Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Entwurf vom 25. Januar 2018

Masterarbeit

Datenschutzfreundliche Speicherung unternehmensinterner Überwachungsdaten mittels Pseudonymisierung und kryptographischer Schwellwertschemata

vorgelegt von

Tom Petersen

geb. am 13. Dezember 1990 in Hannover

Matrikelnummer 3659640

Studiengang Informatik

eingereicht am 25. Januar 2018

Betreuer: Dipl.-Inf. Ephraim Zimmer

Erstgutachter: Prof. Dr. Hannes Federrath

Zweitgutachter: Prof. Dr. Mathias Fischer

Aufgabenstellung

Die technologiegestützte Bekämpfung von Insider-Angriffen im Unternehmenskontext basiert aktuell häufig auf der Analyse des Nutzerverhaltens einzelner Mitarbeiter und der Erkennung von Abweichungen zum erwarteten Normalverhalten. Diese sogenannte Anomalieerkennung benötigt umfassende Überwachungsdaten aller digitalen Endgeräte und Datenkommunikationssysteme zur Erstellung und eindeutigen Zuordnung von Nutzerprofilen. Dabei entsteht ein Konflikt mit dem Datenschutz der Mitarbeiter, da die Erhebung, Verarbeitung und Speicherung von Überwachungsdaten einen schweren Eingriff in die Privatsphäre und die informationelle Selbstbestimmung der Mitarbeiter darstellt. Um diesen Konflikt zu lösen, können auf der einen Seite Datenschutztechniken eingesetzt werden, die den unmittelbaren Personenbezug gesammelter Daten entfernen. Auf der anderen Seite kann mithílfe von Kryptographie die Rückgewinnung des Personenbezugs im Verdachtsfall und unter der Voraussetzung einer mehrseitigen Kollaboration ermöglicht werden.

Das Ziel der Masterarbeit ist die konzepzionelle Erarbeitung einer solchen datenschutzfreundlichen und mehrseitig sicheren Erhebung, Verarbeitung und Speicherung von Überwachungsdaten sowie die prototypische Implementierung auf Basis eines Security Information Event Management Systems. Dabei sollen insbesondere die folgenden Punkte bearbeitet werden:

- Wie ist der aktuelle Stand sowohl der Technik als auch der Wissenschaft im Bereich der Pseudonymisierung und der kryptographischen Schwellwertschemata?
- An welcher Stelle des konzipierten Systems können die Überwachungsdaten entsprechend des Datenschutzes und der späteren möglicherweise erforderlichen Rückgewinnung des Personenbezugs verarbeitet werden und welche Auswirkungen können entstehen?
- Wie und an welcher Stelle muss das Schlüsselmanagement der benötigten kryptographischen Funktionen erfolgen?
- Welche Alternativen gibt es neben der Pseudonymisierung und den kryptographischen Schwellwertschemata zur Lösung des genannten Zielkonflikts und wie können diese in das Konzept und die prototypische Implementierung integriert werden?

Todo list

Zusammenfassung

Inhaltsverzeichnis

1 Einführung

Wo passen Abschnitte zu folgenden Stichworten hin? - Verschiedene Datenarten: Identifizierend, Traffic, nicht relevant, ... - Grundlegende Definition Insiderangriff

Liest man von erfolgreichen Angriffen auf Unternehmensnetzwerke, so ist die implizite Annahme von außenstehenden, unternehmensfremden Angreifern weit verbreitet. Doch häufig sind die Angreifer bereits im Netzwerk ansässig. Es handelt sich um (ehemalige) Mitarbeiter oder zumindest Personen mit legitimem Zugriff auf das Netzwerk, wie Geschäftspartnern oder Kunden. Hierbei geht es keineswegs lediglich um Einzelfälle.

In dem *IBM Cyber Security Intelligence Report* von 2015 werden 55% der Angriffe als aus dem internen Netz stammend angegeben [Bra+15]. Zu beachten ist allerdings, dass nicht nur mit Absicht ausgeführte Angriffe hierunter erfasst wurden, sondern auch unbeabsichtigte wie das versehentliche Veröffentlichen schützenswerter Kundendaten.

Auch der Branchenverband bitkom führt in seiner *Spezialstudie Wirtschaftsschutz* aus dem Jahr 2016 nach einer Befragung von über 1000 Unternehmen aus, dass etwa 60% der erfolgten Handlungen aus dem Bereich Datendiebstahl, Industriespionage oder Sabotage durch (ehemalige) Mitarbeiter erfolgten [bit16].

Schadenshöhe (siehe Antrag)?

Auch wenn die genauen Zahlen aufgrund von unterschiedlichen Annahmen und der in diesem Bereich nicht zu vernachlässigenden Dunkelziffer[1] mit Vorsicht zu betrachten sind, so geben sie doch Hinweise darauf, dass Angriffe von Innentätern weit verbreitet sind und ein hohes Schadenspotenzial aufweisen. Die Erkennung und Verhinderung solcher Angriffe sollte daher ein wichtiger Teil des IT-Sicherheitskonzepts eines Unternehmens sein.

Zur Erkennung von Angriffen in Netzwerken können SIEM-Systeme eingesetzt werden (siehe Abschnitt **??**). Diese sind jedoch in erster Linie auf das Erkennen von externen Angriffen ausgelegt und in ihrer derzeitigen Form kaum sinnvoll für das Erkennen von Innentätern zu nutzen.
Hierfür würden zusätzliche Datenquellen und Erkennungslogiken nötig sein.Zusätzlich spielen auch datenschutzrechtliche Bedenken im Bezug auf das Sammeln von großen Datenmengen über Mitarbeiter des eigenen Unternehmens hier eine entscheidende Rolle. Details hierzu sind im folgenden Kapitel **??** zu finden.

EZ: Warum nicht? Einige werben damit, Innentaeter erkennen zu koennen, z.B. IBM QRadar

EZ: Warum? Welche?

Ein Ansatz, der diese Bedenken ausräumen oder zumindest lindern könnte, ist die Nutzung von Pseudonymen bei der Datenerfassung (siehe Abschnitt **??**). Anstatt direkt identifizierende Merkmale eines Arbeitnehmers abzuspeichern, werden diese Merkmale durch ein Pseudonym ersetzt. Eine Liste dieser Ersetzungen wird verschlüsselt abgelegt. Im Fall eines Angriffs durch einen Innentäter kann die Liste entschlüsselt werden und relevante Ereignisse de-pseudonymisiert, also ihrem ursprünglichen Verursacher wieder zweifelsfrei zugeordnet, werden.
Um die Entschlüsselung nicht einzelnen (möglicherweise bösartig agierenden) Personen zu

1. Insbesondere die Angst vor Imageschäden, die auch in der *Spezialstudie Wirtschaftsschutz* erwähnt wird, könnte ein Grund für das Geheimhalten von Vorfällen sein.

ermöglichen, können sogenannte Schwellwertschemata eingesetzt werden (siehe Abschnitt **??**). Durch sie wird die Entschlüsselung erst durch die Kooperation mehrerer Parteien möglich gemacht.

Bei diesem Ansatz muss jedoch auch beachtet werden, dass durch den Einsatz von Pseudonymen die Erkennung von Angriffen erschwert werden könnte. Beispielsweise könnte das Ändern von Pseudonymen in regelmäßigen Zeitintervallen und die dadurch entstehende Nicht-Verkettbarkeit von Ereignissen dafür sorgen, dass längfristig angelegte Angriffe nicht aufgedeckt werden.

Ziele der Arbeit

In dieser Arbeit soll es darum gehen, prototypisch ein solches Szenario auf Basis eines Open-Source-SIEM-Systems umzusetzen.Hierbei müssen einige Fragen betrachtet werden:

EZ: Was genau ist das Szenario? Liegt dein Fokus nun auf den zusaetzlichen Datenquellen und Erkennungslogiken oder auf den datenschutzrechtlichen Bedenken?

- An welcher Stelle des Systems kann eingegriffen werden, um die erfassten Daten zu verändern, und welche Auswirkungen hat dies?
- Wie erfolgt die angesprochene Pseudonymisierung technisch?
- Welche kryptographischen Schwellwertschemata können genutzt werden? Gibt es bereits quelloffene Implementierungen? Was muss selbst entwickelt werden? Wie erfolgt das Schlüsselmanagement?
- Können neben der Pseudonymisierung noch weitere Funktionen zur Veränderung von Daten sinnvoll sein und wie könnten diese umgesetzt werden?

Gerade die letzte Frage sorgt dafür, dass zusätzliche Anforderungen an den zu entwickelnden Prototypen gestellt werden. Es sollte möglich sein, abhängig von den eingehenden Daten die entsprechend gewünschten Funktionen konfigurieren und den Prototypen in eventuell aufbauenden Arbeiten auch um zusätzliche Funktionen ergänzen zu können.

Zu ergänzen: Aufbau der Arbeit, Literatur

1.1 Related work

2 Grundlagen

Dieses Kapitel widmet sich den Grundlagen der in dieser Arbeit verwendeten Konzepte, Verfahren und Systeme. Zu Beginn werden für den Verlauf der Arbeit notwendige Definitionen gegeben. Anschließend werden die juristischen Hintergründe des Arbeitnehmerdatenschutzes erläutert, die den rechtlichen Rahmen für das Thema dieser Arbeit bilden.

Es folgen Erläuterungen zu SIEM-Systemen, in die - wie bereits in der Einleitung erläutert - die prototypische Umsetzung der datenschutzfreundlichen Speicherung erfolgen soll, sowie zu den zu verwendenden Datenschutztechniken Pseudonymisierung, kryptographische Schwellwertschemata und Searchable Encryption.

2.1 Definitionen und Notationen

- Insider-Angriff

- Datenarten

- Syslog?

- Hashfunktion?

- ...

2.1.1 Mathematische Notationen

p, q

$\mathbb{Z}_p$

$\mathbb{Z}_p^*$

2.2 Arbeitnehmerdatenschutz

Gesetzestexte als „Quellen„ in Fußnoten?

Der Begriff des Arbeitnehmerdatenschutzes[1] beschreibt die Rechte von Arbeitnehmern im Beschäftigungsverhältnis im Bezug auf den Umgang mit personenbezogenen Daten. In diesem Abschnitt soll ein kompakter Überblick über aktuell geltende und in nächster Zeit in Kraft tretende gesetzliche Regelungen im Bezug hierauf gegeben werden, wobei der Fokus auf zum Thema der Arbeit passenden Regelungen liegt.

1. In manchen Veröffentlichungen wird der Arbeitnehmerdatenschutz auch als Mitarbeiterdatenschutz, Beschäftigtendatenschutz, Personaldatenschutz oder Betriebsdatenschutz bezeichnet.

Zu Beginn soll kurz auf das Recht auf informationelle Selbstbestimmung eingegangen werden, das die Grundlage für alle folgenden Betrachtungen zum Arbeitnehmerdatenschutz bildet.

2.2.1 Das Recht auf informationelle Selbstbestimmung

Im sogenannten Volkszählungsurteil aus dem Jahr 1983 wurde das Recht auf informationelle Selbstbestimmung als Grundrecht anerkannt [**TODO**]. Es handelt sich um eine Ausprägung des allgemeinen Persönlichkeitsrechts[2] nach Artikel 2, Absatz 1 in Verbindung mit Artikel 1, Absatz 1 des Grundgesetzes. Es beschreibt das Recht des Einzelnen, selber über den Umgang mit seinen personenbezogenen Daten entscheiden zu können.

Mit dem vermehrten Aufkommen automatisierter Datenverarbeitung stellten die Richter des Bundesverfassungsgerichts damals die besondere Schutzbedürftigkeit der Selbstbestimmung des Einzelnen im Bezug auf die Offenbarung von Lebenssachverhalten heraus. Sie betonten die Notwendigkeit dieser Selbstbestimmung als Voraussetzung für eine freie Entfaltung der Persönlichkeit und auch für die Ausübung bestimmter Grundrechte wie der Versammlungsfreiheit. Damit sei das Recht auf informationelle Selbstbestimmung auch „eine elementare Funktionsbedingung eines auf Handlungs- und Mitwirkungsfähigkeit seiner Bürger begründeten freiheitlichen demokratischen Gemeinwesens“ [**TODO**] .

Einschränkungen dieses Rechts sind dem Urteil nach möglich, jedoch in Gesetzen festzuhalten. Hierbei müssen das Geheimhaltungsinteresse des Betroffenen und das öffentliche Informationsinteresse verarbeitender Stellen gegeneinander abgewogen werden.

Auch wenn sich das Urteil des Bundesverfassungsgerichts nur auf die Rechte des Betroffenen gegenüber staatlichen Akteuren bezieht, so bildet die Intention des Rechts auf informationelle Selbstbestimmung doch die Grundlage für das heutige Bundesdatenschutzgesetz und auch die Datenschutzgrundverordnung der EU, die auch für nicht-staatliche Akteure Gültigkeit besitzen.

Zusätzlich findet sich das Recht auf informationelle Selbstbestimmung auch in der Grundrechtecharta der EU: „Jede Person hat das Recht auf Schutz der sie betreffenden personenbezogenen Daten.“[3] Quelle

2.2.2 Datenschutz im Beschäftigungsverhältnis

Eine besondere Situtation ergibt sich im Unternehmenskontext. Hier muss das Recht des Arbeitnehmers auf informationelle Selbstbestimmung gegenüber dem berechtigten Interesse des Arbeitgebers an der Aufklärung von Straftaten im Beschäftigungsverhältnis abgewogen werden.

Im zur Zeit gültigen Bundesdatenschutzgesetz (BDSG) wird in § 4 die Erhebung, Verarbeitung und Nutzung personenbezogener Daten nur als zulässig angesehen, falls der Betroffene einwilligt oder ein Gesetz dieses erlaubt. Personenbezogene Daten werden in § 3 hierbei als „Einzelangaben über [...] Verhältnisse einer bestimmten oder bestimmbaren natürlichen Person“ definiert. Quelle

2. Das allgemeine Persönlichkeitsrecht beschreibt den Schutz der Persönlichkeit einer Person vor Eingriffen in ihren Lebens- und Freiheitsbereich.
3. Artikel 8, Absatz 1

§ 32 beschreibt die Datenerhebung, -verarbeitung und -nutzung für Zwecke des Beschäftigungsverhältnisses:

> Personenbezogene Daten eines Beschäftigten dürfen für Zwecke des Beschäftigungsverhältnisses erhoben, verarbeitet oder genutzt werden, wenn dies für die Entscheidung über die Begründung eines Beschäftigungsverhältnisses oder nach Begründung des Beschäftigungsverhältnisses für dessen Durchführung oder Beendigung erforderlich ist.
> Zur Aufdeckung von Straftaten dürfen personenbezogene Daten eines Beschäftigten nur dann erhoben, verarbeitet oder genutzt werden, wenn zu dokumentierende tatsächliche Anhaltspunkte den Verdacht begründen, dass der Betroffene im Beschäftigungsverhältnis eine Straftat begangen hat, die Erhebung, Verarbeitung oder Nutzung zur Aufdeckung erforderlich ist und das schutzwürdige Interesse des Beschäftigten an dem Ausschluss der Erhebung, Verarbeitung oder Nutzung nicht überwiegt, insbesondere Art und Ausmaß im Hinblick auf den Anlass nicht unverhältnismäßig sind.[4]

Während sich der erste Satz auf den Umgang mit personenbezogenen Daten in einem normalen Beschäftigungsverhältnis befasst und bezogen auf das Thema dieser Arbeit beispielsweise den Rahmen für erforderliche Datenverarbeitung zur Aufdeckung von Vertragsbrüchen unterhalb der Straftatgrenze darstellt, behandelt der zweite Satz den Straftatfall. Hier sind insbesondere der notwendige Anfangsverdacht als Voraussetzung und die Verhältnismäßigkeit der Datennutzung zu beachten.

