Privacy in sozialen Netzwerken: Eine Informationsflussanalyse

Peter Amthor · Winfried E. Kühnhauser

Technische Universität Ilmenau
{peter.amthor | winfried.kuehnhauser}@tu-ilmenau.de

Zusammenfassung

Seit vielen Jahren kommen Zugriffssteuerungssysteme zur Wahrung von Sicherheitseigenschaften wie Vertraulichkeit und Integrität in IT-Systemen zum Einsatz. Deren grundlegende Konzepte sind innerhalb traditioneller Domänen wie die der Betriebssysteme oder Datenbank-Managementsysteme etabliert und werden im Allgemeinen als ausgereift und verlässlich angesehen, aus diesem Grund sind sie vermehrt auch bei jüngeren Anwendungsdomänen wie den sozialen Online-Netzwerken anzutreffen. Zugleich können bei der Durchsetzung von Zugriffssteuerungspolitiken in solchen Systemen Schwachstellen in Form unerwünschter, indirekter Informationsflüsse auftreten, welche wiederum in der Domäne der Betriebssysteme wohlbekannt und erforscht sind – deren Konsequenzen für soziale Online-Netzwerke jedoch vergleichsweise neu sind.
Vor diesem Hintergrund verfolgt das vorliegende Papier das Anliegen, die Auswirkungen indirekter Informationsflüsse für die Nutzer sozialer Online-Netzwerke zu beleuchten. Hierzu betrachten wir sowohl ein traditionelles, Betriebssystem-basiertes Zugriffssteuerungsszenario, als auch das Szenario eines sozialen Online-Netzwerkes aus dem Blickwinkel der Informationsflussanalysen. Wir werden Ursachen und Folgen von Informationsflussphänomenen vergleichen, um schließlich Empfehlungen hinsichtlich der Weiterentwicklung entsprechender Zugriffssteuerungssysteme im Sinne der Privatsphäre ihrer Nutzer zu geben.

1 Einleitung

Wann immer die Privatsphäre (*Privacy*) der Nutzer eines IT-Systems zur Debatte steht, kommt der Betrachtung potenzieller Informationsflüsse eine wesentliche Bedeutung zu [JHL02, CH04, EK08, XRAG13]. Außerhalb traditioneller Domänen wie die der Betriebssysteme oder Datenbank-Managementsysteme gilt dies insbesondere für jüngere Ausprägungen Internet-basierter Systeme – etwa der sozialen Online-Netzwerke (*OSNs, Online Social Networks*). In beiden Domänen ist die Beantwortung der Frage, welche Informationen von einem Nutzer zu einem anderen fließen können, von fundamentalem Interesse für die Umsetzung jeglicher Privacy-Eigenschaften.

Tatsächlich haben jahrzehntelange Erfahrungen mit diesem Thema im Kontext der Betriebssystemsicherheit gezeigt, dass solche Fragen immer dann besonders schwierig zu beantworten sind, wenn die Sicherheitspolitik eines Systems – im Sinne der Gesamtmenge an Regeln, welche die intendierten Sicherheitseigenschaften umsetzen – ganz oder teilweise vom Verhalten seiner Nutzer abhängig ist (*DAC, Discretionary Access Control*). Der Grund hierfür liegt in einem fundamentalen Widerspruch: DAC-Systeme werden auf Basis von (zwischenmenschlichem) Vertrauen bedient und konfiguriert; wann immer ein Nutzer einem anderen Zugriff auf

P. Schartner (Hrsg.) · D•A•CH Security 2016 · syssec (2016) 224-240.

seine persönlichen Informationen gewährt, ist dies ein Ausdruck von (wenigstens einseitigem) Vertrauen. Während dieses Vertrauen nur direkt zwischen zwei Nutzern bestehen kann, sind Informationsflüsse sind ihrer Natur nach transitiv, sie können also auch auf indirektem Wege zustande kommen.

Um die Frage nach Sicherheitseigenschaften eines IT-Systems möglichst präzise zu beantworten, haben die vergangenen Jahrzehnte eine Vielzahl formaler Methoden zur ihrer Implementierung und Analyse hervorgebracht. Eines der grundlegenden formalen Modelle zur Beschreibung eines Zugriffssteuerungssystems ist die Zugriffsmatrix (*ACM, Access Control Matrix*) [Lam74], welche konzeptionell Eingang in praktisch alle Modelle moderner Zugriffssteuerungssysteme gefunden hat [HRU76, San92, ZLN05, AKP11]. Die ACM beschreibt, welche Operationen ein Subjekt (etwa ein Nutzer) auf einem Objekt (etwa eine Datei oder ein Datensatz) ausführen darf. Betriebssysteme implementieren eine ACM typischerweise in Form von Zugriffssteuerungslisten (*ACLs, Access Control Lists*) ihrer einzelnen Objekte, welche für jedes Subjekt nur diejenigen Operationen (in Form von Zugriffsrechten) auflisten, für die es gemäß der Sicherheitspolitik des Systems autorisiert ist. Auf diese Weise kann die ACL eines Objekts o direkte Informationsflüsse zu einem Subjekt s unterbinden – nicht jedoch indirekte: falls nämlich ein Subjekt s' mit Leserechten an o und Schreibrechten an einem weiteren Objekt o' existiert, welches wiederum von s gelesen werden darf, bedeutet dies einen potenziellen Informationsfluss von o nach s: $o \to s' \to o' \to s$. Dieser Informationsfluss kann weder in der ACL von o noch in der ACM des Zugriffssteuerungssystems unmittelbar beobachtet werden.

Zur Lösung dieses Problems wurden spezialisierte formale Modelle entwickelt, welche anstelle von Zugriffsrechten die Informationsflüsse innerhalb eines Zugriffssteuerungssystems beschreiben [BL73, MLCJ11, KG13]. Um die transitive Eigenschaft letzterer zu modellieren, kommen hierbei mathematische Konstrukte wie Verbände und Graphen zum Einsatz – welche allerdings die angenehme semantische Nähe zur praktischen Implementierung solcher Systeme (etwa durch ACLs) opfern. Damit verbietet sich wiederum eine effiziente Analyse von Informationsflussmodellen für reale Systeme.

So wie bei den traditionellen Domänen der Zugriffssteuerungssysteme ist auch für soziale Online-Netzwerke die formale Analyse von Sicherheitseigenschaften innerhalb der letzten Jahre zunehmend in den Mittelpunkt des Interesses von Forschungs- und Entwicklungsarbeiten gerückt [FS11, CPS12]. Den grundlegenden Paradigmen dieser Systeme entsprechend kommen bei ihrer Implementierung Relationen-basierte Sicherheitspolitiken zum Einsatz, welche die Semantik des „Teilens“ von Informationen zwischen Nutzern widerspiegeln. Wie schon die ACMs erlauben diese Sicherheitspolitiken keine unmittelbare Beobachtung von indirekten Informationsflüssen, wie schon in der Domäne der Betriebssysteme bieten die existierenden Informationsflussmodelle keine adäquate Semantik zur Repräsentation ihrer Relationen-basierten Zugriffsregeln. Offenbar sind indirekte Informationsflüsse in beiden Domänen nur mit hohem Aufwand zu identifizieren, nämlich durch Begutachtung der vollständigen Zugriffsmatrix bzw. der vollständigen Menge aller Relationen zwischen Nutzern.

