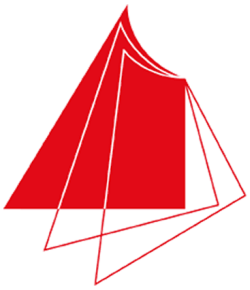

Hochschule Karlsruhe
Technik und Wirtschaft
UNIVERSITY OF APPLIED SCIENCES

Log Anonymization

Christian Kolodziej
Hochschule Karlsruhe - Technik und Wirtschaft
University of Applied Sciences
Moltkestraße 30, D-76133 Karlsruhe

7. Juli 2009

Inhaltsverzeichnis

Deutlich öfter als früher erscheint das Thema Datenschutz heutzutage in den Medien und der öffentlichen Diskussion. Gründe liegen sicherlich in den neuerlichen offenkundigen Datenskandalen bei großen Unternehmen wie der Deutschen Telekom oder der Deutschen Bahn, aber auch in der zunehmenden Durchdringung unseres Alltags durch IT-Systeme, die in großem Maße Daten erfassen. Diese sog. Logdaten (engl. *to log* - aufzeichnen) enthalten Informationen über Vorgänge des realen Lebens beispielsweise bei der Nutzung von Software aber auch der Benutzung von Transportmitteln oder Zugangsanlagen mit IT-technischer Unterstützung.

Man spricht in diesem Zusammenhang auch von personenbezogen oder sensitiven Daten, also solchen Informationen, die mit mehr oder weniger hohem Aufwand einer Person eindeutig zugeordnet werden können. Auch wenn die Rechtsprechung zum Thema IP-Adressen noch nicht eindeutig ist und sich die Gerichte teilweise noch widersprechen, so ist doch davon abzusehen, dass auch IP-Adresse langfristet auch als personenbezogenen Daten gelten. Da auf diese Weise auch eine ständige Überwachung möglich wäre stellen schon das deutsche Grundgesetz im Allgemeinen sowie das Bundesdatenschutzgesetz (BDSG) im Speziellen strenge Richtlinien auf, wenn es um den Umgang mit Personendaten und der Zuordenbarkeit von Daten zu Personen geht. Das BDSG regelt den Umgang aller datenschutzrechtlich relevanten Handlungen, dazu zählen auch die Erhebung, die Verarbeitung sowie die Nutzung personenbezogener Daten.

Die vorliegende Ausarbeitung soll sich dabei speziell auf das Thema der IP-Adresse beschränken. Mit der zunehmenden Selbstverständlichkeit von Internet-Anwendungen wird diese Information ständig und unbemerkt gespeichert. Zunächst stellt sich die Frage, inwiefern dies mit deutschen Recht vereinbar ist. Die Gefahren, die von diesen personenbezogenen Daten ausgehen, sollen anschließend am Beispiel von Google veranschaulicht werden bevor abschließend gezeigt wird, wie sich mit Hilfe bestimmter Tools durch Logdaten-Anonymisierung (engl. *log anonymization*) Probleme mit dem deutschen Datenschutz vermeiden lassen.

Kapitel 1

Rechtsgrundlagen beim Umgang mit personenbezogen Daten

In Deutschland gilt juristisch gesehen das Prinzip des Verbots mit Erlaubnisvorbehalt. Für die Speicherung personenbezogener Daten bedeutet das, dass diese erst und nur nach Einwilligung des betreffenden Benutzers erlaubt ist. Dies ist jedoch speziell bei IT-Systemen oft nicht so einfach möglich, da hier jeder Zugriff mitprotokolliert wird, so auch der erste Aufruf eines Nutzers, bevor dieser der Speicherung personenbezogener Daten zustimmen oder ihr widersprechen konnte.

Nach erteilter Zustimmung dürfen personenbezogene Daten zwar erhoben und gespeichert werden, allerdings sind auch hierbei weitere Auflagen zu beachten. Zum einen darf die Speicherung nur die für die jeweilige Aufgabe nötigen Daten umfassen. Beim Online-Shopping ist offensichtlich, dass die Postadresse obligatorisch ist. Während beim Geburtsdatum schon Zweifel an der Erforderlichkeit angebracht sind, so sind Augenfarbe oder Hobbys offensichtlich nicht erforderlich, ihre Speicherung also unrechtmäßig. Auch darf es rechtlich keine Pflicht sein die Kontoverbindung anzugeben wenn man per Rechnung oder Vorkasse zahlt während dies beim Lastschriftverfahren wiederum zwingend erforderlich ist. Darüber hinaus dürfen die erhoben Daten nur zweckgebunden verwendet werden. Im Falle des geschilderten Online-Händlers ist dies die Abwicklung der Bestellung sowie auch im weiteren Sinne das Anbieten spezieller Angebote aufgrund des früheren Kaufverhaltens. Nicht zulässig wäre natürlich die Weitergabe der Adressdaten oder Kauftransaktionen an Dritte. Müssen Daten dennoch außer Haus gegeben werden, so darf dies ausschließlich in anonymisierter Form geschehen. Die Details zum Thema Logdaten-Anonymisierung finden sich in Kapitel 3.