Weiterhin insbesondere im Rahmen dieser Arbeit entscheidend ist die Ausrichtung des BDSG auf personenbezogene Daten, die wie bereits definiert einer bestimmbaren Person zugeordnet können werden müssen. Das in dieser Arbeit angestrebte System wird durch Pseudonymisierung und erst durch Kollaboration ermöglichte De-Pseudonymisierung den direkten Personenbezug verhindern und erst im durch mehrere Instanzen bestätigten Straftatverdacht ermöglichen[5].

2018 tritt die EU-Verordnung 2016/679, besser bekannt als Datenschutzgrundverordnung, in Kraft. In Deutschland wird das bestehende BDSG durch das Datenschutz-Anpassungs- und Umsetzungsgesetz grundlegend überarbeitet und an die Verordnung angepasst, um diese zu ergänzen. Hier finden sich in § 26 die Bestimmungen zur Datenverarbeitung für Zwecke des Beschäftigungsverhältnisses. Der zitierte Absatz aus dem BDSG ist dort in ähnlicher Form zu finden, wird also auch weiterhin seine Gültigkeit behalten.

Wie sollten Gesetzestexte zitiert werden?

Beispiele für Überwachungsskandale

2.3 SIEM-Systeme

SIEM-Systeme dienen dazu Daten in Netzwerken zu sammeln, um so einen zentralisierten Überblick über das Netzwerk zu erhalten und Bedrohungen erkennen und verhindern zu können.

4. § 32, Absatz 1, Bundesdatenschutzgesetz
5. Der Autor maßt sich an dieser Stelle allerdings keine Beurteilung über die tatsächliche rechtliche Bewertung dieser Lösung an.

Der Begriff *Security Information and Event Management* (SIEM) wurde von zwei Analysten des IT-Marktforschungsunternehmens Gartner geprägt, das auch jährlich einen Bericht über aktuelle Trends im Bereich der SIEM-Systeme veröffentlicht. Er setzt sich zusammen aus *Security Event Management* (SEM), das sich mit Echtzeitüberwachung und Ereigniskorrelation befasst, sowie *Security Information Management* (SIM), in dessen Fokus Langzeiterfassung und Analyse von Log-Daten steht [NK11].

Ein SIEM-System sollte nach [Det+15] die folgenden Aufgaben erfüllen können:

- **Network Behaviour Anomaly Detection:** Beschreibt die Erkennung von Anomalien auf Netzwerkebene durch die Erkennung von vom Normalzustand abweichenden Kommunikationsverhalten.
- **Identity Mapping:** Abbildung von Netzwerkadressen auf Nutzeridentitäten.
- **Key Performance Indication:** Zentrale Analyse sicherheitsrelevanter Informationen und Netzwerkdetails.
- **Compliance Reporting:** Überprüfung der Einhaltung von durch Regelungen vorgeschriebenen Anforderungen wie Integrität, Risiko und Effektivität.
- **API:** Bereitstellung von Schnittstellen zur Integration heterogener Systeme im Netzwerk.
- **Role based access control:** Zuständigkeitsabhängige Sichten auf sicherheitsrelevante Ereignisse.
- **Event Correlation:** im Folgenden näher erläutert.

Eine besondere Bedeutung im Kontext dieser Arbeit kommt hier der Behandlung von sicherheitsrelevanten Ereignissen (Events) zu, die beispielsweise von Intrusion-Detectionen-Systemen oder aus den Log-Daten von Firewalls, Switches oder anderen Netzwerkgeräten stammen können.

Um diese Ereignisse zu erhalten, muss ein SIEM-System nach [DRS14] vor ihrer Speicherung insbesondere drei Aufgaben wahrnehmen. Zu Beginn werden die Daten aus Logeinträgen oder empfangenen Systemmeldungen herausgelesen (Extraktion).
Anschließend müssen die extrahierten Daten in ein SIEM-spezifisches Format übersetzt werden, um eine sinnvolle Weiterverarbeitung zu gewährleisten (Homogenisierung). Hierbei werden relevante Felder eines SIEM-Events wie Datumsangaben, Adressen oder Aktionen aus den empfangenen Daten befüllt. Dieser Schritt wird in anderen Quellen auch als Normalisierung oder Mapping bezeichnet.
Optional können darauf folgend gleichartige Events in bestimmten Fällen anschließend zusammengefasst werden, um aussagekräftigere Informationen zu erhalten (Aggregation).

Liegen die Events nun in einem vorgebenen Format im System vor, so können sie weiterhin mit dem System bekannten Umgebungsdaten über Benutzer, Geräte oder Bedrohungen verknüpft werden, um ihre Relevanz besser einschätzen zu können.

Anschließend lassen sich vorgegebene Regeln anwenden, um aus der Korrelation von Ereignissen aus verschiedenen Datenquellen auf eine Bedrohung schließen zu können, die in den einzelnen Events nicht erkennbar wäre (Event Correlation).

Wäre hier ein Beispiel für Event Correlation nötig/hilfreich?

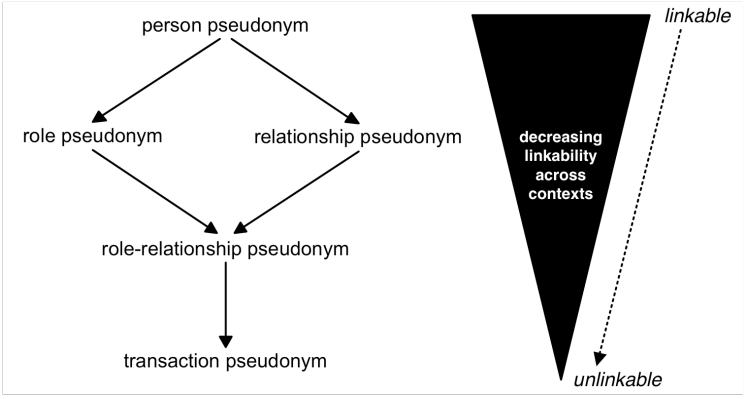

Abbildung 2.1: Pseudonym-Verband entsprechend ihrer Nutzung in verschiedenen Kontexten. Entnommen aus [PH10].

2.4 Pseudonymisierung

Ein Pseudonym[6] bezeichnet nach [PH10] einen Identifikator eines Subjekts ungleich seinem echten Bezeichner. In §3, Absatz 6a des BDSG wird zusätzlich noch der „Zweck [eines Pseudonyms], die Bestimmung des Betroffenen auszuschließen oder wesentlich zu erschweren“[**TODO**] ergänzt. Beispiele für Pseudonyme lassen sich in verschiedensten Bereichen finden: E-Mail-Adressen, Sozialversicherungsnummern oder auch Autoren, die unter einem Pseudonym ihre Schriften veröffentlichen.

Evtl. auch Telemediengesetz erwähnen - Nutzung eines Systems unter einem Pseudonym

Es lassen sich nach [PWP90] unterschiedliche Arten von Pseudonymen unterscheiden. Eine Eigenschaft, die zur Unterscheidung herangezogen werden kann, ist die (Un-)Kenntnis des Zusammenhangs zwischen dem Pseudonym und dem zugehörigen Subjekt zu Beginn seiner Verwendung. Dieser Zusammenhang wird auch als Zuordnungsvorschrift bezeichnet und kann beispielsweise als Funktion oder in Tabellenform vorliegen.
Hier kann zwischen öffentlichen, nicht-öffentlichen und anonymen Pseudonymen unterschieden werden. Ein öffentliches Pseudonym ist beispielsweise die Telefonnummer einer Person, die von Beginn an im Telefonbuch mit der Identität der Person verknüpft ist. Eine Kontonummer zu einem Bankkonto, dessen Inhaber bei der Kontoeröffnung nur der Bank bekannt ist, bildet ein nicht-öffentliches Pseudonym. Ein anonymes Pseudonym bildet beispielsweise die DNA-Sequenz eines Menschen, die oftmals nicht einmal ihm selbst bekannt ist. Diese Eigenschaft eines Pseudonyms kann sich im Laufe der Zeit ändern, wenn Informationen über den Zusammenhang zwischen Pseudonym und zugehörigem Subjekt veröffentlicht werden.

Eng in Verbindung mit Pseudonymen steht auch der Begriffe der Verkettbarkeit. Verkettbarkeit bezeichnet dabei die Eigenschaft, dass ein Außenstehender mit hoher Wahrscheinlichkeit entscheiden kann, ob zwei Objekte in einem System unabhängig sind[PH10].

Auf Basis dieser Eigenschaft lassen sich nun ebenfalls verschiedene Arten von Pseudonymen ausmachen: Diese Arten unterscheiden sich durch Nutzung des Pseudonyms in verschiedenen Kontexten. Personen-, Rollen-, Beziehungs-, Rollen-Beziehungs- und Transaktionspseudonym sind mögliche Ausprägungen dieser Eigenschaft. Eine Übersicht über diese Pseudonymarten und deren Zusammenhang zur Verkettbarkeit ist Abbildung 2.1 zu entnehmen.

6. ursprünglich aus dem Griechischen stammend: *pseudonumon* - falsch benannt

Personenpseudonyme beschreiben Pseudonyme wie beispielsweise die Sozialversicherungsnummer, die stellvertretend für die eindeutige Identität des Subjekts in der Gesellschaft stehen. Ereignisse, die mit solchen Pseudonymen verbunden sind, lassen sich über die gesamte Gültigkeitsspanne des Pseudonyms verketten.
Rollen-, Beziehungs- und Rollen-Beziehungspseudonyme stellen Pseudonyme da, die das Subjekt nur in einer besonderen Funktion oder mit einem bestimmten Kommunikationspartner bzw. in der Kombination beider Möglichkeiten nutzt. Ein Beispiel hierfür wäre eine E-Mail-Adresse wie *administrator@unternehmen.de* in einem Unternehmen, die nicht das Subjekt als solches sondern nur in seiner Rolle in dem Unternehmen beschreibt. Ereignisse verbunden mit diesen Pseudonymarten lassen sich nur in bestimmten Kontexten, jedoch nicht über Kontextgrenzen hinweg, verknüpfen. Beispielsweise könnten zwei Kommunikationspartner mit derselben Person kommunizieren, ohne dies feststellen zu können, wenn die Person ihnen gegenüber unter unterschiedlichen Beziehungspseudonymen auftritt.
Transaktionspseudonyme stellen die stärkste Form der Unverkettbarkeit da. Bei dieser Art von Pseudonymen wird für jedes Ereignis ein neues Pseudonym verwendet, das daher nur ein einziges Mal auftritt und Verkettbarkeit verhindert.

Abgrenzung zur Anonymität sinnvoll?

Ist der „Schnitt“ zwischen Kapitel 2 und 3 hier sinnvoll?

2.5 Schwellwertschemata

Mit der Verbreitung technischer Systeme, die kryptographische Verfahren nutzen, in den 70er Jahren musste auch das Problem der sicheren Aufbewahrung und Verteilung kryptographischer Schlüssel betrachtet werden. Die Sicherheit dieser Schlüssel ist essentiell für den Betrieb solcher Systemen. Das einfache Speichern eines Schlüssels an einem einzigen Ort resultiert in einer hohen Verlustwahrscheinlichkeit, da ein einzelner Fehler wie unbeabsichtigtes Löschen oder Speichermedienausfall den Schlüssel unwiederbringlich verloren gehen lassen kann. Das mehrfache Speichern eines Schlüssels an verschiedenen Orten erhöht hingegen die Gefahr eines Schlüsseldiebstahls oder -missbrauchs, da auch die Angriffsoberfläche vergrößert wird. Bei möglichen Lösungen dieses Problems müssen also immer die Integrität und die Vertraulichkeit eines Schlüssels gegeneinander abgewogen werden. [Gem97]

Ausgehend von diesen Überlegungen entwickelte Shamir das erste (t,n)-Schwellwertschema: Ein Geheimnis D wird so in n Teile $D_1,\ldots,D_n$ (*Shares*) zerlegt, dass durch Kenntnis von mindestens t Teilen das Geheimnis wieder aufgedeckt werden kann, aber jede Kombination aus höchstens $t-1$ Teilen keine Informationen über D liefert [Sha79].[7]

Auf Basis dieses Verfahrens kann also die Integrität eines Schlüssels erhöht werden, da nun selbst bei Verlust von $n-t$ Teilen der Schlüssel noch wiederhergestellt werden kann. Auf der anderen Seite ist die Vertraulichkeit jedoch höher als bei der mehrfachen Speicherung des Schlüssels im Original, da mindestens t Teile des Schlüssels zur Wiederherstellung vorliegen müssen.

7. Im selben Jahr veröffentlichte auch Blakley eine Lösung dieses Problems, die auf den Schnittpunkten von Hyperebenen über endlichen Feldern beruht [Bla79].

Shamirs Verfahren wird im Folgenden im Detail beschrieben, da es auch im später erläuterten und verwendeten Schwellwertschema eine wichtige Rolle spielt.

2.5.1 Shamir's Secret Sharing

Die Menge aller Ganzzahlen modulo einer Primzahl p bilden den (endlichen) Körper $\mathbb{Z}_p$, dessen Eigenschaften für das Verfahren entscheidend sind. Soll das Geheimnis D (das o.B.d.A. als Ganzzahl angenommen wird) aufgeteilt werden, so wird eine Primzahl p mit $p > D$ und $p > n$ gewählt. Weiterhin wird ein Polynom

$$q(x) = a_0 + a_1 x + \cdots + a_{t-1} x^{t-1} \text{ mit } a_0 = D$$

derart gewählt, dass $a_1, \ldots, a_{t-1}$ zufällig gleichverteilt aus der Menge $\mathbb{Z}_p \setminus \{0\}$ stammen. Die einzelnen *Shares* werden nun als

$$D_1 = (x_1, q(x_1)), \ldots, D_i = (x_i, q(x_i)), \ldots, D_n = (x_n, q(x_n))$$

jeweils modulo p berechnet, wobei die x_i paarweise unterschiedlich aus $\mathbb{Z}_p$ gewählt werden können. Beispielsweise kann schlicht $x_i = i$ gelten.

Vielleicht diese Werte direkt nutzen?

Um nun aus diesen einzelnen Teilen wieder das ursprüngliche Geheimnis zu erhalten, wird das Verfahren der Langrange'schen Polynominterpolation verwendet, das ausgehend von einer Menge von Punkten ein Polynom findet, das durch diese Punke verläuft. Hierbei wird die Tatsache ausgenutzt, dass jedes Polynom vom maximalen Grad $t-1$ in einem mathematischen Körper durch t Punkte exakt bestimmt wird.

Für die zur Rekonstruktion verwendeten t Teile $D'_1 = (x'_1, q(x'_1)), \ldots, D'_t = (x'_t, q(x'_t))$ werden t Werte

mod p irgendwo erwähnen

$$\lambda_i := \prod_{j=1, j\neq i}^{t} \frac{-x'_j}{x'_i - x'_j} \text{ für } i \in \{1, \ldots, t\}$$

definiert, die auch als Lagrange-Koeffizienten bezeichnet werden. Das gesuchte Geheimnis D kann nun als

$$D = \sum_{i=1}^{t} \lambda_i \cdot q(x'_i)$$

berechnet werden. Da λ_i nicht von $q(x_i)$ abhängt, können diese Werte in der Praxis bereits vorberechnet werden. Details zu der Korrektheit dieses Verfahrens sind [BS16] zu entnehmen.

(Informationstheoretisc Sicherheit erwähnen

Das Problem dieser Lösung bezogen auf den in dieser Arbeit behandelten Anwendungsfall ist jedoch, dass das Geheimnis nach erstmaligem Aufdecken bekannt ist. Wünschenswert wäre ein Verfahren, bei dem nur ein entsprechend verschlüsseltes Datum (bspw. der gesuchte Eintrag in einer Pseudonym-Tabelle) aufgedeckt werden kann, ohne dass der kombinierte Schlüssel selbst bekannt wird.