Vor diesem Hintergrund soll das vorliegende Papier einen neuen Blickwinkel auf die Frage ermöglichen, welche Bedeutung dem Problem der verdeckten Informationsflüsse in der Praxis zukommt – unter besonderer Berücksichtigung sozialer Online-Netzwerke neben den traditionellen Betriebssystem-Zugriffssteuerungssystemen. Dabei werden wir zunächst, in Abschnitt 2, das grundlegende Problem durch ein praktisches Beispiel illustrieren. Im Anschluss folgen zwei eingehende Fallstudien, welche das Informationsflussproblem einerseits aus Sicht

```
rw- --- --- Anna StaffX 2016-09-26 04:41 ProjectXCode
rw- r-- --- Anna StaffX 2013-09-21 21:19 ProjectXBoard
rw- r-- --- Bernd Sales 2013-08-30 13:50 SalesBoard
rw- --- --- Chris Sales 2013-09-26 17:33 SalesFlyer
```

(a) in `ls`-Notation

ACM	ProjectXCode	ProjectXBoard	SalesBoard	SalesFlyer
Anna	r w	r w		
Bernd		r	r w	
Chris			r	r w

(b) als ACM

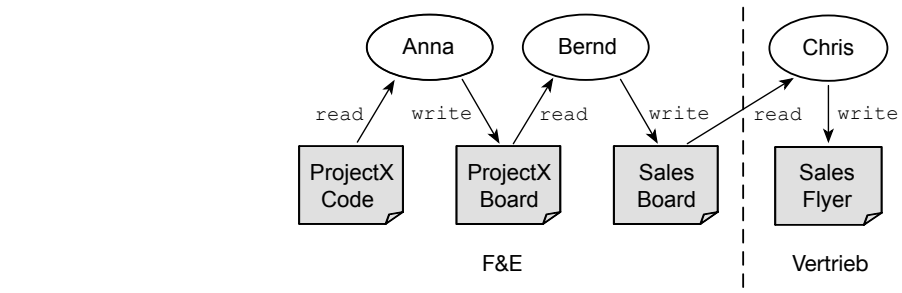
Abb. 1: Zugriffssteuerungsregeln eines Unix-Betriebssystems

eines Dateiserver-Betriebssystems (Abschnitt 3), andererseits aus Sicht eines sozialen Online-Netzwerkes beleuchten (Abschnitt 4). Dabei soll die erste Fallstudie in erster Linie dazu dienen, weithin bekannte Erkenntnisse der Forschung hinsichtlich Informationsflüssen in Betriebssystemen praktisch zu verorten. Die zweite Fallstudie soll einerseits zeigen, in welchem Maße moderne Internet-Dienstleistungen wie OSNs Probleme vergleichbaren Schweregrads aufweisen; andererseits liefert sie Anhaltspunkte für signifikante Verbesserungen der Privacy von OSN-Nutzern durch neue Kontroll- und Informationsmechanismen (beschrieben in Unterabschnitt 4.3). In beiden Studien wird eine graphbasierte Analysemethode zum Auffinden indirekter Informationsflüsse vorgestellt und implementiert, sowie die praktischen Implikationen ihrer jeweiligen Ergebnisse diskutiert. Unsere Schlussfolgerungen und Ausblicke für die Entwicklung moderner Zugriffssteuerungssysteme fassen wir in Abschnitt 5 zusammen.

2 Ein Beispielszenario

Betrachten wir das Szenario eines Unternehmens, dessen IT-System in einzelne Abteilungen partitioniert ist – unter anderem Forschung und Entwicklung (F&E) and Vertrieb. Anna ist eine Projektmanagerin in F&E, Bernd ein Mitglied ihres Teams. Anna ist innerhalb des IT-Systems ihrer Abteilung Eigentümerin des streng vertraulichen Software-Repositories *ProjectXCode* sowie eines internen Entwicklerboards *ProjectXBoard* (hier zur Illustration jeweils als einzelnes Objekt dargestellt). Anna hat auf beide Objekte Lese- und Schreibrechte, wodurch ein potenzieller Informationsfluss von diesen zu Anna und umgekehrt besteht. Ihrem Projektteam (Nutzergruppe *StaffX*) gewährt Anna nur Lesezugriff auf *ProjectXBoard*. Bernd, der als Mitglied von *StaffX* das Board lesen darf, ist außerdem für die Abstimmung mit dem Vertrieb zuständig, für die er ein weiteres elektronisches Nachrichtenforum *SalesBoard* mit Lese- und Schreibrechten für sich selbst eingerichtet hat. Chris schließlich, der als Vertriebsmitarbeiter (Nutzergruppe *Sales*) momentan an einem Ankündigungstext *SalesFlyer* für das in Entwicklung befindliche Produkt arbeitet, darf wie alle Vertriebsmitarbeiter in Bernds *SalesBoard* Statusberichte und -aktualisierungen lesen (etwa um diese für Produktankündigungen zu verwenden); gleichzeitig stellen die Regeln des Zugriffssteuerungssystems sicher, dass er keinerlei Zugriff auf *ProjectXCode* hat. Das Rechtemanagement Unix-artiger Dateisysteme würde die beschriebene Konfiguration darstellen wir in Abb. 1(a), Abb. 1(b) zeigt die zugrunde liegende ACM.

Obwohl hier sämtliche Rechte direkt aus der oben beschriebenen Sicherheitspolitik abgeleitet wurden, enthüllt ein ganzheitlicher Blick auf das Zugriffssteuerungssystem ein schwerwiegendes Informationsleck: der in Abb. 2(a) dargestellte, unerwünschte Fluss vertraulicher Informationen aus Annas *ProjectXCode* zu Chris' *SalesFlyer* kann stattfinden, indem (korrekt gesetzte)



(a) Unerwünschter, indirekter Informationsfluss

(b) Äquivalenter Informationsflussgraph

Abb. 2: Informationsflüsse im Beispielszenario

Rechte von zwischengeschalteten Subjekten und Objekten ausgenutzt werden.

Zwar setzt dieser Informationsfluss die Kooperation der beteiligten Nutzer voraus – man beachte aber, dass diese in der Praxis nicht unbedingt freiwillig erfolgen muss. Da beispielsweise Anna, Bernd und Chris einander allein durch ihren Kollegenstatus ein gewisses Maß an gegenseitigem Vertrauen entgegen bringen, wären die Aussichten für Chris, manipulierte Spähsoftware durch Social Engineering auf Bernds Rechner zu hinterlassen, vergleichsweise gut. Der entscheidende Punkt ist, dass hier ein *existierender* Kanal für Informationsflüsse genutzt wird, der keinerlei direkte Autorisation hinsichtlich des Zielobjekts benötigt – welches durchaus anforderungsgemäß vom Zugriffssteuerungssystem geschützt wird.

Durch die Transitivität von Informationsflüssen einerseits und die Implementierung der Zugriffssteuerungsmatrix andererseits ist es Insidern möglich, ohne Überwältigung des eigentlichen Zugriffssteuerungssystems die Vertraulichkeit von Informationen erfolgreich anzugreifen: lediglich die Kenntnis indirekter, potenzieller Informationsflüsse und deren Ausnutzung ist erforderlich. Gleichwohl sind spezifische indirekte Informationsflüsse alles andere als leicht zu finden (daher ist in diesem Papiers auch von verborgenen Informationsflüssen die Rede). Gründe hierfür sind der gewaltige Suchraum, bei realen Serversystemen beispielsweise in Form einer Milliarden Zellen umfassenden Matrix, gepaart mit der Laufzeitkomplexität entsprechender Suchalgorithmen: Die Berechnung direkter Informationsflüsse zwischen Subjekten und Objekten in einer ACM hat bereits Komplexität $\mathcal{O}(|Subjekte| \times |Objekte|)$, und die Laufzeitkosten zum Auffinden transitiver Pfade sind exponentiell von deren Länge abhängig.

Die Frage nach der tatsächlichen Anfälligkeit realer Zugriffssteuerungssysteme gegenüber der hier geschilderten Angriffsform bleibt im Einzelfall zu klären – im Folgenden stellen wir hierzu zwei Fallstudien vor.

3 Fallstudie: Dateiserver

Die erste Fallstudie basiert auf früheren Arbeiten zur Informationsflussanalyse von Betriebssystemen [AFK09] und betrachtet ein in dieser Domäne typisches Szenario: ein Dateiserver einer Universität mit einem professionell administrierten Debian Linux und einem *ext4*-Dateisystem; die Informationen auf diesem Server werden durch die DAC-Standardmechanismen von Linux geschützt. Sämtliche Nutzer des Systems besitzen einen fachlich soliden IT-Hintergrund.

3.1 Analyseansatz

Das zugrunde liegende Zugriffssteuerungsmodell des Dateiservers ist eine ACM, die als Spaltenvektoren in Form der ACLs sämtlicher Objekte des Dateisystems realisiert ist. ACMs können durch eine Funktion $acm : S \times O \to 2^R$ modelliert werden, wobei S, O und R die Mengen aller Subjekte, Objekte und Rechte eines Systems repräsentieren. Wenn zudem alle Rechte auf entweder eine Lese- oder eine Schreibsemantik reduziert werden können ($R = \{read, write\}$), kann jede ACM als bipartiter, gerichteter Informationsflussgraph $IFG = (V, E)$ dargestellt werden. Dabei gilt $V = S \cup O, E \subseteq (S \times O) \cup (O \times S)$ mit

$$(s,o) \in E \quad \Leftrightarrow \quad write \in acm(s,o) \text{ und} \tag{1}$$
$$(o,s) \in E \quad \Leftrightarrow \quad read \in acm(s,o). \tag{2}$$

Eine ACM und ihr korrespondierender IFG sind dann isomorph: zu jeder in der einen Struktur nachweisbaren Eigenschaft existiert eine direkte Entsprechung in der anderen. Als Beispiel ist in Abb. 2(b) der IFG dargestellt, welcher aus der ACM in Abb. 1(b) resultiert.[1]

Diese Isomorphie kann nun genutzt werden, um für Informationsflussanalysen die passende Modellrepräsentation zu wählen: Wenn beispielsweise im IFG ein Pfad zwischen Knoten o und s existiert, so muss die entsprechende ACM die notwendigen Rechte enthalten, damit Informationen des Objekts o zum Subjekt s fließen können. Zwischenknoten auf diesem Pfad sind dann weitere Subjekte und Objekte, mittels derer transitive Informationsflüsse entstehen (etwa Anna, Bernd oder *SalesBoard* in Abb. 2(a)). Jeder auf diese Weise entdeckte Informationsfluss kann direkt auf die isomorphe ACM zurückgeführt und somit seine Ursache in den ACLs realer Zugriffssteuerungssysteme identifiziert werden.

