Analyse von Zugriffssteuerungssystemen

Peter Amthor · Anja Fischer · Winfried E. Kühnhauser

Technische Universität Ilmenau
[a.fischer | winfried.kuehnhauser]@tu-ilmenau.de

Zusammenfassung

Zugriffsschutzsysteme spielen beim Schutz von Informationen in IT-Systemen eine kritische Rolle; sie verhindern auch dann noch den Zugriff auf Informationen, wenn virtuelle private Netzwerke, Firewalls oder Intrusion-Detection-Systeme nicht mehr greifen können.
Entgegen ersten Anscheins ist die korrekte Konfiguration von Zugriffsschutzsystemen sehr subtil: während direkte Zugriffe von Benutzern auf Dokumente sehr einfach zu steuern sind, ist es dagegen sehr schwierig auszuschließen, dass durch Umwege über Zwischenobjekte verdeckte Informationsflussmöglichkeiten bestehen; und dies sind dann exakt diejenigen Wege, über die Insider Zugriff auf Informationen erhalten, zu denen sie gemäß der intendierten Sicherheitspolitik keinen Zugang haben dürfen.
Dieses Papier stellt eine Methode zum Auffinden von Informationsflusspotenzialen vor, die auf der Transformation einer Zugriffsmatrix in einen Informationsflussgraphen und seiner Reduktion auf Informationsfluss-Äquivalenzklassen beruht. Hierauf aufbauend wurde in ein Werkzeug entwickelt, welches die Analyse von Zugriffssteuerungssystemen hinsichtlich verdeckter Informationsflusspotenziale weitgehend automatisiert. Die Ergebnisse der Anwendung von Methode und Werkzeug auf einen mittelgroßen Fileserver mit einer diskreten, Unix-artigen Zugriffssteuerungssemantik weisen einige Überraschungen auf.

Schlüsselworte: Zugriffssteuerung, Zugriffsmatrix, Informationsfluss, Informationsflussgraph, obligatorische und diskrete Zugriffssteuerung

1 Einführung

Studien der Corporate Trust Business Risk & Crisis Management GmbH und der Universität Lüneburg [Gmb07, KM04] beziffern die durch Wirtschaftsspionage verursachten finanziellen Schäden für die deutsche Wirtschaft auf Beträge zwischen 2,8 und 7 Milliarden Euro pro Jahr; die Tendenz ist steigend.

Nach einer weiteren Studie des Verizon Business Risk Teams sind für mehr als 40% des Schadensvolumens aller sicherheitsrelevanten Vorfälle mit IT-Systemen Insider verantwortlich [Tea08]; Menschen also, die bereits durch ihre berufliche Stellung legalen Zugang zu den IT-Systemen ihres Unternehmens besitzen. Techniken wie Firewall-Architekturen, virtuelle private Netzwerke oder Intrusion-Detection-Systeme sind gegenüber Insidern wirkungslos; zwischen ihnen und den sicherheitskritischen Informationen auf den IT-Systemen liegt lediglich eine letzte Hürde: die Zugriffsschutzsysteme der IT-Systeme.

Zugriffsschutzsysteme heutiger IT-Systeme basieren auf Zugriffssteuerungs- oder Capabilitylisten, die konkrete Formen der Realisierung Lampson'scher Zugriffsmatrizen [Lam74] darstellen. So finden sich beispielsweise in Systemen aus der Unix-Familie die bekannten 3er-Tupel aus (*read/write/execute*)-Attributen, die mit jedem Dokument assoziiert sind und die Rechte

beschreiben, die individuelle Benutzer oder Benutzergruppen an diesem Dokument besitzen. Die Menge der Rechtetupel sämtlicher Dokumente eines IT-Systems ist ein vollständiges und präzises Bild der Zugriffsrechtesituation in einem IT-System und bildet in der Terminologie der Sicherheitsmodelle dessen Zugriffsmatrix.

Eine solche Zugriffsmatrix ist äußerst aufschlussreich. Zum einen gibt sie – trivialerweise – Antworten auf die Frage, in welcher Weise ein gegebenes Subjekt (ein Benutzer oder ein Mitglieder einer Benutzergruppe) unmittelbar Zugriff auf ein gegebenes Objekt besitzt; eine entsprechende Prüfung erfolgt bei einem Zugriff bereits durch die elementaren Zugriffsschutzmechanismen der Betriebssysteme. Zum anderen ermöglicht eine Zugriffsmatrix weit darüber hinaus auch Analysen, ob etwa über Umwege über zwischengeschaltete Subjekte oder Objekte indirekte Informationsflüsse möglich sind, mittels derer ein Benutzer letztendlich doch Informationen aus einen Dokument erhalten kann, obwohl ihm ein direktes Lesen untersagt ist. Derart verdeckte Informationsflusspotenziale sind dann exakt diejenigen Wege, über die Insider Zugriffe zu Informationen in IT-Systemen erhalten, zu denen sie gemäß der intendierten Sicherheitspolitik tatsächlich keinen Zugang haben dürfen; dabei müssen sie nicht einmal Techniken einsetzen, Zugriffsschutzsysteme außer Kraft zu setzen – sie müssen lediglich die verdeckten Informationsflusspotenziale aufdecken und nutzen.

Analysen realer Systeme mit dem Ziel des Findens verdeckter Informationsflusspotenziale zeigen hier aufschlussreiche Ergebnisse; einige dieser Ergebnisse über unsere eigenen, durchaus sicherheitsbewusst administrierten Server werden wir im Laufe dieses Papiers vorstellen. Während nahezu immer die *direkten* Lese/Schreibrechte zwischen Benutzern und Dokumenten korrekt den Regeln der intendierten Sicherheitspolitik genügen, wurden zahlreiche ungewollte Informationsflusspotenziale aufgedeckt, deren Ursache in der Transitivitätseigenschaft von Informationsflüssen und einem wenig transparenten Benutzergruppenkonzept zu suchen ist.