1.1 Gesetz zur Vorratsdatenspeicherung

Großes Medienecho fand das am 9. November 2007 erlassene Gesetz zur Vorratsdatenspeicherung. Bis Ende 2007 durften Telekommunikationsanbieter wie im Kapitel zuvor beschrieben nur die für die Abrechnung nötigen Daten speichern, die Kunden hatten zudem das Recht, dass alle diese Informationen nach Rechnungsstellung wieder gelöscht werden.

Mit dem neuen Gesetz sollen die Firmen die Daten nun in zunehmenden Umfang und längerfristig speichern, damit Strafverfolgungsbehörden im Fall der Fälle auf diese Daten zurückgreifen können. Auch wenn diese Daten nur in sehr begrenztem Maße und auch nur auf richterlichen Beschluss von einigen wenigen Behörden verwendet werden dürfen, so stellt doch allein das Vorhandensein ein potentielles Risiko dar. Nicht zu Unrecht sieht der Arbeitskreis Vorratsdatenspeicherung[1] darin einen unverhältnismäßigen Eingriff in die Privatsphäre sowie eine Beeinträchtigung bestimmter Berufsgruppen und hat am 31. Dezember 2007 Verfassungsbeschwerde eingereicht.

Aus rein technischer Sicht ist es nun möglich nachzuvollziehen, welche Benutzer zu welcher Zeit sich auf welchen Internetseiten bewegt hat, da der Anbieter persistent gespeichert hat welche IP-Adresse zum fraglichen Zeitpunkt welchem Benutzer zugewiesen war. Was datenschutzrechtlich den Super-GAU und in der Tat einen massiven Eingriff in die Privatsphäre darstellt, hat der deutsche Gesetzgeber damit quasi für die eigenen Zwecke legalisiert während es beispielsweise von den Datenschutzbeauftragten Schlewig-Holsteins im Juli 2008 etliche Betreiber von Internetseiten die Nutzung der Analyse-Software Google Analytics[2] untersagt wurde. Welche Problematik von Google Analytics ausgeht soll das folgende Kapitel zeigen.

[1] `http://www.vorratsdatenspeicherung.de`
[2] `http://analytics.google.com`

Kapitel 2

Google und der Datenschutz

Neben dem ursprünglichen Suchmaschinengeschäft bietet der Internet-Gigant Google[1] aus dem amerikanischen Mountain View (Kalifornien) mittlerweile zahlreiche weitere Dienste an, darunter u.a. den eMail-Dienst Google Mail[2], den Kalender Google Calendar[3] sowie die bereits angesprochene Online-Applikation Google Analytics zur Analyse der Zugriffe und des Besucherverhaltens auf der eigenen Internetseite. Während die Nutzer unentgeltlich ein riesiges Funktionsspektrum bekommen, laufen Datenschützer Sturm.

Der Grund liegt wieder in der Speicherung personenbezogener Daten, die in diesem Fall zudem auf Servern geschieht, die außerhalb Deutschlands stehen Dies wird zumindest angenommen, da Google keine Informationen über die genauen Standorte ihrer Server gibt.. Neben der IP-Adresse eines Besuchers werden auch seine Suchanfragen, eMails und Termine gespeichert und Google ist so in der Lage zwischen allen diesen Daten eine Verbindung herzustellen. Was bei Nutzung mehrerer Google-Dienste für den Benutzer zunächst einen Vorteil darstellt Stichwort Single-Sign-On ist datenschutzrechtlich bedenklich weil Google so bei jeder Suchanfrage Werbung oder andere benutzerspezifische Inhalte einzublenden.

Für Google selber sind diese Informationen Grundlage für ihre Einnahmen, die sich zu nahezu 100 % aus dem Verkauf von Werbeanzeigen ergeben. Google ist bei jeder Suchanfrage in der Lage die passendste Werbung einzublenden, die zugleich die höchste Erfolgschancen garantiert und den eigenen Verdienst in die Höhe

[1] `http://www.google.com`
[2] `http://mail.google.com`
[3] `http://calendar.google.com`

treibt. Offensichtlich handelt es sich hierbei nicht um eine zweckgebundene Nutzung von privaten bei Google gespeicherten eMails und Terminen. Auch wenn Google stets betont, dass das System vollkommen transparent für die Werbekunden funktioniert, so bleibt doch ein gewisses Restrisiko was den Zugang Dritter zu persönlichen und personenbezogen Daten angeht, die bei Google zuhauf gespeichert sind - sei es aus Profitstreben von Google oder aufgrund von Sicherheitslücken in den Systemen. Ebenso wenig nachkontrollieren lässt sich, ob Google auch wirklich nach 90 Tagen personenbezogen Daten anonymisiert, wie den Konzern verlauten lässt. Auch musst Google mittlerweile zugeben, bereits Nutzerdaten an die US-Regierung weitergegeben zu haben (siehe [Win08]).