Äquivalenzklassenbildung

Eine Besonderheit stellen Kreise im IFG dar: jeder Knoten auf einem solchen Kreis liegt auf einem gemeinsamen Pfad mit jedem anderen Knoten des Kreises, kann also insbesondere sowohl von jedem anderen solchen Knoten lesen als auch in diesen schreiben. Wir bezeichnen die Knotenmenge auf einem solchen Kreis daher als *Informationsfluss-Äquivalenzklasse*. Das Auffinden dieser Äquivalenzklassen unterstützt die Informationsflussanalyse insofern, als die Pfadsuche in den umfangreichen IFGn realer Systeme damit wesentlich beschleunigt werden kann.

Das Beispiel in Abb. 2(b) enthält drei solcher Kreise und kann dem zufolge auf einen IFG mit nur drei Knoten reduziert werden: Die Äquivalenzklasse von *ProjectXCode*, Anna und *ProjectXBoard* bildet den ersten, die von Bernd und *SalesBoard* den zweiten und die von Chris und *SalesFlyer* den dritten Knoten.

Implementierung

Zum Auffinden unerwünschter Informationsflusspfade eines Zugriffssteuerungssystems wird ein Pfadsuchalgorithmus auf den zur ACM isomorphen IFG angewandt. Dazu wird in unserem Beispielszenario zunächst die ACM des Linux-Dateiservers extrahiert und anschließend in einen IFG transformiert.

[1] Im Sinne der Übersichtlichkeit verzichten wir auf eine Graphdarstellung der Eigenschaften Reflexivität und Transitivität der Informationsflussrelation.

Nutzer und Gruppen des Systems bilden die Subjektmenge der Matrix, Dateien und dateiartige Systemressourcen (Verzeichnisse, Sockets etc.) die Objektmenge. Die Informationen über die Subjektmenge finden sich in Konfigurationsdateien wie */etc/shadow* oder */etc/group*, die Objektmenge sowie die Inhalte der Matrixzellen ergeben sich aus einem Durchlauf des baumartigen Namensraums des Dateisystems und den individuellen Schutzinformationen in den Inodes. Die resultierende ACM wird dann gemäß der Gleichungen 1 und 2 in einen IFG überführt.

Für beide Schritte wurden Werkzeuge entwickelt, die Extraktion und Graphtransformation auf Basis eines realen Systems performant automatisieren (vgl. [AFK09, AKP14]) und sowohl auf einem Dateisystem im Live-Betrieb als auch auf ein eingefrorenes Image desselben anwendbar sind. Für die Graphtransformation und Äquivalenzklassenbildung kommen etablierte Standardalgorithmen zum Einsatz [Flo62, War62].

Informationsflussanalyse

Neben effizientem Pfadsuchen ermöglichen Informationsfluss-Äquivalenzklassen eine direkte Bestimmung des *Informationsfluss-Perimeters* jedes Subjekts oder Objekts – derjenigen Menge an Knoten, in die Informationen von einem gegebenen Subjekt oder Objekt maximal gelangen können (Privacy-Perimeter) bzw. von denen maximal Informationen zu diesem Subjekt gelangen können (Integritäts-Perimeter). In praktischen Szenarien sind diese Perimeter von zentraler Bedeutung, wenn es beispielsweise um die Vertraulichkeit sensiblen Quellcodes in einem geschützten Software-Repository oder um die Integrität von Server-Konfigurationsdateien geht; darüber hinaus lässt sich auf diese Weise unmittelbar die korrekte Implementierung von Domänen-Isolationspolitiken verifizieren (wie etwa [ST13], [GM82] oder [LS01]).

3.2 Ergebnisse

Der Dateiserver unseres Beispielszenarios wird professionell von technischen Mitarbeitern einer Universität administriert und typischerweise von Studenten und Mitarbeitern einer Informatik-Gruppe benutzt. Bei der ACM-Extraktion wurden 377.520 Objekte (Dateien und Verzeichnisse) von 40 verschiedenen Nutzern gefunden, aus denen eine Matrix mit etwa 15.000.000 Zellen erstellt wurde. Im Interesse der Handhabbarkeit des recht großen Suchraums wurde diese ACM weiter auf 7 ausgewählte Subjekte mit guten Fachkenntnissen hinsichtlich sicherer Informationssysteme reduziert. Die endgültige ACM enthielt etwa 2.643.000 Zellen, wurde auf einer Standard-Hardware (Intel Core i7 3,4 GHz CPU, 16 GB RAM) in ca. 14 Sekunden extrahiert und in weiteren 400 Sekunden in einen reduzierten IFG transformiert.

Aufgrund der sicherheitsbewussten Administration und Benutzung des Systems erwarteten wir im Ergebnis, für jeden Nutzer mit seinen jeweils eigenen Objekten (innerhalb seines *home*-Verzeichnisses) disjunkte Informationsfluss-Äquivalenzklassen vorzufinden. Zusätzlich war eine Äquivalenzklasse für Systemobjekte zu erwarten, in der sich etwa das Betriebssystem-Image, Konfigurationsdateien, Bibliotheks- und Anwendungsbinärdateien etc. wiederfinden. Tatsächlich zeigt die Informationsflussanalyse ein deutlich von den Erwartungen abweichendes Bild: neben der System-Äquivalenzklasse – welche in der Tat perfekt von Schreibzugriffen regulärer Nutzer isoliert ist – findet sich lediglich eine einzelne weitere Klasse, in der sämtliche Nutzer und ihre Dateiobjekte enthalten sind (Abb. 3). Wesentliche Ursache hierfür sind zwei Eigenschaften des untersuchten Zugriffssteuerungssystems [AFK09]:

Discretionary Access Control: Im Gegensatz zu obligatorischen Sicherheitspolitiken (*MAC, Mandatory Access Control*), welche zunehmend in modernen Betriebsystemen zu finden sind

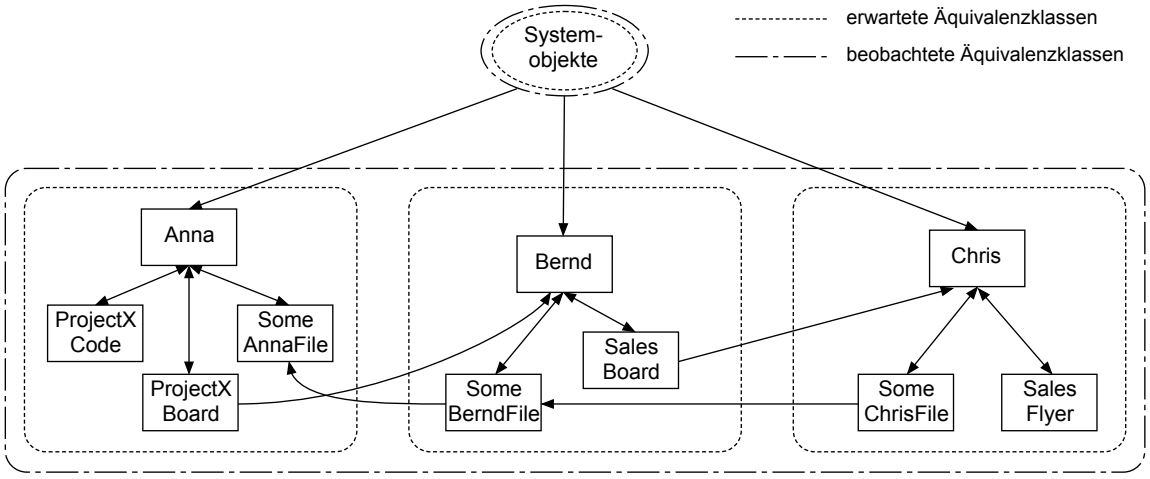


Abb. 3: Ergebnis der Informationsflussanalyse

[LS01, EK08, BHS13], lastet durch die DAC-Mechanismen des Standard-Linux-Serversystems ein wesentlicher Teil der Verantwortung für die Durchsetzung von Sicherheitseigenschaften auf individuellen Nutzern. Deren Aktionen bergen jedoch das permanente Risiko unabsichtlicher oder absichtlicher Miskonfigurationen.

