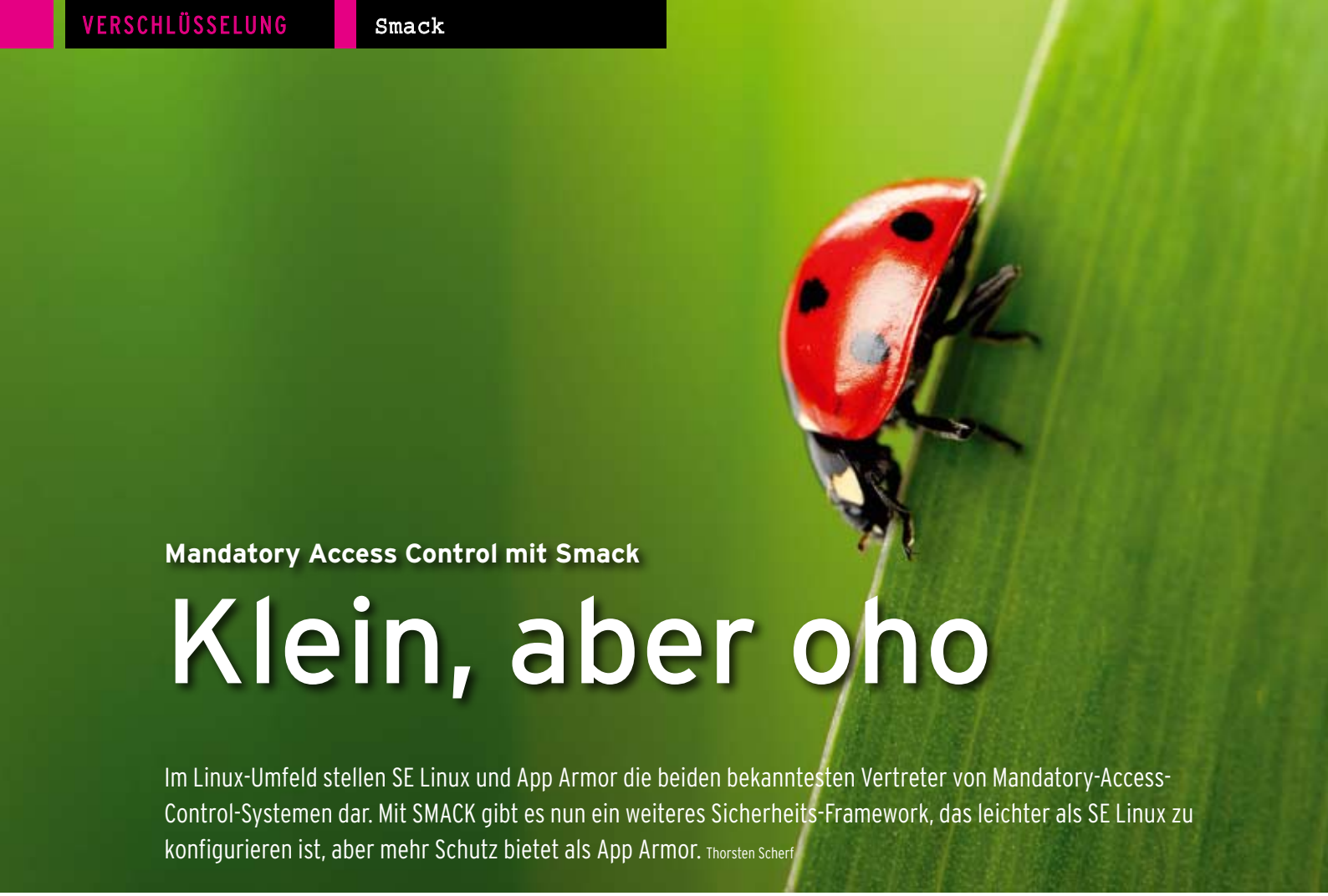

Mandatory Access Control mit Smack

Klein, aber oho

Im Linux-Umfeld stellen SE Linux und App Armor die beiden bekanntesten Vertreter von Mandatory-Access-Control-Systemen dar. Mit SMACK gibt es nun ein weiteres Sicherheits-Framework, das leichter als SE Linux zu konfigurieren ist, aber mehr Schutz bietet als App Armor. Thorsten Scherf

Auf klassischen Linux-Systemen mit Discretionary-Access-Control (DAC) hat ein Benutzer üblicherweise komplette Kontrolle über seine eigenen Objekte. Ob es sich bei dabei um Dateien, Verzeichnisse oder Netzwerkports handelt, spielt keine Rolle. Auf MAC-Systemen funktioniert dies anders. Hier ist ein zentrales Regelwerk dafür verantwortlich, die Zugriffe von Benutzern oder auch Prozessen, allgemein als Subjekte bezeichnet, auf die einzelnen Objekte zu regeln. Ist ein Zugriff nicht explizit erlaubt, so ist er nicht gestattet und wird somit unterbunden. Somit sind Mandatory-Access-Control Systeme (MAC) von Haus aus besser gegen die immer wieder auftauchenden Schwachstellen in Software gewappnet.