Kapitel 3

Logdaten-Anonymisierung

Wie zu sehen ist, ist die Speicherung von IP-Adresse datenschutzrechtlich bedenklich, sicherheitstechnisch kritisch und rechtlich gesehen bewegen sich die Betreiber von Internetanwendungen und Servern mangels letztinstanzlicher und stattdessen sich widersprechender Urteile in einer Grauzone. Das Dilemma besteht darin, dass die Daten einerseits benötigt werden um die Nutzung der Internetseiten zu analysieren und diese auf Basis der gewonnenen Erkenntnisse weiterzuentwickeln und zu verbessern, ihre Erhebung aber ein rechtliches Risiko darstellt, auch wenn es (noch?) sehr unwahrscheinlich sein dürfte, deshalb in Konflikt mit der deutschen Rechtssprechung zu geraten. Webmaster und Betreiber, die dennoch auf Nummer sicher gehen wollen haben nun die Möglichkeit komplett auf Logdaten zu verzichten oder die unter Zuhilfenahme von Logdaten-Anonymisierung weiterhin zu nutzen.

3.1 Logdaten-Anonymisierung in der Theorie

Jede Logdaten-Anonymisierung muss sicherstellen, dass die Orginaldaten aus den anonymisierten Daten nicht wiederhergestellt werden können, die Abbildung darf demnach nicht umkehrbar sein. Beispielhaft soll die folgende Tabelle 3.1 betrachtet werden, die als sensitive Information die Augenfarbe der Personen beinhaltet. Ein solche Tabelle könnte beispielsweise für statistische Auswertungen benötigt werden, wie sie etwa das Statistische Bundesamt veröffentlicht.

Nun könnte man argumentieren, dass die Information über die eigene Augenfarbe für Dritte nicht sehr wertvoll sei. Enthielte die Tabelle nun aber anstatt der Au-

genfarbe Krankheiten, so offenbart sich die Notwendigkeit der Anonymisierung.

Name	**Geburtstag**	**Geschlecht**	**Geburtsort**	**Augenfarbe**
Klaus Müller	17.04.1955	M	76227	blau
Martin Schneider	31.07.1955	M	76228	grau
Dietmer Ludwig	17.01.1955	M	76227	blau-grau
Karl Schütz	05.07.1961	M	76133	braun
Konrad Fischer	31.12.1961	M	76139	blau-grün
Heike Schäfer	05.07.1963	W	76133	grün
Elke Friedrich	31.10.1963	W	76131	grün

Tabelle 3.1: **Tabelle mit Informationen zur Augenfarbe von Personen**

Der erste Schritt ist die Entfernung der Namen, sodass wir die Tabelle 3.2 erhalten. Trotzdem ist es immer noch möglich jeder Person eindeutig die Augenfarbe zuzuordnen insofern man Details über den Geburtstag, das Geschlecht sowie die Geburtsort kennt. Diese drei Merkmale fungieren aus diesem Grunde als Quasi-Identifier Q_t. Aspekte des Datenschutzes werden weiterhin berührt.

Geburtstag	**Geschlecht**	**Geburtsort**	**Augenfarbe**
17.04.1955	M	76227	blau
31.07.1955	M	76228	grau
17.01.1955	M	76227	blau-grau
05.07.1961	M	76133	braun
31.12.1961	M	76139	blau-grün
05.07.1963	W	76133	grün
31.10.1963	W	76131	grün

Tabelle 3.2: **Anonymisierte Tabelle mit Informationen zur Augenfarbe von Personen**

Um die Daten unter Bewahrung ihrer statistischen Eigenheiten trotzdem veröffentlichen zu können, gibt es in der Theorie die Prinzipien der *k*-Anonymity, *l*-Diversity sowie *t*-Closeness.

3.1.1 *k*-Anonymity

Anonymität bezeichnet der Grad der Geheimhaltung einer speziellen Identität innerhalb einer Gruppe von Identitäten. Die *k*-Anonymity „verhindert die Reindizierung einer Person in einem Datenbestand durch Anonymisierung dieser in einer

Gruppe von anderen Personen des Datenbestandes“ [Hau07]. Unter Angabe von *k* ist gewährleistet, dass es mindestent *k*-1 Instanzen mit derselben Wertkombionation gibt. Je größer *k*, desto größer die Gruppe und desto besser die Anonymität des Individuums.

Geburtstag	**Geschlecht**	**Geburtsort**	**Augenfarbe**
..1955	M	7622*	blau
..1955	M	7622*	grau
..1955	M	7622*	blau-grau
..1961	W	7613*	braun
..1961	W	7613*	blau-grün
..1963	W	7613*	grün
..1963	W	7613*	grün

Tabelle 3.3: ***k*=2-anonymisierte Tabelle mit Informationen zur Augenfarbe von Personen**

Die Tabelle 3.3 ist nun *k*-anonym mit *k*=2. Der Quasi-Identifier Q_t = {Geburtsdatum, Geschlecht, PLZ} teilt den Datenbestand nun in drei Blöcke. Für jedes mögliche Tupel gibt es nun mindestens zwei Datensätze, die eindeutige Zuordnung zu einem bestimmten Individuum wird dadurch verhindert.

Allerdings ist die Anonymität immer noch gefährdet wennbeispielsweise auch die Tabelle 3.4 mit einer anderen *k*-Anonymisierung bei gleicher Sortierung veröffentlicht würde. In den meisten Fällen wäre dann die Ergänzung einzelner Datensätze und damit die Wiederherstellung von Informationen möglich. Die zufällige Sortierung von *k*-anonymisierten Tabellen vor einer Veröffentlichung sollte deshalb obligatorisch sein. Aber auch in diesem Fall ist die Wiederherstellung von Informationen möglich wenn mehrere Versionen der Ursprungstabelle veröffentlicht werden.