Intransparente und komplexe Semantik: Sicherheitsbewusste Nutzer haben einerseits die Pflicht, Berechtigungen an ihren eigenen Objekten so zu setzen, dass Sicherheitseigenschaften gewahrt bleiben; andererseits haben selbst Nutzer mit hohem Fachwissen nicht die nötigen Informationen, um dieser Pflicht nachzukommen: die systemweiten Auswirkungen von Rechten gegenüber Gruppen (deren Mitglieder sich unbemerkt ändern können) oder gar dem „Rest der Welt“ sind nicht transparent nachvollziehbar.

Die Ergebnisse dieser Fallstudie bestätigen und konkretisieren die wohlbekannten Schwächen von DAC-Sicherheitspolitiken in Betriebssystemen: trotz professioneller Administration und sicherheitsbewussten Nutzern enthüllt die Informationsflussanalyse zahlreiche verborgene Informationsflüsse, die die Zugriffssteuerungsmechanismen des Systems aushebeln.

4 Fallstudie: Soziale Online-Netzwerke

Soziale Online-Netzwerke (*OSNs, Online Social Networks*) stellen einen der erfolgreichsten Trends im Bereich der Internet-basierten Anwendungssysteme dar. Es existiert eine Reihe von Plattformen, welche OSN-Dienste für verschiedene Zielgruppen bereitstellen – die bekanntesten Beispiele hierfür sind Facebook, Google+, Twitter, Myspace oder LinkedIn. Einige OSNs bieten in erster Linie Plattformen für soziale Kommunikation und Unterhaltung, während andere, etwa LinkedIn oder Myspace, auch oder in erster Linie als Karriereportal sowie zur beruflichen Kommunikation dienen. Da der Schutz persönlicher Informationen eines Nutzers sowie beruflicher, wirtschaftlich relevanter Informationen in solchen Systemen gleichermaßen kritisch für die Wahrung von Privacy ist, kommen hier Zugriffssteuerungskonzepte zum Einsatz, welche den bereits besprochenen in traditionellen Anwendungsdomänen höchst ähnlich sind.

Da es sich aufgrund der interaktiven Nutzungsprofile eines OSN hierbei fast ausschließlich um DAC-Mechanismen handelt, sind OSN Zugriffssteuerungssysteme bereits zum Gegenstand einer Reihe von Forschungsarbeiten geworden. Existierende Arbeiten adressieren eine Reihe si-

cherheitsrelevanter Fragestellungen hinsichtlich der praktischen Benutzung von OSNs, so etwa Anonymisierung Privacy-relevanter Informationen [CW14], Risikomodellierung [WSAL11], Data-Mining-Angriffe [GA11] oder nicht vertrauenswürdige Drittanbieter-Apps [SBL09]. Was die Analyse ihrer inhärenten Sicherheitspolitiken angeht, konzentriert sich die Forschung in der OSN-Domäne vorrangig auf Relationen-basierte Zugriffssteuerungsmodelle zur Beschreibung der sicherheitsrelevanten Beziehungen zwischen Nutzern [FS11, CPS12]. Als Ergänzung zu diesen Arbeiten ist hier unser Ziel, diese Politiken aus einem anderen Blickwinkel – dem der Informationsflussmodelle – zu betrachten. In Abschnitt 3 haben wir dargestellt, wie Miskonfigurationen typischer DAC-Mechanismen von Betriebssystemen zu uneingeschränkten verdeckten Informationsflüssen führen können. In diesem Abschnitt soll die Frage beantwortet werden, in welchem Ausmaß OSN-Systeme demselben Phänomen unterworfen sind.

4.1 Analyseansatz

Vor Betrachtung der Zugriffssteuerungskonzepte von OSNs sollen zunächst einige ihrer wesentlichen Charakteristika diskutiert werden. Anschließend werden wir eine Methode zur Informationsflussanalyse und ihre Anwendung auf ein reales OSN vorstellen.

Zwei grundlegende Charakteristika von OSNs sind zum einen das Teilen von Informationen (in Form multimedialer Inhalte) durch einen Nutzer mit einem oder mehreren anderen Nutzern (dies können bereits öffentlich zugängliche persönliche Informationen des Nutzerprofils sein), zum anderen der Ausdruck und die Pflege bestimmter Beziehungen zwischen je zwei Nutzern, welche auf Basis sozialer Interaktion zustande kommen [BE07]. In einer typischen OSN-Zugriffssteuerungssemantik sind Schreibrechte an Inhalten auf ihren jeweiligen Eigentümer (und Urheber) beschränkt, während Leserechte potenziell für alle OSN-Nutzer gelten können (in diesem Fall spricht man üblicherweise von „öffentlichem Teilen“). Zur weiteren Einschränkung von Leserechten an ihren jeweils eigenen Inhalten können Nutzer feingranulare Zugriffsregeln festlegen, welche durch das OSN-System durchgesetzt werden. Die Spezifikation dieser Regeln erfolgt üblicherweise, indem Nutzer andere, mit ihnen in Beziehung stehende Nutzer in Gruppen einteilen (im Falle von Google+ z.B. als *Kreise* bekannt), welche die konkrete Art der sozialen Beziehung ausdrücken: übliche Beispiele für Gruppen sind „Bekannte“, „Freunde“, „Kollegen“, „Familie“ etc. Gleichermaßen reflektiert das Spektrum geteilter Inhalte all diese sozialen Dimensionen und diverse Medientypen: Beispiele reichen von persönlichen Informationen wie Alter, Familienstand oder Einkommen in Textform bis hin zu Fotos und Videos von Freizeitaktivitäten, umfassen aber auch geschäftliche oder karrierebezogene Informationen wie Stellenangebote oder Kooperationsvereinbarungen.

Dieses Spektrum an Privacy-kritischen Informationen rechtfertigt eine grundlegende Analyse der gebräuchlichen OSN Zugriffssteuerungsmechanismen im Hinblick auf verdeckte Informationsflüsse. Betrachtet man die oben genannten Charakteristiken, finden sich tatsächlich einige wichtige Gemeinsamkeiten mit traditionellen DAC-Anwendungsdomänen: Subjekte (OSN Nutzer), Objekte (deren Inhalte), Eigentümer von Objekten (deren Urheber) sowie Gruppen von Subjekten (bspw. Kreise). Dies ermöglicht es, die IFG-basierte Analysemethode aus Abschnitt 3 anzuwenden, sowie ein adäquates formales OSN-Modell gefunden ist.

Modellbildung

Im Gegensatz zu Betriebssystemen bieten OSNs im Allgemeinen keine expliziten, präzise abgegrenzten Zugriffssteuerungsmechanismen wie ACLs, welche unmittelbar als Vektoren und

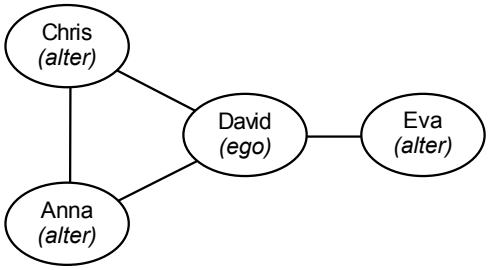


Abb. 4: Beispiel eines Ego-Netzwerks für einen Nutzer „David“

Matrizen modelliert werden können. Die entsprechenden Semantiken in OSNs sind stattdessen über die Relationen zwischen Nutzern definiert. Um diese zu modellieren, kommen in der Forschung traditionell sog. Ego-Netzwerke zum Einsatz. Ein Ego-Netzwerk ist eine grafische Repräsentation der Beziehungen einer bestimmten Bezugsperson, genannt *Ego*, mit anderen Personen (ihren *Alters*); zusätzlich beinhaltet es alle Beziehungen zwischen diesen Alters (s. Beispiel in Abb. 4)[2]. Ein Ego-Netzwerk kann als mikroskopische Sicht auf einen Teil eines sozialen Graphen angesehen werden, welcher (im vorliegenden Fall) ein vollständiges Abbild sämtlicher Beziehungen zwischen allen Nutzern eines OSNs ist (vgl. [dN12]).