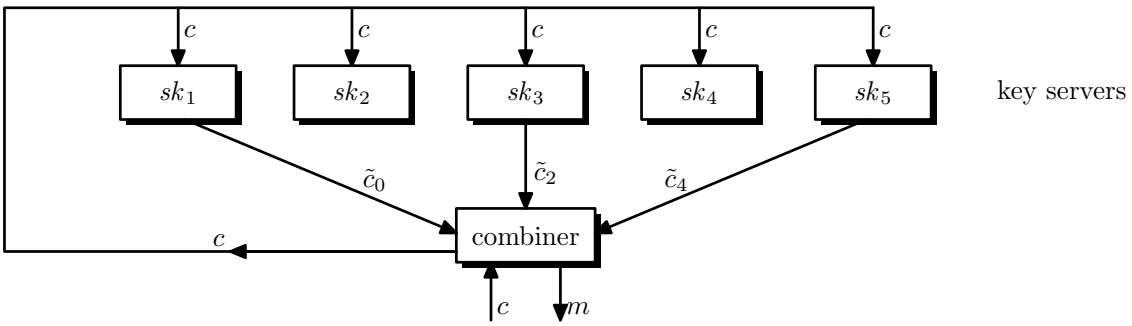

Abbildung 2.2: Übersicht über den Entschlüsselungsvorgang bei der Nutzung eines (3,5)-Schwellwertschemas. Entnommen aus [BS16].

2.5.2 Threshold Decryption

In [Des87] wird das erwähnte Problem das erste Mal im Kontext von verschlüsselten Nachrichten an Gruppen betrachtet: Ein Sender möchte eine Nachricht an eine Gruppe von Empfängern senden, die nur in Zusammenarbeit die Nachricht entschlüsseln können sollen. Hier wird auch die zentrale Forderung an mögliche Lösungen des Problems aufgestellt, den mehrfachen Nachrichtenaustausch zwischen Sender und Empfänger(n) bei der Entschlüsselung (sogenannte Ping-Pong-Protokolle) zu vermeiden.

In [Des93] spricht der Autor bei dieser Klasse von Verfahren von *Threshold Decryption* und fordert weiterhin, dass praktisch einsatzbare Systeme auch *non-interactive* sein sollten, also bei der Entschlüsselung keinen aufwendigen Datenaustausch zwischen den Besitzern der *Shares* notwendig machen.

In [BS16] werden diese Systeme formalisiert. Ein *Threshold-Public-Key-Decryption*-Schema $\varepsilon = (G, E, D, C)$ besteht aus vier Algorithmen:

- $G(t, n, r)$ ist der Algorithmus zur Generation des öffentlichen Schlüssels pk und der n *Shares* des geheimen Schlüssels $\{sk_1, \ldots, sk_n\}$. t steht für die Anzahl der zur Entschlüsselung benötigten *Shares*, wie bereits beschrieben. r ist als stellvertretend für die einfließenden Zufallswerte zu betrachten.
- $E(pk, m, r)$ steht für den Algorithmus, der der Verschlüsselung eines Klartexts m mit dem öffentlichen Schlüssel pk dient. r dient der Vermeidung von *Known-Ciphertext-Angriffen*.
- $D(sk_i, c)$ ist der Algorithmus der für einen bestimmten *Share* und einen Schlüsseltext c eine partielle Entschlüsselung c'_j liefert.
- $C(c, c'_1, \ldots, c'_t)$ ist der Algorithmus, der aus dem Schlüsseltext c und aus t durch D generierten partiellen Verschlüsselungen wieder den Klartext m liefert. Dieser Algorithmus wird auch *Combiner* genannt.

Vmtl.? Mehr Details...

Zusätzlich wird von diesen Algorithmen die folgende Eigenschaft verlangt. Sie beschreibt die korrekte Entschlüsselung von validen Schlüsseltexten im Kontext eines Schwellwertschemas: Für alle möglichen Ergebnisse $(pk, \{sk_1, \ldots, sk_n\})$ von G, alle möglichen Nachrichten m und alle t-elementigen Teilmengen der *Shares* $\{sk'_1, \ldots, sk'_t\}$ soll für alle möglichen Schlüsseltexte $c = E(pk, m, r)$ gelten: $C(c, D(sk'_1, c), \ldots, D(sk'_t, c)) = m$.

Eine Übersicht über den Entschlüsselungsvorgang ist in Abbildung 2.2 zu finden. Dort sind die partiellen Entschlüsselungen und der *Combine*-Vorgang eines $(3,5)$-Schwellwertschemas dargestellt. Der Algorithmus D für die partielle Entschlüsselung läuft dabei auf den einzelnen *Key-Servern* ab.

In [BBH06] werden diese Algorithmen noch um einen fünften erweitert, der dazu dient, einzelne partielle Entschlüsselungen auf Validität zu überprüfen. Hierdurch können fehlerhaft handelnde *Key-Server* aufgedeckt werden. Hierzu wird auch der Algorithmus G verändert, der zusätzlich einen Validierungsschlüssel vk liefert:

1. $G(t,n,r)$ liefert nun $(pk, vk, \{sk_1, \ldots, sk_n\})$.

...

5. $V(pk, vk, c, c'_j)$ überprüft die j-te partielle Entschlüsselungen auf Validität.

Weiterhin wird für den neuen Algorithmus eine weitere Eigenschaft verlangt. Für jeden Schlüsseltext c und $c'_j = D(sk_i, c)$, wobei sk_i der i-te von G erstellte *Share* ist, gelte: $V(pk, vk, c, c'_j)$ liefert ein valides Ergebnis.

2.5.3 ElGamal-Kryptosystem

Das in Abschnitt 3.3 vorgestellte und in dieser Arbeit verwendete kryptographische Schwellwertschema basiert auf dem bereits erläuterten Secret Sharing von Shamir und einem von Taher ElGamal entwickelten asymmetrischen Verschlüsselungsverfahren [ElG85]. Die Grundlagen dieses Verfahrens sollen daher hier kurz vorgestellt werden.

Im Folgenden sei $\mathbb{G}$ eine zyklische Gruppe der primen Ordnung p und g ein Generator dieser Gruppe. Diese Parameter können öffentlich bekannt gegeben werden. Alle folgenden Berechnungen werden in $\mathbb{G}$ (also modulo p) ausgeführt.

Ein Teilnehmer wählt nun ein zufälliges Element $x \in \mathbb{Z}_p$. Dies ist der private Schlüssel des Teilnehmers. Er berechnet zusätzlich seinen öffentlichen Schlüssel $h = g^x$.
Um eine Nachricht m, die an den Teilnehmer geschickt werden soll, zu verschlüsseln, wird zuerst ein zufälliges Element $y \in \mathbb{Z}_p$ gewählt. Anschließend kann die Nachricht verschlüsselt als $(v,c) = (g^y, h^y \cdot m)$ versendet werden.
Zur Entschlüsselung berechnet der Empfänger $k' = (v^x)^{(-1)}$ und kann die Nachricht $m = c \cdot k'$ entschlüsseln. Dies funktioniert, da

$$c \cdot k' = (h^y \cdot m) \cdot (v^x)^{(-1)} = g^{xy} \cdot m \cdot g^{(-yx)} = m$$

gilt.

Die Sicherheit des Verfahrens beruht auf dem Diskreten-Logarithmus-Problem. Details und Beweise hierzu sind beispielsweise in [KL14] zu finden.

Erweitern, um später darauf zurückgreifen zu können

2.6 Searchable Encryption

Ist es nötig Searchable Encryption zu verwenden?
Reicht eine lokale Zuordnungstabelle (User -> Pseudonym)? Kompromittierung des Systems würde Schlüssel für Tokengenerierung oder äquivalent dazu Tabelle betreffen, oder?
Möglicher Vorteil: Eindeutige Pseudonymgenerierung kann in Store stattfinden, aber das wäre auch durch Create(enc-user), das Pseudonym liefert, möglich?

Einführung SE (aus dem DB-Bereich) und s.u.

Ein allgemeines SSE-Schema besteht aus vier effizient berechenbaren Algorithmen [WWC16]:

- **GenerateKey**(k) generiert einen geheimen Schlüssel K anhand eines (verfahrensabhängigen) Sicherheitsparameters k.
- **BuildIndex**(K, D) erstellt einen Suchwort-Index I aus dem generierten Schlüssel K und einer Dokumentenmenge D.
- **GenerateTrapdoor**(K, w) erstellt für ein spezielles Suchwort w mithilfe des Schlüssels K das Trapdoor-Element T_w für die Suche nach w.
- **Search**(I, T_w) liefert eine Menge von Dokumenten basierend auf einem Suchwort-Index I und einem Trapdoor-Element T_w.

Der Besitzer der Daten erstellt sich einen Schlüssel mithilfe von **GenerateKey** und generiert durch **BuildIndex** einen Suchwort-Index für seine Dokumente. Anschließend lädt er diese Dokumente in verschlüsselter Form zusammen mit dem Index auf den Server. Möchte der Besitzer nun alle Dokumente erhalten, auf die ein spezielles Suchwort zutrifft, so erstellt er für dieses Suchwort mithilfe von **GenerateTrapdoor** ein Trapdoor-Element und sendet dieses an den Server. Dort wird nun auf dem Suchwort-Index durch **Search** die Suche nach dem Trapdoor-Element ausgeführt, die eine Menge von verschlüsselten Dokumenten liefert, auf die das Suchwort zutrifft. Diese können zurück an den Besitzer gesendet werden, der sie lokal entschlüsseln kann.

Übersichtsdiagramm angelehnt an overview-sse, p.3

3 Stand der Wissenschaft und Auswahl von Verfahren

In diesem Kapitel soll der aktuelle Stand der Wissenschaftbezogen auf die in dieser Arbeit verwendeten Klassen von Verfahren betrachtet werden sowie ausgehend von den Anforderungen an das zu entwickelnde System ein passendes Verfahren ausgewählt und im Detail beschrieben werden. An geeigneten Stellen wird auf die im letzten Kapitel dargelegten Grundlagen zurückgegriffen.

Naja SIEM eher weniger Wissenschaft

Vielleicht als Übergang eine ein bisschen ausführlichere Beschreibung des Systems, um das Zusammenspiel der verschiedenen Verfahren besser zu verstehen? Dieser Teil könnte dann in der Einführung mit Verweis kürzer ausfallen.

3.1 SIEM-Systeme

Zur Zeit gibt es eine vielfältige Auswahl an SIEM-Systemen auf dem Markt, die die grundsätzlichen Aufgaben eines SIEM-Systems erfüllen und über diese hinausgehen: Splunk[1], QRadar von IBM[2] oder ArcSight von Micro Focus[3] sind nur einige Beispiele aus diesem Bereich.

Die Auswahl an Open-Source-Software in diesem Bereich ist jedoch sehr gering. Eine der wenigen Ausnahmen stellt OSSIM - ein SIEM-System der Firma AlienVault[4] - dar, das auf Basis weiterer quelloffener Lösungen aus dem Netzwerksicherheits-Bereich unter anderem die in Abschnitt 2.3 beschriebenen Funktionen bereitstellt. AlienVault bietet zusätzliche eine kommerzielle Variante seines Produkts namens USM an, das insbesondere in den Bereichen Event-Korrelation und Compliance-Reporting die Funktionalität von OSSIM übersteigt. Von der Entwicklungsarbeit die in USM fließt, profitiert jedoch auch OSSIM, beispielsweise durch die Aktualisierung von Plugins für die Einbindung von aktuellen Netzwerkgeräten.

3.1.1 OSSIM-Überblick

Im Folgenden soll eine Übersicht über die für diese Arbeit relevanten Komponenten von OSSIM und deren Zusammenspiel gegeben werden. Diese ist auch in Abbildung 3.1 dargestellt.

Den Kern des SIEM-Systems bildet der OSSIM-Server. Hier werden Events gespeichert sowie aggregiert und es findet die Korrelation von Events statt, die der Erkennung von Angriffen oder ungewöhnlichem Netzverhalten dient. Events und generierte Meldungen können über ein Web Interface betrachtet werden. Weiterhin können hier unter anderem Angaben zur Netzinfrastruktur bereitgestellt, Netzwerk- und Schwachstellenscanner bedient und sämtliche Informationen über den Netzwerkstatus eingesehen werden.

1. https://www.splunk.com
2. https://www.ibm.com/us-en/marketplace/ibm-qradar-siem
3. https://software.microfocus.com/en-us/software/siem-security-information-event-management
4. AlienVault OSSIM: The World's Most Widely Used Open Source SIEM
   https://www.alienvault.com/products/ossim

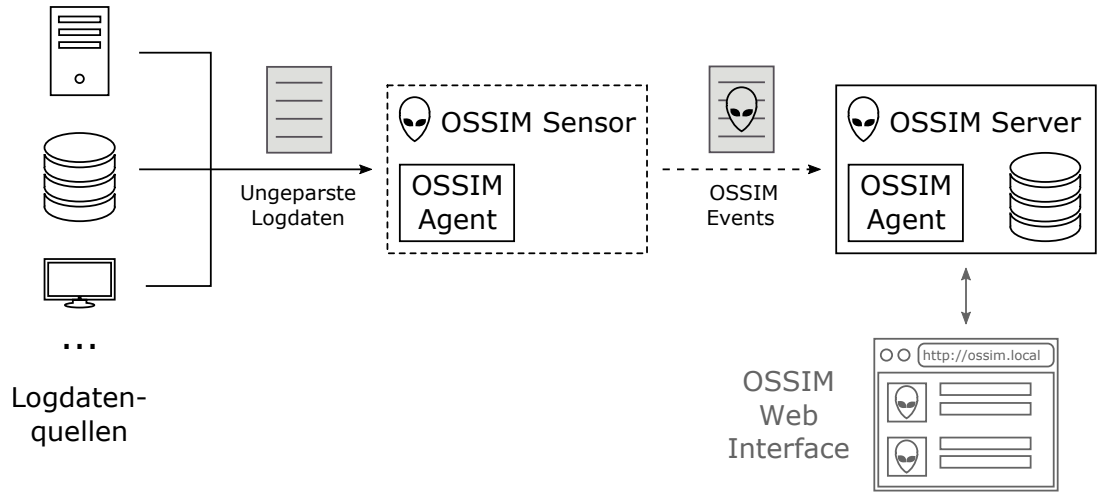

Abbildung 3.1: High-Level-Übersicht über die OSSIM-Architektur und den Datenfluss.

Der OSSIM-Agent ist dafür zuständig, vorliegende Logdaten zu parsen und in ein OSSIM-spezifisches Event-Format zu übersetzen. Auf diesen Vorgang wird im nächsten Abschnitt genauer eingegangen. Die erzeugten Events werden anschließend an den Server weitergeleitet. Der Agent befindet sich sowohl direkt auf dem Server als auch auf jedem installierten Sensor.

Eine OSSIM-Umgebung kann optional ein oder mehrere Sensoren nutzen, auf denen jeweils ein Agent seine Arbeit verrichtet. Dies wird im Folgenden verteilte Installation genannt. Der Vorteil dieser Lösung besteht darin, dass das aufwendige Parsen und Normalisieren von Logdaten verteilt staffinden und dadurch die Serverlast in großen Umgebungen reduziert werden kann. Kommt kein externer Sensor zum Einsatz, so spricht man von einer All-In-One-Installation.

3.1.2 Parsen von Logdaten in OSSIM

Besonders von Bedeutung für diese Arbeit ist die Verarbeitung von Logdaten. OSSIM ermöglicht es, Logdaten aus unterschiedlichen Quellen entgegenzunehmen bzw. aktiv selber abzurufen und in ein gemeinsames Event-Format zu übersetzen. Hierzu stehen verschiedene Möglichkeiten zur Verfügung:

- Entgegennehmen von Daten über das Syslog-Protokoll
- Beschaffen von Daten über das SNMP-Protokoll
- Entgegennehmen von Daten über proprietäre Protokoll wie SDEE oder WMI
- Beschaffen von Daten durch Datenbankabfragen

Unabhängig von der Datenquelle funktioniert die Verarbeitung der Logdaten nach dem immer gleichen Schema. OSSIM bietet die Möglichkeit mitgelieferte oder selber entwickelte Plugins für verschiedene Datenquellen zu aktivieren. Für eintreffende Logdaten überprüft der Agent anhand von regulären Audrücken, ob ein Plugin für das entsprechende Datum zuständig ist. Ist so ein Plugin gefunden, so wird ein neues OSSIM-Event angelegt und anhand der Angaben im Plugin die entsprechenden vorgegebenen Felder des Events gesetzt. Hierbei kann es sich beispielsweise um den Zeitpunkt des Events, IP-Adresse und Port der Datenquelle, einen zu

dem Event gehörigen Netzwerkbenutzer oder ereignisabhängige selbstgesetzte Felder handeln. Anschließend folgt die Weiterleitung des Events an den Server.

