BIND 9 Security

(Part 2 - AppArmor, SecompBPF and Firejail)

Carsten Strotmann and the ISC Team

Welcome

Welcome to part two of our BIND 9 security webinar series

In this Webinar

- Linux security modules
- AppArmor
- The AppArmor Policy for BIND 9
- Fixing AppArmor file permission issues
- Syscall "Firewall" SecompBPF
- Securing BIND 9 with Firejail
- Systemd Security for BIND 9
- Hands-On lab

Process Hardening

Motivation of process hardening

- BIND 9, as all network enabled software, had security issues in the past
  - There are likely new security issues being found in the future
  - The process hardening recipes contained in this webinar series do minimize the impact of security issues in networked software
  - However the process hardening increases the complexity of the setup and might prevent certain features of BIND 9 to work
  - Do not apply these hardening recipes without doing throughout research and a good understanding how they work and what they restrict

Detecting attacks

- The process hardening methods also enable the operator to detect attacks on the BIND 9 process
  - If the policy gets violated, the BIND 9 process is terminated and additional information is logged (for example into the Systemd journal or the audit log)
  - To catch attacks, it is recommended that the log-files, and also the BIND 9 `named` process is monitored for restarts
    - An unusual restart can be evidence of an attack

When BIND 9 acts strangely

- The process hardening alters the way how a Linux system works
    - If BIND 9 starts to act strangely, try to back-out some or all process hardening steps until the strangeness goes away
    - Do not open a ticket with the operating system vendor (Debian, Suse, Canonical) or ISC before testing the misbehaving function on a non-hardened install

Linux security modules

LSM (Linux Security Modules)

- LSMs are extensions of the Linux kernel.
  - We've discussed the place of LSMs in the Linux Kernel in the last webinar
- Major LSMs
  - SELinux (last webinar)
  - AppArmor (this webinar)
  - SMACK (mostly used in embedded systems)
  - TOMOYO (powerful, but less popular)

AppArmor

AppArmor LSM

- AppArmor is a Linux Security Module that implements *Mandatory Access Control*
  - The MAC policy works on the file path of objects (in contrast to *SELinux*, where the policy is selected by security label on the file-system independent of the path)
  - AppArmor is available in Linux since Kernel version 2.6.36 (October 2010)
  - AppArmor is enabled by default in Debian 10/11. It is also available in Suse Linux, Ubuntu, Gentoo and Arch-Linux

AppArmor status

- The command `aa-status` can be used to check if AppArmor is active (example from a default Debian 11 install, the `named` profile is loaded and AppArmor is active)

```
# aa-status
apparmor module is loaded.
8 profiles are loaded.
8 profiles are in enforce mode.
   /usr/bin/man
   firejail-default
   lsb_release
   man_filter
   man_groff
   named
   nvidia_modprobe
   nvidia_modprobe//kmod
0 profiles are in complain mode.
0 processes have profiles defined.
0 processes are in enforce mode.
0 processes are in complain mode.
0 processes are unconfined but have a profile defined.
```

AppArmor Logging

- AppArmor (like SELinux) uses the Linux Audit Subsystem for logging.
  - It is recommended to have the `auditd` service installed and running

```
# apt install auditd
```

AppArmor Logging

- AppArmor security policy violations will be written to `/var/log/audit/audit.log`
  - Unfortunately AppArmor messages are currently not compatible with the format expected by the `ausearch` tool
  - The audit log file must be filtered manually

```
# grep -i denied /var/log/audit/audit.log
type=AVC msg=audit(1634287113.597:118): apparmor="DENIED" operation="open"
    profile="named" name="/srv/bind/zones/example.com"
    pid=41687 comm="isc-worker0000" requested_mask="r"
    denied_mask="r" fsuid=106 ouid=0FSUID="bind" OUID="root"
```

AppArmor on Processes

- The `-Z` parameter to the `ps` command can be used to list processes that are secured by AppArmor:

```
# ps -efZ | grep -v unconfined
LABEL                           UID          PID    PPID  C STIME TTY          TIME CMD
named (enforce)                 bind       41687       1  0 10:38 ?        00:00:00 /usr/sbin/named -f
```

AppArmor Tools

- Additional AppArmor tools (such as `aa-unconfined`) can be found in the Debian packet `apparmor-utils`

```
apt install apparmor-utils
```

- The command `aa-unconfined` will check for running processes listening on a network socket and will print their AppArmor security status:

```
# aa-unconfined
451 /usr/sbin/dhclient (/sbin/dhclient) not confined
675 /usr/sbin/sshd (sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups) not confined
41687 /usr/sbin/named confined by 'named (enforce)'
```