Somit bietet *k*-Anonymity keinen ausreichenden Schutz der sensitiven Informationen, die Schwächen sollen durch das Konzept der *l*-Diversity gelöst werden.

3.1.2 *l*-Diversity (auch *l*-Sensitivity

) „Eine Datentabelle ist *l*-sensitiv, wenn die Instanzen einer Äquivalenzklasse mindestens *l* verschiedene Werte eines sensitiven Merkmals enthalten.“ [Mar08]

Geburtstag	Geschlecht	Geburtsort	Augenfarbe
17.**.1955	M	76227	blau
31.**.19**	*	76***	grau
17.**.1955	M	76227	blau-grau
05.07.196*	*	76133	braun
31.**.19**	*	76***	blau-grün
05.07.19**	*	76133	grün
31.**.19**	*	76***	grün

Tabelle 3.4: ***k*=2-anonymisierte Tabelle mit Informationen zur Augenfarbe von Personen**

In Tabelle 3.3 wäre es noch möglich gewesen schlusszufolgern, dass alle 1963 geborenen Frauen eine grüne Augenfarbe besitzen. Die *l*-Diversity schickt sich nun an, dies zu vermeiden. Die Definition einer Entropy-*l*-Entropy (siehe [Hau07], Seite 9) besagt, dass jeder q*-Block mindestens *l* unterschiedliche Werte des senstiven Merkmals (hier: Augenfarbe) aufweist. Dieser Forderung wird die Tabelle 3.5 gerecht.

Geburtstag	Geschlecht	Geburtsort	Augenfarbe
..1955	M	7622*	blau
..1955	M	7622*	grau
..1955	M	7622*	blau-grau
..196*	*	7613*	braun
..196*	*	7613*	blau-grün
..196*	*	7613*	grün
..196*	*	7613*	grün

Tabelle 3.5: ***k*=2-anonymisierte Tabelle mit einer Entropy von *l*=2**

Allerdings gibt es immer noch Szenarien, in denen auch eine *l*-diverse Tabelle noch sensitive Informationen preisgeben kann. D.h. es sind immer noch Schlussfolgerungen auf Details möglich, was eigentlich verhindert werden sollte. Ohne diese Szenarien aus Komplexitätsgründen an dieser Stelle detailiert beschreiben zu können, soll nun im Folgenden noch das Konzept der *t*-Closeness vorgestellt werden – eine Erweiterung und gleichzeitig ein Ersatz für die *l*-Diversity. Die Grenzen der *l*-Diversity sowie die möglichen (Angriffs-)Szenarien lassen sich unter [Hau07] (Seite 11f) nachlesen.

3.1.3 *t*-Closeness

„Ein q*-Block besitzt *t*-Closeness, wenn die Distanz zwischen den zu veröffentlichen sensitiven Attributen dieses Block zur gesamten Tabelle nicht mehr als ein Grenzwert von *t* beträgt. Eine Tabelle besitzt *t*-Closeness, wenn alle q*-Blöcke von ihr *t*-Closeness besitzen.“ ([Hau07], Seite 12).

Intention der *t*-Closeness ist es, den möglichen Wissensgewinn für einen Angreifer zu minimieren. Im Idealfall gibt jede Gruppe nicht mehr Informationen preis als die gesamte Distribution. Dieser Forderung wird die Tabelle 3.6 gerecht

Geburtstag	**Geschlecht**	**Geburtsort**	**Gruppe**	**Augenfarbe**	**Gruppe**
17.04.1955	M	76227	2	blau	2
..1955	M	76228	1	grau	1
..1955	M	76227	2	blau-grau	2
..196*	*	7613*	2	braun	2
..196*	*	7613*	1	blau-grün	1
..196*	*	7613*	2	grün	2
..196*	*	7613*	1	grün	1

Tabelle 3.6: ***k*=2-anonymisierte Tabelle mit einer Entropy von *l*=2 und *t*-Closeness mit *t*=0,29**

Nach der theoretischen Betrachtung der Prinzipien der Logdaten-Anonymisierung soll nun im praktischen Teil Logdaten-Anonymisierung am Beispiel von Logdaten des freien und verbreiteten Webservers Apache gezeigt werden. Der Fokus liegt dabei nicht auf der Umsetzung der vorgestellten Prinzipien, sondern um die praktische Anwendung von Logdaten-Anonymisierung im Alltag, um Datenschutzprobleme zu vermeiden.

Kapitel 4

Apache-Logdaten anonymisieren

Auch wenn in der letzten Zeit der Marktanteil des freien Webservers von einst fast 80 % auf 46,49 % im Mai 2009 geschrumpft ist, ist dieser immer noch deutlich Markführer mit großem Abstand vom Zweitplatzierten Microsoft IIS (28,35 %). Damit gehört der Apache aber auch zu den größten Datensammlern, wird doch jeder Aufruf über das HTTP-Protokoll erfasst und gespeichert. Da jedes einzelne Bild und jede einzelne auf einer Webseite eingebundene Datei einen separaten Aufruf erzeugt, kommen so innerhalb kurzer Zeit immense Datenvolumina zusammen.