Verdeckte Informationsflusspotenziale sind ihrer Natur gemäß nicht auf den ersten Blick zu erkennen. Die Ursache hierfür ist in der Kombination zweier Fakten zu suchen. Zum einen sind bereits Zugriffsmatrizen kleiner IT-Systeme sehr groß; Zugriffsmatrizen etwa von Laptop-Systemen besitzen typischerweise bereits mehrere Millionen Zellen, und Serversysteme überschreiten diese Zahl um Größenordnungen. Ein derart großer Suchraum für verdeckte Informationsflusspotenziale ist manuell nicht annähernd beherrschbar. Automatisierte Suchverfahren, die nur für ein einziges Subjekt dessen sämtliche *direkten* Rechte innerhalb eines IT-Systems aufsammeln, haben bereits eine linear mit der Objektanzahl wachsende Laufzeitkomlexität; eine Ausdehnung dieser Analyse auf sämtliche Subjekte eines IT-Systems liegt bereits in $O(Subjektanzahl \times Objektanzahl)$. Von entsprechend höherer Komplexität ist die Suche nach Informationsflusspotenzialen über alle möglichen Zwischenstationen. Ursache für die Schwierigkeiten des Auffindens verdeckter Informationsflusspotenziale ist somit die Kombination aus der Laufzeitkomplexität der Suchalgorithmen und den großen Suchräumen.

Dieses Papier leistet die folgenden Beiträge: Zum Umgang mit der Suchkomplexität wird eine Methode vorgestellt, die auf der Transformation einer Zugriffsmatrix in einen Informationsflussgraphen und seiner Reduktion auf Informationsfluss-Äquivalenzklassen beruht. Es wird ein Werkzeug vorgestellt, welches auf dieser Methode beruht und die Analyse von Unix-Zugriffssteuerungssystemen weitgehend automatisiert. Die Ergebnisse der Anwendung von Methode und Werkzeug auf einen mittelgroßen Server (ca. 6,5 Millionen Matrixzellen) werden vorgestellt; den Namen dieses Servers nennt dieses Papier allerdings nicht.

2 Die Methode

Zugriffssteuerungssysteme in IT-Systemen der heutigen Generation basieren typischerweise auf Zugriffssteuerungslisten (ZSLs) oder Capabilitylisten, die Regeln der Form „Benutzer a erhält ein Leserecht auf Datei b" ermöglichen. Fehlt dieses Recht in der ZSL von b, so kann a die Datei b zwar nicht direkt lesen; allerdings ist damit bei weitem noch nicht sichergestellt, dass Informationen in b nicht auch durch einen Umweg über andere Objekte doch nach a gelangen können. Die ursprüngliche Intention der Wahrung der Vertraulichkeit der Informationen in b gegenüber a kann also allein durch die genannte Zugriffsregel nicht erreicht werden. Die Aussage, dass „Informationen in b nicht nach a fließen dürfen" ist somit wesentlich tiefgründiger und dem Problem der Vertraulichkeitswahrung angemessener; eine geeignete Ausdrucksform hierfür ist eine ZSL allerdings nicht.

Sicherheitspolitiken, die Aussagen über zulässige Informationsflüsse treffen, verwenden daher auch zu deren Darstellung Informationsflussgraphen; die Frage, ob ein Informationsfluss von b nach a möglich ist, ist damit äquivalent zur Frage nach der Existenz eines Weges von b nach a.

Dieses Kapitel beschreibt die Ziele der Analyse von Zugriffssteuerungssystemen in dieser Arbeit und stellt die methodischen Grundlagen der Analyse vor, die wesentlich auf einer bijektiven Transformation einer Zugriffsmatrix in einen Informationsflussgraphen beruhen.

2.1 Analyseziele

Sicherheitsanalysen von IT-Systemen untersuchen, inwieweit konkrete Zugriffssteuerungssysteme eine intendierte Sicherheitspolitik und somit unabdingbare Sicherheitseigenschaften korrekt durchsetzen. Von besonderem Interesse sind hierbei Fragen wie:

- Ist die Vertraulichkeit bestimmter Daten gewährleistet? Ist es beispielsweise möglich, dass ein bestimmtes Subjekt Informationen aus einem Verzeichnis wie beispielsweise „Gehaltsabrechnungen" erhält?
- Ist die Integrität von Daten in einem speziellen Verzeichnis gewahrt? Ist es beispielsweise möglich, dass Informationen von einem beliebigen Subjekt zu einem Systemverzeichnis fließen?
- Welche Informationen können zu einem Benutzer fließen?
- Wohin können Informationen eines Benutzers fließen?

Antworten auf Fragestellungen dieser Art lassen sich im Vergleich zu Zugriffsmatrizen aus Informationsflussgraphen (IGs) [Den76] sehr viel effizienter herleiten. Zur Analyse der Vertraulichkeitseigenschaften einer Information I wird der Informationsflussgraph auf Wege untersucht, die von Knoten wegführen, die I enthalten. Analog lassen sich Aussagen über Integritätseigenschaften von I aufzeigen durch Betrachtung der Wege, die zu Knoten mit I hinführen.

Neben einer Wegeanalyse kann auch die Suche nach informationellen Äquivalenzklassen interessante Ergebnisse liefern; solche Klassen stellen Knotenmengen dar, die das selbe Informationspotenzial besitzen.

Zum einen ist die Zugehörigkeit eines Knotens zu einer Äquivalenzklasse bereits eine Aussage an sich, aus der sich Vertraulichkeits- und Integritätsaussagen ableiten lassen; zum anderen lassen sich alle Knoten einer Klasse im Rahmen einer Informationsflussanalyse zu einem einzigen Knoten Knoten zusammenfassen, was in Analysen realer Systeme zu erheblichen Reduktionen der Knotenzahlen und damit der Laufzeit der Graphalgorithmen führt.