Formal lässt sich ein Ego-Netzwerk durch einen Graphen $EGO = (V, E \subseteq V \times V)$ beschreiben, dessen Knoten Nutzer und dessen Kanten Beziehungen zwischen diesen Nutzern repräsentieren. Aufgrund der nicht zwangsläufig symmetrischen Natur solcher Beziehungen werden wir im Folgenden gerichtete Graphen benutzen – dies deckt sowohl den Fall symmetrischer Relationen, wie etwa „Freundschaften“ in Facebook (die von beiden Nutzern explizit bestätigt werden müssen), als auch asymmetrischer Relationen wie etwa „Folgen“ in Google+ ab. In unserer Notation repräsentiert weiterhin ein ausgezeichneter Knoten $u \in V$ Ego, während $A = V \setminus \{u\}$ die Menge seiner Alters bezeichnet, mit $a \in A \Leftrightarrow (u,a) \in E$.

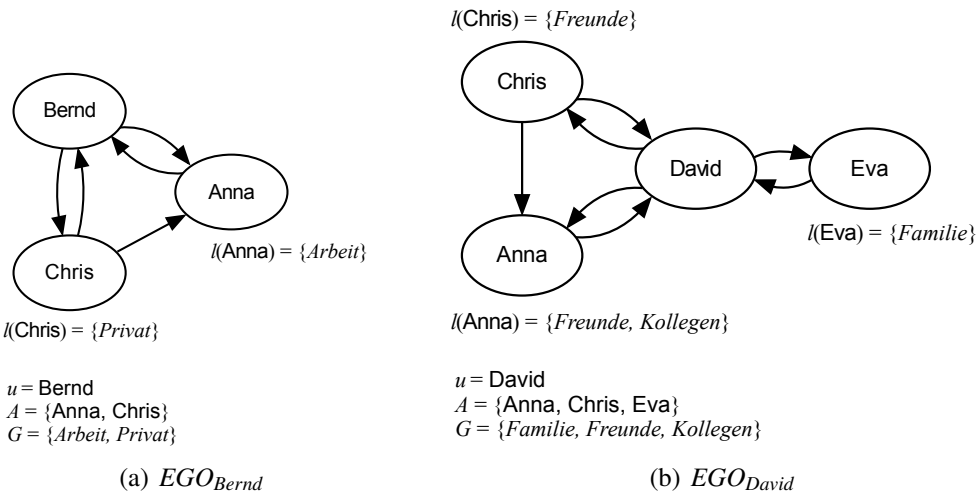
Zur Festlegung von Zugriffssteuerungsregeln kann Ego nun geteilte Inhalte für bestimmte Gruppen von Alters zugänglich machen. Dies modellieren wir durch eine Erweiterung der traditionellen Ego-Netzwerke wie folgt: Jedem Alter-Knoten $a \in A$ wird ein Label $l(a)$ zugewiesen, welches diejenigen Gruppen von Ego bezeichnet, in denen a enthalten ist. Solche Labels sind also Mengen von Gruppenbezeichnern, etwa $\{Arbeit, Familie, Freunde\}$. So erweitert definieren wir das Ego-Netzwerk eines Nutzers u als Tupel $EGO_u = (V_u, E_u, G_u, l_u)$, wobei V_u, E_u den Beziehungsgraphen, G_u eine Menge von Gruppenbezeichnern und $l_u : A_u \to 2^{G_u}$ die Label-Zuweisungsfunktion an die Alters bezeichnet. Beispielhafte Ego-Netzwerke für zwei Nutzer Bernd and David sind in Abb. 5 illustriert. Wir können nun ein OSN als Menge von Ego-Netzwerken seiner Nutzer modellieren:

$$OSN = (U, \{EGO_u | u \in U\}).$$

Um dieses Modell für ein reales Szenario zu instantiieren, stehen anonymisierte Nutzerdaten für große OSN-Anbieter wie Facebook oder Google+ öffentlich zur Verfügung. In dieser Studie wurden die anonymisierten Ego-Netzwerke eines Ausschnitts des Google+ Nutzerbestands verwendet, welche durch die *Stanford Large Network Dataset Collection*[3] gesammelt und bereit gestellt wurden [ML12].

[2] Man beachte, dass hier nur *direkte* Beziehungen gemeint sind.

[3] Verfügbar unter `http://snap.stanford.edu/data/index.html`.



(a) EGO_{Bernd} (b) EGO_{David}

Abb. 5: Zwei beispielhafte Ego-Netzwerke in formaler Notation

Transformation in Informationsflussgraphen

Da Informationen in OSNs nur zwischen Nutzern mit einer gemeinsamen Beziehung fließen können, müssen die Kanten in Ego-Netzwerken zu einem gewissen Grad denen in einem IFG entsprechend. Um folglich die in Abschnitt 3.1 beschriebene, graphbasierte Analysemethode nutzen zu können, ist zunächst eine isomorphe Transformation von Ego-Netzwerken in IFGs erforderlich (so wie zuvor von ACMs).

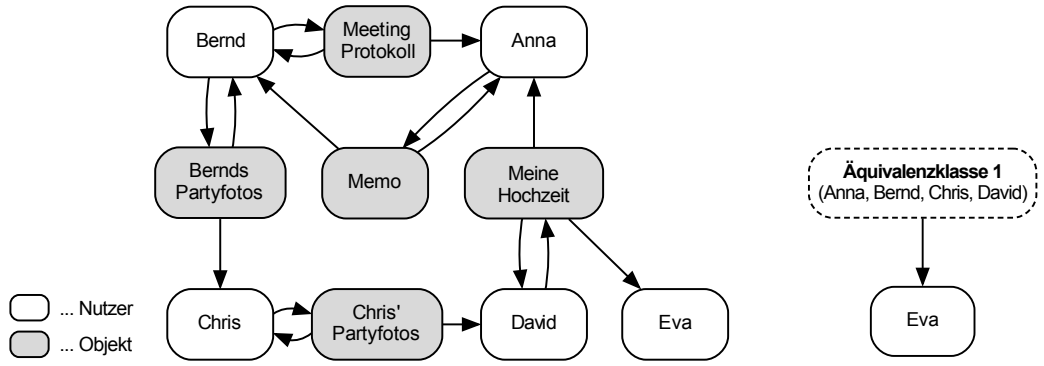
Da jede Kante zwischen einem Nutzer u und einem Alter a im Ego-Netzwerk einen potenziellen Informationsfluss von u's Objekten zu a impliziert, muss im IFG auch ein korrespondierender Pfad zwischen beiden Subjekten existieren. Da hierbei nach verschiedenen Objekttypen mit ihren jeweiligen Freigabeoptionen basierend auf der Nutzergruppe von a zu unterscheiden ist, gibt es im Allgemeinen n solcher Pfade für n mit a geteilte Objekte. Basierend auf dieser Idee kann ein Ego-Netzwerk EGO_u als gerichteter Informationsflussgraph $IFG_u = (V'_u, E'_u)$ durch einfache Kantenunterteilung wie folgt dargestellt werden: Sei $cl_u(o) = \{g_0, \ldots, g_n\}$ die Menge von Gruppen, mit denen u ein Objekt o teilt. Dann wird E'_u so konstruiert, dass für jedes geteilte Objekt o gilt

$$(o,a) \in E'_u \Leftrightarrow cl_u(o) \cap l_u(a) \neq \emptyset. \tag{3}$$

Da der Eigentümer u von o das Objekt sowohl lesen als auch schreiben darf, gilt weiterhin $\{(u,o),(o,u)\} \subseteq E'_u$. Wir bezeichnen O_u als Menge aller von u geteilten Objekte und $cl_u : O_u \to 2^{G_u}$ als Freigabe-Label eines Objekts, d.h. die Menge aller Gruppen von u, deren Mitglieder o lesen dürfen. Wie die IFG zuvor ist IFG_u bipartit, seine Kontenmenge ist daher definiert als $V'_u = V_u \cup O_u$.

Um schließlich den vollständigen sozialen Graphen eines OSN in Form eines Informationsflussgraphen $IFG = (V', E')$ darzustellen genügt es, sämtliche IFG_u durch simple Mengenvereinigung zu kombinieren: $V' = \bigcup_{u \in U} V'_u$ und $E' = \bigcup_{u \in U} E'_u$.