3.2 Pseudonymisierung

Der Begriff der Pseudonymisierung beschreibt die Benutzung von Pseudonymen zur Identifizierung von Subjekten. Ein Pseudonym (im technischen Sinne) kann nach [PH10] als einfache Bitkette betrachtet werden. Es sollte zufällig generiert werden, d.h. vollkommen unabhängig von dem zugehörigen Subjekt oder es betreffenden Eigenschaften sein, um keine Rückschlüsse aus dem Pseudonym selbst zu ermöglichen. Ein Negativbeispiel wäre ein nutzervergebenes Pseudonym, das den Namen des Haustiers enthält. Aber beispielsweise auch eine aufsteigende Nummerierung als Pseudonym könnte durch den hierdurch genauer spezifizierten Erstellungszeitpunkt Rückschlüsse auf das Subjekt hinter dem Pseudonym ermöglichen.

Pseudonymisierung sagt erst einmal lediglich etwas über die Verwendung eines Verfahrens aus, jedoch nichts über die daraus entstehenden Auswirkungen auf die Identifizierbarkeit eines Subjekts oder die Zurechenbarkeit bestimmter Aktionen. Hierfür spielen nach [PK01] weitere Eigenschaften von Pseudonymen wie die folgenden eine Rolle:

- garantierte Eindeutigkeit von Pseudonymen
- Möglichkeit von Pseudonymänderungen
- begrenzt häufige Verwendung von Pseudonymen
- zeitlich begrenzte Verwendung von Pseudonymen
- Art der Pseudonymserstellung

Um die Auswirkungen dieser Eigenschaften einordnen zu können, soll im Folgenden kurz die Pseudonymisierung in zwei Systemen betrachtet und die Relevanz der eben genannten Eigenschaften verdeutlicht werden: Pseudonyme in Mobilfunknetzen und in der Fahrzeug-zu-Fahrzeug-Kommunikation.

3.2.1 Mobilfunknetze

In Mobilfunknetzen wird zur Identifikation eines Teilnehmers anstelle seiner identifizierenden *International Mobile Subscriber Identity* meist ein Pseudonym – die *Temporary Mobile Subscriber Identity* (TMSI) – genutzt, das in bestimmten Situationen gewechselt wird und so Ortung und Bewegungsprofile der Teilnehmer verhindern soll. In [Ara+14] beschreiben die Autoren Schwächen der Implementierungen von Mobilfunkstandards in Netzen bei der (Neu-)Vergabe einer TIMSI. Bestimmte Eigenschaften im Bezug auf die Unverkettbarkeit von Pseudonymen und damit auf die Privatsphäre der Nutzer werden in vielen Netzen aufgrund einiger Schwächen nicht erreicht:

- Zu seltene Änderung der Pseudonyme
- Keine nutzungsabhängige Änderung von Pseudonymen
- Pseudonyme werden in verschiedenen Bereichen beibehalten

- Anfälligkeit der Neuvergabe für Replay-Angriffe

Die letzten beiden Schwächen sind für den Anwendungskontext dieser Arbeit nicht relevant, aber die zeit- und aktivitätsabhängige Neuvergabe von Pseudonymen müssen auch hier beachtet und umgesetzt werden.

3.2.2 VANets

Ein anderer Bereich, der sich besonders mit der Nutzung von Pseudonymen beschäftigt hat, ist die Forschung an Vehicular Ad Hoc Networks (VANets). Hierbei handelt es sich um Netzwerke für die Kommunikation zwischen Fahrzeugen, die beispielweise für die Datenübermittlung zur Bremserkennung naher Fahrzeuge oder die Stauerkennung genutzt werden können. Um die Privatsphäre der Fahrzeughalter zu schützen, wird für die Kommunikation in vielen Ansätzen auf die Verwendung von Pseudonymen gesetzt. So soll sich beispielsweise das Anlegen von Bewegungsprofilen verhindern lassen.
Unter anderem in [Dö05] und [Pet+15] widmen sich die Autoren der Nutzung von Pseudonymen in VANets und den besonderen Anforderungen, die diese erfüllen müssen – insbesondere auch im Hinblick auf die Häufigkeit von Pseudonymwechseln. Es ergibt sich, dass die Häufigkeit und Situation[5], in der Pseudonymwechsel stattfinden sollten, abhängig vom gewünschten Grad an Anonymität bzw. Angreifermodell sind und außerdem gegenüber Sicherheitsanwendungen[6] abgewogen werden müssen.
Bei der Nutzung von Pseudonymen in VANets handelt es sich natürlich um eine Anwendung mit anders gelagerten Prioritäten im Vergleich zu dem Kontext dieser Arbeit. Dennoch wird deutlich, dass die Strategie zum Pseudonymwechsel stark von der Anwendungssituation abhängig ist. Bezogen auf den hier vorliegenden Anwendungsfall werden insbesondere Besonderheiten der Datenquelle, wie die Häufigkeit von auftretenden Überwachungsdaten, und Anforderungen an die Verknüpfbarkeit von Ereignissen der auf den Daten beruhenden Anomalieerkennung zu beachten sein.

In [SMK09] stellt der Autor eine weitere Anforderung an die Nutzung von Pseudonymen in VANets, die jedoch nicht nur für diesen speziellen Anwendungsfall relevant ist: Er verlangt, dass die Aufdeckung eines Pseudonyms keine Informationen über die Identität eines Nutzers im Bezug auf andere Pseudonyme ermöglichen sollte. Diese Eigenschaft bezeichnet er als *Perfect Forward Privacy*[7].

5. Es werden beispielweise Lösungen vorgestellt, die abhängig von Geschwindigkeitsänderungen, einer gewissen Anzahl anderer Fahrzeuge oder besonderen Verkehrssituationen wie Kreuzungen die Pseudonymänderung vornehmen. Das Ziel ist hier immer die Möglichkeit der Pseudonymverkettung bzw. der Bewegungsprofilerstellung durch die äußere Situation der Pseudonymänderung zu erschweren.
6. Beispielsweise wäre zur VANet-basierten Kollisionsvermeidung eine Verkettung von Orten, an denen sich ein Fahrzeug zu verschiedenen Zeitpunkten befindet, erstrebenswert.
7. Die Bezeichnung ist an *Perfect Forward Secrecy* angelehnt. Diese Eigenschaft beschreibt ein ähnliches Verhalten bei der verschlüsselten Kommunikation: Ein Angreifer, der in den Besitz des Langzeitschlüssels eines Kommunikationspartners kommt, sollte trotzdem nicht in der Lage sein, bereits aufgezeichnete Nachrichten entschlüsseln zu können.

3.2.3 Pseudonymisierung im zu entwickelnden System

Aus diesen Vorüberlegungen können nun die Rahmenbedingungen der in dieser Arbeit verwendeten Pseudonymisierung aufgestellt werden. Pseudonyme sollten als zufällig gewählte Bitketten hinreichender Länge gewählt werden. Ihre Eindeutigkeit muss sichergestellt werden.

Wie auch in den Beispielen deutlich wurde, müssen Pseudonyme abhängig von dem Anwendungsszenario in bestimmten Fällen für einen Benutzer gewechselt werden. In dem hier vorliegenden Anwendungsfall, in dem Pseudonyme für die Zuordnung von eintreffenden Überwachungsdaten in Unternehmensnetzen genutzt werden, sind insbesondere die Zeitabhängigkeit sowie die Abhängigkeit von der Nutzungshäufigkeit für die Pseudonymwechselstrategie ausschlaggebend. Verschiedene Nutzeraktionen sollten nur in einem gewissen zeitlichen Rahmen und nur in einer gewissen Häufigkeit verkettbar sein.

Eine über diese generelle Aussage hinausgehende Bewertung davon, wie diese Abhängigkeiten konkret zu implementieren sind, ist jedoch im Rahmen dieser Arbeit nicht zu leisten. Hierfür sind zwei Gründe ausschlaggebend:

- Sie hängen stark von den Eigenschaften der Datenquellen ab, die die Überwachungsdaten liefern. Beispielsweise wäre das Datenprofil, das von einem elektrischen Türschließsystem geliefert wird, sehr unterschiedlich zu dem, das beispielsweise Zugriffe auf einen Netzwerkspeicher protokolliert. Im ersten Fall würden im Allgemeinen selten Daten anfallen, die zudem durch die Anwendung von Hintergrundwissen (Benutzer wird beim Betreten eines Raumes beobachtet) eher zur Aufdeckung eines Pseudonyms führen könnten. Hier wären wahrscheinlich häufige nutzungsabhängige Wechsel angebracht. Eventuell wäre sogar der Extremfall einer einmaligen Pseudonymvergabe pro Aktion in Erwägung zu ziehen.
  Im zweiten Fall hingegen würden im Allgemeinen häufig Daten anfallen und erst die Verkettung dieser Daten könnte hilfreiche Rückschlüsse auf vorliegende Anomalien liefern. Ein einzelner Datenzugriff hätte meist wenig Aussagekraft, wohingegen ein massenhafter Zugriff beispielsweise auf die Kundendatenbank eines Unternehmens durch einen gekündigten Mitarbeiter möglicherweise auf Datendiebstahl schließen lassen könnte.
- Weiterhin muss die Pseudonymwechselstrategie auch abhängig von der später auf den pseudonymisierten Überwachungsdaten auszuführenden automatisierten Anomalieerkennung sein. Je nachdem welche Verfahren auf Daten aus welchen Datenquellen eingesetzt werden sollen, könnte auch hier unterschiedliche Verknüpfbarkeit der Daten erforderlich sein. Hieraus ergibt sich auch ein Spannungsfeld zwischen den Anforderungen der Anomalieerkennung gegenüber der Verknüpfbarkeit der Daten und damit der Privatsphäre der Arbeitnehmer.

Aus diesen Gründen wird eine parameterabhängige Pseudonymwechselstrategie implementiert, die sowohl zeit- als auch die nutzungsabhängige Wechsel ermöglicht. Wie lange bzw. häufig ein Pseudonym verwendet wird, kann so in konkreten Anwendungen mit gesetzten Rahmenbedingungen beurteilt und gesetzt werden. Dieses Vorgehen wird auch in den *Leitlinien für die rechtssichere Nutzung von Pseudonymisierungslösungen unter Berücksichtigung der Datenschutz-Grundverordnung* beschrieben: „Abhängig vom Anwendungsfall sind – zeit- oder datenvolumenabhängig – geeignete Intervalle zu definieren, in denen ein Wechsel [...] erfolgt.“[SW17]

Weiterhin wird angestrebt für die Pseudonyme bzw. ihre Aufdeckung die erwähnte Perfect Forward Privacy zu ermöglichen. Die konkrete Umsetzung dieser Eigenschaft wird in einem späteren Abschnitt beschrieben werden.

Art des gewählten Pseudonymverfahrens basierend auf basics-pseudonymity erwähnen

3.3 Schwellwertschemata

Aufbauend auf den Ideen von Shamir und Blakley und den ersten Ideen zu kryptographischen Schwellwertschemata wurden für verschiedene Algorithmen und Anwendungsfälle Schemata mit unterschiedlichen Eigenschaften entwickelt.

3.3.1 Übersicht

Eine Vielzahl von Veröffentlichungen behandlen das Problem der verteilten Erstellung von Signaturen: Die in [Sho00] entwickelte Lösung basiert auf dem RSA-Verfahren, [Gen+96b] erweitert den DSS-Standard um ein Schwellwertschema und [SS01] entwickelt ein Schema zur verteilten Signatur mittels Schnorr-Signaturen.
Weitere Forschungen haben sich mit der Entwicklung von RSA-basierten Schwellwertschemata zur verteilten Entschlüsselung beschäftigt, die im Kontext dieser Arbeit genutzt werden [Fra+97; Gen+96a; Rab98].
Ein zusätzliches Verfahren, das im Zusammenhang mit verteilter Entschlüsselung Aufmerksamkeit erfuhr, ist das Paillier-Kryptosystem. In [DJ01] und [FPS00] entwickelten die Autoren auf diesem System basierte Schwellwertschemata, die insbesondere durch ihre homomorphe Eigenschaft hervorstechen und dadurch im Bereich der elektronischen Wahlsysteme genutzt werden können.
Einen Überblick über weitere Veröffentlichungen in diesem Bereich bieten beispielsweise [Des97], [Gem97] und [Des93].

3.3.2 ElGamal-basiertes Schwellwertschema

Ein Verfahren zur *Threshold Decryption*, das auf auf einer geschickten Kombination von Shamir's Secret Sharing (Abschnitt 2.5.1) und des ElGamal-Kryptosystems (Abschnitt 2.5.3) basiert, veröffentlichten die Autoren in [DF90]. Aufbereitete Darstellungen lassen sich in [KL14] und [BS16] finden.
Es ist eines der ersten veröffentlichten Schwellwertschemas und erfuhr dadurch viel Beachtung und entsprechend viele aufbauende Arbeiten, die Verbesserungen vorschlugen. Durch die hinterliegende Mathematik bietet das Schema einfachere Umsetzbarkeit (auch von Erweiterungen wie dezentraler Schlüsselgenerierung) gegenüber RSA[8]. Dies gilt ebenso gegenüber den Paillier-basierten Schemata – deren homomorphe Eigenschaften in dieser Arbeit nicht benötigt werden. Aus diesen Gründen fiel die Wahl des in dieser Arbeit umzusetzenden Schemas auf das genannte Verfahren.

8. Das ElGamal-Verfahren nutzt zur Berechnung eine öffentlich bekannter Ordnung (sie ist Teil des öffenltichen Schlüssels). Im Gegensatz dazu werden Berechnungen bei RSA in $\Phi(n)$ ausgeführt, das jedoch nicht öffentlich vorliegen darf [Ngu05].

Der Rest dieses Abschnitts stellt das Verfahren nun entsprechend den in Abschnitt 2.5.2 aufgeführten Algorithmen eines Threshold-Public-Key-Decryption-Systems im Detail vor.

Algorithmus G: Schlüsselgenerierung

In dem Verfahren wird für die Schlüsselgenerierung eine zentrale, vertrauenswürdige Instanz vorausgesetzt, die den öffentlichen Schlüssel und die später benötigten Shares des geheimen Schlüssels erzeugt und verteilt.

Zur Erzeugung werden zwei Primzahlen p und q mit der Eigenschaft $p = 2q + 1$ - bekannt als sichere Primzahl bzw. Sophie-Germain-Primzahl - benötigt. Weiterhin ist ein Generator der Untergruppe der Ordnung q von $\mathbb{Z}_p^*$ notwendig.

Der (temporär erstellte) geheime Schlüssel $a \in \mathbb{Z}_q$ wird analog zu der Schlüsselgenerierung im ElGamal-Verfahren zufällig gewählt. Aus ihm wird der öffentliche Schlüssel $pk = g^a \mod p$ berechnet.
Der geheime Schlüssel wird anschließend analog zu Shamirs Secret Sharing in $\mathbb{Z}_q$ in einzelne Shares $(x_i, y_i) = (x_i, q(x_i))$ aufgeteilt und diese an die Teilnehmer verteilt. Anschließend werden diese Werte gelöscht, so dass nur noch die Teilnehmer im Besitz ihrer Shares und damit in der Lage sind, Schlüsseltexte zu entschlüsseln.

Algorithmus E: Verschlüsselung

Anschließend kann ein Klartext mithilfe von pk analog zu dem ElGamal-Verfahren verschlüsselt werden. So erhält man $(v, c) = (g^k, m \cdot g^{ak})$ für ein durch den Sender zufällig gewähltes $k \in \mathbb{Z}_q$.