2.2 Zugriffssteuerungslisten und Zugriffsmatrix

Zugriffsrechte werden in derzeitigen IT-Systemen auf der Ebene von Betriebs- oder Dateisystemen durch Zugriffssteuerungslisten repräsentiert, die den Objekten des Systems (Dateien, Verzeichnisse etc.) assoziiert sind. ZSLs sind eine ökonomische Implementierung der Zugriffsmatrix (ZM), wobei die ZSLs den Spaltenvektoren der ZM entsprechen. Abbildung 1 verdeutlicht diesen Zusammenhang: die ZSL des Objekts *uebung.doc* bildet in der Zugriffsmatrix einen Spaltenvektor; die Matrixzelle $ZM(user1, uebung.doc) = \{r\}$ reflektiert die Tatsache, dass Nutzer *user1* dieses Dokument lesen darf.

Aus diesem Zusammenhang ergibt sich nun eine formale Äquivalenz von ZM und ZSLs durch $ZM = \{ZSL_o | o \in O\}$, O die Objektmenge des IT-Systems. Zur Vollständigkeit sei erwähnt, dass bei der Rekonstruktion der ZM heutiger IT-Systeme weitere Rechteeinschränkungen zu berücksichtigen sind, die sich aus der Einbettung eines Objekts in die Namenshierarchie ergeben: Objekte sind grundsätzlich nur dann für ein Subjekt sichtbar (eine prinzipielle Voraussetzung für einen Objektzugriff), wenn dieses Subjekt entsprechende Rechte entlang des gesamten Objektpfadnamens besitzt.

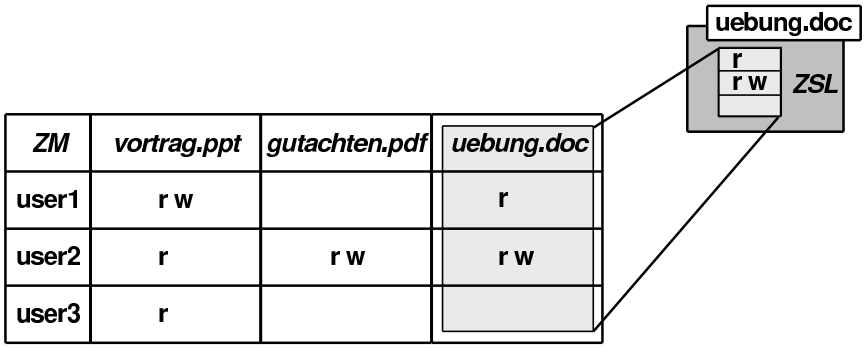

Abb. 1: Zugriffsmatrix und Zugriffssteuerungslisten

Die formale Äquivalenz zwischen der Menge sämtlicher ZSLs eines IT-Systems und der Zugriffsmatrix lässt sich nun nutzen, um aus den ZSLs die ZM des IT-Systems rückzugewinnen; diese stellt dann eine präzise Momentaufname sämtlicher in einem IT-System vorhandenen Zugriffsrechte dar und kann anschließend zur Ableitung eines Informationsflussgraphen genutzt werden.

2.3 Zugriffsmatrix und Informationsflussgraph

Informationsflussgraphen sind zunächst einmal lediglich gerichtete, nicht notwendigerweise zusammenhängende und auch nicht notwendigerweise kreisfreie Graphen. Zur Repräsentation von Informationsflüssen in einem IT-System mittels eines Informationsflussgraphen (IG) wird dessen Knotenmenge aus der Subjektmenge S und der Objektmenge O des Systems gebildet:

$$IG = (N, E), N = S \cup O.$$

Die Kantenmenge repräsentiert direkte Informationsflüsse zwischen den Subjekten und Objekten; eine Kante von einem Knoten a zu einem Knoten b existiert genau dann, wenn entweder b ein Subjekt ist und ein Leserecht auf das Objekt a besitzt oder wenn a ein Subjekt ist und ein Schreibrecht auf das Objekt b besitzt:

$$\forall a, b \in S \cup O : [a, b] \in E \Leftrightarrow (r \in ZM(b, a) \vee w \in ZM(a, b)).$$

ZM und IG sind somit isomorph, d.h. es lässt sich eine bijektive (eineindeutige) Abbildung zwischen ZM und IG angeben, so dass Analyseergebnisse auf der Basis des IG eindeutig auf Zugriffsrechte in der ZM rückführbar sind, und diese sind wiederum auf Grund der formalen Äquivalenz von ZM und ZSLs auf die individuellen ZSLs der einzelnen Objekte rückführbar.

Unter Ausnutzung dieser Zusammenhänge lässt sich die Zugriffsmatrix aus Abbildung 1 wie folgt als Informationsflussgraph darstellen (Abbildung 2).

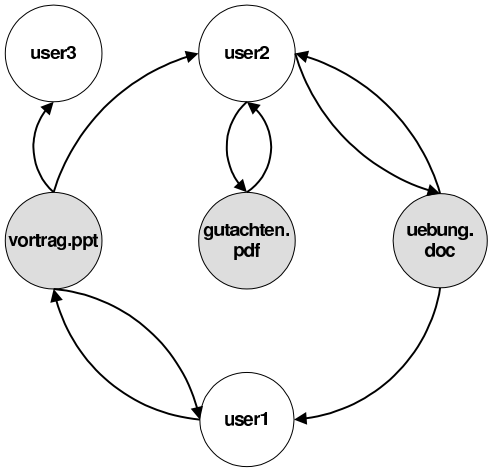

Abb. 2: Abgeleiteter Informationsflussgraph

2.4 Analysemethode

Auf der Grundlage der bisher dargestellten Zusammenhänge besteht die Methode zur Analyse von Zugriffssteuerungssystemen aus vier Schritten - der Gewinnung der ZSLs des zu analysierenden IT-Systems, der Konstruktion der ZM aus den ZSLs, der Transformation der ZM in einen IG und schließlich der eigentlichen Informationsfluss. Diese Schritte werden im Folgenden anhand eines Unix-basierten IT-Systems näher erläutert.

