

# Hardening Linux

and introducing Securix GNU/Linux





# Hardening basics

- ✦ From lowest to highest level
- ✦ Tune every part of system and applications
- ✦ Follow standards and security policies
- ✦ Regularly check system health
- ✦ Install security patches when possible
- ✦ Log and audit every action





# Physical security

- ✦ Locked rack
- ✦ BIOS setup password
- ✦ Console in different rack

// How it helps?

- ✦ Avoid unauthorized access where somebody can:
  - shutdown server
  - reboot server into single-user mode and change password
  - boot live CD and access data
  - sniff data on ethernet cable
  - steal hard disks





# Encrypting partitions

- its essential to have encrypted all partitions except /boot and swap ( / , usr, home, var, opt, tmp)

- no impact on resources where is HW acceleration possible

- my recommendation is LUKS

```
cryptsetup -c aes-xts-plain -y -s 512 -h whirlpool luksFormat /dev/xyz
```

// How it helps?

- data stored are unreadable for attacker
- passwords can't be changed during boot from live CD





# Securing Grub

- protect grub using password  
`password --md5 $1$U10TR0$fK/7jE2gCbkBAnzBQWWYf/`
- generate hash using grub-md5-crypt
- protect single-user boot via password as well
- setup fallback option

// How it helps?

- avoid unauthorized single-user mode boot
- fallback in case of problems with new kernel





# Kernel configuration

- ✦ enabled only options which are really needed
  - smaller/faster kernel
  - secure - big piece of code isn't really needed
- ✦ enabled PaX and Grsecurity
  - no LSM
  - robust multi-level security system
- ✦ Securix have minimalistic predefined kernel setup which should boot on many systems by default





# /etc/securetty

- ✦ limit root access to console and serial port only

```
# file: /etc/securetty
# limit root access
console
vc/1
tty1
tty2
# serial console access
ttyS0
ttyS1
...
```


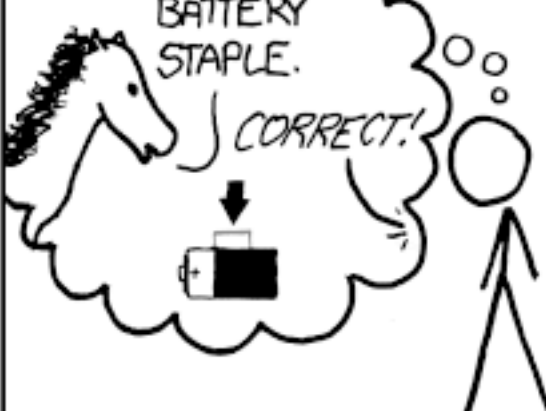




Use phrases, not passwords



# Use phrases, not passwords

<p>□□□□□□□□□□□□□□□□</p> <p>UNCOMMON (NON-GIBBERISH) BASE WORD</p> <p>ORDER UNKNOWN</p> <p>Tr0ub4dor &amp;3</p> <p>CAPS? □</p> <p>COMMON SUBSTITUTIONS □□□</p> <p>NUMERAL □□□</p> <p>PUNCTUATION □□□□</p> <p>(YOU CAN ADD A FEW MORE BITS TO ACCOUNT FOR THE FACT THAT THIS IS ONLY ONE OF A FEW COMMON FORMATS.)</p>	<p>~28 BITS OF ENTROPY</p> <p>□□□□□□□□ □□□□□□□□ □□□ □□□□</p> <p><math>2^{28} = 3 \text{ DAYS AT } 1000 \text{ GUESSES/SEC}</math></p> <p>(PLAUSIBLE ATTACK ON A WEAK REMOTE WEB SERVICE. YES, CRACKING A STOLEN HASH IS FASTER, BUT IT'S NOT WHAT THE AVERAGE USER SHOULD WORRY ABOUT.)</p> <p>DIFFICULTY TO GUESS: <b>EASY</b></p>	<p>WAS IT TROMBONE? NO, TROUBADOR. AND ONE OF THE 0s WAS A ZERO?</p> <p>AND THERE WAS SOME SYMBOL...</p>  <p>DIFFICULTY TO REMEMBER: <b>HARD</b></p>
<p>correct horse battery staple</p> <p>□□□□□□ □□□□□□ □□□□□□ □□□□□□ □□□□□□ □□□□□□ □□□□□□ □□□□□□</p> <p>FOUR RANDOM COMMON WORDS</p>	<p>~44 BITS OF ENTROPY</p> <p>□□□□□□□□□□ □□□□□□□□□□ □□□□□□□□□□ □□□□□□□□□□</p> <p><math>2^{44} = 550 \text{ YEARS AT } 1000 \text{ GUESSES/SEC}</math></p> <p>DIFFICULTY TO GUESS: <b>HARD</b></p>	<p>THAT'S A BATTERY STAPLE.</p> <p>CORRECT!</p>  <p>DIFFICULTY TO REMEMBER: YOU'VE ALREADY MEMORIZED IT</p>

THROUGH 20 YEARS OF EFFORT, WE'VE SUCCESSFULLY TRAINED  
EVERYONE TO USE PASSWORDS THAT ARE HARD FOR HUMANS  
TO REMEMBER, BUT EASY FOR COMPUTERS TO GUESS.