AppArmor Modes

AppArmor Modes

- AppArmor can work in several modes
  - The modes can be activated per AppArmor profile (service/application)
  - Mode *disabled* - AppArmor is not active
  - Mode *complain* - AppArmor is active but not enforcing the policy
  - Mode *enforce* - AppArmor is active and is enforcing the policy
  - Mode *audit* - AppArmor is active, enforcing and is auditing system calls

Setting AppArmor Modes

- The tools `aa-complain`, `aa-enforce`, `aa-disable` and `aa-audit` can be used to switch the AppArmor mode on a program.
  - The process needs to be restarted after changing the AppArmor mode:

```
# aa-complain /usr/sbin/named
Setting /usr/sbin/named to complain mode.
Warning: profile named represents multiple programs
# systemctl restart bind9
```

AppArmor and BIND 9

The AppArmor Policy for BIND 9

- The AppArmor Policy file for BIND 9 can be found in `/etc/apparmor.d/usr.sbin.named`:

```
# vim:syntax=apparmor
# Last Modified: Fri Jun  1 16:43:22 2007
#include <tunables/global>

profile named /usr/sbin/named flags=(attach_disconnected) {
  #include <abstractions/base>
  #include <abstractions/nameservice>

  capability net_bind_service,
  capability setgid,
  capability setuid,
[...]
  # /etc/bind should be read-only for bind
  # /var/lib/bind is for dynamically updated zone (and journal) files.
  # /var/cache/bind is for slave/stub data, since we're not the origin of it.
  # See /usr/share/doc/bind9/README.Debian.gz
  /etc/bind/** r,
  /var/lib/bind/** rw,
  /var/lib/bind/ rw,
  /var/cache/bind/** lrw,
  /var/cache/bind/ rw,

  # Database file used by allow-new-zones
  /var/cache/bind/_default.nzd-lock rwk,
[...]
```

The AppArmor Profile for BIND 9

- This profile is called `named` and it is for the process started from `/usr/sbin/named`
    - The `attach_disconnected` flag tells AppArmor how to handle file access to *disconnected* files. Disconnected files are open files where the application still has access to the file handle, but the file cannot be looked up by name anymore.

```
profile named /usr/sbin/named flags=(attach_disconnected) {
```

The AppArmor Profile for BIND 9

- The profile specifies the Linux capabilities available to the `named` process

```
capability net_bind_service, # open a network socket
capability setgid,           # change the group ID of the process
capability setuid,           # change the user ID of the process
capability sys_chroot,       # use the chroot syscall
capability sys_resource,     # change resource restrictions on the process
```

The AppArmor Profile for BIND 9

- The profile defines the file access permissions that might overwrite the file-system permissions:

```
/etc/bind/** r,             # read access including sub-directories
/var/lib/bind/** rw,        # read/write access incl. sub-directories
/var/lib/bind/ rw,          # read/write access for this directory
/var/cache/bind/** lrw,     # read/write and link permission
/var/cache/bind/ rw,        # read/write access for this directory
```

AppArmor Profile for BIND 9

- Local additions and overwrites to the system AppArmor profile can be stored in `/etc/apparmor.d/local/usr.sbin.named`. This file is included at the end of the mail profile:

```
  #include <local/usr.sbin.named>
}
```

Finding and fixing AppArmor issues

Fixing AppArmor file permission issues

- File permission issues are the main problem type with AppArmor installs
- The command `aa-logprof` can be used to find AppArmor permission issues and to create an extension to the AppArmor profile:

```
# aa-logprof
Reading log entries from /var/log/audit/audit.log.
Updating AppArmor profiles in /etc/apparmor.d.
Complain-mode changes:

Profile:  named
Path:     /srv/bind/zones/example.com
New Mode: r
Severity: 4

 [1 - #include <abstractions/ubuntu-browsers.d/user-files>]
  2 - /srv/bind/zones/example.com r,
(A)llow / [(D)eny] / (I)gnore / (G)lob / Glob with (E)xtension / (N)ew / Audi(t) / Abo(r)t / (F)inish
```

Fixing AppArmor file permission issues

- The changes will overwrite the profile provides by the Linux system
  - Overwriting the system delivered profile is not recommended, as it might get replaced by an system update
  - Best is to view the changes and manually add them to `/etc/apparmor.d/local`

```
Adding /srv/bind/zones/example.com r, to profile.
Enforce-mode changes:

= Changed Local Profiles =

The following local profiles were changed. Would you like to save them?

 [1 - named]
(S)ave Changes / Save Selec(t)ed Profile / [(V)iew Changes] / View Changes b/w (C)lean profiles / Abo(r
```

