CHKROOTKIT
EATING APTs SINCE 1997
AGENDA

- Background
- Why chkrootkit?
- Project definitions
- Features
- Usage modes
- Future
- Demo
$ WHO I AM

- Security Analyst since 1992 and Unix Admin since 1986
- Worked and still working in some infosec areas, such as Incident Response, Forensics, Cyber Security Intel, Network & Wireless security and New Technologies Evaluation
- Talks in Blackhat, Defcon, Auscert, Bluehat, Virus Bulletin and now SECTOR!
- Wrote books about security in Portuguese
- Author of some open source tools like beholder (Wireless IDS), bluetooth tools and others.
- Author and maintainer of Chkrootkit
BACKGROUND

Rootkit definition:

Code or any piece of code, used after an intrusion, to hide the invader's presence on the victim's box.
BACKGROUND

Rootkit definition:

Code or any piece of code, used after an intrusion, to hide the invader's presence on the victim's box.

1989 – PHRACK MAGAZINE PUBLISHES CODES TO HIDDEN INVASORS'S PRESENCE
1994 – CERT ADVISORIES INCLUDE TERM ROOTKIT
1995 – TOOLKIT CALLED ROOTKIT BECAME POPULAR
1997 – REFERENCES ABOUT MALICIOUS LKM CODES

CHKROOTIK BEGINS IT'S DEVELOPMENT
In 1995 Internet became commercial in Brazil

Linux have a attractive price, but no one knew Linux

Several compromised boxes with similar toolkits

During the investigations, I ran a series of repetitive commands

The most part of that process could be automated
**O QUE O CHKROOTKIT IS/IS NOT**

- Malware unix-based host scanner
- forensics tool
- Suite of programs
- Intrusion Detection tool
- Network security tool
- Removal tool
PROJECT CHALLENGES

- Run in most Unix as possible (*not* only on Linux distros)
- Easy to install
- Comprehensive, easy to use
- As few dependencies as possible
- Widely deployable
- Written in a portable languages
  - Posix Shell
  - C Ansi
- Easy to contribute
SECURITY BY DESIGN (OR THE ART OF SAYING NO)

- NO Databases
- NO Config files
- NO PATH dependence
  Distos are doing it wrong!
- NO network services
- NO log files
- Monolithic structure
CAVEATS?

Almost 3k shell script lines

```
$ ./chkrootkit
./chkrootkit needs root privileges
$ 
```
CAVEATS?

REMEMBER: CHKROOTKIT IS A USER SPACE APPLICATION
FEATURES

- Runs in most Unix in the market
  - Tru64, HPUX, AIX, etc.
- Can run the shell module independently
  - C compiler is not always available
  - Compilation of C programs eventually fail
- Modes
  - HIDS - Realtime
  - Forensics - Post-mortem

detecting more LKMs
Features

- Checks are also based on signatures, **BUT**

  Behavior checks were added in the 90's

```bash
#... Suspects PHP files
if [ "$QUIET" != "t" ]; then
  printf "Searching for suspect PHP files... ": fi

#... shell history anomalies
if [ "$QUIET" != "t" ]; then \
  printf "Searching for anomalies in shell history files... ": fi

#... suspicious files and dirs
DIR="${ROOTDIR}usr/lib"
[ -d ${ROOTDIR}usr/man ] &&
[ -d ${ROOTDIR}usr/share ]

./chkrootkit lkm
ROOTDIR is `/'
Checking `lkm'...
2 process hidden for readdir command
2 process hidden for ps command
chkproc: Warning: Possible LKM Trojan installed
chkdirs: nothing detected
```
COMPONENTS

CHKROOTKIT
SHELL SCRIPT

CHkdirs
CHKPROC
IFPROMISC

CHKUTMP
CHKWTMP
CHKLASTLOG

STRING-STATIC
REAL TIME MODES

- Real time
- Post-mortem
INDICATORS OF COMPROMISE (IOC)

- Malicious Loadable Kernel Modules (LKM)
- Trojaned binaries
- The suspects:
  - Files and directories
  - Ports and services
  - Processes
  - Kernel memory
Double check - `chkdirs.c`

```
# ls -la | head -3
total 196
drwxr-xr-x 2 root root 4096 Mai 22 06:27 .
drwxr-xr-x 20 nelson nelson 4096 Mar 11 19:38 ..
# cd ..
# ls -la | egrep ^d | wc -l
20
#
```
Double check - `chkproc.c`

Diagram:
- Hardware
- Kernel
- `/proc`
- `ps`
USAGE MODES

- Real time
- Post-mortem
POSTMORTEM USAGE
POSTMORTEM USAGE

./chkrootkit -r mounted_suspectDisk

-\r dir use dir as the root directory
-\p dir1:dir2:dirN path for the external commands used by chkrootkit

./chkrootkit -p mounted_readonlyDiskorCD
FUTURE
After 20 years Chkrootkit, is, surprisingly, still efficient

Some challengers to keep tool working

- Changes in kernel architecture, subsystems, syscalls, commands, etc.
- Still useful for legacy systems
- New behaviors checks are being increasingly added