Diese Informationen sind natürlich die obligatorische Grundlage für die Statistikerstellung, um Kenntnis über die Art und Weise der Nutzung eines Webangebots zu erhalten. Nichts desto trotz ist man gleichzeitig mit der bereits mehrfach angesprochenen Datenschutzproblematik konfrontiert. Um auch rechtlich auf der sicheren Seite zu sein, können die Daten direkt anonymisiert in die Logfiles geschrieben werden.

4.1 Logdaten-Anonymisierung per Shell-Skript

Apache lässt sich beispielsweise über ein einfaches Perl-Skript (siehe Anhang A) der ZENDAS[1] (Zentrale Datenschutzstelle der baden-württembergischen Universitäten) für diese Anforderung einstellen. Innerhalb der Konfiguration wird dem Apache Server mitgeteilt, was und in welchem Format und Umfang er mitprotokollieren soll. Für die Funktion muss das Log-Format wie folgt definiert sein:

[1] `http://www.zendas.de`

```
1 LogFormat "%a - - %t \"%r\" %>s %b \"%{Referer}i\""
    combined
```

Listing 4.1: **Angabe des Log-Formats in der Apache-Konfiguration**

Während Listing 4.2 dem klassischen Logging ohne Anonymisierung entspricht, werden die Datensätze im Listing 4.3 zunächst durch das Perl-Skript geschickt. Dementsprechend unterscheiden sich auch die Einträge in den Log-Dateien.

```
1 # Eintrag in der Apache-Konfigurationsdatei
2 CustomLog /path/to/access_log combined
3
4 # Auszug aus der Log-Datei
5 217.227.170.61 - - [30/Jun/2009:20:35:05 +0200] "GET /
    index.html HTTP/1.1" 200 12812 "-" "Mozilla/5.0 (
    Macintosh; U; Intel Mac OS X 10.5; de; rv:1.9.0.11)
    Gecko/2009060214 Firefox/3.0.11"
6 217.227.179.61 - - [30/Jun/2009:20:35:06 +0200] "GET /js
    /lightbox.js HTTP/1.1" 200 20158 "http://www.christian
    -kolodziej.de/index.html" "Mozilla/5.0 (Macintosh; U;
    Intel Mac OS X 10.5; de; rv:1.9.0.11) Gecko/2009060214
     Firefox/3.0.11"
```

Listing 4.2: **Normales Logging im Apache**

```
1 # Eintrag in der Apache-Konfigurationsdatei
2 CustomLog "|/path/to/aplog-anon /path/to/access_log"
    combined
3
4 # Auszug aus der Log-Datei
5 217.227.0.0 - - [30/Jun/2009:20:45:01 +0200] "GET /index
    .html HTTP/1.1" 200 12812 "-" "Mozilla/5.0 (Macintosh;
     U; Intel Mac OS X 10.5; de; rv:1.9.0.11) Gecko
    /2009060214 Firefox/3.0.11"
6 217.227.0.0 - - [30/Jun/2009:20:45:02 +0200] "GET /js/
    lightbox.js HTTP/1.1" 200 20158 "http://www.christian-
    kolodziej.de/index.html" "Mozilla/5.0 (Macintosh; U;
    Intel Mac OS X 10.5; de; rv:1.9.0.11) Gecko/2009060214
     Firefox/3.0.11"
```

Listing 4.3: **Anonymisiertes Logging im Apache**

4.2 Logdaten-Anonymisierung per Modul

Der Apache-Webserver ist nach einem sehr modularen System aufgebaut, sein Funktionsumfang lässt sich auf diese Weise einfach um neue Features erweitern oder auch aus Performancegründen auf die nötigsten Funktionen beschränken.

Für die Anonymisierung von IP-Adressen lässt sich das Modul *removeip* verwenden. Die Installation gestaltet sich auf einem System mit Debian Linux denkbar einfach:

```
1 apt-get install libapache2-mod-removeip
2 a2enmod removeip
3 /etc/init.d/apache2 force-reload
```

Listing 4.4: **Installation des Moduls *removeip***

Auf anderen Systemen gestalltet sich die Installation ein wenig anders, aber generell geht es darum das Modul in den Apache-Modul-Ordner zu kopieren, es innerhalb der Apache-Konfiguration bekannt zu machen und schließlich den Webserver neu zu starten damit die geänderten Konfigurationseinstellungen gültig werden. Anschließend werden IP-Adressen nicht mehr mitgeloggt und stehen auch innerhalb von Webanwendungen beispielsweise als PHP-Umgebungsvariablen nicht mehr zur Verfügung. Stattdessen wird nun die IP-Adresse 127.0.0.0 (localhost) geloggt und weitergegeben. Der Seitenaufruf bewirkt fortan folgende Einträge in der Log-Datei:

```
1 # Auszug aus der Log-Datei
2 127.0.0.0 - - [30/Jun/2009:20:52:40 +0200] "GET /index.
    html HTTP/1.1" 200 12812 "-" "Mozilla/5.0 (Macintosh;
    U; Intel Mac OS X 10.5; de; rv:1.9.0.11) Gecko
    /2009060214 Firefox/3.0.11"
3 127.0.0.0 - - [30/Jun/2009:20:52:41 +0200] "GET /js/
    lightbox.js HTTP/1.1" 200 20158 "http://www.christian-
    kolodziej.de/index.html" "Mozilla/5.0 (Macintosh; U;
    Intel Mac OS X 10.5; de; rv:1.9.0.11) Gecko/2009060214
     Firefox/3.0.11"
```