Abb. 6(a) zeigt einen exemplarischen Auszug aus dem IFG eines OSN, der aus dem Ego-Netzwerk in Abb. 5 transformiert wurde, mit einigen beispielhaften Objekten. Vor dem Hintergrund einer Informationsflussanalyse kann das abgebildete Szenario wie folgt interpretiert werden: Anna ist Bernds Vorgesetzte in einem Unternehmen, welches OSN-basierte soziale



(a) Voller Graph (b) Reduzierter Graph

Abb. 6: Ein möglicher IFG zu Abb. 5

Medien zur Kommunikation nicht-sensibler alltäglicher Dokumente nutzt, etwa Bernds Protokoll eines Meetings oder Annas Memo. Zugleich nutzt Bernd denselben OSN-Dienst, um private Fotos mit einigen seiner Freunde zu teilen, unter anderem Chris. Chris wiederum teilt grundsätzlich seine gesamte Fotosammlung mit den meisten seiner Freunde, darunter David. Letzterer schließlich arbeitet im Management des Unternehmens mit Anna zusammen, mit der er auch privat befreundet ist (vgl. Abb. 5).

Im beschriebenen Szenario hat nun David ein Fotoalbum seiner Hochzeit angelegt, welches er mit all seinen Freunden und Familienmitgliedern über das OSN teilt. Es ist damit auch Anna zugänglich, welche daraus an Informationen über Bernd gelangen kann (bspw. Fotos, Foto-Tags, Kommentare etc.), welche ihr gemäß der von Bernd spezifizierten Zugriffsregeln nicht zugänglich sein dürften. Damit ist wiederum, so wie bereits im Dateiserver-Szenario, ein – aus Bernds Sicht – unerwünschter Informationsfluss möglich. Wiederum hat Bernd nur begrenzte Mittel, um dieses Problem zu vermeiden: Während angenommen werden darf, dass er Chris soweit vertraut, keine seiner privaten Informationen an Anna weiterzugeben, hat er mit David keine gemeinsamen Beziehungen (reflektiert in der Abwesenheit einer gemeinsamen Kante im Ego-Netzwerk). Dennoch ist es David, welcher genau diesen unerwünschten Informationsflusspfad etabliert. Reduziert man schließlich den IFG in diesem Beispiel durch Äquivalenzklassenbildung (Abb. 6(b)), so wird deutlich, dass durch Bernd zugriffsbeschränkte Informationen letztlich sogar Nutzer erreichen können, welche ihm völlig unbekannt sind – in diesem Fall Davids Ehefrau Eva.

Im Folgenden werden wir die Analyse eines realen OSNs beschreiben, deren Ziel es ist, eine quantitative Aussage über den Einfluss unerwünschter indirekter Informationsflüsse auf die Privacy eines Nutzers zu ermöglichen. Hierfür werden Daten über Ego-Netzwerke in Google+ aus [ML12] verwendet, um den Umfang und das Wachstum von Informationsflusspfaden durch das Teilen von Inhalten zu untersuchen.

Informationsflussanalyse

Analog zur ACM in Abschnitt 3 ist in einem ersten Analyseschritt der Zustand des OSN-Zugriffssteuerungssystems zu extrahieren. Hierfür sind die verfügbaren Daten über Ego-Netzwerke der Ausgangspunkt. Da diese jedoch ausschließlich Nutzer und ihre jeweiligen Relationen berücksichtigen, nicht jedoch die von ihnen geteilten Inhalte, können daraus weder Informationen über Objekte noch ihre gruppenbasierten Zugriffsregeln gewonnen werden. Den-

noch sind die bereitgestellten Informationen zur Nutzerstruktur eines realen OSN ausreichend, um im Rahmen einer quantitativen Analyse die Evolution seines Zustands unter Annahme höchst sicherheitsbewusster Nutzer zu simulieren – welche hier wiederum im Interesse einer kritischen Analyse steht. Wir nehmen daher folgendes Nutzerverhalten an: Wann immer ein neues Objekt geteilt wird, nutzt der Eigentümer die verfügbaren Gruppen so restriktiv wie möglich; insbesondere wird niemals ein Objekt öffentlich geteilt (obwohl dies in der Praxis häufig vorkommt). Infolge dieser Annahmen betrachten wir für diese Analyse nur solche Informationsflüsse, die über mindestens einen Alter jedes Nutzers zustande kommen.

Auf Grundlage dieses Verhaltensmusters besteht der erste Analyseschritt in der Generierung eines Ausgangszustands für das OSN: Für jeden Nutzer u wird ein Objekt o_u erstellt und für genau eine Gruppe $cl_u(o_u) = \{g_u\}$ freigegeben, welche zufällig aus G_u ausgewählt wird (während A_u, G_u und l_u durch das Google+ Ego-Netzwerk vorgegeben sind). Da hier ein System zum Datenaustausch modelliert wird, erscheint es zulässig anzunehmen, dass jeder teilnehmende Nutzer zumindest ein Objekt teilt. Zugleich setzen wir die Annahme sicherheitsbewusster Nutzer um, indem die maximale Anzahl geteilter Objekte zu einem Zeitpunkt minimal gehalten wird.

Im zweiten Schritt, nach der Generierung eines OSN-Zustands, wird der IFG wie oben beschrieben konstruiert und soweit wie möglich auf Äquivalenzklassen reduziert (wie in Abschnitt 3.1 beschrieben). Nachdem auf diese Weise der Privacy-Perimeter jedes Nutzers bestimmt wurde, fahren wir iterativ mit der Generierung und Analyse von Folgezuständen fort, bei der jeweils neue Objekte oder Freigaben existierender Objekte (beides maximal einmal pro Nutzer) hinzu kommen. Die Anzahl der Iterationen unserer Simulation ist hinsichtlich des entstehenden IFG vergleichbar mit der Dichte einer ACM-Belegung in der ersten Fallstudie.

Zur Interpretation der Simulationsergebnisse schließlich definieren wir Analyseziele basierend auf den angenommenen Intentionen unserer OSN-Nutzer (und deren möglicher Verletzung). Solche Verletzungen identifizieren wir basierend auf der Annahme, dass ein Nutzer grundsätzlich nicht beabsichtigt, geteilte Objekte anderen Nutzern zugänglich zu machen, welche sich in keiner seiner Gruppen befinden; dies kann quantifiziert werden durch die durchschnittliche Anzahl unbekannter Nutzer, welche die Informationen eines geteilten Objekts maximal erreichen können.

4.2 Ergebnisse

Für unsere Fallstudie benutzen wir einen Google+ Datensatz von 131 Ego-Netzwerken und insgesamt 468 Gruppen. Diesen haben wir, nach Transformation in einen IFG, in drei Zuständen untersucht: nach einer, zwei und drei Iterationen der oben beschriebenen Simulation. Berücksichtigt man die Größe eines OSN (selbst auf dem benutzten Datensatz), so ist zu erwarten, dass eine signifikante Teilmenge der Nutzer von potenziell unerwünschte Informationsflüssen in oben beschriebenem Sinne betroffen ist – wenn zunächst auch vorwiegend durch kurze Pfade von nur wenigen unbekannten Nutzern. Dabei sollten bei dem angenommenen Verhaltensmuster dennoch die meisten Nutzer in einer eigenen Äquivalenzklasse zusammen mit ihren eigenen Objekten isoliert sein, vergleichbar mit den *home*-Verzeichnissen im Szenario aus Abschnitt 3.2.

Dem gegenüber zeigen die Analyseergebnisse (Tabelle 1), dass auch diese letztgenannte Annahme in der Praxis nicht haltbar ist: Nach nur zwei Iterationen der Simulation sind bereits ausreichend viele indirekte Informationsflusspfade etabliert, um ein Viertel aller Nutzer in derselben Äquivalenzklasse kollabieren zu lassen; nach nur einer weiteren Iteration betrifft dies bereits über 37%. Auch hinsichtlich einzelner unerwünschter Informationsflusspfade sind die

Tab. 1: Ergebnisse der quantitativen Informationsflussanalyse eines Google+ Datensatzes

Iteration	1	2	3
Anzahl unbekannter Nutzer auf längstem Pfad	7	10	10
Durchschnittliche Anzahl unbekannter Nutzer auf allen Pfaden	2,23	3,3	3,19
Anteil von Nutzern in der größten Äquivalenzklasse	3%	25,9%	37,4%

Ergebnisse als ernüchternd anzusehen: Nach der ersten Iteration haben Pfade, welche zu unbekannten Nutzern führen, bereits eine länge von mehr als 2 – reichen jedoch bis hin zu 7 unbekannten Nutzern. Dies bedeutet, dass bereits das erste Objekt, welches ein Nutzer mit irgendeinem seiner Alters teilt, im Durchschnitt für mehr als zwei weitere, ihm völlig unbekannte Nutzer sichtbar wird. Dabei soll nochmals erwähnt werden, dass zum Zwecke dieser Studie nur ein Bruchteil des tatsächlichen IFG eines OSN verwendet, sowie ein für reale Bedingungen extrem rigoroses Nutzerverhalten angenommen wurde.