Auf klassischen Unix-Systemen wird diese Form der Zugriffssteuerung meistens durch das so genannte Bell & La-Padula-Modell implementiert. Hierbei kommen dann unterschiedliche Sensitivitäten und Kategorien zum Einsatz, mit denen der Administrator die Zugriffssteuerung zwischen den einzelnen Subjekten und Objekten sehr fein granuliert regeln kann. Auf Linux-Systemen hat sich in den letzten Jahren SE Linux als MAC-System durchgesetzt. So verwenden es mittlerweile auch Anbieter, die bisher auf App Armor gesetzt hatten.

SE Linux kennt unterschiedliche Formen der MAC-Implementierung. Meistens kommt das so genannte Type-Enforcement (TE) oder die Role-based-Access-Control (RBAC) zum Einsatz. Auch wenn das Mehr an Sicherheit, das durch den Einsatz von SE Linux erreichbar ist, außer Frage steht, so ist gleichzeitig die Konfiguration eines SE-Linux-Systems relativ kompliziert.

Die Einführung von App Armor versprach hier Abhilfe, da der Admin hierbei lediglich auf Pfad-Ebene des eingesetzten Dateisystems eine Zugriffsentscheidung trifft. Auf der anderen Seite gilt diese Implementierung auch nicht als sehr sicher. Der ehemalige Chefentwickler von App Armor steht heute übrigens auf der Gehaltsliste von Microsoft und ist für die Verwaltung des MAC-Subsystems auf aktuellen Windows-Plattformen zuständig.

Saubere Arbeit

Mit der Simplified Mandatory Access Control (SMACK) geht nun ein weiteres MAC-System ins Rennen um das beste Security-Subsystem. Dass SMACK es nach nur sehr kurzer Anlaufzeit in den offiziellen Linux-Kernel geschafft hat, spricht für die gute Code-Qualität. Dies ist nicht weiter verwunderlich, wenn man sich die früheren Projekte des Hauptentwicklers Casey Schaufler etwas näher ansieht: Angefangen von der ersten Unix-Portierung auf 32-Bit-Systeme, über Trusted Irix bis hin zur Mitarbeit am POSIX P1003.1e/2c-Draft.

Mit SMACK [1] stellt Casey Schaufler nun sein neuestes Projekt vor. Es besteht aus einem Linux-Kernel mit SMACK als Linux-Security-Modul (LSM), einem Start-Skript zum Laden der entsprechenden Regelsätze, Konfigurationsdaten, einem Administrationsinterface in Form des Pseudodateisystems »smackfs« und einigen angepassten Applikationen. Da nun auch SMACK (neben SE Linux, App Armor oder Tomoyo [5]) auf die LSMs im Linux-Kernel zurückgreift, dürfte sich nun endgültig die Diskussion erledigt haben, ob die LSM-API überhaupt noch eine Existenzberechtigung hat. Lange Zeit war es so, dass lediglich SELinux auf sie zurückgriffen hat. Zu Recht hatten sich da einige Kernel-Entwickler gefragt, ob es nicht sinnvoller wäre, die LSM-API abzuschaffen und SE Linux als einziges Sicherheitsframework fix im Kernel zu verankern.

Anders als SE Linux implementiert SMACK jedoch kein Domain Type Enforcement (DTE), sondern arbeitet lediglich mit einfachen Labeln, die der

Administrator den einzelnen Objekten und Subjekten zuweisen kann. Bei Benutzern werden diese zur Laufzeit aus einer Konfigurationsdatei ausgelesen und dynamisch zugewiesen, Prozesse erben grundsätzlich das Label vom Benutzer, der den Prozess gestartet hat. Bei Datei- und Verzeichnis-Objekten stehen diese Label in den erweiterten Attributen. Hier existiert mit »security.SMACK64« ein eigenes Feld für das SMACK-Label, das bis zu 23 Zeichen lang sein kann. Einige Label bringt SMACK von Haus aus mit:

- _ („Floor“)
- ^ („Hat“)
- * („Star“)
- ? („Huh“)
- @ („Internet“)

In der Standardeinstellung erhalten sowohl Subjekte wie auch Objekte das interne Label Floor (»_«). Über die Datei »/etc/smack/user« kann man einzelnen Benutzern eigene Label zuweisen. Allerdings sind momentan keine Anwendungen, außer ein von Casey Schaufler modifizierter SSH-Daemon, in der Lage, auf diese Datei zuzugreifen, um Benutzern ein differenziertes Label zuzuweisen. Somit beschränkt sich das Einsatzgebiet von SMACK derzeit auf Remote-Logins über SSH.