Listing 4.5: **Log-Datei-Einträge mit dem Modul *removeip***

4.3 Fazit der Live-Anonymisierung

Sowohl bei Benutzung des Perl-Skripts aks auch bei der Verwendung des Apache-Moduls liegen nun anonymisierte Logdaten vor, die aber nicht in gleichem Maße dieselben Eigenarten vorweisen wie ihre Ursprungsdaten, einige statische Auswertungen sind deshalb eventuell nicht mit dem selben Ergebnis reproduzierbar. Durch die Live-Anonymisierung werden die Daten sofort anonymisiert gespeichert und liegen auch temporär nie in einer Form vor, die datenschutztechnisch kritisch sein könnte.

Unterschiede zwischen beiden vorgestellten Varianten finden sich vor allem in der Flexibilität. Während beim Shell-Skript durch Änderungen am regulären Ausdruck leicht Änderungen an der Art und Weise der Anonymisierung möglich sind – sollen andere, weniger oder mehr als die voreingestellten zwei Adressblöcke auf 0 gesetzt werden – so ist dies beim Apache-Modul *removeip* nicht möglich. Auch wenn die Ergebnisse auf den ersten Blick sehr ähnlich aussehen, liegt der Unterschied im Detail. Während beide Varianten für die Anonymisierung der Log-Datei an sich sorgen, garantiert lediglich die Nutzung des Apache-Moduls, dass auch auf Seite der Web-Anwendung die IP-Adresse anonymisiert vorliegt. Bei Benutzung des Perl-Skripts steht beispielsweise innerhalb einer PHP-Anwendung weiterhin die Original-IP-Adresse zur Verfügung und könnte dort entsprechend verarbeitet oder gespeichert werden.

Kapitel 5

Log-Anonymisierung mit FLAIM

Als Alternative zur Live-Anonymisierung von Apache-Logdaten bietet sich die Benutzung eines flexiblen Tools an, beispielsweise das Open-Source-Programm FLAIM[1] der LAIM Working Group[2] am National Center for Supercomputing Application (NCSA) der Universität von Illinois in den USA. FLAIM ist ein Framework, das den Rahmen zur Anonymisierung verschiedenartiger Logdaten und -formate bietet. Mit Hilfe sog. Module lassen sich diverse Formate anonymisieren - die Art und Weise der Anonymisierung wird dabei durch eine *Anonymization Policy* festgelegt. FLAIM liegt momentan in der Version 0.7.0 vom 29. Februar 2008 vor. Leider war seitdem keine Aktivität des Open-Source-Projekts mehr zu verzeichnen, auch die Community ist nur sehr klein und bei Fragen und Problemen bleibt man auf sich alleine gestellt, wie sich nachher noch zeigen wird

Anders als bei der Live-Anonymisierung im vorherigen Kapitel arbeitet FLAIM nachgelagert und anonymisiert bestehende Log-Dateien. Eine synchrone Anonymisierung ist nicht möglich, d.h. FLAIM kann sich nicht in der Datenstrom einer Anwendung einklinken, um die Daten direkt zu anonymisieren und so zu verhindern, dass zumindest temporär nicht-anonymisierte Daten gespeichert werden.

[1] `http://flaim.ncsa.illinois.edu`
[2] `http://www.ncsa.illinois.edu`

5.1 Installation des Kerns

Zunächst einmal gilt es, sich die aktuellste Version von der Download-Seite[3] zu besorgen. Anschließend wird das Archiv am gewünschten Ort entpackt, konfiguriert und letztlich installiert. Für die Linux-Distributionen Debian und RedHat stehen zudem spezielle Installationspakete zur Verfügung.

```
1 tar -xzf flaim-core-0.7.0.tgz
2 cd flaim-core-0.7.0
3 ./configure
4 make
5 make install
```

Listing 5.1: **Installation von FLAIM**

5.2 Installation der Module

Noch einfacher als die Installation des Kerns ist die Installation der Module. Für jedes zu anonymisierende Log-Format benötigt man das entsprechende Modul. Auf der offiziellen Internetseite erhält man nach Angabe seiner E-Mail-Adresse die Download-Links für die Module der Log-Formate *iptable*, *nfdump*, *pacct* sowie *pcap*. Das jeweilige Modul muss nun noch in das Verzeichnis *modules* im Installationsverzeichnis von Flaim entpackt werden bevor nun im nächsten Schritt die Anonymisierung durchgeführt werden kann.