Algorithmus D: Partielle Entschlüsselung

Jeder Besitzer eines Shares (x_i, y_i) kann nun für den zu entschlüsselnden Schlüsseltext (v, c) seine partielle Entschlüsselung (x_i, v^{y_i}) berechnen und diese an eine zentrale Instanz, den Combiner, senden. Empfängt dieser mindestens t partielle Entschlüsselungen[9], so kann er den Klartext wiederherstellen.

Algorithmus C: Kombination

Hierzu berechnet der Combiner die Lagrange-Koeffizienten $\lambda_i \in \mathbb{Z}_q$ wie in Shamir's Secret Sharing beschrieben[10]. Anschließend kann

$$g^{ak} = \prod_{i=1}^{k} (v^{y_i})^{\lambda_i}$$

berechnet werden. Dies funktioniert, da

$$\prod_{i=1}^{k} (v^{y_i})^{\lambda_i} = \prod_{i=1}^{k} (g^k)^{y_i \cdot \lambda_i} = (g^k)^{\sum_{i=1}^{k} y_i \cdot \lambda_i} \overset{(*)}{=} (g^k)^a$$

gilt. Der letzte Schritt $(*)$ folgt direkt aus dem zugrundeliegenden Secret Sharing und ist in dieser Form bereits in Abschnitt 2.5.1 zu finden.

Anschließend kann der Klartext als $m = c \cdot (g^{ak})^{(-1)}$ wiederhergestellt werden.

9. Zur Erinnerung: t beschreibt die Mindestzahl zur Entschlüsselung benötigter Shares des Schwellwertschemas.
10. In diesem Abschnitt gilt $i \in C$. C stellt dabei die Menge der Indizes der beteiligten Sharebesitzer dar. Es gilt also $C \subseteq \{1, \ldots, n\}$ und $|C| \geq t$.

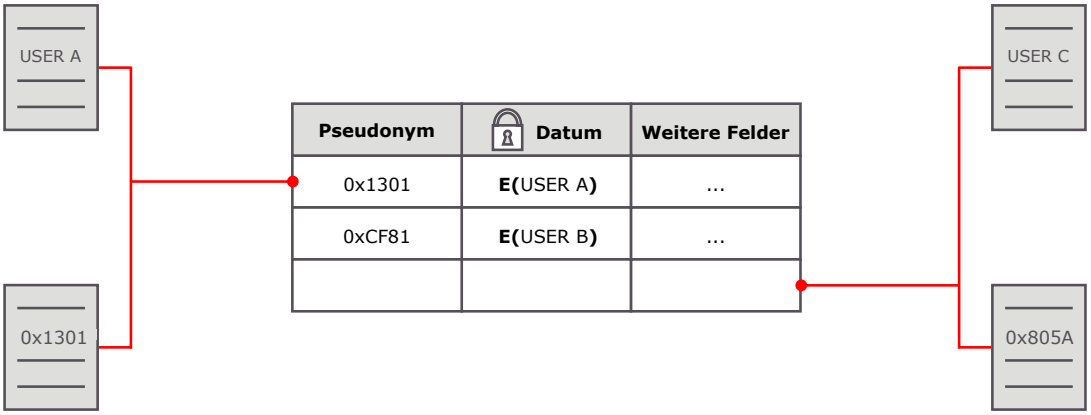

Abbildung 3.2: Erhalt von Pseudonymen aus der Zuordnungstabelle: Links für den Fall eines bereits bekannten Datums (hier Benutzer), rechts für ein unbekanntes Datum.

3.3.3 Existierende Implementierungen

Auch nach umfangreicher Recherche ließ sich keine quelloffene, kryptographisch überprüfte und lizenzrechtlich nutzbare Bibliothek finden, die das gewünschte Schwellwertschema implementiert. Es gab verschiedene verwandte Lösungen wie Civitas[11] oder Helios[12], die jedoch alle eng verknüft mit dem Anwendungskontext der elektronischen Wahl verknüpft waren und dadurch andere Anforderungen erfüllten als sie für diese Arbeit erforderlich sind. Aus diesem Grund wird das Schwellwertschema gezwungenermaßen eigenständig implementiert.

3.4 Searchable Encryption

Trifft ein neues Datum in dem System ein und soll pseudonymisiert werden, so muss überprüft werden, ob bereits ein Pseudonym für das Datum vergeben wurde. Da die Daten jedoch in verschlüsselter Form vorliegen, stellt sich die Frage, wie diese Überprüfung erreicht werden kann. In Abbildung 3.2 ist dieses Problem noch einmal anhand eines Beispiels dargestellt. Zu einem Zeitpunkt liegen in der Pseudonymtabelle Pseudonyme für zwei Benutzer *User A* und *User B* vor. Das diese Benutzer zu den Pseudonymen gehören, ist jedoch nicht offensichtlich, da die Daten verschlüsselt gespeichert sind. Sollen nun neue Logdaten verarbeitet werden, liegen zwei mögliche Fälle vor:

- Es liegt bereits ein Pseudonym für den Benutzer vor (linker Bereich in der Darstellung). Hier muss das bereits vergebene Pseudonym *0x1301* verwendet werden.
- Es liegt noch kein Pseudonym für den Benutzer vor (rechter Bereich in der Darstellung). Nun wird ein neues Pseudonym *0x805A* angelegt und verwendet werden.

Verschiedene Lösungmöglichkeiten inklusive ihrer Vor- und Nachteile für das Problem, wie nun trotz der Verschlüsselung der Daten dieses Verhalten erreicht werden kann, werden in diesem Abschnitt beleuchtet.

11. Civitas – A secure voting system. http://www.cs.cornell.edu/projects/civitas/
12. Helios Voting. https://heliosvoting.org/

3.4.1 Entschlüsseln aller Datensätze

Da für die Verschlüsselung ein kryptographisches Schwellwertschema verwendet wird, scheiden zwei triviale Möglichkeiten aus: Das Entschlüsseln aller Datensätze zur Überprüfung wäre nicht nur unter Performance-Gesichtspunkten nicht wünschenswert. Es darf auch nicht möglich sein, da die *Shares* zur Entschlüsselung nicht am Ort der Verschlüsselung vorliegen dürfen. Dies ist eine der Grundannahmen, die der Sicherheit des Systems zugrundeliegen.

3.4.2 Deterministische Verschlüsselung

Die zweite ausscheidende Möglichkeit wäre das Überprüfen aller Einträge auf Schlüsseltextgleichheit. Bei Gleichheit eines Eintrages könnte das entsprechende Pseudonym zurückgeliefert werden. Hierzu müsste ein deterministisches Verschlüsselungsverfahren genutzt werden, das ein Datum immer auf den gleichen Schlüsseltext abbildet. Diese Möglichkeit scheidet jedoch aus, da es sich bei dem verwendeten Schwellwertschema um ein Public-Key-Verfahren, bei dem bei der Verschlüsselung ein Zufallswert einfließt – folglich ein nicht-deterministisches Verfahren, handelt.
Dieser Nicht-Determinismus ist notwendig, da ansonsten zur Aufdeckung eines Pseudonyms auch ein Wörterbuch-Angriff mithilfe des öffentlichen Schlüssels des Schwellwertschemas genutzt werden könnte. Ein Angreifer würde alle möglichen Werte, die ein Datum annehmen kann, mit dem öffentlichen Schlüssel verschlüsseln und mit dem aufzudeckenden Eintrag vergleichen. Gleichheit der Schlüsseltexte würde den gesuchten Klartext liefern. Der im Kontext dieser Arbeit vorliegende kleine und bekannte Wertebereich (wie beispielsweise Mitarbeiternamen) würde diesen Angriff relativ effizient machen. Aus diesem Grund muss ein nicht-deterministisches Verschlüsselungsverfahren genutzt werden und damit ist die Überprüfung auf Schlüsseltextgleichheit nicht möglich.

3.4.3 Nutzung von Hashfunktionen

Ein weiterer Ansatz, der ebenfalls anfällig für diese Art von Wörterbuch-Angriff wäre, ist die Verwendung von (kryptographisch sicheren) Hashfunktionen zur Suche: Neben dem Pseudonym und dem verschlüsselten Datum wird ein Hash des Datums abgespeichert. Muss nun für ein neues Datum überprüft werden, ob bereits ein Pseudonym vorliegt, kann der Hash des Datums gebildet und mit allen vorliegenden Hashes verglichen werden. Bei Übereinstimmung wäre das Datum (mit großer Wahrscheinlichkeit) gleich dem verschlüsselten Datum und das Pseudonym könnte genutzt werden. Diese Möglichkeit ist jedoch durch den bereits erwähnten kleinen Wertebereich ebenso anfällig für einen Wörterbuch-Angriff: Ein Angreifer könnte mithilfe der bekannten Hashfunktion die Hashes aller Werte berechnen und mit dem des aufzudeckenden Pseudonyms vergleichen.

3.4.4 Lokale Zuordnung

Eine andere Möglichkeit ist die Anlage einer vor externem Zugriff geschützten Zuordungstabelle zwischen Datum und Pseudonym am Ort der Ersetzung. Bei Eintreffen eines neuen Datums kann in der Tabelle das zugehörige Pseudonym ermittelt werden oder falls es noch nicht existiert, ein

neues Pseudonym erstellt und zusammen mit dem verschlüsselten Datum gespeichert werden. Diese Lösung wirda auch in [Goh03] erwähnt.

Nachteil bezogen auf das für diese Arbeit zu entwickelnde System ist jedoch die notwendige Generierung von Pseudonymen und Speicherung der Zuordnungstabelle an der Stelle, an der neue Daten eintreffen. Hierdurch wird ein relativ leichtgewichtiger Proxy-Server wie er in der Architektur vorhergesehen ist (ein Vorgriff auf Abschnitt 4.2) verhindert. Außerdem würde eine Kompromittierung dieser Komponente direkt zur Aufdeckung des Pseudonymzusammenhanges führen.

3.4.5 Message Authentication Codes

Aufbauend auf der Hash-basierten Lösung lässt sich jedoch auch eine nicht für einen Wörterbuch-Angriff anfällige Lösung entwickeln. Dazu wird der verwendete Hash durch einen schlüsselabhängigen Message Authentication Code (MAC) ersetzt. Beim Speichern eines neuen Eintrags wird dazu unter Zuhilfename eines zufällig generierten Schlüssels ein MAC über das Datum berechnet und mit dem Eintrag gespeichert. Für ein Datum kann jetzt durch Überprüfen aller MACs bestimmt werden, ob bereits ein Pseudonym vergeben wurde. Ein Angreifer kann den beschriebenen Wörterbuch-Angriff jedoch ohne Kenntnis des Schlüssels nicht ausführen.

Bei dieser Lösung handelt es sich um eine einfache Form der Searchable Symmetric Encryption wie sie in Abschnitt 2.6 dargestellt ist. Die durch Pseudonyme zu ersetzenden Daten bilden die zu durchsuchenden Dokumente. Der MAC bildet den Suchwort-Index für jeden verschlüsselten Eintrag und wird so auch als Trapdoor-Element für die Suche nach einem passenden Eintrag genutzt. Ausgehend von den Anforderungen des umzusetzenden Systems eignet sich dieser Ansatz: Er ermöglicht einer Komponente die Zuordnung eines Datums zu einem Pseudonym, ohne dass dass diese Zuordnung direkt gespeichert werden muss oder der Datenbank bei der Abfrage bekannt wird. Auch ein Wörterbuchangriff, der bei dem erwähnten kleinen Wertebereich geringen Aufwand bedeutete, wird verhindert. Aus diesen Gründen wird dieser Lösungsansatz in einem späteren Schritt im zu entwickelnden System umgesetzt.

Die Nutzung von deterministischer Verschlüsselung (mit der hier beschriebenen Verwendung eines MACs als Sonderfall) wird erstmals in [BBO07] beschrieben. Dort werden auch einige Schwächen dieser Lösung diskutiert: Die Datenbank erfährt durch den Suchindex bereits einiges über die gespeicherten Dokumente, da durch die deterministische Struktur gleiche Suchwörter auf gleiche Trapdoor-Elemente abgebildet werden. Diese Schwäche ist im Bezug auf den besonderen Anwendungsfall dieser Arbeit jedoch zu vernachlässigen, da Dokumente (meint Benutzernamen, ...) nur einmalig vorliegen dürfen. Eine weitere Schwäche, die auch in dieser Arbeit beachtet werden muss, ist, dass die Datenbank etwas über die Häufigkeit verschiedener Suchanfragen erfährt, da die Trapdoor-Elemente für ein bestimmtes Datum immer gleich sind. Durch Korrelation mit Hintergrundwissen könnte so möglicherweise in bestimmten Fällen der Inhaber eines Pseudonyms herausgefunden werden.

Hash-Kollisionen und Geburtstags-Paradoxon hier erwähnen?

3.4.6 Weitere Möglichkeiten der Searchable Encryption

Die im letzten Abschnitt betrachtete MAC-basierte Lösung funktioniert für den in dieser Arbeit behandelten Anwendungsfall. Bei der Abbildung Pseudonym zu Datum (wie den Benutzer-

name) handelt es sich um eine 1:1-Abbildung, die lediglich von einer Komponente – dem zu entwickelnden Syslog-Proxy – abgefragt wird.

Referenz auf XY?

In anderen Umgebungen bzw. Erweiterungen des in dieser Arbeit betrachteten Anwendungsfalls kann es jedoch auch andere Anforderungen geben. Vorstellbar wäre beispielsweise die verteilte Abfrage der Datenbank nach existierenden Pseudonymen. Für den MAC-basierten Ansatz müsste dazu zumindest der genutzte Schlüssel verteilt werden, was Kommunikation innerhalb der verteilten abfragenden Komponenten erfordert. Zusätzlich müssten auch die Sicherheitsauswirkungen dieser Lösung betrachtet werden.
Eine andere Erweiterung könnte die Mehrfachverwendung von Pseudonymen für verschiedene Merkmale eines Benutzers (Name, IP-Adresse, Signaturschlüssel, ...) sein, um die Erkennung von Insiderangriffen zu verbessern. Auch diese Erweiterung wäre mit dem MAC-basierten Ansatz nicht direkt abbildbar.

Andere Lösungsansätze für die in diesen Fällen entstehenden Probleme könnten Forschungsergebnisse aus dem Bereich der *Searchable Encryption* bieten. In [SWP00] wurde von den Autoren das erste Searchable-Symmetric-Encryption-Schema entwickelt (siehe auch Abschnitt 2.6). Verbesserte Schemata folgten in [Goh03] und [CM05]. Durch diese sind variable Dokumentenlängen mit beliebig vielen Suchwörtern möglich, die auch nachträglich erweiterbar sind. Für den Aspekt der verteilten Abfrage könnten die Ergebnisse aus [Bon+04] genutzt werden, in dem sich die Autoren mit der Suche in mit asymmetrischen Verfahren verschlüsselten Daten befassen.
Eine Übersicht über weitere Ergebnisse in diesem Bereich lässt sich in [WWC16] finden.

4 Implementierung

Einführender Absatz

4.1 Anforderungen

Der umgesetzte Prototyp soll es ermöglichen, aufbauend auf dem bestehenden Open-Source-SIEM-System OSSIM Logdaten mittels Pseudonymisierung und Schwellwertschemata so zu verändern, dass diese erst durch Kollaboration einer bestimmten Anzahl an Teilnehmern wieder aufgedeckt werden können. Neben dieser primären Anforderung sollte er noch weitere wie die Erweiterbarkeit um weitere Dateschutztechniken erfüllen. Alle diese Anforderungen sollen im folgenden Abschnitt näher erläutert werden. Zuerst wird jedoch ein Angreifermodell für die prototypische Umsetzung aufgestellt.

4.1.1 Zugrundeliegendes Angreifermodell

Im Kontext dieser Arbeit, die sich mit der datenschutzfreundlichen Speicherung von Überwachungsdaten beschäftigt, muss zuerst folgende Vorüberlegung getroffen werden: Logdaten erreichen das verwendete SIEM-System abhängig von den verwendeten Protokollen im allgemeinen nicht-pseudonymisiert und oftmals weder verschlüsselt noch mit geschützer Integrität über das Netzwerk. Diese Angriffsmöglichkeiten zu verhindern, ist ausdrücklich kein Ziel dieser Arbeit. Daher bezieht sich das Angreifermodell auf die Bearbeitung und Speicherung von Logdaten, erst sobald sie das zu entwickelnde System erreichen, jedoch nicht vorher.