Neben diesen Standardlabeln kennt SMACK auch einige Regeln die es von Hause aus mitbringt. Diese lauten:

- Jeder Zugriff von einem Subjekt mit dem Label »*« ist verboten.
- Jeder Zugriff (rx) von einem Subjekt mit dem Label »^« ist gestattet.
- Jeder Zugriff auf Objekte mit dem Label »*« ist erlaubt.
- Jeder Zugriff (rx) auf Objekte mit dem Label »_« ist gestattet.
- Jeder Zugriff von einem Subjekt auf ein Objekt mit identischem Label ist erlaubt.
- Alle Zugriffe aus der Datei »/etc/smack/accesses« sind erlaubt.
- Alle weiteren Zugriffe ohne explizite Regel sind verboten.

Eigene Regeln kann der Admin SMACK über die bereits erwähnte Datei »/etc/smack/accesses« hinzufügen. SMACK kennt dabei, anders als SELinux, lediglich vier unterschiedliche Berechtigungen: lesen (r-read), schreiben (w-write), ausführen (x-execute) und anhängen (a-append).

Um die Funktionsweise von SMACK zu demonstrieren, zeigen Listing 1 bis 5 ein kleines Beispiel. Hier legt der zuerst der Benutzer »foo« im Ordner »/data« eine Datei »orders« an (Listing 1). Diese Datei erbt den Security-Kontext »foo« vom gleichnamigen Benutzer. Verantwortlich für diese Labelzuweisung ist der Eintrag in der Datei »/etc/smack/user« (Listing 2). Meldet sich der Benutzer »foo« über den modifizierten SSH-Server an, so bekommt er nicht das Standard-Label »floor - (_)« zugewiesen, sondern das Label »foo«.

Ein Benutzer mit einem anderen Label, hier »tscherf«, darf auf diese Datei nicht zugreifen, da hierfür keine ausdrückliche Regel in der Datei »/etc/smack/accesses« existiert (Listing 3). Möchte man den Zugriff gestatten, so ist hierfür eine entsprechende Regel notwendig (Listing 4). Fügt der Admin eine solche Regel der Datei »/etc/smack/accesses« hinzu und liest diesen Regelsatz anschließend mittels »smackload < /etc/smack/accesses« ein, so klappt der Zugriff auf die gewünschte Datei (Listing 5). SMACK leitet die aktiven Regeln über die Datei »/smack/load« an den Kernel weiter. Es ist auch möglich, diese Datei direkt zu beschreiben.

Damit der Zugriff funktioniert, ist natürlich sicherzustellen, dass der Benutzer beim Login das Label »tscherf« zugewiesen bekommt. Das geht, wie bereits erwähnt, derzeit nur mit dem modifizierten SSH-Daemon. Dieser ist, zusammen mit anderen SMACK-Tools, auf der Projektwebsite [2] erhältlich.

Ein Zugriff auf Dateien im Home-Verzeichnis des Benutzers »foo«, das nach einem Labeling ja auch das Label »foo« enthält, funktioniert übrigens nicht, da hierfür die notwendigen Regeln auf Basis der Discretionary-Access-Control (DAC) fehlen. Genau wie bei SE Linux ist es so, dass die MAC-Regeln erst dann zum Tragen kommen, wenn ein Zugriff aufgrund der DAC-Regeln gestattet ist. Die einzige Ausnahme stellen hier privilegierte Prozesse oder Benutzer und entsprechend vorhandene Capabilties dar. Beispielsweise verfügt der Superuser Root über die Linux-Capability »cap_mac_override«. Hiermit könnte dieser auch auf Objekte zugreifen, auf die er anhand der bestehenden MAC-Regeln eigentlich gar keinen Zugriff hat. Seit dem Linux-Kernel 2.6.28 existiert für dieses Problem jedoch eine recht elegante Lösung. Über das SMACK-Dateisystem »/smack« kann der Administrator in der Datei »onlycap« ein Star-Label (*) hinterlegen. Somit ist auch der Zugrif durch privilegierte Prozesse oder Benutzern mit bestimmten Linux-Capabilities nur auf Grundlage der bestehenden MAC-Regeln möglich.