5.3 Log-Datei anonymisieren

Neben einer zu anonymisierenden Log-Datei benötigt man nun noch die sog. *Anonymization Policy*, die die genauen Informationen über die Art und Weise der durchzuführenden Anonymisierung enthält. Auch diese findet sich auf dem Webauftritt des Projekts und hat am Beispiel des Formats *iptable* folgenden Inhalt (Auszug):

[3] `http://flaim.ncsa.illinois.edu/download.html`

```
<policy>
  <field name="IPV4_SRC_IP">
    <NumericTruncation>
      <numShifts>4</numShifts>
      <radix>2</radix>
    </NumericTruncation>
  </field>
  <field name="ETHER_DST_MAC">
    <BinaryBlackMarker>
      <numMarks>3</numMarks>
      <replacement>1</replacement>
    </BinaryBlackMarker>
  </field>
  <field name="TCP_DST_PORT">
    <Classify>
      <configString>9:9,99:99,1024:1024</configString>
    </Classify>
  </field>
  <field name="PCKT_MACHINE_NAME">
    <Substitution>
      <substitute>delta.beta.com</substitute>
    </Substitution>
  </field>
</policy>
```

Listing 5.2: **Anonymoization Policy für das Logdaten-Format *iptable***

Im vorliegenden Auszug der *Anonymization Policy* werden ein paar der zu Verfügung stehenden Anonymisierungs-Mechanismen genutzt, eine vollständige Referenz findet sich in der offiziellen Dokumentation[4].

Der vorliegende Auszug aus der *Anonymization Policy* sorgt für die folgenden Manipulationen innerhalb der Log-Datei:

IPV4_SRC_IP Die *NumericTruncation* sorgt für eine Rechts-Verschiebung von *numShifts* Bits, wobei *radix* angibt, ob es sich dabei um eine Zahl zur Basis 2 oder 10 handelt. Optional könnte an dieser Stelle noch über die Variable *direction* eine Links-Verschiebung eingestellt werden, standardmäßig ist die Rechts-Verschiebung eingestellt.

[4] `http://flaim.ncsa.illinois.edu/downloads/FLAIM-Core-UG.pdf`

ETHER_DST_MAC Per *BinaryBlackMarkers* werden die *numMarks* (hier: 3) letzten Bits durch *replacement* (hier: 1) ersetzt. Damit ist für die IP-Adresse eine analoge Anonymisierung möglich, wie sie bereits per Skript in Kapitel 4.1 vorgenommen wurde.

TCP_DST_PORT Die Portnummern werden in Äquivalenzklassen eingeteilt.

PCKT_MACHINE_NAME Der ursprüngliche Hostname wird durch *delta.beta.com* ersetzt.

Die nötigen Vorarbeiten sind nun abgeschlossen. Der folgende Befehl lädt nun die Datei *sample.iptable.log* (ebenfalls als Download auf der FLAIM-Internetseite verfügbar), um ihre Inhalte gemäß dem in Listing 5.2 definierten Schema zu anonymisieren.

```
1 flaim -m iptable -p sample-iptable.apolicy.xml -i sample
    .iptable.log -o anonymized.iptable.log
```

Listing 5.3: **Ausführung des Anonymisierungs-Vorgangs**

Als Ergebnis sollte nun eigentlich die Datei *anonymized.iptable.log* geschrieben werden, die den dann anonymisierten Inhalt enthält. Doch leider zwingt uns die Software dazu im Konjunktiv zu bleiben, denn jeder Versuch der Anonymisierung wird mit der folgenden Fehlermeldung quittiert ohne dass ein brauchbares Ergebnis in Form einer anonymisierten Log-Datei erzeugt wird:

```
1 flaim: could not load module library: /usr/local/flaim/
    modules/iptable/libiptable\_flaim.so: cannot open
    shared object file: No such file or directory}
```

Listing 5.4: **FLAIM-Fehlermeldung beim Ausführen der Anonymisierung**

Dass die genannte Datei nicht vorhanden ist, darin hat FLAIM Recht. Allerdings bleibt die Frage nach dem Warum des Fehlens offen und unbeantwortet. Dies gilt leider auch für die anderen Module, in alle Fällen bestand das Ergebnis lediglich aus der genannten (modulspezifischen) Fehlermeldung. Auch die Recherchen im Internet blieben erfolglos, was auch auf das schon angesprochene Fehlen einer aktiven Community zurückzuführen ist.

Aus diesem Grunde muss an dieser Stelle mit der Vorstellung von FLAIM abgeschlossen werden, ohne ein Ergebnis der Anonymisierung zeigen zu können. Auch die Auswirkungen von Manipulationen an der *Anonymization Policy* muss diese Ausarbeitung deshalb schuldig bleiben.

Es bleibt ein fader Beigeschmack beim Open-Source-Tool FLAIM, das einen sehr interessanten Ansatz verfolgt und auch durchaus Potential hat. Dass mit der Version 0.7.0 ein noch sehr früher Versionsstand vorliegt (und dies seit fast anderthalb Jahren) ist ebenso wenig förderlich wie die quasi nicht vorhandene Community. Als potentieller und interessierter Benutzer bleibt man bei den ersten Schritten auf sich alleine gestellt und scheitert dabei eventuell wie im Beispiel (ohne eigenes Verschuldung) schon sehr früh.