# /etc/sysctl.conf

- ✦ build with focus on security and network stack tuning
- ✦ some examples:

**# Kernel EXEC shield**

**kernel.exec-shield = 1**

**# Make the addresses of mmap base, stack, heap and VDSO page randomized**

**kernel.randomize\_va\_space = 2**

**# Reboot system when kernel panic occur, oops will wait 30 seconds untill call panic()**

**kernel.panic = 30**

**kernel.panic\_on\_oops = 30**

**# Disable magic-sysrq key**

**kernel.sysrq = 0**





# /etc/sysctl.conf

- **# Prevent SYN attack**  
net.ipv4.tcp\_syncookies = 1  
net.ipv4.tcp\_max\_syn\_backlog = 4096  
net.ipv4.tcp\_syn\_retries = 5  
net.ipv4.tcp\_synack\_retries = 2  
  
**# Enable IP spoofing protection, turn on source route verification**  
net.ipv4.conf.all.rp\_filter = 1  
net.ipv4.conf.default.rp\_filter = 1  
  
**# Increase allowed local port range**  
net.ipv4.ip\_local\_port\_range = 1024 64000  
  
**# Enable tcp\_window\_scaling**  
net.ipv4.tcp\_window\_scaling = 1
- ...and much more





# iptables configuration

- ✦ script for pushing/editing/saving firewall rules
- ✦ port scan detection
- ✦ access logging
- ✦ traffic divided into chains
- ✦ strict default policy





# /etc/security/limits.conf

```
✱ # default limits
*   soft  core    0
*   hard  core    0
*   hard  nproc   200
*   hard  nofile  2000
*   hard  fsize   1024000
*   -     maxlogins 3

# operator group limits
@operators  hard  nproc    2000
@operators  hard  nofile   4000
@operators  hard  fsize    10240000
@operators  -     maxlogins 10

# service group limits
@services   hard  nproc    40000
@services   hard  nofile   45000
@services   hard  fsize    1024000000
@services   -     maxlogins 100
```





# /etc/ssh/sshd\_config

- **Port 5522**  
**Protocol 2**  
**ServerKeyBits 2048**  
**Ciphers aes256-cbc,aes256-ctr,blowfish-cbc**  
**Compression yes**  
**ClientAliveInterval 15**  
**ClientAliveCountMax 3**  
**TCPKeepAlive no**  
  
**PermitRootLogin no**  
**PermitEmptyPasswords no**  
**UsePrivilegeSeparation yes**  
**StrictModes yes**  
  
**AllowTcpForwarding no**  
**X11Forwarding no**  
**GatewayPorts no**  
**...**





# PaX - NOEXEC

- ✦ The goal of NOEXEC is to prevent the injection and execution of code into a task's address space and render this exploit technique unusable under PaX.
  - segmentation based non-executable page protection [IA-32 only] (SEGMEXEC)
  - paging based non-executable page protection (PAGEEXEC)
  - mmap() and mprotect() restrictions (MPROTECT)





# Pax - ASLR

- decrease success rate of attacks that require advanced knowledge of memory addresses via Address Space Layout Randomization (ASLR)
  - Main executable code/data/bss segment base address randomization (RANDEXEC) - ET\_EXEC ELF
  - mmap() and brk() managed memory [libraries, heap, thread stacks, shared memory, etc] address randomization (RANDMMAP) ET\_DYN ELF
  - User stack base address randomization (RANDUSTACK)
  - Kernel stack base address randomization (RANDKSTACK)





# Grsecurity

- “Unlike other expensive security “solutions” that pretend to achieve security through known-vulnerability patching, signature-based detection, or other reactive methods, grsecurity provides real proactive security. The only solution that hardens both your applications and operating system, grsecurity is essential for public-facing servers and shared-hosting environments.

Grsecurity increased authentication for administrators, audit important system events, and confine your system with no manual configuration through advanced Role-Based Access Control.”





