Sicherheitsmechanismen des Betriebssystems

Andre Ufer, *RZRS,* Oliver Burger, *IMR*

Abstract—The abstract goes here.

Index Terms—Computer Society, IEEEtran, journal, LaTeX, paper, template.

1 LINUX

1.1 Grundlegende Sicherheitsmechanismen

ALS der grundlegende Sicherheitsmechanismus in GNU/Linux-Systemen ist die strikte Trennung zwischen dem Administrationsbenutzer *root* und den normalen Benutzern zu sehen.

Die meisten GNU/Linux-Distributionen gehen hier sogar so weit, dass der graphische Login als *root* unterbunden wird.
Auch die meisten beim Bootvorgang gestarteten Dienste (wie ein Mail- oder Web-Server etc.) werden zwar von *root* gestartet, jedoch sofort an einen dedizierten Benutzer für den jeweiligen Dienst abgegeben. Diese Benutzer haben - in den meisten Fällen - auch keine Login-Shell, so dass Angriffe in diese Richtung fehlschlagen.
Aufgrund der oben angesprochenen Trennung ist es notwendig eine Möglichkeit für Benutzer zu schaffen, auf Systemressourcen - wie Laufwerke und Peripherie-Hardware - zugreifen zu können.

1.2 Benutzerverwaltung

DIE lokale, also nicht auf ldap und ähnliche Systeme aufsetzende Benutzerverwaltung wird über drei Konfigurationsdateien implementiert. Dies sind die */etc/passwd*, die */etc/group* und */etc/shadow*. Die genaue Syntax dieser Dateien kann den jeweiligen Man-Pages entnommen werden. [4] [5] [6]

Auch wenn der Name etwas anderes implementiert, enthält die *passwd*-Datei heute keine Passwörter mehr, da diese über den *shadow*-Mechanismus in die entsprechende Datei ausgelagert werden. Auch dort befinden sich keine Klartextpasswörter mehr, sondern nur Passwort-Hashes, die mit Hilfe der *crypt*-Funktion erzeugt werden. [7] Setzte *crypt* früher auf ein DES-basiertes System auf, das einen Key auf Grundlage des Passwortes nutzte um damit einen Nullvektor zu verschlüsseln, der dann als Hashwert gespeichert wurde, kommen heute üblicherweise md5- sha1 oder in einigen Fällen blowfish-basierte Verfahren zum Zuge. Die blowfish-Verfahren sind nicht in der normalen *crypt*-Funktion implementiert, wurden aber von einigen Linux-Distributionen hineingepatcht. Um eine Kompatibilität mit Passwörtern, die durch andere Verfahren erzeugt wurden, zu erreichen, werden die gespeicherten Hashwerte mit einem Präfix versehen, anhand dessen *crypt* erkennt, welches Verfahren verwendet wurde.

1.3 Dateirechte

DAS allgemeine Berechtigungskonzept für den Zugriff auf Dateien - und somit auch Geräte, die als Dateien im */dev/*-Dateisystem umgesetzt sind - ist sehr simpel gehalten. Das Dateisystem fügt jeder Datei ein Tupel aus drei Bytes an, das die grundlegende Berechtigung enthält. [8]

Hierbei steht das erste Byte für die Rechte des Eigentümers, das zweite Byte für die Rechte der Eigentümer-Gruppe und das dritte Byte für alle anderen im System befindlichen Benutzer.
Ein Byte setzt sich hierbei aus den Rechten *Lesen*, *Schreiben* und *Ausführen* zusammen. Diese drei rechte werden nun als Binärzahl mit der gerade angegebenen Reihenfolge abgebildet.

Das zugrundeliegende Dateisystem muss diese Art der Rechteverwaltung unterstützen. Momentan wird von den meisten Distributionen das *ext4*-Dateisystem als Standard verwendet. Früher waren hier auch die Vorgänger *ext2* - das Urgestein im Linuxbereich - und *ext3* üblich sowie *reiserfs*, *xfs* und *jfs*.
Linux ist zwar in der Lage, auch auf vielen anderen Dateisystemen zu schreiben und von ihnen zu lesen, jedoch funktioniert dort die Rechteverwaltung nicht. Da viele Systemdienste bei gewissen Konfigurationsdateien verlangen, dass nicht zu hohe Rechte vergeben sind, würden sie nicht funktionieren, wenn diese Dateien auf einem *fat-* oder *ntfs*-Dateisystem liegen. Dies betrifft zum Beispiel die Benutzerkonfiguration von *ssh* aber auch Grundlegendes wie die *shadow*-Datei.

