restrictions
- solves a few problems, but hard to configure properly (cf. proprietary BLOBs)
- competing technologies include
  - SELinux: not invulnerable (see PA case)
  - OpenVZ: still vulnerable in case of kernel bugs

Thanks for your attention!