SMACK-Konfiguration

Der Einsatz von SMACK setzt die Kernel-Quellen ab Version 2.6.25 mit folgenden Einstellungen voraus:

```
CONFIG_NETLABEL=y
CONFIG_EXT3_FS_XATTR=y
CONFIG_EXT3_FS_SECURITY=y
CONFIG_SECURITY_SMACK=y
```

Hierbei ist zu beachten, dass nicht mehrere Linux-Security-Module parallel lau-

Listing 1: »orders«

```
01 $ cat /proc/self/attr/current; echo
02 foo
03 $ ls -ldM /data/orders
04 foo drwxrwxrwx 2 root root 4096 2009-05-04 23:16 /
   data/
05 $ touch /data/orders
06 $ ls -lM /data/orders
07 foo -rw-rw-r-- 1 foo foo 0 2009-05-04 23:20 /data/
   orders
```

Listing 2: »/etc/smack/user«

```
01 # Benutzername Label-Zuweisung
02 tscherf tscherf
03 foo foo
```

Listing 3: Zugriff verboten

```
01 $ cat /proc/self/attr/current; echo
02 tscherf
03 $ ls -lM /data/orders
04 ls: cannot access /data/orders: Permission denied
```

Listing 4: »/etc/smack/accesses«

```
01 # Subjektlabel Objektlabel Berechtigungen
02 tscherf foo rx
```

Listing 5: Zugriff erlaubt

```
01 $ cat /proc/self/attr/current; echo
02 tscherf
03 $ ls -lM /data/orders
04 foo -rw-rw-r-- 1 foo foo 0 2009-05-04 23:20 /data/
   orders
```

fen können. Wer sich also für SMACK als MAC-System entscheidet, muss andere aktive LSMs (SELinux, AppArmor, TOMOYO) zuerst deaktivieren. Nach dem Kompilieren des so konfigurierten Kernels ist das Pseudo-Dateisystem »smackfs« der Datei »/etc/fstab« hinzuzufügen:

```
echo "smackfs  /smack  smackfs smackfsdef=?
*  0 0" >> /etc/fstab
```

Hiermit bekommen alle Objekte die auf Dateisystemen liegen, welche keine erweiterten Attribute unterstützen, das Label »star - (*)« zugewiesen. Das ist besonders beim Einsatz von Wechselmedien interessant. Auf der Projektwebseite findet sich unter [2] ein Archiv mit einigen Smack-Tools sowie dem bereits angesprochenen SSH-Server. Auch eine modifizierte Version des »ls«-Kommandos ist im Archiv enthalten. Hiermit kann der Anwender sich, wie in den Listings oben zu sehen, das SMACK-Label der einzelnen Objekte auflisten lassen. Zuständig hierfür ist die neue Option »-M«.

Die Datei »/proc/ < pid > /attr/current« zeigt das SMACK-Label von Prozessen an. Benutzer fragen ihr Label über »/proc/self/attr/current« ab. Daneben enthält das Archiv ein Init-Skript, das die entsprechenden SMACK-Regeln beim Systemstart aktiviert. Sind die Tools und Konfigurationsdateien alle an die richtige Stelle kopiert (ein Blick in das init-Skript verrät wo es die einzelnen Programme erwartet), kann man die einzelnen Dateien auf dem Dateisysteme mit dem gewünschten Label versehen. In der Standard-Einstellungen verwenden alle Objekte und Subjekte das Standard-Label »floor - (_)«, deswegen ist ein Zugriff grundsätzlich auch auf alle Objekte möglich – siehe Builtin-Regeln. Erst das manuelle Zuweisen eigener Label schränkt den Zugriff auf die jeweiligen Objekte ein. Entsprechende Label an Datei- und Ordner-Objekten setzt das Tool »attr«:

```
attr -S -s SMACK64 -V "foo" /data/orders
```

Über die Standardregeln ist ein Zugriff auf die Datei mit dem Label »foo« jetzt nur von Benutzern oder Prozessen möglich, die entweder über das gleiche Label (»foo«) oder ein Label »hat -(^)« verfügen. Findet ein Zugriff von einem Benutzer mit einem anderen Label statt, so überprüft SMACK ob dieser Benutzer über eine explizite Regeln in der Datei »/etc/smack/accesses« verfügt. Ist dies nicht der Fall, ist der Zugriff verboten (Listing 3). Leider ist der aktuelle Linux-Kernel 2.6.29 noch nicht an das Auditing-Subsystem angeschlossen, sodass sich ein nicht erlaubter Zugriff nicht im Log auftaucht. Dieses Problem wird aber in Kürze gelöst sein, da der aktuelle Entwicklerzweig des Kernels, linux-next, bereits über eine solche Erweiterung verfügt [3]. Es bleibt also zu hoffen das Linus Torvalds diese Patches bald in den offiziellen Kernel aufnimmt.