Angriffsmöglichkeiten erläutern: Mitschneiden aller Daten und damit Aufdecken von Pseudonymen, ...

Das Ziel des Systems bezogen auf seine Sicherheit lässt sich folgendermaßen definieren: Das Pseudonym eines Nutzers erlaubt (ohne Anwendung von Hintergrundwissen) keinen Rückschluss auf die Identität eines Nutzers. Erst die Kollaboration berechtigter Akteure ermöglicht das Aufdecken eines Pseudonyms.

? außerdem erweitern um aktive Angreifer?

Ein Angreifermodell beschreibt die maximale Stärke eines Angreifers im Bezug auf verschiedene Faktoren, gegen die ein System abgesichert ist. Enthalten sind die Rolle eines Angreifers, seine Verbreitung im System, aktives oder passives Verhalten und die Rechenkapazität, die der Angreifer zum Überwinden der eingesetzten Schutzmaßnahmen aufbringen kann. Es bildet die Basis für alle Folgeüberlegungen im Bezug auf die Sicherheit des zu entwickelnden Systems.

Für das Angreifermodell werden folgende Annahmen getroffen: Bei dem Angreifer kann es sich um einen Außenstehenden, um einen Berechtigten mit Zugriff auf das SIEM-System oder sogar um einen Administrator mit physischem Zugriff auf Rechner im Netz handeln. Wie bereits beschrieben, wird die ungesicherte Übertragung nicht-pseudonymisierter Daten zu dem System nicht betrachtet. Insofern wird von einem Angreifer ausgegangen, der erst Zugriff auf das zu entwickelnde System oder das SIEM-System beseitzt oder erlangt. Das zu entwickelnde System soll in der Lage sein, sich auch gegen aktive Angreifer zur Wehr zu setzen. Bezogen auf die verfügbare Rechenleistung des Angreifers sollen verbreitete und nach heutigem Wissensstand für

Denial of Service etc?

sicher befundene kryptographische Algorithmen als nicht mit vertretbarem Aufwand zu brechen angesehen werden. Es handelt sich daher um die Annahme von komplexitätstheoretischer Sicherheit.

4.1.2 Anforderungen an die Integration in OSSIM

Um zu beurteilen, an welcher Stelle in den Datenfluss der Logdaten in OSSIM (vergleiche Abschnitt 3.1) eingegriffen wird, um die Daten zu verändern, müssen Vor- bzw. Nachteile der verschiedenen Möglichkeiten gegeneinander abgewogen werden. Folgende Eigenschaften einer Möglichkeit sollten betrachtet werden:

- **Abhängigkeit von der OSSIM-Konfiguration:** Ist die Lösung sowohl in der verteilten als auch in der All-in-One-Installation von OSSIM umsetzbar? Eine Lösung, die unabhängig von der gewählten Installationsmethode funktioniert, ist zu bevorzugen, da sie universell einsetzbar ist.
- **Veränderung des SIEM-Systems:** Muss OSSIM für die Umsetzung der Lösung verändert werden? Dies wäre im Hinblick auf zukünftige Updates, die OSSIM durch den Entwickler erfährt, nicht wünschenswert, da jedes dieser Updates dafür sorgen könnte, dass die umgesetzte Lösung angepasst werden muss. Hierbei können sowohl direkte Änderungen an dem OSSIM-Quellcode als auch Eingriff in die interne Kommunikation betrachtet werden.
- **Nicht-pseudonymisierte Daten im SIEM-System:** Um das Ziel der Arbeit - die Pseudonymisierung, die nur durch Kollaboration aufgedeckt werden kann - zu erreichen, muss sichergestellt sein, dass Logdaten nirgendwo in nicht pseudonymisierter Form vorliegen. Da insbesondere das zukünftige Verhalten von OSSIM nicht beeinflusst werden kann, wäre es wünschenswert, dass die Logdaten bereits in pseudonymisierter Form das SIEM-System erreichen.
  Die Relevanz dieser Eigenschaft lässt sich bereits an einem einfachen Beispiel erkennen: Wird das Syslog-Protokoll genutzt, um Logdaten in OSSIM aufzunehmen, so werden die Einträge im Sensor erst in eine Logdatei gespeichert und von dort aus geparst, normalisiert und in der Datenbank gespeichert. Das Datum verbleibt in der Logdatei. Kommen die Daten in nicht-pseudonymisierter Form in dem OSSIM-Sensor an, so muss sichergestellt werden, dass verarbeitete Einträge gelöscht/verändert werden.
- **Mehrfaches Parsen von Logdaten:** Durch den OSSIM-Agenten werden die Logdaten - wie in Abschnitt 3.1 beschrieben - geparst und normalisiert. Aus Performancegründen ist eine Lösung zu bevorzugen, die dieses Parsen nicht mehrfach voraussetzt.

4.1.3 Anforderungen an die Pseudonymisierung

Die Pseudonymisierung muss es ermöglichen nach Aufdecken eines Eintrags wieder auf den ursprünglichen Dateninhalt schließen zu können. Daher müssen die Pseudonyme für die Zeit ihrer Speicherung eindeutig sein, d.h. es darf zu keiner gleichzeitigen Mehrfachverwendung von Pseudonymen kommen.

Weiterhin muss es eine Möglichkeit beim Pseudonymisieren von Logeinträgen geben, zu überprüfen, ob für ein Datum bereits ein Pseudonym vergeben wurde. So kann sichergestellt werden,

dass in einem bestimmten Zeitraum Logeinträge zu einer Person mit dem gleichen Pseudonym versehen werden, um über die Verknüpfung von Einträgen die gewünschte Erkennung von Insider-Angriffen erreichen zu können.

Auf 3.4 beziehen

Außerdem muss es eine Möglichkeit geben, die Parameter der Pseudonymisierung wie den Zeitraum ihrer Verwendung (vergleiche Abschnitt 3.2) konfigurierbar zu machen.

Mit Blick auf 3.2 nochmal überprüfen und anpassen

4.1.4 Anforderungen im Bezug auf den Einsatz eines kryptographischen Schwellwertschemas

Anpassen nach Kapitel 3 - Threshold-Abschnitt?

Der Einsatz eines kryptographischen Schwellwertschemas setzt eine verteilte Anwendung voraus, die den Zugriff für die Pseudonymisierungskomponente sowie für die bei der Entschlüsselung eines Eintrags beteiligten Akteure bereitstellt. Die für das Schwellwertschema nötigen Parameter t und n sowie die beteiligten Akteure müssen anpassbar bzw auswählbar sein.

In der Phase der Schlüsselgenerierung muss das System die Kommunikation und Koordination aller Beteiligten unterstützen. Die hier erstellten Schlüssel und *Shares* müssen an geeigneten Stelle sicher gespeichert und abrufbar sein.

Der für die Verschlüsselung erforderliche öffentliche Schlüssel muss so vorliegen, dass er bei der Verschlüsselung eines Pseudonym-Datensatzes genutzt werden kann.

Bei der Entschlüsselung eines Eintrags, also der Aufdeckung eines Pseudonyms, muss das System wiederum die beteiligten Aktuere koordinieren und anschließend die Rolle des *Combiners* übernehmen, so dass anschließend der entschlüsselte Datensatz zentral vorliegt.

4.1.5 Benutzerinteraktion

Die zu entwickelnde verteilte Anwendung wird an verschiedenen Stellen Benutzerinteraktion erfordern.

Das Konfigurieren des Systems zur Integration verschiedener Datenquellen und deren Beschreibung muss einem berechtigten Nutzer zugänglich gemacht werden. Ebenso sollte es im Hinblick auf die in der Aufgabenstellung geforderte Erweiterbarkeit im Bezug auf weitere Datenschutztechniken relativ leicht sein, diese Techniken im System nutzen zu können.

Für pseudonymisierte Datensätze muss es berechtigten Benutzern ermöglicht werden, Anfragen zur Aufdeckung eines Pseudonyms zu stellen und sich über ihren Status informiert zu halten.

Für die Benutzung eines kryptographischen Schwellwertschemas sollte es einem Administrator des Systems ermöglicht werden, grundlegende Parameter des Systems wie die Schwellwertparameter und die beteiligten Nutzer auszuwählen sowie die Initialisierung des Schemas anzustoßen. Die am Schwellwertschema beteiligten Nutzer müssen die Möglichkeit erhalten, eine Übersicht über sie betreffende Anfragen zur Aufdeckung eines Pseudonym-Datensatzes zu bekommen sowie einzelne Anfragen abzulehnen oder sich an m Prozess des Aufdeckens mithilfe des Schwellwertschemas zu beteiligen.

Performance?

4.1.6 Erweiterbarkeit um neue Datenquellen

Das umzusetzende System sollte es ermöglichen, Daten aus verschiedenen Quellen und (abhängig vom gewählten Eingriffspunkt in OSSIM auch) in verschiedenen Formaten entgegenzunehmen und mithilfe der umgesetzten Datenschutztechniken verändern zu können. Dabei muss das Format der Logdaten grundsätzlich beibehalten werden, um die Behandlung der Daten in OSSIM weiterhin zu ermöglichen.

4.1.7 Erweiterbarkeit um neue Datenschutztechniken

Neben der im Fokus dieser Arbeit stehenden Pseudonymisierung und dem Einsatz von kryptographischen Schwellwertschemata zum Schutz der Logdaten gibt es weitere Datenschutztechniken, die für den Anwendungsfall genutzt werden könnten (siehe Kapitel 5). Der umgesetzte Prototyp sollte leicht um diese Techniken erweiterbar sein, d.h. so gestaltet sein, dass andere Techniken ohne große Änderungen am System integriert und auf eingehende Logdaten angewendet werden können.

4.2 Entwurf

Einführender Absatz

4.2.1 Eingriff in den OSSIM-Datenfluss

Hier oder in Anforderungen?

Für den Eingriff zur Pseudonymisierung der Logdaten bieten sich verschiedene Stellen im Datenfluss von OSSIM an. In diesem Abschnitt sollen die verschiedenen Möglichkeiten dargestellt und gegeneinander abgewogen werden. Eine Übersicht über die verschiedenen Stellen bietet Abbildung 4.1. Im Folgenden sollen die verschiedenen Möglichkeiten bezogen auf die in Abschnitt 4.1.2 dargestellten Eigenschaften bewertet werden.

1. **In der Quelle der Logdaten**
   Bei diesem Ansatz werden die Daten bereits pseudonymisiert, bevor sie die Datenquelle verlassen. Auch wenn dieser Ansatz aus datenschutztechnischer Sicht die beste Möglichkeit darstellen würde, so ist er doch nicht umsetzbar, da hierzu jede mögliche Quelle von Logdaten universell verändert werden müsste.
2. **Syslog-Proxy**
   Dieser Ansatz pseudonymisiert die Daten vor dem ersten Kontakt mit einer OSSIM-Komponente, indem Datenquellen ihre Logdaten an einen Proxy senden, der die Daten pseudonymisiert und erst anschließend an OSSIM weiterreicht. Hierdurch wird erreicht, dass die Daten zu keiner Zeit nicht-pseudonymisiert in OSSIM vorliegen. Ein Nachteil dieser Lösung ist, dass sie das Parsen und Neuzusammensetzen der Logdaten im Proxy erfordert.

Beschreibung erweitern - analog zu Plugins in OSSIM

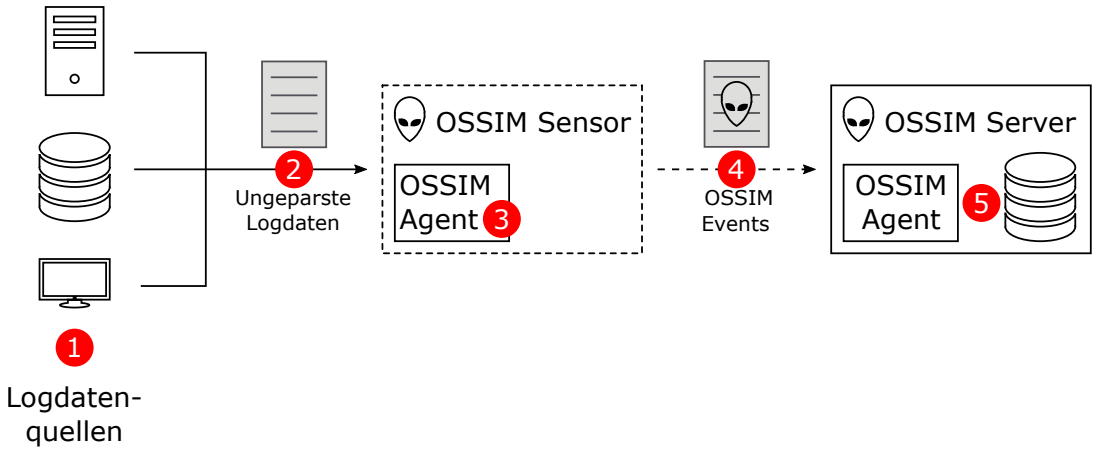

Abbildung 4.1: Mögliche Eingriffspunkte in den OSSIM-Datenfluss.

3. **Patchen des OSSIM-Sensor-Agents**
   Bei dieser Lösung müsste der OSSIM-Agent des Sensors so verändert werden, dass vor dem Senden der Events an den Server die Pseudonymisierung stattfindet. Daten erreichen den OSSIM-Server nur pseudonymisiert und mehrfaches Parsen wie in der zweiten Lösung wird verhindert. Auf der anderen Seite erfordert diese Lösung einen Eingriff in die Funktionsweise von OSSIM, was beispielsweise bei Updates von OSSIM zu Problemen führen kann. Außerdem liegen die Daten zu Beginn in nicht-pseudonymisierter Form im Sensor vor. Zusätzlich erfordert diese Lösung die verteilte Installation von OSSIM-Sensor und -Server, schließt also die All-In-One-Installation aus.

   Syslog-Problematik erwähnen

4. **Sensor-Server-Proxy**
   Hier wird ein Proxy zwischen Sensor und Server geschaltet, der bereits geparste Events pseudonymisiert und anschließend an den Server sendet. Dieser Ansatz würde mehrfaches Parsen verhindern und dafür sorgen, dass nur pseudonymisierte Logdaten den OSSIM-Server erreichen. Wie die vorhergehende Lösung würde er jedoch nur in der verteilten Installation funktionieren und zusätzlich in die Kommunikation zwischen Sensor und Server aktiv eingreifen, was im Hinblick auf die Nachrichtenintegrität[1] und auch auf geändertes Verhalten nach Updates von OSSIM einen Nachteil darstellt.

5. **Patchen des OSSIM-Servers**
   Die letzte Möglichkeit ist das Verändern des OSSIM-Servers selbst. Diese Lösung ist vergleichbar mit der dritten Möglichkeit. Zusätzlich würde sie bei der verteilten sowie bei der All-In-One-Installation funktionieren, auf der anderen Seite aber zulassen, dass nicht-pseudonymisierte Events sogar noch direkt auf dem Server vorliegen.

Insbesondere der aus datenschutztechnischer Sicht relevante Vorteil, dass die Daten bereits pseudonymisiert in allen OSSIM-Komponenten eintreffen, ließ die Entscheidung auf die **zweite Möglichkeit** fallen. Dass die Lösung außerdem noch für beide Varianten der OSSIM-Installation möglich ist und keine Anpassungen an OSSIM selbst benötigt, wiegt den Nachteil des zusätzlichen Parsens und wieder Zusammensetzens der Lognachricht bei Weitem auf.

Digitalgipfel: Die Pseudonymisierung ist im Verarbeitungsprozess so früh wie möglich durchzuführen.

Auch auf Angreifermodell beziehen

1. In der aktuellen Version von OSSIM werden Nachrichten unverschlüsselt und nicht signiert zwischen Sensor und Server versendet, aber zu hoffen ist, dass dieser Zustand sich in zukünftigen Versionen noch ändert.