Extraktion: Von der Zugriffssteuerungsliste zur Zugriffsmatrix

In dem ersten Schritt wird die aktuelle Rechtesituation eines Unix-artigen Systems aus seinen Zugriffssteuerungslisten extrahiert und in eine äquivalente Zugriffsmatrix überführt.

In einem Unix-artigen System enthält das Dateisystem die für die Extraktion benötigen Informationen über die Objekte und ihre Zugriffssteuerungslisten. Diese Informationen können über entsprechende Systemaufrufe (z.B. *stat()* der Unix API) ermittelt werden. Die in einem System registrierten Benutzer sowie ihre Gruppenzugehörigkeiten liefern alle notwendigen Informationen über die Subjekte. Aus der Komposition dieser Informationen entsteht dann die Zugriffsmatrix (Abbildung 3).

Hierbei ist es notwendig, für jedes Objekt das 3er-Tupel aus (*read/write/execute*)-Rechtemengen für sämtliche Subjekte hinsichtlich der Rechte *read* und *write* zu untersuchen. Der Benutzer *root* besitzt prinzipiell für alle Objekte Lese- und Schreibzugriff; für sämtliche anderen Subjekte müssen die Rechte der Zugriffssteuerungsliste hinsichtlich *Eigentümer-*, *Gruppen-* oder *all-others*-Status ausgewertet werden. Die entstehende Zugriffsmatrix enthält am Ende dieses Extraktionsschritts alle Rechte, die ein Nutzer zu diesem Zeitpunkt besitzt. Um

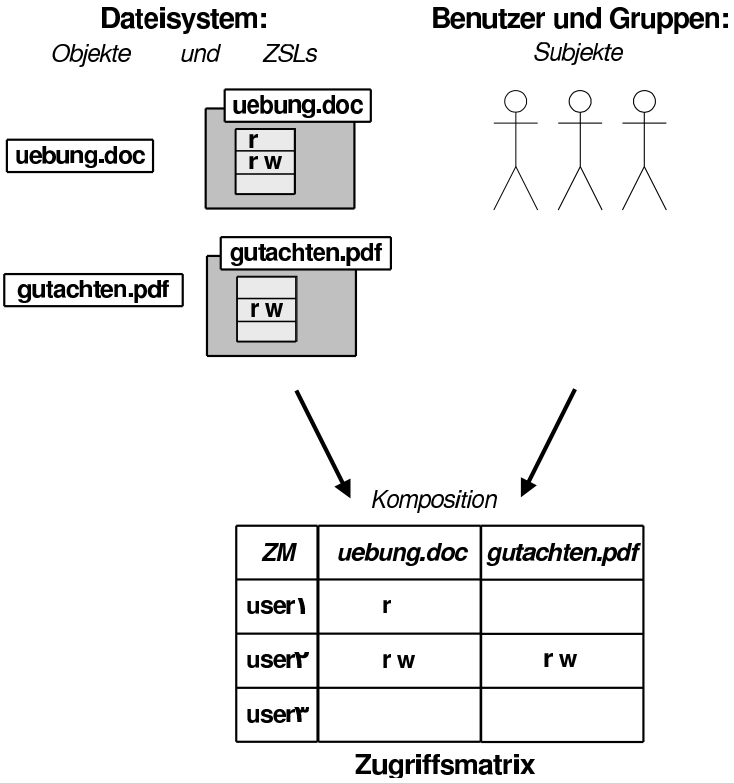

Abb. 3: Überführen der ZSLs in eine ZM

anschließend aussagekräftige Analyseergebnisse zu erhalten dürfen hierbei die Informationen über den Ursprung eines Rechtes (Besitzerrecht, Gruppenrecht oder all-others) nicht verloren gehen.

Transformation: Von der Zugriffsmatrix zum Informationsflussgraphen

Im nächsten Schritt erfolgt die Transformation der Zugriffsmatrix in den Informationsflussgraphen gemäß der in 2.2 definierten bijektiven Abbildung. Hierzu werden für alle Subjekte und Objekte Knoten erzeugt, die über gerichtete Kanten entsprechend der Rechte in der Zugriffsmatrix verbunden werden. Im Ergebnis entsteht ein die Zugriffsmatrix widerspiegelnder Informationsflussgraph.

Analyse

Im Anschluss an die Transformation erfolgt die zweiphasige Informationsflussanalyse. In der ersten Phase werden Auffälligkeiten anhand von Äquivalenzklassen und Wegen im Informationsflussgraphen gesucht (vgl. Kapitel 2.1) . Diese richten sich nach den definierten Analysezielen und der intendierten Zugriffssteuerungspolitik.

Wurden Auffälligkeiten gefunden, können in einer zweiten Phase deren Ursachen schärfer analysiert werden. Hierzu werden die Informationen über den Ursprung der Rechte benötigt, aufgrund derer im IG ein Weg oder die Zugehörigkeit zu einer Äquivalenzklasse entstanden ist. Die Bijektivität der ZM/IG-Transformation erlaubt es dabei, Kanten im IG eindeutig auf ihre Ursache, einem Recht in einer ZSL, zurückzuführen und auf diese Weise die direkte Ursache eines ungewollten Informationsfluss zu finden. Typische, in unseren eigenen experimentellen

Analysen vorkommende Maßnahmen zur Abhilfe sind

- das Entfernen von Nutzern aus Gruppen
- das Ändern von Gruppenrechte,
- das Ändern von Gruppenhierarchien oder
- das Ändern der Rechte von *all-others.*

Erwähnenwert erscheint in diesem Zusammenhang, dass ein Nutzer durch das Entfernen aus einer Gruppe den Status *all-others* für das betroffene Objekt erhält und somit unter Umständen durchaus nun *mehr* Rechte als zuvor besitzen kann.