Sicherheit auch für IP-Pakete

SMACK unterstützt ebenfalls den CIPSO-2.2-IETF-Draft aus dem Jahr 1993. Hiernach kann man Prozesse die auf verschiedenen Systemen in einem Netzwerk laufen, zu einer Gruppe, der so genannten Domain of Interpretion (DOI), zusammenfassen. Jedes IP-Paket, das ein solcher Prozess versendet – der Sender ist hierbei das Subjekt – bekommt ein Tag im IP-Header zugewiesen. In den Zugriffsregeln kann der Administrator dann über passende Tags bestimmen, welcher Prozess mit welchem anderen Prozess – der empfangene Prozess ist hier das Objekt – auf einem entfernten System kommunizieren darf. Unter SMACK kann man die DOI und die Tags über die beiden Dateien »/etc/smack/doi« und »/etc/smack/cipso« setzen. Ist keine DOI angegeben, so verwendet SMACK standardmäßig die Gruppe 3. Bei der Angabe der Tags über die Datei »/etc/smack/cipso« gibt man zuerst einen Sensitivitätslevel gefolgt von einer oder mehreren Kategorien an.

Empfängt SMACK Pakete von Systemen, die keine CIPSO-Label benutzen, so verwendet SMACK hierfür das Default-Label »floor - (_)« aus der Datei »/smack/ambient«. Somit ist der Zugriff auf diese Pakete grundsätzlich für alle Prozesse möglich. Möchte man CIPSO nur das eigene Subnetz verwenden und zur Kommunikation mit der Außenwelt keine CIPSO-Label benutzen, so kann man hierfür das Default-Label »Internet - (@)« verwenden. Die folgenden Zeilen richten eine CIPSO-Konfiguration für das Subnetz »192.168.0.0/24« und Localhost ein, fügen aber den IP-Paketen für andere Netze keine CIPSO-Label hinzu:

```
echo 127.0.0.1       -CIPSO > /smack/netlabel
echo 192.168.0.0/16 -CIPSO >> /smack/netlabel
echo 0.0.0.0/0       @      >> /smack/netlabel
```

Im Vergleich zu SE Linux [4], das ebenfalls CIPSO unterstützt, gestaltet sich die Konfiguration unter SMACK um einiges einfacher. Abschließend sei noch angemerkt, dass CIPSO wohl nur für wenige Implementierungen interessant ist. Wer jedoch Linux-Systeme mit anderen Trusted-Unix-Varianten, wie beispielsweise Trusted Solaris kombinieren möchte, um einen leichten und verständlichen Einstieg in die CIPSO-Welt zu bekommen, der ist für die doch recht simplen Konfigurationsmöglichkeiten unter SMACK vermutlich dankbar.

Fazit

SMACK ist ein leicht zu konfigurierendes MAC-System für Linux, das aber nicht den Funktionsumfang von SE Linux bietet. Somit ist SMACK interessant für Umgebungen, in denen es lediglich um eine Trennung zwischen verschiedenen Benutzern geht, für den unternehmenskritischen Einsatz reicht es aber (noch) nicht aus. Dies liegt zum einen an den wenigen SMACK-fähigen Anwendungen, zum anderen am fehlenden Logging. Hier ist jedoch baldige Besserung zu erwarten, da der dafür notwendige Code bereits im Repository der kommenden Linux-Kernel-Version liegt. Bleibt zu hoffen das auch andere Entwickler auf den SMACK-Zug aufspringen, dann wird es vielleicht bald eine ernsthafte Konkurrenz zu SE Linux darstellen. *(ofr)* ■

Infos

[1] Smack-Projektseite: [http://www.schaufler-ca.com]

[2] Smack-Utilities: [http://www.schaufler-ca.com/data/080616/smack-util-0.1-x86.tar]

[3] Auditing für SMACK, [git://git.kernel.org/pub/scm/linux/kernel/git/jmorris/security-testing-2.6#next]

[4] SE Linux mit GUI, Thorsten Scherf, Linux-Magazin-Sonderheft 01/2009, S. 102

[5] Tomoyo, Name-based MAC extension for the Linux kernel: [http://tomoyo.sourceforge.jp]