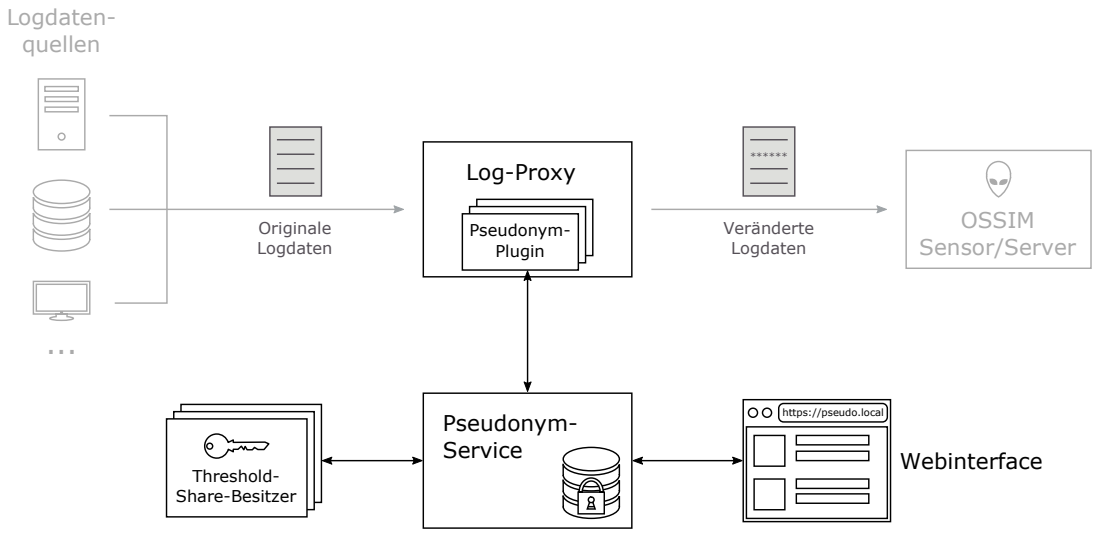

Abbildung 4.2: Ein Überblick über die entworfene Architektur.

4.2.2 Architektur

Ausgehend von diesen Überlegungen wurde ein verteiltes System entworfen, dass die Anforderungen aus Abschnitt 4.1 erfüllt und an der beschriebenen Stelle in den Datenfluss eingreift. Einen Überblick bietet Abbildung 4.2. Das System besteht aus verschiedenen Komponenten, die im Folgenden näher beschrieben werden.

Ein **Log-Proxy**, der die Daten über das Syslog-Protokoll entgegennimmt, verändert und anschließend an OSSIM weiterleitet. Das Verändern der Daten kann mit verschiedenen Plugins geschehen, so dass neben der umzusetzenden Pseudonymisierung auch weitere Datenschutztechniken eingesetzt werden können, was die geforderte Erweiterbarkeit aus Abschnitt 4.1.7 ermöglicht. Der Proxy leistet die Behandlung von Logdaten aus verschiedenen Quellen (siehe Abschnitt 4.1.6), was wie bereits im vorhergehenden Abschnitt beschrieben durch Parsen und Wiederzusammensetzen der Daten geschehen muss. Die Konfiguration des quellenabhängigen Vorgehens bei der Logdatenverarbeitung erfolgt ebenfalls hier.

Ein in dem Proxy enhaltenes Plugin wird für die Pseudonymisierung von Daten zuständig und kommuniziert dazu mit einer externen Komponente – dem Pseudonym-Service. Die Kommunikation mit dem Proxy erfolgt über einen Webservice-basierten Ansatz. Das Plugin kann für eingehende Daten ein Pseudonym anfordern und dieses anschließend in den Logdaten verwenden.

Der **Pseudonym-Service** erfüllt zwei Aufgaben: das Speichern und Verwalten der Pseudonyme sowie die Integrierung des kryptographischen Schwellwertschemas. Initial muss die Schlüsselgenerierung des Schwellwertschemas (bei zentraler Schlüsselgenerierung) oder die Koordinierung der teilnehmenden Benutzer (bei dezentraler Schlüsselgenerierung) durch den Service geleistet werden. Es können während des Betriebs neue Pseudonyme angelegt und zusammen mit ihrem durch das Schwellwertschema verschlüsselten Datum abgelegt werden. Sie werden durch geeignete Maßnahmen durchsuchbar gehalten, um für ein Datum überprüfen zu können, ob bereits ein Pseudonym vergeben wurde. Über ein Webinterface kann ein berechtiger Benutzer die Aufdeckung eines bestimmten Pseudonyms fordern und den Status seiner Forderung bzw. im Erfolgsfall das aufgedeckte Datum betrachten. Dieses Datum wird durch das Kombinieren der

partiellen Entschlüsselungen erhalten, die von den entsprechenden *Share*-Besitzern berechnet werden.

Benutzer, die zuständig für die Bewertung von Anfragen zur Aufdeckung eines Pseudonyms sind, erhalten die Möglichkeit zur Interaktion mit dem System über eine **Client-Anwendung**, für die der Pseudonym-Service ebenfalls als Webservice agiert. Diese Anwendungen leisten im Falle der dezentralen Schlüsselgenerierung die initiale Generierung, verwalten den *Share* des Benutzers und können nach der Bestätigung des Benutzers zu der Aufdeckung eines Pseudonyms partiell beitragen.

4.3 Einbindung in OSSIM

Proxy-Aufbau und Vorgehensweise erläutern, insgesamt Abschnitte besser verknüpfen

Für die in dieser Arbeit zu leistende prototypische Umsetzung des Systems wurde sich auf den am häufigsten[2] genutzten Weg des Datenerhalts in OSSIM (siehe Abschnitt 3.1.2) beschränkt: das Entgegennehmen der Daten über das Syslog-Protokoll.

Die Behandlung von verschiedenen Datenquellen wird durch Konfigurationsdateien ermöglicht:

```
1  [general]
2  active=True
3
4  [group1]
5  pattern=^(?P<time>\w+ *\d{1,2} \d{2}:\d{2}:\d{2}) (?P<device>[^:]+):
       Testing my device USER=(?P<user>.+)$
6  time=Substitute(substitute = 'somevalue_time')
7  device=Substitute(substitute = 'somevalue_device')
8  user=Pseudonymize()
9
10 [group2]
11 pattern=^(?P<test>.*)$
12 test=Pseudonymize()
```

Eine Konfigurationsdatei kann aus mehreren Bereichen bestehen. Der `general`-Bereich enthält allgemeine Angaben über das Plugin. Um unterschiedliche Lognachrichten eines Gerätes bündeln zu können, kann eine Konfigurationsdatei weiterhin mehrere Bereiche enthalten, die jeweils die Verarbeitung einer bestimmten Lognachricht beschreiben. Angegeben werden muss jeweils ein regulärer Audruck, der die Nachricht beschreibt und mehrere Gruppen (`(?P<name>...)`) enthalten kann. Für jede dieser Gruppen muss eine Angabe zu dem Plugin inklusive notwendiger Parameter gemacht werden, dass die Gruppe verarbeiten soll. Durch diese Konfigurationsdateien können Nachrichten unbekannter Formate aus neuen Datenquellen leicht in das bestehende System eingebunden werden.

Die Erweiterbarkeit um neue Datenschutztechniken wird durch leicht erweiterbare Plugins ermöglicht. Ein Plugin muss lediglich die Methode `handle_data` implementieren, die die

2. Die in OSSIM bereits mitgelieferten Plugins bestätigen dies. Unter den Hunderten Plugins ist nur ein einziges, das einen anderen Mechanismus als das Syslog-Protokoll nutzt. Auch wenn beispielsweise Plugins, die eine Datenbankabfrage enthalten, immer dem Anwendungsfall angepasst und daher nicht in OSSIM inkludiert werden können, so unterstützt dies doch die Annahme des Syslog-Protokolls als am häufigsten genutzten Weg und damit als geeignet für den Fokus dieser Arbeit.

originalen Daten und alle in der Konfiguration angegebenen Parameter erhält. Ein einfaches Plugin, das die Daten durch ein in der Konfiguration angegebenen Wert ersetzt, könnte so aussehen:

```
1 class Substitute(AbstractPlugin):
2
3     def handle_data(self, data: str, **kwargs) -> str:
4         if 'substitute' in kwargs:
5             return kwargs['substitute']
6         else:
7             raise MissingSubstituteError
```

4.4 Umsetzung der Pseudonymisierung

Um Implementation erweitern (welche Parameter in welchen Systemteilen, Setup, ...)

Für die Pseudonymisierung wurde ein Plugin für den Proxy entwickelt. Bei jedem eintreffenden Logdatum kann abhängig von dem Datenformat für ein entsprechendes Datenfeld ein Pseudonym als Ersatz für das echte Datum gesetzt werden. Dazu stellt der Pseudonym-Service eine Schnittstelle bereit, über die für ein Datum ein Pseudonym erhalten werden kann. Durch diese Trennung wird eine höhere Sicherheit der Zuordnung zwischen Datum und Pseudonym erreicht: In dem Proxy kommen die Logdaten in unveränderter Form an und werden verändert weitergesendet, daher ist die Zuordnung hier implizit bekannt und muss in Kauf genommen werden. Die Speicherung dieser Zuordnung erfolgt jedoch nur in dem Pseudonym-Service. Durch die Verschlüsselung und die in den folgenden Abschnitten beschriebene MAC-abhängige Pseudonymgenerierung erfährt der Pseudonym-Service nichts über das Datum, das durch das Pseudonym beschrieben wird. So führt unberechtigter Zugriff auf die Datenbank des Pseudonym-Service nicht zu mehr Informationen über das Datum, das ein Pseudonym beschreibt.

4.4.1 Proxy

Auf der Proxy-Seite wird das verschlüsselte Datum (siehe dazu Abschnitt 4.5) zusammen mit einem generierten MAC, der wie in Abschnitt 3.4 beschrieben für die Überprüfung auf bereits bestehende Pseudonyme genutzt wird, an den Pseudonym-Service gesendet. Der Schlüssel, der für die Generierung des MACs verwendet wird, wird immer nach einer bestimmten Zeitspanne neu generiert. Diese Zeitspanne kann mittels eines Parameters an das Anwendungsszenario angepasst werden (vgl. 3.2). Durch diesen Schlüsselwechsel wird erreicht, das für gleiche Daten, für die der MAC mit einem neuen Schlüssel erstellt wird, auch neue Pseudonyme erhalten werden.
Da der Schlüsselwechsel nicht Pseudonym-abhängig geschieht, ist die Zeitspanne global für alle Pseudonyme gültig und somit als maximale Zeitspanne zu verstehen. Dies kann für die Anomalieerkennung evtl. Probleme bereiten, wenn nicht genügend lange Überwachungsdaten verkettet werden können. Auf der anderen Seite würde eine Verweildauer für einzelne Pseudonyme ein Erfassen des Erstellungszeitpunkts in der Datenbank erfordern, was wie in Abschnitt 3.2 beschrieben Rückschlüsse auf das ursprüngliche Datum des Pseudonyms liefern könnte. Daher wurde sich gegen diesen Ansatz entschieden.

4.4.2 Service

Auf der Service-Seite wird nun anhand des empfangenen MACs durch Vergleich mit in der Datenbank vorliegenden MACs überprüft, ob bereits ein Pseudonym für das Datum vergeben wurde, das noch nicht zu häufig verwendet wurde. Ist dies noch nicht der Fall, so wird ein noch nicht verwendetes, zufälliges Pseudonym erstellt und zusammen mit dem MAC in der Datenbank gespeichert. Anderenfalls wird das bereits vergebene Pseudonym zurückgeliefert. Die maximale Anzahl an Nutzungen kann analog zur maximalen Zeitspanne ebenfalls mittels eines Parameter gesetzt werden.

4.4.3 Perfect Forward Privacy

Im Zusammenspiel dieser Parameter kann jedoch noch ein Problem entstehen. Für neu vergebene Pseudonyme, die innerhalb eines Zeitabschnitts durch Überschreiten der maximalen Nutzungsanzahl entstanden sind, liegen in der Datenbank Einträge mit gleichem MAC vor. So wird die Verknüpfung verschiedener Pseudonyme ermöglicht, wenn jemand (berechtigt oder unberechtigt) Zugriff auf die Daten erhält. Das Aufdecken eines Pseudonyms deckt auch alle anderen in diesem Zeitintervall erstellten Pseudonyme implizit auf, was dem in Abschnitt 3.2 beschriebenen Prinzip der *Perfect Forward Privacy* widerspricht. Dieses Problem kann durch eine zusätzliche MAC-Berechnung auf der Service-Seite verhindert werden, die zusätzlich noch einen Pseudonym-abhängigen Zufallswert einbeziehen muss. Der hierzu verwendete Schlüssel muss dann ebenfalls abhängig von dem bereits beschriebenen Parameter neu generiert werden. Hierdurch enthalten Datenbankeinträge, die innerhalb eines Zeitintervalls zu dem gleichen Datum gehören, durch den einfließenden Zufallswert trotzdem unterschiedliche MACs und die Verkettbarkeit verschiedener Pseudonyme ist verhindert. Jedoch erfordert dieser Ansatz eine MAC-Berechnung pro Datenbankeintrag für jede Anfrage und ist daher aus Performance-Sicht kritisch zu betrachten. Daher wird diese Möglichkeit nicht implementiert. Das bestehende Problem ist jedoch für ein konkretes Anwendungsszenario und bei der Wahl der Parameter – insbesondere des Zeitintervalls – zu beachten.

Inhaltliche Aussage überprüfen... würde zusätzlicher MAC auf Serverseite ohne SALT etwas bringen?

4.5 Implementierung und Integration des Schwellwertschemas

Einführender Text

4.5.1 ThresholdCrypto „Lib“

Um die Funktionen des Schwellwertschemas, die in unterschiedlichen Systemteilen benötigt werden, einfach zur Verfügung zu stellen, wurde eine Bibliothek entwickelt, die in den verschiedenen Komponenten genutzt werden kann. Das öffentliche Interface der Bibliothek besteht aus folgenden Funktionen:

- **Parametergenerierung:** Diese Funktion dient dem Erhalt der benötigten sicheren Primzahl bzw. des Generator (vgl.). Hierbei kann zwischen Neugenerierung und Verwendung vorberechneter Parameter verschiedener Schlüsselstärken entschieden werden (siehe auch Abschnitt 4.5.1).

ergänzen

- **Schlüssel- und Sharegenerierung:** Durch diese Funktion wird ein zufälliger geheimer Schlüssel erzeugt und aus ihm der öffentliche Schlüssel sowie die einzelnen *Shares* erzeugt.
- **Verschlüsselung einer Nachricht:** Diese Funktion verschlüsselt mithilfe des öffentlichen Schlüssels eine Nachricht (siehe auch Abschnitt 4.5.1). Sie wird im Proxy verwendet.
- **Berechnung einer partiellen Verschlüsselung:** Mithilfe eines *Shares* wird die zugehörige partielle Entschlüsselung einer verschlüsselten Nachricht berechnet. Diese Funktion wird in den Threshold-Clients genutzt.
- **Kombinieren von partiellen Verschlüsselungen:** Aus ausreichend partiellen Verschlüsselungen kann die Nachricht wiederhergestellt bzw. entschlüsselt werden (siehe wiederum Abschnitt 4.5.1).

Neben den Funktionen werden noch einige Klassen zur Verfügung gestellt, die das Arbeiten mit den Ergebnissen der Funktionen erleichtern sowie Funktionalität wie Serialisierbarkeit ermöglichen:

- **ThresholdParameters:** Enthalten die Werte t und n des Schwellwertschemas.
- **KeyParameters:** Enthalten die benötigten Primzahlen bzw. den Generator der zugrundeliegenden Gruppe.
- **PublicKey:** Enthält den öffentlichen Schlüssel der zum Verschlüsseln einer Nachricht verwendet werden kann.
- **KeyShare:** Enthält die Werte des Shares eines Teilnehmers am Schwellwertschema.
- **EncryptedMessage:** Enthält die Daten einer verschlüsselten Nachricht (vgl. auch Abschnitt 4.5.1).
- **PartialDecryption:** Enthält die zu einer partiellen Entschlüsselung gehörenden Werte, die zum Entschlüsseln der vollständigen Nachricht genutzt werden.