- *A. Ufer ist angestellt beim Rechenzentrum Region Stuttgart*
- *O. Burger ist angestellt bei der in medias res GmbH.*

Erstellt im Rahmen der Vorlesung Datensicherheit and der DHBW Stuttgart - Campus Horb.

In manchen Fällen ist es notwendig, etwas feiner granulierte Rechte vergeben zu können, zum Beispiel um ganz bestimmten Personen Lese- und Schreibrechte einzuräumen. Hierzu kann der *Extended Access Control Layer* verwendet werden.

1.4 Das alte Zugriffskonzept

In früheren Linux-Systemen wurde der Zugriff normaler Benutzer auf die Hardware - wie zum Beispiel DC-/DVD-Brenner und Netzwerkgeräte durch das Gruppenkonzept umgesetzt.
Eine Einführung hierfür kann man in [1] einsehen. Da diese Technik seit einigen Jahren nicht mehr in Benutzung ist, ist hierfür wenig wirkliche Dokumentation zu finden.

In vielen GNU/Linux-Distributionen können noch heute Gruppen wie z.B. *dialout* gefunden werden, da sie Teil der LSB[3] sind.
Die Vorgehensweise war hier, dass alle Benutzer, die eine Verbindung nach außen aufbauen können mussten Mitglieder dieser Gruppe waren.

1.5 Policykit

Wie in [2] beschrieben wird heutzutage ein anderer Weg hierfür gewählt. Policykit wird über den Systemnachrichtenbus (DBUS) angesprochen und erteilt auf einer Benutzersession-Basis Freigaben zur Benutzung einer bestimmten Systemressource.
Hierfür stellt Policykit eine API bereit, die von anderen Programmen genutzt werden kann, um diesen Effekt zu erreichen.
Es ist dem Systemadministrator durch diesen Mechanismus möglich, fein granulierte Zugriffsrechte auf bestimmte Systemressourcen an einzelne Benutzer zu vergeben.

Die Implementation von Policykit besteht hierbei aus zwei Teilen, der *Authority*, implementiert als Dienst im Systemnachrichtenbus und dem *Authentication Agent*.
Die Vorgehensweise ist hierbei folgende:
Ein nicht-privilegiertes Programm der Benutzer-Session, bei Policykit als *Client* bezeichnet, bittet ein privilegiertes Programm auf Systemebene, bei Policykit als *Mechanism* bezeichnet um Zugriffsrechte auf eine gewisse Ressource. Der *Mechanism* fragt über den Systemnachrichtenbus die *Authority* an, die dann vom jeweiligen *Authentication Agent* die Berechtigung des Clients abfragt und diese an den *Mechanism* zurückgibt.
Es ist hiermit - im Gegensatz zum früheren System - auch möglich einmalige Berechtigungen zu vergeben, einen Benutzer also nur zum jeweiligen Zeitpunkt in die Lage zu versetzen, auf Systemressourcen zuzugreifen.
Dies wird unter anderem genutzt, wenn ein Benutzer ein Systemadministrationswerkzeug der graphischen Oberfläche zum Beispiel im KDE-Kontrollzentrum aufruft.

Zur Zeit befindet sich dieser gesamte Bereich in einem Umbau, da der *Hardware Abstraction Layer* (kurz *HAL*) entfällt. Seine Funktionen werden zukünftig direkt über *udev* implementiert. Dies hat eine vollständige Reimplementierung von *policykit* zur Folge. Wie allgemein üblich, befindet sich die Dokumentation hierzu aber leider nicht auf dem aktuellen Stand beziehungsweise ist noch nicht vorhanden.

Wie die genaue Funktionsweise in Zukunft sein wird, kann daher in dieser Arbeit noch nicht behandelt werden.