4.3 Praktische Konsequenzen

Die Ergebnisse dieser Fallstudie stützen die Schlussfolgerung, dass typische OSN-Systeme ähnliche Schwachstellen hinsichtlich unerwünschter Informationsflusspfade aufweisen wie die bekannten, traditionellen DAC-Anwendungen. Trotz dieser ernüchternden Erkenntnis könnten moderne OSN-Systeme von einer Integration der beschriebenen Analysemethode in ihre Zugriffssteuerungsmechanismen profitieren.

Die Idee hierbei ist, das wohlbekannte Paradigma der Referenzmonitorarchitekturen auszunutzen. Zwar wurde dieses ursprünglich zur Durchsetzung obligatorischer Sicherheitspolitiken (*MAC*) entwickelt, kann aber auch bei DAC zur Überwachung von Zugriffen zur Laufzeit eingesetzt werden. Aus Implementierungssicht besteht die Architektur aus einer zentralen Instanz zur Entscheidungsfindung (*PDP, Policy Decision Point*) und einer Anzahl Interzeptoren (auch *PEP, Policy Enforcement Point*), welche die Autorisation für jeden durch sie überwachten Zugriff beim PDP anfragen. In der Betriebssystem-Domäne, in der dieses Paradigma seit seiner ersten praktischen Implementierung [SSL+99] zunehmend eingesetzt wird, unterstützen derartige Architekturen seit einiger Zeit bereits Echtzeit-Informationsflussanalysen [EGC+10, NT13].

Um das Referenzmonitorparadigma für die Informationsflussüberwachung von OSN-Systemen anzupassen, muss die entsprechende Analysemethode in Form eines Middleware-basierten PDP implementiert werden. Zu diesem Zweck werden mehrere PEPs der Backend-Funktionalität zum Teilen von Inhalten vorgeschaltet (um feingranulare Zugriffssteuerung für verschiedene Inhaltstypen wie Bilder, Blogposts, Profilinformationen etc. zu unterstützen). Eine graphbasierte Informationsfluss-Datenbank wird immer dann durch den PDP aktualisiert, wenn ein Objekt geteilt bzw. eine Freigabe verändert wurde; zugleich wird das Ergebnis einer quantitativen Informationsflussanalyse auf dem aktualisierten Graphen an den anfragenden PEP zurückgegeben. Auf diese Weise ist die OSN-Oberfläche in der Lage, dem Nutzer konkrete Informationen über die Folgen seiner beabsichtigten Aktion zur Verfügung zu stellen, vergleichbar mit den Angaben in Tabelle 1. Ein solcherart modifiziertes OSN-System kann die Bedingungen, unter denen DAC-Nutzer Zugriffsregeln für ihre Informationen festlegen, wesentlich transparenter gestalten.[4]

[4] Dabei sind freilich Datenschutzinteressen hinsichtlich der etwaig offengelegten Aktivitäten anderer Nutzer abzuwägen; dieses Problem muss bei einer praktische Umsetzung der hier skizzierten Lösung berücksichtigt werden.

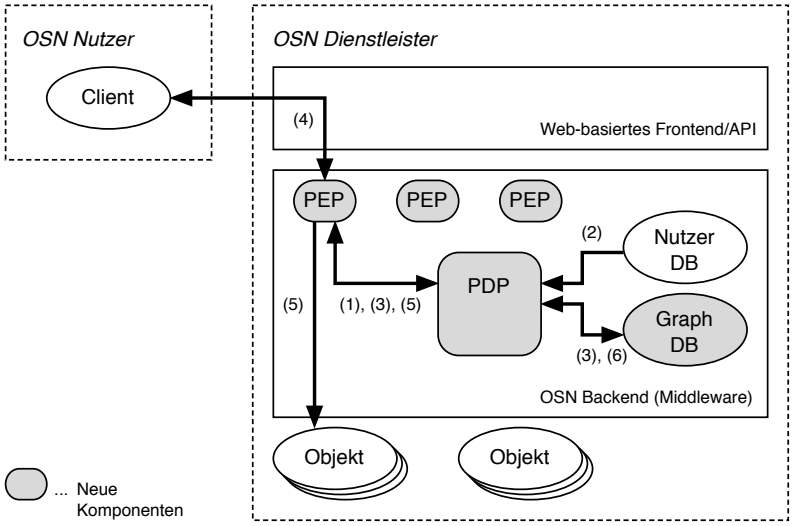


Abb. 7: Sicherheitsarchitektur eines Informationsfluss-Monitoring-Systems

Implementierung

Basierend auf der IFG-Konstruktionsregel (Gleichung (3)) müssen alle Nutzeroperationen, welche entweder cl_u oder l_u modifizieren, eine PEP-Interaktion auslösen. Grundsätzlich sind dies:

- *share(o, g)* und *unshare(o, g)* zum Freigeben bzw. Sperren eines Objekts o für den Lesezugriff durch Alters in Gruppe g.
- *addgroup(a, g)* und *rmgroup(a, g)* zum Hinzufügen bzw. Entfernen eines Alters a zu bzw. aus einer Gruppe g.

Man beachte, dass Änderungen am globalen IFG vom PDP nur vorläufig berechnet werden und erst nach expliziter Bestätigung des jeweiligen Nutzers für andere Nutzer beobachtbar sein dürfen; jede Ausführung einer der obigen Operationen hat daher transaktionalen Charakter.

In der Praxis würde ein solches System den Aufruf von *share(o, g)* wie folgt behandeln (vgl. Abb. 7):

1. Der zuständige PEP leitet den Aufruf an den PDP weiter.
2. Der PDP greift auf das Ego-Netzwerk und Informationen über die Objekte von u zu (all dies ist Teil der Standard-OSN-Nutzerdatenbank), um zu bestimmen ob eine neue Kante im IFG erzeugt werden muss.
3. Der PDP aktualisiert vorläufig die Graph-Datenbank und führt analysiert diese auf Informationsflüsse. Die Ergebnisse werden zusammen mit einer frischen Transaktions-ID für die Ausführung der Operation an den PEP returniert.
4. Durch ein spezielles Frontend werden die Analyseergebnisse dem Nutzer präsentiert, welcher die Operation explizit bestätigen muss.
5. Unter Nutzung der Transaktions-ID leitet der PEP die Nutzerentscheidung an den PDP weiter. Sofern es sich um eine Bestätigung handelt, löst er zugleich die eigentlichen Mechanismen zum Teilen des Objekts im OSN Backend aus.
6. Abhängig von der Nutzerentscheidung bestätigt der PDP entweder die Aktualisierung des globalen IFG oder verwirft sie.

Zum effizienten Management im Falle eines umfangreichen, realen OSN kann es in der Praxis erforderlich sein, den zentralisierten PDP auf mehrere nutzerspezifische PDPs aufzuteilen – jeder davon verantwortlich für einen Teilgraph IFG_u des globalen IFG. Schritt 3 beinhaltet dann zusätzlich einen verteilten Algorithmus zur Graphsynchronisation zwischen mehreren beteiligten PDPs, basierend etwa auf wohlbekannten Lösungen für die Verklemmungsanalyse auf verteilten Ressourcengraphen (vgl. [CMH83]). Aus technischer Sicht könnte eine Implementierung dieses Systems auf einer verbreiteten Web Services Middleware wie Apache Axis2 [PHE+06] und einem graphbasierten Datenbanksystem wie Neo4j [Neo14] basieren.

5 Zusammenfassung

Dieses Papier beleuchtet die prinzipbedingten Schwachstellen moderner Zugriffssteuerungssysteme hinsichtlich indirekter Informationsflüsse, welche sich neben den wohlbekannten Beispielen aus der traditionellen Betriebssystemdomäne auch in der Domäne sozialer Online-Netzwerke finden: hier stellen verdeckte Informationsflüsse insbesondere ein Risiko für die Privacy einzelner Nutzer dar. Zum Beleg dieser Aussage stellen wir eine Methode, Werkzeuge sowie Ergebnisse einer graphbasierten Informationsflussanalyse in beiden Domänen vor. Insbesondere werden die spezifischen Eigenschaften einer DAC-Sicherheitspolitik als Ursache der beobachtbaren Schwachstellen identifiziert.