Example Extension of the BIND 9 AppArmor Profile

- This extension in `/etc/apparmor.d/local/usr.sbin.named` allows the BIND 9 process `named` to read files below `/srv/bind/zones/`:

```
/srv/bind/zones/** r,
```

Securing a non-system BIND 9 install

- AppArmor secures processes based on the path of their binaries
  - If the program binary is stored under a non-default path, the process is not protected by AppArmor
  - For example this can happen for BIND 9 versions that have been compiled from source and that are located under `/usr/local` or `/opt`
- The command line tool `aa-exec` can be used to start any binary under the control of AppArmor

```
# aa-exec -p named /opt/bind/sbin/named -f -u bind
```

Securing a non-system BIND 9 install

- Adjust the Systemd unit for BIND 9 to execute BIND 9 through `aa-exec`
- Check that the BIND 9 process is confined by AppArmor:

```
# ps auxZ | grep named
named (enforce)                 bind      102252  0.4  0.9 241200 19388 pts/9    Sl+  12:39   0:00 /opt
unconfined                      root      102257  0.0  0.0   6256   720 pts/3    S+   12:39   0:00 grep
```

Syscall "Firewall" SecompBPF

SecompBPF

- SecompBPF (SECure COMPuting with filters) is a security technology available in modern Linux systems
    - It's a *Firewall* for system-calls
- By default, every process can issue any system-call towards the Linux kernel
    - but not every process needs access to all the system-calls
    - BIND 9 for example does not need to load kernel modules, change the system time or set a new host-name on the server

SecompBPF

- SecompBPF can be used to restrict the system-calls available to a process
- Systemd and Process-Sandbox tools like *Firejail* can be used to configure SecompBPF

Securing BIND 9 with Firejail

Firejail Sandbox

- Firejail (https://firejail.wordpress.com/) is a security sandbox for Linux processes
  - It was originally developed for Desktop applications (such as the Firefox browser), but it can be used for background server applications as well
  - Firejail creates a secure sandbox around the process using Linux name-spaces (container) and SecompBPF

Installing Firejail

- The Debian packet `firejail` contains the Firejail command line tools, `firejail-profiles` contains the security profiles for popular applications

```
# apt install firejail firejail-profiles
```

- The main configuration file for Firejail is `/etc/firejail/firejail.config`
- Firejail integrates with AppArmor if available

Firejail Profile for BIND 9

- The Firejail profiles are stored in `/etc/firejail`

```
# ls -l /etc/firejail/ | head
total 4460
-rw-r--r-- 1 root root  1146 Feb 27  2021 0ad.profile
-rw-r--r-- 1 root root   841 Feb 27  2021 2048-qt.profile
-rw-r--r-- 1 root root   310 Feb 27  2021 7za.profile
-rw-r--r-- 1 root root   310 Feb 27  2021 7z.profile
-rw-r--r-- 1 root root   310 Feb 27  2021 7zr.profile
-rw-r--r-- 1 root root   906 Feb 27  2021 abiword.profile
-rw-r--r-- 1 root root   521 Feb 27  2021 abrowser.profile
-rw-r--r-- 1 root root   258 Feb 27  2021 acat.profile
-rw-r--r-- 1 root root   260 Feb 27  2021 adiff.profile
```

- Firejail provides profiles for `dig`, `host` and `nslookup`, but not for the BIND 9 process
- A template for a BIND 9 Firejail profile can be found at https://webinar.defaultroutes.de/webinar/07-firejail.html

List Firejail processes

- The command `firejail --list` can be used to list all processes that are protected by Firejail

```
# firejail --list
49757:root::firejail named
```

Firejail Top

- The command `firejail --top` will display the resource usage of all processes running under Firejail

```
PID   User      RES(KiB) SHR(KiB) CPU%  Prcs Uptime    Command
49757 root      17740    8324     0.0   3    00:02:50  firejail named -f -u bind
```

- The command `firejail --netstats` will display network statistics for all processes running under Firejail

Firemon

- The command `firemon` can be used to monitor the execution of a process under Firejail control:

```
# firemon

12:25:10 exec 49413 (root) NEW SANDBOX: firejail /usr/sbin/named -g
12:25:10 fork 49413 (root) firejail /usr/sbin/named -g
        child 49414 firejail /usr/sbin/named -g
12:25:10 fork 49413 (root) firejail /usr/sbin/named -g
        child 49415 firejail /usr/sbin/named -g
12:25:10 exit 49415 (root)
12:25:10 fork 49414 (root) firejail /usr/sbin/named -g
        child 49416 firejail /usr/sbin/named -g
12:25:10 exec 49416 (root) /run/firejail/lib/fseccomp protocol build inet,inet6 /run/firejail/mnt/secco
12:25:10 exit 49416 (root)
12:25:10 fork 49414 (root) firejail /usr/sbin/named -g
        child 49417 firejail /usr/sbin/named -g
12:25:10 exit 49417 (root)
12:25:10 fork 49414 (root) firejail /usr/sbin/named -g
        child 49418 firejail /usr/sbin/named -g
12:25:10 exec 49418 (root) /run/firejail/lib/fseccomp drop /run/firejail/mnt/seccomp/seccomp /run/firej
```