3 Das Werkzeug

Die Analyse der Zugriffssteuerungssysteme realer IT-Systeme nach der in Abschnitt 2.4 vorgestellten Methode stößt i. d. R. auf die Schwierigkeit, eine Matrix erheblicher Größe auszuwerten – für einen mittelgroßen Fileserver beispielsweise kann diese leicht einige Milliarden Zellen enthalten. Manuelle Erstellung und Analyse des IGs sind daher ausgeschlossen; aus diesem Grund wurde das Softwaretool *I-Graphoscope* [Amt08] entwickelt.

3.1 Die Analysemethode in der Praxis

I-Graphoscope ist ein Werkzeug zur automatischen Analyse von Zugriffssteuerungsmatrizen. Eingabe ist eine Zugriffsmatrix, die zunächst in einen Informationsflussgraphen transformiert wird; dieser wird anschließend durch Graphalgorithmen auf seine Eigenschaften untersucht. Zur internen Darstellung von ZM und IG wurden in Bezug auf Adressraumbedarf und Zugriffszeit hocheffiziente Datenstrukturen entwickelt, die eine Analyse selbst auf den derzeit noch verbreiteten 32-Bit-Arbeitsplatzsystemen erlauben.

Im ersten Analyseschritt wird der IG auf vorhandene Äquivalenzklassen untersucht; in der Praxis kann dies automatisiert und pauschal für den gesamten Graphen realisiert werden. Anders verhält es sich mit dem Finden bestimmter Wege: Hier ist die Software auf menschliches Eingreifen angewiesen, da die reine ZM als Eingabe keine ausreichenden Informationen zur automatischen Beurteilung von sicherheitskritischen Wegen bereitstellt (wie etwa die Auszeichnung besonders lese- oder schreibsensibler Objekte). Bei der Analyse der ZM aus Abschnitt 2 etwa könnten Fragen zu beantworten sein wie: „Gibt es einen Pfad von *gutachten.pdf* zu *user3*?“ oder „Gibt es einen Pfad von *user1* zu *uebung.doc*?“.

Die softwaregestützte Auswertung ist also aus Nutzersicht in zwei Schritte eingeteilt: Am Beginn steht das Einlesen einer ZM mit anschließender Erzeugung des IGs, bei der noch im selben Schritt eine Reduktion auf alle verfügbaren Informationsklassen erfolgt. Anschließend ermöglicht *I-Graphoscope* das bedarfsgerechte Finden von Wegen in diesem Graphen, wobei sich der Anwender an Sicherheitszielen der intendierten Sicherheitspolitik sowie den schützenswerten Subjekten und Objekten des IT-Systems orientieren wird.

3.2 I-Graphoscope

Eingabe

Zur Gewinnung einer ZM aus den ZSLs eines zu analysierenden IT-Systems ist zuvor eine systemspezifische Software erforderlich. Für Unix-artige Systeme wurde hierzu ein Unix-

spezifisches Werkzeug entwickelt, das auf Unix-artige Rechte- und Gruppensemantiken zugeschnitten ist. Alternativ lässt *I-Graphoscope* auch die manuelle Erstellung neuer sowie eine nachträgliche Bearbeitung bereits eingelesener Zugriffsmatrizen über eine GUI zu, die auch das Abspeichern und Laden einmal eingelesener bzw. erstellter Matrizen ermöglicht. Die Darstellung einer (sehr kleinen) ZM zeigt exemplarisch Abbildung 4.

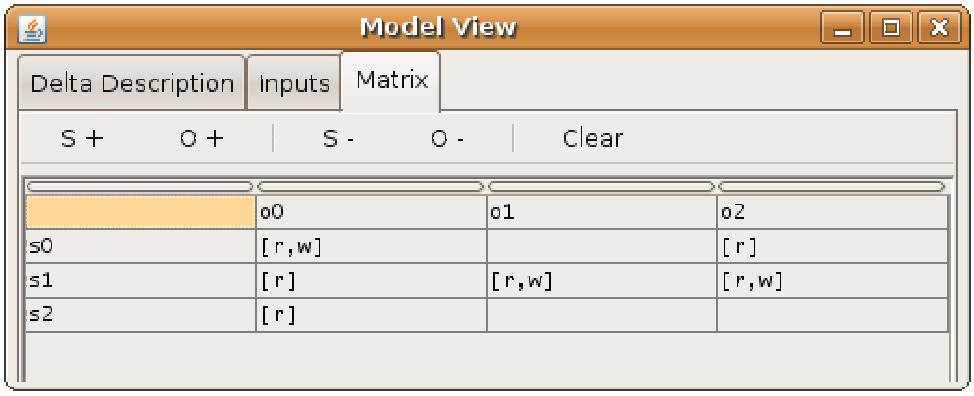

Abb. 4: Zugriffsmatrix in *I-Graphoscope*

Ausgabe

Abhängig von ihrer Größe können Informationsflussgraphen in zwei verschiedenen Modi bearbeitet werden: graphisch und textbasiert.

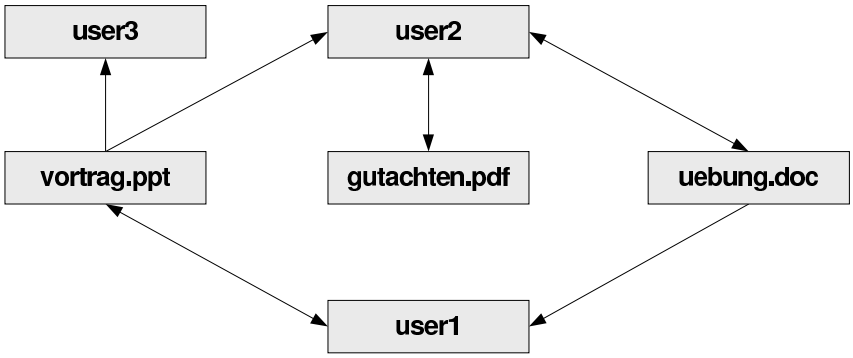

Abb. 5: Graphische Darstellung eines expandierten IGs

graphisch: Zur Veranschaulichung der Existenz und Lage von Wegen und Informationsklassen kann eine interaktive visuelle Graphdarstellung generiert werden. Kreise werden automatisch detektiert und per Mausklick zu Informationsklassen zusammengefasst; auf diese Weise kann ein minimaler Graph im Sinne der Informationspotenziale konstruiert werden. Abbildungen 5 und 6 zeigen einen solchen IG in seiner expandierten und reduzierten Form. Einzelne Subjekte und Objekte sind dabei durch dunkle, Informationsklassen entsprechend Abschnitt 2.4 durch hellere Knoten dargestellt. Eine Eingabemaske ermöglicht das Finden von Wegen, indem u. a. gezielt nach der Existenz eines Pfades zwischen einem Anfangsknoten (etwa *gutachten.pdf*) und einem Endknoten (etwa *user1*) gesucht werden kann.