2 Windows

Windows ist das auf Desktopsystemen am vebreitetste Betriebssystem mit einem Marktanteil von ca. 90%. Das macht es zur attraktivsten Plattform für Schadsoftware und Angriffe. Dieses Problem ist auch dem Hersteller Microsoft bekannt, weshalb er von Version zu Version mehr Sicherheitsmechanismen Entwickelt und einbaut, die es Angreifern jeglicher Art schwerer machen soll, Zugriff auf das angegriffene System zu erlangen oder Schadcode auszuführen.
Vor allem mit Windows XP und den darauf folgenden Versionen nahm die Anzahl dieser Sicherheitsmechanismen deutlich zu. Diese Mechanismen setzen an verschiedenen Stellen an. Im Rahmen dieser Seminararbeit sollen die Sicherheitsfunktionen von NTFS sowie die Mandatory Integrity Control genauer beschrieben werden.

2.1 NTFS

Zusammen mit Windows NT lieferte Microsoft das Dateisystem *Microsoft NTFS* ("New Technology File System") aus, welches von allen darauffolgenden Windows-Versionen (ausgenommen Windows 9x Produkte) unterstützt wird und sich als Standard-Dateisystem unter Windows etabliert hat. Microsoft hat das Dateisystem seitdem in mehreren Versionen weiterentwickelt und dabei vor allem Funktionen für Verschlüsselung und Dateimanagement hinzugefügt.[12]

NTFS ist das erste Dateisystem für Windows, welches ein Metadaten-Journaling verwendet. Damit wird sichergestellt, dass sich jederzeit ein konsistenter Zustand des Dateisystems (re-)konstruieren lässt. Daten selbst lassen sich jedoch nicht wiederherstellen, diese Informationen werden von einem Meta-Journaling nicht gespeichert.

Mit NTFS 3.x erhält das Dateisystem erstmals eine Mechnismus zur Verschlüsselung von Dateien und Ordnern. Der EFS (Encrypting File System) genannte Mechanismus verschlüsselt unkomprimierte Daten und wird von allen Windows-Versionen ab Windows XP

unterstützt[14]. Die Home-Versionen sind davon jedoch ausgenommen. Der Benutzer selbst merkt von der Ver- und Entschlüsselung seiner Daten nichts, da dies on-the-fly beim Zugriff geschieht. Damit dient der Schutz der Dateien nur einem Zugriff auf die Festplatte außerhalb des Windows-Systems. Schadsoftware und andere Prozesse und Anwendungen, die mit Benutzerrechten ausgeführt werden, haben weiterhin Zugriff auf die verschlüsselten Daten.
Die Verschlüsselung war bis Windows 2000 ein erweiterter DES (DESX), der die Schlüssellänger erhöhte, ohne die Arbeitsweise des DES-Algorithmus zu ändern. DESX wurde ab Windows XP SP1 von einem 128-bit starken AES Verschlüsselungs-Algorithmus abgelöst.[14]

NTFS liefert als erstes Microsoft-Dateisystem eine Berechtigungsverwaltung für Dateien und Verzeichnisse. Diese basiert auf Access Control Lists (ACL) und dem sogenannten SecurityDescriptor (SD). Jedes Element im Dateisystem (Dateien, Ordner, ...) erhält dabei einen eigenen SD. Der Aufbau des SD ist im Folgenden schematisch dargestellt:

Name	Funktion
Hash(SD)	Hashwert des SDs zur Identifikation
Owner_SID	Identifikation des Besitzers
DACL	Discretionary ACL
SACL	System ACL

Jedes Element im Dateisystem bekommt einen SD zugewiesen. Bei vielen Elementen sind die Informationen im jedoch SD identisch, sodass dafür derselbe SD verwendet werden kann. Um das zu ermöglichen, wird ein Hashwert für die Informationen im SD berechnet und ebenfalls im SD abgelegt. Wird nun für zwei Elemente derselbe Hash berechnet, verweisen sie auch auf denselben SD.
Die wichtigsten Bestandteile des SD sind die DACL und die SACL. Die DACL enthält die Zugriffsrechte von berechtigten Benutzern und Gruppen auf ein bestimmtes Element. Diese Liste wird beim Zugriff auf ein Element abgefragt.
Die SACL enthält u.a. Informationen darüber, welche Zugriffe protokolliert werden sollen. Außerdem wird der Integritätslevel von Objekten in der SACL gespeichert.[13]

2.2 Mandatory Integrity Control

Die Mandatory Integrity Control ist ein mit Windows Vista eingeführter Sicherheitsmechanismus. Er stellt sicher, dass die dem Betriebssystem zur Verfuegung stehenden Informationen integer/vertrauenswürdig sind und greift bei Dateizugriff noch vor der DACL-Prüfung. Dafür wurde eine Klassifizierung der Subjekte eingeführt, die über sogenannte Integritätslevel realisiert wurde[9]. Als Subjekte kommen Benutzer, Prozesse und Anwendungen infrage.