Kapitel 6

Fazit

Datenschutz geht uns alle etwas an und wird mit der zunehmenden Durchdringung unseres Alltags mit IT-Systemen auch immer wichtiger. Zum einen liegt es an jedem Benutzer selbst, sich um den Schutz der eigenen personenbezogenen Daten zu kümmern um nicht zum „gläsernen“ Benutzer zu werden. Das Bundesdatenschutzgesetz liefert hierzu die Grundlage, die auch von Betreibern von Webanwendungen und IT-Systemebn eingehalten werden müssen. Nicht alles, das technisch möglich ist, ist auch rechtlich zulässigund moralisch tragbar. Trotz stetig sinkender Speicherpreise gilt es sich gesetzesztreu zu verhalten und auch wirklich nur die nötigen Daten zu erfassen und zu speichern sowie diese auch ausschließlich zweckgebunden einzusetzen.

Mit Logdaten-Anonymisierung kann der Spagat zwischen der Notwendigkeit des Vorhandenseins von Informationen über Benutzerverhalten auf der einen und den datenschutzrechtlichen Gesichtspunkten auf der anderen Seite gelingen. Am Beispiel von Logdaten des Webservers Apache wurde veranschaulicht in welcher Weise Logdaten-Anonymisierung mit Hilfe von Open-Source-Software möglich ist. Leider ließ das Programm FLAIM keinen Vergleich zu. Trotzden sollten auch in anderen Systemen und Bereichen solche Möglichkeiten zu Verfügung stehen und dann auch genutzt werden, um nicht früher oder später mit dem Datenschutzesetz in Konflikt zu geraten.

Anhang A

Shell-Script zur Live-Anonymisierung im Webserver Apache

```
#!/usr/bin/perl
# aplog-anon -- Real-Time-Anonymisierung von
# Apache-Log-Daten
# Copyright (C) 2003 by Z E N D A S,
# Universität Stuttgart.
#
# Die Anonymisierung dier Client-Adressen wird auf
# /16er-Netzmasken durchgeführt. Eine andere Möglichkeit
# wären /24er-Netzmasken, aber dadurch ist nicht völlig
# sichergestellt, daß der Personenbezug verloren geht.
# Zusätzlich wird der Referer-URL beim ersten "?"
# abgeschnitten.
#
# In der allgemeinen Apache-Konfiguration ist folgende
# Zeile einzufügen:
#
# LogFormat "%a - - %t \"%r\" %>s %b \"%{Referer}i\""
   combined
#
# In der Konfiguration für virtuelle Server wird das
# Logging wie folgt aktiviert:
#
# ErrorLog "|/path/to/aplog-anon /path/to/error_log"
# CustomLog "|/path/to/aplog-anon /path/to/access_log"
   combined
#
# Änderungen an den Skripten für die Log-Rotation sind
# üblicherweise nicht erforderlich. Das Logging über
# eine Pipe erfordert ein paar zusätzliche Context
# Switches; dies sollte aber nur unter absolutem
# Hochlastbetrieb ein Problem darstellen; das
```

```
30 # Anonymsieren funktioniert auf Hardware aus dem Jahr
31 # 2002 auch bei mehreren Requests pro Sekunde ohne
32 # Probleme.
33 #
34 # Vorsicht: aplog-anon schreibt typischerweise die
35 # Log-Datei mit root-Rechten.
36
37 use strict;
38 use warnings;
39
40 use IO::Handle;
41
42 if (@ARGV != 1) {
43     exit 1;
44 }
45
46 my $LOG;
47 open $LOG, ">>$ARGV[0]";
48
49 # Daten sofort in die Zieldatei schreiben, ohne
50 # Pufferung auf Perl-Seite.
51 $LOG->autoflush(1);
52
53 while (my $Line = <STDIN>) {
54     chomp $Line;
55
56     # Der erste Fall behandelt Fehlermeldungen,
57     # die eine andere Syntax haben.
58     if ($Line =~ /^\[.*/) {
59         $Line =~ s/^(.*?\[client \d+\.\d+)\.\d+\.\d
              +(\].*)/$1.0.0]$2/;
60     } else {
61         $Line =~ s/^(\d+\.\d+)\.\d+\.\d+ (.*" (?:\d+|-)
              (?:\d+|-) "[^?]*).*/$1.0.0 $2\"/;
62         $Line =~ s/""$/\"/;
63     }
64
65     print $LOG "$Line\n";
66 }
```

Listing A.1: **Echtzeit-Anonymisierung von Apache-Log-Daten mittels Shell-Skript**

Literaturverzeichnis

Hau07 Dietmar Hauf. Allgemeine Konzepte - K-Anonymity, l-Diversity and T-Closeness. http://dbis.ipd.uni-karlsruhe.de/img/content/SS07Hauf_kAnonym.pdf, 2007. Letzte Abfrage: 30. Juni 2009.

Mar08 Sven Martin. Bericht über das Fortbildungssemester im Sommersemester 2008, 2008.

Win08 WinFuture.de. Google gibt Nutzerdaten an US-Regierung weiter. http://winfuture.de/news,38201.html, 2008. Letzte Abfrage: 01. Juli 2009.