textbasiert: Im Batchverarbeitungsmodus sind alle o. g. Auswertungen in derselben Weise möglich, jedoch tritt hier eine Textausgabe an die Stelle des visualisierten IGs. Diese liefert die im reduzierten Graphen enthaltenen Informationsklassen und deren angehörige Subjekte / Objekte sowie ggf. existierende Pfade zwischen spezifizierten Graphknoten. Dieser Modus ist vor allem interessant, wenn die eingelesene ZM die Größe realer Systeme annimmt und damit ein Visualisierung wenig sinnvoll wird.

Anhand dieser Ausgaben legt *I-Graphoscope* den Grundstein für den zweite Analyseschritt aus Abschnitt 2.4.

Abb. 6: Graphische Darstellung eines reduzierten IGs im *I-Graphoscope*

Auf diese Weise erschließt sich neben der Analyse von akademischen Beispielen überschaubarer Größe auch die Analyse realer Systeme mit tausenden von Nutzern und Millionen von Objekten. Um eine sinnvolle Darstellung zu erhalten, ist die Matrixgröße bei grafischer Ausgabe des IGs auf einige hundert Zellen beschränkt; bei Überschreiten dieser Grenze schaltet das Werkzeug automatisch (unterhalb der Grenze auch manuell) auf Batchverarbeitung und Textausgabe um. Laufzeitlimitationen in Abhängigkeit von der Matrixgröße sind in erster Linie für diesen Modus relevant. Konkrete Werte nennt Kapitel 4.

4 Ergebnisse

Dieses Kapitel stellt die Ergebnisse der Anwendung von Analysemethode und -werkzeug auf einen unserer eigenen Fileserver vor. Vorweggeschickt sei, dass sowohl die Administratoren als auch ein Großteil der Benutzer dieses Servers – darunter auch die Autoren – Menschen sind, die aufgrund ihres Umfeldes eine vermutlich überdurchschnittliche Sensibilität für IT-Sicherheitsproblematiken besitzen.

Der analysierte Server stellt Dateidienste und eine Auswahl an Internetdiensten für etwa 330 Benutzer bereit. Im Kontext dieser Analyse lassen sich die Benutzer in zwei Kategorien einteilen: Benutzer, die entweder in keinem oder aber in mindestens einem gemeinsamen Projekt arbeiten. Benutzer, die in keinem Projekt tätig sind, arbeiten lediglich in ihrem Home-Verzeichnis; Benutzer, die mit anderen Benutzern in Projekten zusammenarbeiten, haben zusätzlich Zugriff auf die entsprechenden Projektverzeichnisse. Diese Kategorisierung erlaubt es, im Kontext der Evaluation die Analyse der Zugriffsmatrix auf eine vergleichsweise kleine Anzahl von Benutzern zu beschränken, sofern diese aus beiden Kategorien gewählt werden.

Das Betriebssystem des Servers ist Linux; entsprechend besitzt das Dateisystem die Standard-Unix-Zugriffssteuerungssemantik: diskreter, Eigentümer basierter Zugriffsschutz mittels einfacher, den Objekten assoziierten Zugriffssteuerungslisten, die jeweils Lese-, Schreib- und Ausführungsrechte für den Eigentümer, eine Benutzergruppe und alle sonstigen Benutzer beschreiben. Unter Ausschluss aller Administratoren (d.h. alle Benutzer der Gruppe *root*) besaß die Zugriffsmatrix zum Zeitpunkt der Analyse ca. 6,5 Millionen Matrixzellen.

Das Extrahieren der Zugriffssteuerungslisten aus unserem Server sowie das Erstellen der kompletten Zugriffsmatrix (330 Benutzer, 650.000 Dateien) mit ca. 214.500.000 Matrixzellen benötigte 245 Sekunden. Diese Zeit ist abhängig von der Hardware und des Betriebssystems des

zu analysierenden Servers. Das Analysewerkzeug ist plattformunabhängig und wurde im Rahmen unserer Tests auf einem Intel Core Two Duo Prozessor (jeweils 2,5 GHz Taktfrequenz) mit insgesamt 3,0 Gigabyte RAM mit dem Betriebssystem Microsoft Windows XP Professional, Service Pack 3, ausgeführt. Das Anwenden der Analysesoftware auf die eingeschränkte Zugriffsmatrix mit 6,5 Millionen Matrixzellen, das Generieren des IGs sowie das Identifizieren von Äquivalenzklassen benötigte insgesamt 680 Sekunden.

Aufgrund des überdurchschnittlichen Sicherheitsbewusstseins der Benutzer unseres Servers erwarten wir, dass jeder Benutzer die Rechte an seinen Dateien sorgfältig verwaltet und somit vor Zugriffen anderer Benutzer schützt. In diesem Fall würde jeder Benutzer zusammen mit den Objekten seines Home-Verzeichnisses eine eigene Informationsfluss-Äquivalenzklasse bilden, so dass der IG bei elf Benutzern idealerweise in elf wohl-isolierte Äquivalenzklassen zerfiele. Bei einem gut administrierten System, wie es in Abbildung 7 zu sehen ist, besäße der IG eine weitere Äquivalenzklasse, die ausschließlich Systemdateien und -benutzer (daemon, sys usw.) enthielte. Ein Informationsfluss in die einzelnen Äquivalenzklassen wäre ausschließlich über die entsprechenden Benutzer der Home-Verzeichnisse (z.B. $s_1, s_2, ..., s_n$) möglich.