# Grsecurity - features

- ✦ high scalable Role-Based Access Control
- ✦ learning mode for RBAC policies
- ✦ restrict access to /dev/kmem, /proc/<pid>/maps, /proc, kernel processes, linking, FIFO, dmesg, IO, ptrace, ...
- ✦ trusted path execution
- ✦ hardened chroot environment
- ✦ fork bomb protection & randomized PIDs





# Grsecurity - features 2

- network protections - randomized IP IDs, larger entropy pools, random TCP sequence numbers, randomized TCP source ports, tcp/udp blackhole, altered ping payload, socket restrictions, ...
- auditing - exec & chroot exec, ptrace, chdir, un/mount, signal sent, fork failure, time change, append client IP address where possible, denied RWX mmap/mprotect (PaX)
- ...and much more












# Grsecurity benchmark

- Apache Benchmark:  
This test profile measures how many requests per second a given system can sustain when carrying out 700,000 requests with 100 requests being carried out concurrently.

Source: <http://grsecurity.net/~spender/benchmarks.txt>

-  Vanilla
-  All grsecurity/PaX features enabled
-  All but RBAC
-  All but UDEREF
-  All but KERNEXEC/UDEREF
-  All but KERNEXEC/UDEREF/SANITIZE
-  All but KERNEXEC/UDEREF/SANITIZE/STACKLEAK



# Grsecurity benchmark

- Apache Benchmark:  
This test profile measures how many requests per second a given system can sustain when carrying out 700,000 requests with 100 requests being carried out concurrently.

Source: <http://grsecurity.net/~spender/benchmarks.txt>





# Why not SELinux?

- ✦ build as Linux Security Module (LSM)
- ✦ doesn't contain features as Grsecurity (NX, ASLR, DoS and fingerprint mitigation, entropy pools, auditing, ...)
- ✦ complicated policy & no learning mode
- ✦ worse performance in benchmark tests
- ✦ developed and maintained by NSA ("Big Brother")





# Securix GNU/Linux





# Securix GNU/Linux

- follow “best practices” and standards with high focus on security and performance
- all services and system configuration secured by default
- own monitoring system which can fix or inform administrator about problems by email or motd (rkhunter, checksec.sh, secxmon.sh, secxwatch.sh, XCCDF, OVAL, ...)
- update tool which can align system with current Securix configuration
- actively implement protection against new attack vectors (via update tool) where applicable





# Securix GNU/Linux





# Securix GNU/Linux

- anyone can contribute or suggest improvement!
- build good pre-defined RBAC policies
- binaries build with SSP and PIE protection
- proactive security against buffer/heap/stack overflow and exploits
- every new settings must be tested on production environment
- basically “it is still Hardened Gentoo” with great support on forums, IRCs and mailing lists
- ...stay tuned





# Securix Installer





# Securix Installer

- ✦ written in bash under GPLv3 license
- ✦ system environment check and setup
- ✦ advanced auto-partitioning with LVM and LUKS support (noatime, nodev, nosuid, noexec)
- ✦ auto-detection/setup of architecture, bonding, serial console, http proxy, profile, ...
- ✦ all you need is setup some essential variables like root passphrase, hostname, root mail, etc.
- ✦ following methods KISS (Keep It Simple Stupid) and DASQ (Don't Ask Stupid Questions)





# Q&A?





# Thank you!

Martin Čmelík  
[ cm3l1k1 ]

[martin.cmelik \[ at \] security-portal.cz](mailto:martin.cmelik@security-portal.cz)  
[www.security-portal.cz](http://www.security-portal.cz)  
[www.securix.org](http://www.securix.org)





# Links

- ✦ [www.securix.org](http://www.securix.org)
- ✦ [www.grsecurity.net](http://www.grsecurity.net)
- ✦ <http://pax.grsecurity.net/docs/>
- ✦ <http://www.gentoo.org/proj/en/hardened/>
- ✦ <http://en.wikibooks.org/wiki/Grsecurity>
- ✦ <http://www.pjvenda.net/linux/doc/pax-performance/>
- ✦ [www.cs.virginia.edu/~jcg8f/SELinux%20grsecurity%20paper.pdf](http://www.cs.virginia.edu/~jcg8f/SELinux%20grsecurity%20paper.pdf)
- ✦ [www.ece.cmu.edu/~dbrumley/courses/18739c-s11/docs/aslr.pdf](http://www.ece.cmu.edu/~dbrumley/courses/18739c-s11/docs/aslr.pdf)
- ✦ <http://www.cyberciti.biz/>
- ✦ ...





# Top security stories of 2011

(by Kaspersky)





# Top security stories of 2011

(by Kaspersky)

- ✦ Rise of “Hactivism” - Anonymous, LulzSec, ...
- ✦ HBGary Federal Hack
- ✦ RSA, Comodo and DigiNotar incident
- ✦ Duqu (Stuxnet) - undetectable for years
- ✦ Sony Playstation Network hack
- ✦ Rise of Android / Mobile malware
- ✦ CarrierIQ - spy and monitoring tool
- ✦ ...we need Securix