Die MIC entspricht einem vereinfachten Biba-Modell. Während das Biba-Modell zur Sicherung der Integrität sowohl ein ”No-Write-up” als auch ein ”No-Read-Down” vorsieht, implementiert die MIC nur das ”No-Write-up”-Prinzip. Damit soll sichergestellt werden, dass Systemdateien nicht von weniger integeren Quellen (Standard-Benutzer, Administratoren) verändert werden können.

Folgende Integritätslevel sind der MIC bekannt:
- untrusted - wird i.d.R. nicht verwendet
- low - temporäre Internetdateien, IE im protected mode
- medium - IL der Standardbenutzer
- high - IL der Administratoren
- system - Ausschließlich für system-/kernelnahe Prozesse und Dienste

Abhängig vom Betrachtungswinkel sind Elemente des Dateisystems entweder Subjekte oder Objekte. Das heißt, dass jedes Subjekt zum Objekt werden kann, sobald ein anderes Element darauf zugreift. Der Subjekt-IL wird im Anmelde-Token gespeichert, indem dieser Token um einen weiteren Wert erweitert wird. Als Beispiel soll der Token eines normalen Windows-Benutzers dienen: `S-1-16-8192`. Dabei stellt die letzte Zahl den Integritätslevel dar. `8192` steht also für den IL “medium”.
Der IL von Objekten wird in den SACL der jeweiligen Security Descriptors gespeichert.

Ein Beispiel für die Umsetzung der MIC stellt der Protected Mode des Internet Explorers (IE) dar. Im Protected Mode wird der IE-Prozess mit dem IL ”low” ausgeführt und hat somit keinen Zugriff auf Dateien, Prozesse und Registrierungssschlüssel mit einem höheren IL. Damit soll z.B. verhindert werden, dass ein kompromittierter IE-Prozess beispielsweise einen Keylogger in den Windows-Autostart hinzufügt. Aktiviert ist der Protected Mode für die Zonen ”Internet”, ”Intranet” und ”Eingeschränkte Sites”.[10][11]

2.3 Weitere Sicherheitsmechanismen

Neben den bereits genannten Sicherheismechanismen liefert Windows noch weitere Schutzfunktionen mit verschiedenen Anwendungsbereichen. Die bekanntesten und wichtigsten davon sind im Folgenden aufgeführt.
- User Account Control (UAC): Regelt die Anforderung von erhöhten Rechten bei Zugriff auf systemrelevante Daten (z.B. Treiber- und Programminstallation)
- Data Execution Prevention (DEP): gezielte Ausführungsverhinderung von Daten, Hardware-Kompatibilität wird vorausgesetzt.
- Adress Space Layout Randomization (ASLR): soll gezielte Pufferüberläufe verhindern, indem Programmen zufällige Adressbereiche übergben werden.

REFERENCES

[1] http://www-lehre.inf.uos.de/~ainf/2006/dokumentation/SelfLinux-0.12.1/html/nutzer_unter_linux06.html, 10.11.2011.
[2] http://hal.freedesktop.org/docs/polkit/, 10.11.2011.
[3] http://www.linuxfoundation.org/collaborate/workgroups/lsb, 12.11.2011.
[4] http://linux.die.net/man/5/passwd, 03.12.2011.
[5] http://linux.die.net/man/5/group, 03.12.2011.
[6] http://linux.die.net/man/5/shadow, 03.12.2011.
[7] http://www.kernel.org/doc/man-pages/online/pages/man3/crypt.3.html, 03.12.2011.
[8] http://www.comptechdoc.org/os/linux/usersguide/linux_ugfilesp.html, 03.12.2011.
[9] http://msdn.microsoft.com/en-us/library/bb625964.aspx
[10] http://msdn.microsoft.com/en-us/library/bb250462\%28v=vs.85\%29.aspx
[11] http://windows.microsoft.com/de-DE/windows-vista/What-does-Internet-Explorer-protected-mode-do
[12] http://technet.microsoft.com/de-de/library/bb457112\%28en-us\%29.aspx
[13] http://www.ntfs.com
[14] http://www.wintotal.de/artikel/artikel-2006/49.html