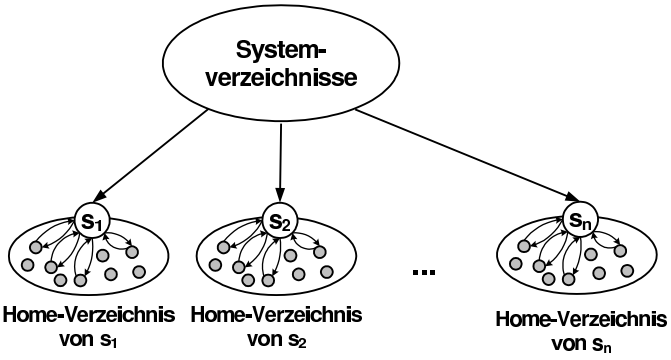

Abb. 7: Erwarteter Zustand einer gut administrierten Systemumgebung

Würden weiterhin gemeinsame Projektverzeichnisse berücksichtigt, ergäben sich in einem gut administrierten System weitere Äquivalenzklassen, deren Schnittstellen diejenigen Benutzer sind, die in diesem Projekt arbeiten. Üblicherweise müssen Projektmitarbeiter sowohl lesenden als auch schreibenden Zugriff auf die gemeinsamen Projektdaten besitzen. Dies führt dazu, dass die einzelnen Äquivalenzklassen der Benutzer und der entsprechenden Projekte kollabieren würden (siehe Abbildung 8). Die Äquivalenzklassen derjenigen Benutzer, die in keinem gemeinsamen Projekt tätig sind, wären weiterhin wohlisoliert. Daraus lässt sich schließen, dass eine Systemumgebung umso besser administriert ist, je höher die Anzahl der Äquivalenzklassen des IGs ist.

Tatsächlich weichen die Analyseergebnisse gravierend von unseren Erwartungen ab. Der IG besitzt lediglich *eine einzige*, sämtliche Benutzer und einen Großteil der Objekte (95%) umfassende Äquivalenzklasse, innerhalb derer sämtliche Knoten dasselbe Informationspotenzial besitzen (siehe Abbildung 9). Innerhalb dieser Klasse bildet jeder einen Benutzer repräsentierende Knoten eine Singularität, über die selbst in dem hier untersuchten Serversystem bescheidenen Ausmaßes Milliarden von Informationsflusspotenzialen führen, über deren Nutzung allein dieser Knoten (d.h. ein individueller Benutzer) entscheiden kann.

Entfernt man im IG alle Kanten zwischen Benutzern und Dateien aus den Home-Verzeichnissen anderer Benutzer, so wird man einen IG mit isolierten Äquivalenzklassen derjenigen Benutzer

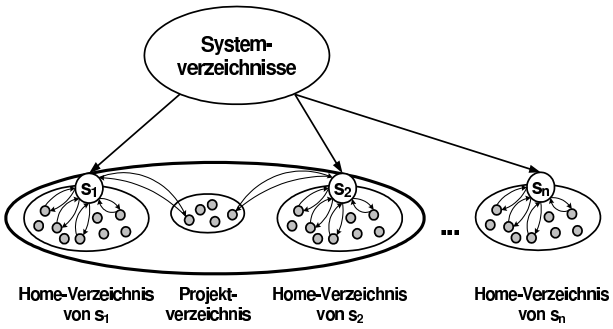

Abb. 8: Erwarteter Zustand einer gut administrierten Systemumgebung mit gemeinsamen Projektverzeichnissen

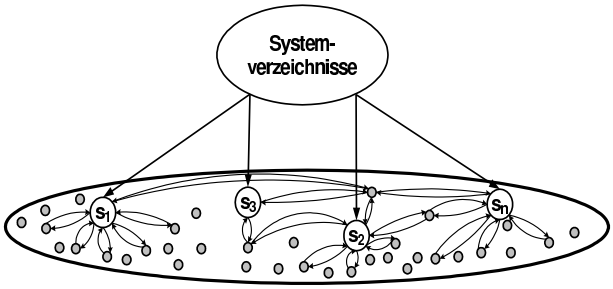

Abb. 9: Tatsächlicher Zustand unserer Systemumgebung

erwarten, die nicht in Projekten tätig sind. Tatsächlich konnte auch dieses Resultat nicht beobachtet werden. Dagegen wurden etliche weitere Informationsflüsse gefunden, die zeigen, dass selbst professionelle und sicherheitsbewusste Benutzer nicht durchgängig in der Lage sind, Informationsflüsse über die Projektverzeichnisse zu verhindern. Ursache hierfür waren in nahezu allen beobachteten Fällen Rechte, die über die Gruppenzugehörigkeit von Benutzern entstanden.

Die Ursachen für derartige Miskonfigurationen sind vielschichtig. Im Unterschied zu obligatorischen Zugriffssteuerungssemantiken, wie diese z.B. für die Realisierung von Informationsfluss-Sicherheitspolitiken [Den76, Fol89] erforderlich und in jüngeren Betriebssystemen auch enthalten sind [SSL+99, LS01, EK08], liegt bei diskreten Zugriffsschutzsemantiken das Recht der Zugriffsrechtevergabe bei den individuellen Eigentümern der Objekte. Mit diesem Recht ist eine hohe Verantwortung verbunden: da eine zentrale, obligatorische Sicherheitpolitik in IT-Systemen mit diskreter Zugriffssteuerungssemantik nicht durchgesetzt werden kann, liegt die Last der Organisation und Durchsetzung der Sicherheitspolitik auf den Schultern jedes einzelnen Benutzers; und dieser ist in diskreten Systemen in der Lage, ob ungewollt oder auch gewollt, durch fehlerhafte Rechtekonfiguration diese ohne weiteres zu unterlaufen. Aufgrund technischer Grenzen bei der Realisierung von diskreten Zugriffssteuerungssemantiken können auch professionelle und sicherheitsbewusste Benutzer die Durchsetzung der Sicherheitspolitik nicht durchgängig garantieren. Da jeder Benutzer lediglich eine eingeschränkte Perspektive auf seine persönlichen Dateien hat, fällt es selbst kompetenten Benutzern sehr schwer, den Überblick über Gruppenzugehörigkeiten und die damit verbundenen Rechtevergabe zu behalten. Darüber hinaus ist das Risiko einer Miskonfiguration enorm, da schon eine einzige Datei mit lesenden und schreibenden Zugriff für alle Benutzer eine Singularität im IG bildet und ausreicht, alle Äquivalenzklassen in einem IG zu einer einzigen kollabieren zu lassen.