Als praktische Konsequenz aus diesen Erkenntnissen nutzen wir die Methode zur Informationsflussanalyse für soziale Online-Netzwerke, um Empfehlungen für die Implementierung eines Informationsfluss-Monitoring-Systems zu geben. In Anlehnung an den Erfolg der MAC-Politiken in Betriebssystemen kann dieses dazu dienen, Nutzerentscheidungen hinsichtlich der Privacy-kritischen Zugriffsregeln zu unterstützen, ohne hierbei die grundsätzlichen DAC-Charakteristika dieser Anwendungsdomäne infrage zu stellen.

Literatur

[AFK09] Peter Amthor, Anja Fischer, and Winfried E. Kühnhauser. Analyse von Zugriffssteuerungssystemen. In Patrick Horster and Peter Schartner, editors, *D·A·CH Security 2009*, pages 49–61. syssec Verlag, 2009.

[AKP11] Peter Amthor, Winfried E. Kühnhauser, and Anja Pölck. Model-based Safety Analysis of SELinux Security Policies. In P. Samarati, S. Foresti, J. Hu, and G. Livraga, editors, *In Proc. of 5th Int. Conference on Network and System Security*, pages 208–215. IEEE, 2011.

[AKP14] Peter Amthor, Winfried E. Kühnhauser, and Anja Pölck. WorSE: A Workbench for Model-based Security Engineering. *Computers & Security*, 42(0):40–55, 2014.

[BE07] Danah M. Boyd and Nicole B. Ellison. Social Network Sites: Definition, History, and Scholarship. *Journal of Computer-Mediated Communication*, 13(1):210–230, 2007.

[BHS13] Sven Bugiel, Stephan Heuser, and Ahmad-Reza Sadeghi. Flexible and Fine-Grained Mandatory Access Control on Android for Diverse Security and Privacy Policies. In *22nd USENIX Security Symposium (USENIX Security '13)*. USENIX, August 2013.

[BL73] D. Elliott Bell and Leonard J. LaPadula. Secure Computer Systems: Mathematical Foundations (Vol.I). Technical Report AD 770 768, MITRE, Bedford, Massachusetts, Nov 1973.

[CH04] Roderick Chapman and Adrian Hilton. Enforcing Security and Safety Models with an Information Flow Analysis Tool. In *Proc. 2004 Annual ACM SIGAda International Conference on Ada*, SIGAda '04, pages 39–46, New York, NY, USA, 2004. ACM.

[CMH83] K. Mani Chandy, Jayadev Misra, and Laura M. Haas. Distributed Deadlock Detection. *ACM Trans. Comput. Syst.*, 1(2):144–156, May 1983.

[CPS12] Yuan Cheng, Jaehong Park, and Ravi Sandhu. A User-to-User Relationship-based Access Control Model for Online Social Networks. In *Proc. 26th Annual IFIP WG 11.3 Conference on Data and Applications Security and Privacy*, DBSec'12, pages 8–24, Berlin, Heidelberg, 2012. Springer-Verlag.

[CW14] Wei Chang and Jie Wu. Strengthening Data Privacy During Propagation. In *Proc. ACM MobiCom Workshop on Security and Privacy in Mobile Environments*, SPME '14, pages 37–42, New York, NY, USA, 2014. ACM.

[dN12] Wouter de Nooy. Graph Theoretical Approaches to Social Network Analysis. In Robert A. Meyers, editor, *Computational Complexity*, pages 2864–2877. Springer New York, 2012.

[EGC+10] William Enck, Peter Gilbert, Byung-Gon Chun, Landon P. Cox, Jaeyeon Jung, Patrick McDaniel, and Anmol N. Sheth. TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones. In *Proc. 9th USENIX Conference on Operating Systems Design and Implementation*, OSDI '10, pages 1–6, Berkeley, CA, USA, 2010.

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

[Flo62] Robert W. Floyd. Algorithm 97: Shortest Path. *CACM*, 5(6):345, June 1962.

[FS11] Philip W.L. Fong and Ida Siahaan. Relationship-based Access Control Policies and Their Policy Languages. In *Proc. 16th ACM Symposium on Access Control Models and Technologies*, SACMAT '11, pages 51–60, New York, NY, USA, 2011. ACM.

[GA11] Daniel Gayo-Avello. All Liaisons Are Dangerous when All Your Friends Are Known to Us. In *Proc. 22nd ACM Conference on Hypertext and Hypermedia*, HT '11, pages 171–180, New York, NY, USA, 2011. ACM.

[GM82] J.A. Goguen and J. Meseguer. Security Policies and Security Models. In *Proc. IEEE Symposium on Security and Privacy*, pages 11–20. IEEE, April 1982.

[HRU76] Michael A. Harrison, Walter L. Ruzzo, and Jeffrey D. Ullman. Protection in Operating Systems. *Communications of the ACM*, 19(8):461–471, August 1976.

[JHL02] Xiaodong Jiang, Jason I. Hong, and James A. Landay. Approximate Information Flows: Socially-Based Modeling of Privacy in Ubiquitous Computing. In *Proc. 4th International Conference on Ubiquitous Computing*, UbiComp '02, pages 176–193, London, UK, UK, 2002. Springer-Verlag.

[KG13] Dennis Kafura and Denis Gracanin. An Information Flow Control Meta-model. In *Proc. 18th ACM Symposium on Access Control Models and Technologies*, SACMAT '13, pages 101–112, New York, NY, USA, 2013. ACM.

[Lam74] Butler W. Lampson. Protection. *ACM SIGOPS 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 *Proc. 2001 Ottawa Linux Symposium*, 2001.

[ML12] Julian J. McAuley and Jure Leskovec. Learning to Discover Social Circles in Ego Networks. In Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, Léon Bottou, and Kilian Q. Weinberger, editors, *Neural Information Processing Systems*, NIPS '12, pages 548–556, 2012.

[MLCJ11] Ziqing Mao, Ninghui Li, Hong Chen, and Xuxian Jiang. Combining Discretionary Policy with Mandatory Information Flow in Operating Systems. *ACM Transactions on Information and System Security*, 14(3):24:1–24:27, November 2011.

[Neo14] Neo Technology. Neo4j Graph Database. `http://neo4j.com/`, 2014.

[NT13] Ben Niu and Gang Tan. Efficient user-space information flow control. In *Proc. 8th ACM SIGSAC Symposium on Information, Computer and Communications Security*, ASIA CCS '13, pages 131–142, New York, NY, USA, 2013. ACM.

[PHE+06] Srinath Perera, Chathura Herath, Jaliya Ekanayake, Eran Chinthaka, Ajith Ranabahu, Deepal Jayasinghe, Sanjiva Weerawarana, and Glen Daniels. Axis2, Middleware for Next Generation Web Services. In *Proc. 2006 IEEE International Conference on Web Services*, ICWS '06, pages 833–840, Washington, DC, USA, Sept 2006. IEEE Press.

[San92] Ravi S. Sandhu. The Typed Access Matrix Model. In *Proc. 1992 IEEE Symposium on Security and Privacy*, SP '92, pages 122–136, Washington, DC, USA, 1992. IEEE.

[SBL09] Kapil Singh, Sumeer Bhola, and Wenke Lee. xBook: Redesigning Privacy Control in Social Networking Platforms. In *Proc. 18th Conference on USENIX Security Symposium*, SSYM'09, pages 249–266, Berkeley, CA, USA, 2009. USENIX Association.

[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. 8th USENIX Security Symposium*, 1999.

[ST13] Alireza Sharifi and Mahesh V. Tripunitara. Least-restrictive Enforcement of the Chinese Wall Security Policy. In *Proc. 18th ACM Symposium on Access Control Models and Technologies*, SACMAT '13, pages 61–72, New York, NY, USA, 2013. ACM.

[War62] Stephen Warshall. A Theorem on Boolean Matrices. *Journal of the ACM*, 9(1):11–12, January 1962.

[WSAL11] Ting Wang, Mudhakar Srivatsa, Dakshi Agrawal, and Ling Liu. Modeling Data Flow in Socio-information Networks: A Risk Estimation Approach. In *Proc. 16th ACM Symposium on Access Control Models and Technologies*, SACMAT '11, pages 113–122, New York, NY, USA, 2011. ACM.

[XRAG13] Xing Xie, Indrakshi Ray, Raman Adaikkalavan, and Rose Gamble. Information Flow Control for Stream Processing in Clouds. In *Proc. 18th ACM Symposium on Access Control Models and Technologies*, SACMAT '13, pages 89–100, New York, NY, USA, 2013. ACM.

[ZLN05] Xinwen Zhang, Yingjiu Li, and Divya Nalla. An Attribute-based Access Matrix Model. In *Proc. 2005 ACM Symposium on Applied Computing*, SAC '05, pages 359–363, New York, NY, USA, 2005. ACM.