Systemd Security for BIND 9

Systemd Unit Security

- Linux `systemd` can apply many security measures to processes started from Systemd units
- The number of security related configuration options increases between Systemd releases
- The command `systemd-analyze security` can be used to check the security *score* on a Systemd unit

Systemd-analyze

- The score for the default BIND 9 unit file delivered with Debian 11 is rather poor (9.6, where 10 is the worst and 0 is the best)

```
# systemd-analyze security bind9
  NAME                                                DESCRIPTION
✗ PrivateNetwork=                                     Service has access to the host's network
✗ User=/DynamicUser=                                  Service runs as root user
✗ CapabilityBoundingSet=~CAP_SET(UID|GID|PCAP)        Service may change UID/GID identities/cap
✗ CapabilityBoundingSet=~CAP_SYS_ADMIN                Service has administrator privileges
✗ CapabilityBoundingSet=~CAP_SYS_PTRACE               Service has ptrace() debugging abilities
✗ RestrictAddressFamilies=~AF_(INET|INET6)            Service may allocate Internet sockets
✗ RestrictNamespaces=~CLONE_NEWUSER                   Service may create user namespaces
✗ RestrictAddressFamilies=~…                          Service may allocate exotic sockets

✗ CapabilityBoundingSet=~CAP_(CHOWN|FSETID|SETFCAP)   Service may change file ownership/access
[...]
✗ CapabilityBoundingSet=~CAP_SYS_TTY_CONFIG           Service may issue vhangup()
✗ CapabilityBoundingSet=~CAP_WAKE_ALARM               Service may program timers that wake up t
✗ RestrictAddressFamilies=~AF_UNIX                    Service may allocate local sockets
✗ ProcSubset=                                         Service has full access to non-process /p

→ Overall exposure level for named.service: 9.6 UNSAFE 😨
```

Hardened Systemd Unit for BIND 9

- A hardened Systemd unit for BIND 9 for Debian 11 can be found at https://webinar.defaultroutes.de/webinar/07-bind9-systemd.html
  - With the hardened unit, the Systemd security score is 3.2 (10 is very bad, 0 is best)
- The hardened systemd unit does conflict with Firejail, use either one but not both
- Systemd security or Firejail can be used together with the Linux Security Module such as AppArmor or SELinux

Systemd Resource usage display

- The command `systemd-cgtop`

```
Control Group                                                      Tasks   %CPU   Memory  Inpu
/                                                                    104  100.4     1.6G
system.slice                                                          35   75.9   200.5M
system.slice/named.service                                             5   29.0    30.1M
system.slice/crowdsec.service                                          7   25.9    37.6M
user.slice                                                            14   23.3     1.3G
user.slice/user-0.slice                                               14   23.3     1.3G
```

Resources

- AppArmor Wiki https://gitlab.com/apparmor/apparmor/-/wikis/home
- AppArmor disconnected_path Flag: https://gitlab.com/apparmor/apparmor/-/wikis/Release_Notes_2.5#path-name-lookup-and-mediation-of
- AppArmor technical documentation http://lkml.iu.edu/hypermail/linux/kernel/0706.1/0805/techdoc.pdf
- SecompBPF https://www.kernel.org/doc/html/latest/userspace-api/seccomp_filter.html

Process hardening - Final words

- SELinux, AppArmor, Firejail and Systemd-Unit Security are alternatives to reach the same goal: restrict the impact of a security issue in networked software
  - It is not required (nor recommended) to deploy all these methods at once - choose the one that suits your deployment best
    - If you OS already comes with SELinux or AppArmor enabled, use it
    - If you Linux is using Systemd, use the extra security settings on the unit files
    - If SELinux, AppArmor or Systemd are not an option, Firejail might be a good alternative

Next webinars

- November 16 - Instrumenting BIND 9 on Linux with BCC/eBPF
- December 15 - DNS Fragmentation: Real-World measurements, impact and mitigation

Questions and Answers

Hands-On

- We have prepared a VM machine for every participant
- This time the sessions does not build upon each other and do not need to be done in order
- find the instructions at https://webinar.defaultroutes.de/webinar/07-apparmor-workshop.html