5 Zusammenfassung

In diesem Beitrag wird eine Methode zum Aufdecken indirekter Informationsflusspotenziale in Zugriffssteuerungssystemen vorgestellt, die die hohe Komplexität der Suche von Informationsflüssen in Zugriffsmatrizen auf das Problem des Findens von Wegen in durch Äquivalenzklassenbildung reduzierten Informationsflussgraphen abbildet. Die vorgestellte Methode wendet eine bijektive Abbildung von Zugriffsmatrizen auf Informationsflussgraphen an, die das Aufdecken indirekter Informationsflusspotenziale eindeutig auf Zugriffsrechte in der Zugriffsmatrix rückführbar macht. Zur Anwendung der Methode auf reale Systeme wurde ein Werkzeug für die Analyse von Systemen mit Unix-Zugriffssteuerungslisten entwickelt, mit dessen Hilfe ein Unix-Server mit einer Zugriffsmatrix von ca. 6,5 Millionen Zellen analysiert wurde.

Ziel der Analyse war es, Informationsflusspotenziale zu finden, die nicht der intendierten Sicherheitspolitik entsprechen. Die Ergebnisse zeigen die grundsätzlichen Schwächen diskreter Zugriffssteuerungssysteme, die Auswirkungen einer daraus resultierenden Überforderung individueller Benutzer und die Gefahren eines wenig transparenten Gruppenmanagements auf. Miskonfigurationen wurden aufgedeckt und ihre Ursachen festgestellt. So hatten beispielsweise einige Nutzer ihre Daten effektiv bzgl. der anderen Nutzer isoliert, während andere bei der Verwaltung ihrer Dateien die Rechte von Gruppen und insbesondere von *all-others* geradezu vernachlässigten. Auf diese Weise entstehen Informationsflusspotenziale, die jedem regulären Systemnutzer Informationen zugänglich machen, auf die er direkt keinen Zugriff besitzt.

Die Ergebnisse machen ein sehr grundsätzliches Problem deutlich: diskrete Zugriffssteuerungssemantiken übertragen einen erheblichen Teil der Pflichten und Verantwortung bei der Durchsetzung einer systemweiten Sicherheitspolitik individuell auf einzelne Nutzer; dagegen haben verdeckte Informationsflüsse in der Regel globale Ursachen, die für einzelne Benutzer gar nicht erkennbar sind.

Grundsätzliche Abhilfe kann hier nur ein Paradigmenwechsel hin zu einer obligatorischen Zugriffssteuerung schaffen, da die Ursachen tief in den diskreten Zugriffssteuerungsmechanismen heutiger Standardbetriebssysteme verankert sind; sie lassen sich folglich auch nicht durch kurzfristige Reparaturmaßnahmen beseitigen. Abhilfe können hier Analysesysteme schaffen, die nicht nur – wie das hier vorgestellte *I-Graphoscope*-Werkzeug – verdeckte Informationsflüsse im Nachhinein aufdecken, sondern in Form eines Live-Monitorings ihr Zustandekommen von vorn herein verhindern. Die Effizienz und Performanz der in unserem Werkzeug eingesetzten Reduktions- und Suchalgorithmen sind ein deutlicher Hinweis auf die Machbarkeit dieses Ansatzes; die Entwicklung einer Live-Variante des Werkzeugs wird daher derzeit diskutiert.

Literatur

[Amt08] Peter Amthor. Generierung von Informationsflussgraphen aus HRU-Modellen, 2008.

[Den76] Dorothy E. Denning. A Lattice Model of Secure Information Flow. *Communications of the ACM*, 19(5):236–242, May 1976.

[EK08] Petros Efstathopoulos and Eddie Kohler. Manageable Fine-Grained Information Flow. In *Proc. of the 2008 EuroSys Conference*, pages 301–313. ACM SIGOPS, April 2008.

[Fol89] Simon N. Foley. A Model for Secure Information Flow. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 248–258. IEEE, May 1989.

[Gmb07] CORPORATE TRUST Business Risk & Crisis Management GmbH. Industriespionage – Die Schäden durch Spione in der deutschen Wirtschaft. Studie, erhältlich unter www.corporate-trust.de/, 2007.

[KM04] Egbert Kahle and Wilma Merkel. Fall- und Schadensanalyse bezüglich Know-how-/Informationsverlusten in Baden-Württemberg ab 1995. Studie, Universität Lüneburg, erhältlich unter www.sicherheitsforum-bw.de, 2004.

[Lam74] Butler W. Lampson. Protection. *Operating Systems Review*, 8(1):18–24, January 1974.

[LS01] Peter A. Loscocco and Stephen D. Smalley. Meeting Critical Security Objectives with Security-Enhanced Linux. In *Proceedings of the 2001 Ottawa Linux Symposium*, 2001.

[SSL+99] Ray Spencer, Stephen Smalley, Peter Loscocco, Mike Hibler, David Andersen, and Jay Lepreau. The Flask Security Architecture: System Support for Diverse Security Policies. In *Proc. of the 8th USENIX security Symposium*, 1999.

[Tea08] Verizon Business Risk Team. 2008 Data Breach Investigations Report. Studie, erhältlich unter www.verizonbusiness.com, 2008.