Im Folgenden soll kurz auf zwei Kernelemente/Schlüsselstellen/... des implementierten Verfahrens eingegangen werden.

Parametergenerierung

Die für das Verfahren benötigten sicheren Primzahlen p und q lassen sich mithilfe eines einfachen Ansatzes finden [MVOV96]: Es wird solange eine Primzahl q im gewünschten Bereich zufällig gewählt, bis $2q+1$ ebenfalls eine Primzahl ist. Zur Überprüfung der Primalität der entsprechenden Zahlen wird der Miller-Rabin-Test genutzt.

Zusätzlich benötigt das Verfahren einen Generator g einer Untergruppe der Ordnung q von $\mathbb{Z}_p^*$. Hierzu wird lediglich solange ein zufälliges Element g aus $\mathbb{Z}_p^*$ gewählt bis

$$g^q \equiv 1 \mod p \text{ und } g^2 \not\equiv 1 \mod p$$

gelten. Da Untergruppen von $\mathbb{Z}_p^*$ nach dem Satz von Lagrange lediglich die Ordnungen $1, 2, q$ oder $2q$ besitzen, werden durch obenstehende Bedingungen lediglich Untergruppen der Ordnung q zugelassen.

Nach [KL14] beeinträchtigt es nicht die Sicherheit des ElGamal-Verfahrens, wenn vorberechnete Parameter von verschiedenen Benutzern geteilt werden. Auch die Kombination dieses Verfahrens mit Shamirs Secret Sharing dürfte hieran nichts ändern, da dieses Verfahren lediglich für die Verteilung des geheimen Schlüssels sorgt, aber an den grundlegenden Eigenschaften der Ver- und Entschlüsselung nichts ändert. Daher wurden verschiedene Parameter bereits vorberechnet und als statische Parameter zur Verfügung gestellt. Weiterhin ist es jedoch auch möglich eigene Parameter in gewünschter Stärke zu generieren. In den meisten Empfehlungen werden heutzutage Schlüssellängen um 2000 Bit empfohlen[3].

evtl. auf Evaluation beziehen.

Hybride Kryptographie

Die Verschlüsselung bzw. Entschlüsselung wurden in Form eines hybriden Verschlüsselungsverfahrens umgesetzt, wie es auch in [KL14] empfohlen und beschrieben wird: Bei der Verschlüsselung wird ein zufälliger Schlüssel k^{symm} für ein symmetrisches Verfahren E^{symm} erzeugt und dazu genutzt den Klartext m zu verschlüsseln. Das kryptographische Schwellwertschema wird lediglich dazu verwendet, diesen erzeugten Schlüssel zu verschlüsseln. Ein Schlüsseltext besteht daher aus drei Teilen:

- $v = g^k$: Der erste Teil der ElGamal-Verschlüsselung des symmetrischen Schlüssels (siehe Abschnitte 2.5.3 und **??**)
- $c^{tc} = k^{symm} \cdot g^{ak}$: Der zweite Teil der ElGamal-Verschlüsselung des symmetrischen Schlüssels (siehe ebenda)
- $c^{symm} = E^{symm}_{k^{symm}}(m)$: Der symmetrisch verschlüsselte Klartext

Bei der Entschlüsselung wird ähnlich vorgegangen: Das kryptographische Schwellwertschema wird dazu genutzt, den symmetrischen Schlüssel wiederherzustellen. Anschließend kann der ursprüngliche Klartext mit diesem Schlüssel wieder entschlüsselt werden. Dieses hybride Vorgehen bietet einige Vorteile:

Symmetrische Verfahren sind im Allgemeinen deutlich schneller als asymmetrische. Durch die Kombination beider Verfahren wird also erreicht, dass die Eigenschaften des kryptographischen Schwellwertschemas im Bezug auf die verteilte Ver- bzw. Entschlüsselung erhalten werden, jedoch mit dem Geschwindigkeitsvorteil der symmetrischen Verschlüsselung.

Das Vorgehen ermöglicht es weiterhin beliebige Nachrichten relativ einfach zu verschlüsseln, da die Beschränkung des asymmetrischen Verfahrens bezogen auf die Nachrichtenlänge (die in kodierter Form kleiner sein muss als der Parameter p) nur noch für den Schlüssel des symmetrischen Verfahrens erfüllt werden muss. Dies stellt kein Problem dar, da symmetrische Verfahren für vergleichbare Sicherheit geringere Schlüssellängen benötigen als asymmetrische Verfahren, die auf dem Diskreten-Logarithmus-Problem beruhen.

Weiterhin kann für die symmetrische Verschlüsselung ein Verfahren genutzt werden, dass neben der Verschlüsselung auch die Validität der Daten überprüft - ein sogenanntes *Authenticated Encryption Scheme*. Hierdurch wird erreicht, dass Änderungen am Schlüsseltext bei der

3. Für einen Vergleich verschiedener Empfehlungen siehe: https://www.keylength.com

Entschlüsselung erkannt werden und der Vorgang abgebrochen werden kann, also neben der Vertraulichkeit auch die Integrität der Nachricht geschützt ist.

In der Implementierung wird die Standardfunktion zur symmetrischen Verschlüsselung aus der Kryptographie-Bibliothek NaCl[4] genutzt. Hierbei handelt es sich um ein *Authenticated Encryption Scheme* auf Basis der Algorithmen Salsa20 und Poly1305. Neben den bereits erwähnten Vorteilen des allgemeinen Ansatzes handelt es sich hierbei zusätzlich um eine weit verbreitete und von erfahrenen Kryptographen entwickelte und überprüfte Bibliothek. Dies erhöht das Vertrauen in eine sichere Umsetzung der Verfahren.

Eine direkte Implementierung des beschriebenen Schemas enthält jedoch noch eine Schwäche: In [BJN00] wird von den Autoren ein Angriff vorgestellt, der bei der direkten Implementierung von auf dem ElGamal-Verfahren basierenden Schemata möglich ist. Der Angriff besteht darin, dass bei der Verschlüsselung symmetrischer Schlüssel durch deren geringere Länge und die Berechnungen in speziellen Untergruppen im ElGamal-Verfahren die Möglichkeit besteht, die symmetrischen Schlüssel mit geringerem Aufwand zu entschlüsseln.

Als Gegenmaßnahme wird die Vorverarbeitung der symmetrischen Schlüssel empfohlen. In [ABR99] wird ein hybrides Schema dargestellt, dass diese Gegenmaßnahme umsetzt. Der entscheidende Schritt im Vorgehen besteht darin, den symmetrischen Schlüssel nicht direkt zufällig zu erzeugen. Stattdessen wird ein zufälliges Untergruppenelement $r \in \mathbb{Z}_q^*$ (das den Nachrichtenraum darstellt) gewählt und der symmetrische Schlüssel mithilfe einer kryptographisch sicheren Hashfunktion H hieraus abgeleitet. Hierdurch verändert sich die Zusammensetzung eines Schlüsseltextes leicht:

- $v = g^k$: wie vorhergehend beschrieben
- $c^{tc} = r \cdot g^{ak}$: Anstelle des symmetrischen Schlüssels wird das Untergruppenelement r durch das ElGamal-Verfahren verschlüsselt.
- $c^{symm} = E^{symm}_{H(r)}(m)$: Als symmetrischer Schlüssel wird nun der durch H berechnete Hashwert von r genutzt

Zusätzlich schreibt das Schema die Benutzung eines MACs zur Sicherung der Integrität der Nachricht vor. Auf diesen Schritt kann in dem in dieser Arbeit implementierten Schema verzichtet werden, da die Integrität bereits durch das verwendete AE-Verfahren geschützt ist. Einen Überblick über das letztlich umgesetzte Schema, das auch diesen Angriff verhindert, bietet Abbildung 4.3.

4.5.2 Service und Setup-Verfahren

Setup und Generierung

Anschließend können berechtigte Benutzer die Aufdeckung eines Pseudonyms über das Webinterface anfragen. Anschließend werden von den teilnehmenden Clients die partiellen Entschlüsselungen entgegengenommen. Liegen ausreichend partielle Verschlüsselungen vor, so werden diese mithilfe der Bibliothek kombiniert und damit der Halter des Pseudonyms entschlüsselt und aufgedeckt.

4. NaCl: Networking and Cryptography library: https://nacl.cr.yp.to

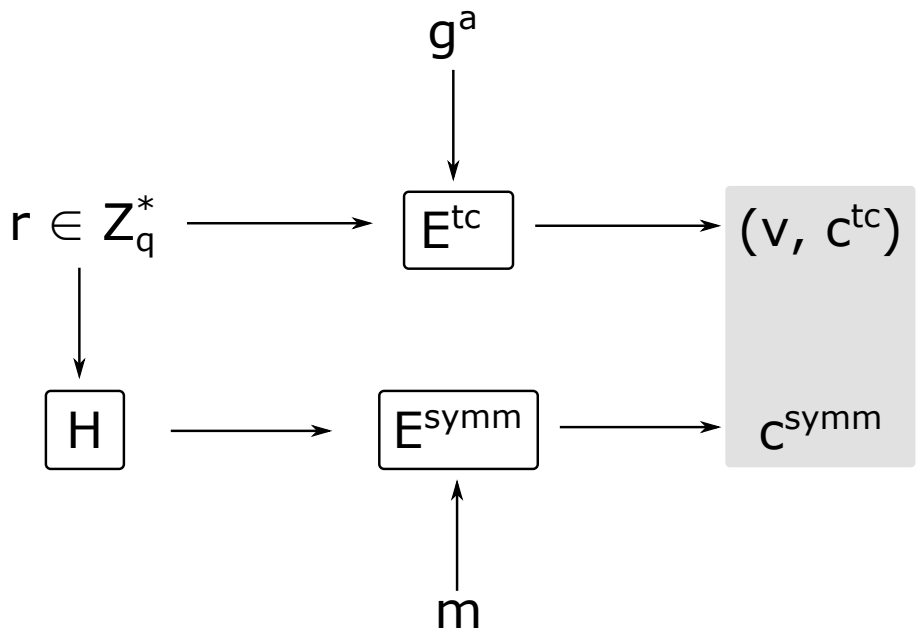

Abbildung 4.3: Übersicht über das umgesetzte hybride Verschlüsselungsschema.

4.5.3 Client-Anwendung

Für Teilnehmer, die an dem Schwellwertschema beteiligt sind, wurde eine konsolenbasierte Anwendung entwickelt, die die folgende Funktionalität bereitstellt:

Clients melden sich zu Beginn an dem Pseudonym-Service an und können in der anschließenden Setup-Phase mit in das Verfahren integriert werden. Jeder teilnehmende Client empfängt nach der Generierung das für ihn bestimmte Share vom Server und speichert dieses verschlüsselt lokal ab. Hierzu enthält die Anwendung einen lokalen Webserver. Dies ermöglicht es dem Pseudonym-Service die *Shares* nicht speichern und nach der Generierung sofort entfernen zu können.

Anschließend können sich Teilnehmer regelmäßig nach neuen Anfragen zur Aufdeckung eines Pseudonyms erkundigen. Wurde eine neue Anfrage empfangen, so kann der Teilnehmen diese annehmen oder ablehnen. Die Annahme führt zur Berechnung einer partiellen Entschlüsselung mithilfe der Bibliotheksfunktion, die anschließend an den Pseudonym-Service gesendet wird.

4.5.4 Proxy

Das im Syslog-Proxy implementierte Plugin zur Pseudonymisierung empfängt während der Initialisierung den während der Schlüsselerzeugung generierten öffentlichen Schlüssel vom Pseudonym-Service. Bei einer eintreffenden Syslog-Nachricht werden die zu pseudonymisierenden Daten mit diesem Schlüssel unter Nutzung der Bibliotheksfunktion verschlüsselt und zusammen mit den bereits beschriebenen weiteren Daten an den Service gesendet.

4.6 Evaluation

Inklusive Problemen, ...

4.6.1 Angriffsmöglichkeiten

- Zentrale Schlüsselgenerierung
- Krypto nicht von Kryptographen überprüft (Sidechannel, ...)

4.6.2 Performance

- Zeitmessungen für einzelne Systemfunktionen

5 Alternativen

- **Alternativen** Welche alternativen oder ergänzenden Vorgehensweisen zu Pseudonymisierung + kryptographisches Schwellwertschema gibt es und welche Eigenschaften, Vor- und Nachteile besitzen sie?
- **Umsetzung** Wie könnten diese Alternativen im Prototypen umgesetzt werden?

5.1 Alternative1

5.2 ...

5.3 Vorgehen zur Integration

6 Fazit

TBW

Literatur

[ABR99] Michel Abdalla, Mihir Bellare und Phillip Rogaway. *DHAES: An Encryption Scheme Based on the Diffie-Hellman Problem*. In: *IACR Cryptology ePrint Archive* 1999 (1999), S. 7.

[Ara+14] Myrto Arapinis u. a. *Privacy through Pseudonymity in Mobile Telephony Systems*. In: *Network and Distributed System Security Symposium*. 2014.

[BBH06] Dan Boneh, Xavier Boyen und Shai Halevi. *Chosen ciphertext secure public key threshold encryption without random oracles*. In: *Topics in Cryptology – CT-RSA 2006*. Hrsg. von David Pointcheval. Springer, 2006, S. 226–243.

[BBO07] Mihir Bellare, Alexandra Boldyreva und Adam O'Neill. *Deterministic and efficiently searchable encryption*. In: *Advances in Cryptology-CRYPTO 2007* (2007), S. 535–552.

[bit16] bitkom. *Spezialstudie Wirtschaftsschutz*. bitkom. 2016. URL: https://www.bitkom.org/Bitkom/Publikationen/Spezialstudie-Wirtschaftsschutz.html (besucht am 24. 06. 2017).

[BJN00] Dan Boneh, Antoine Joux und Phong Q Nguyen. *Why textbook ElGamal and RSA encryption are insecure*. In: *ASIACRYPT*. Bd. 1976. Springer. 2000, S. 30–43.

[Bla79] George Robert Blakley. *Safeguarding cryptographic keys*. In: *Proceedings of the AFIPS 1979 National Computer Conference*. Hrsg. von Richar E. Merwin. AFIPS Press, 1979, S. 313–317.

[Bon+04] Dan Boneh u. a. *Public key encryption with keyword search*. In: *Eurocrypt*. Bd. 3027. Springer. 2004, S. 506–522.

[Bra+15] Nicholas Bradley u. a. *IBM 2015 Cyber Security Intelligence Index*. IBM. 2015. URL: http://www-01.ibm.com/common/ssi/cgi-bin/ssialias?htmlfid=SEW03073USEN (besucht am 24. 06. 2017).

[BS16] Dan Boneh und Victor Shoup. *A graduate course in applied cryptography*. In: (2016). URL: https://crypto.stanford.edu/~dabo/cryptobook/draft_0_3.pdf (besucht am 02. 09. 2017).

[CM05] Yan-Cheng Chang und Michael Mitzenmacher. *Privacy preserving keyword searches on remote encrypted data*. In: *ACNS*. Bd. 5. Springer. 2005, S. 442–455.

[Des87] Yvo Desmedt. *Society and Group Oriented Cryptography: a New Concept*. In: *Conference on the Theory and Application of Cryptographic Techniques 1987*. Hrsg. von Carl Pomerance. Springer, 1987, S. 120–127.

[Des93] Yvo Desmedt. *Threshold cryptosystems*. In: *Advances in Cryptology — AUSCRYPT '92*. Hrsg. von Jennifer Seberry und Yuliang Zheng. Springer, 1993, S. 1–14.

[Des97] Yvo Desmedt. *Some recent research aspects of threshold cryptography*. In: *International Workshop on Information Security*. Springer. 1997, S. 158–173.

[Det+15] Kai-Oliver Detken u. a. *SIEM approach for a higher level of IT security in enterprise networks*. In: *2015 IEEE 8th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*. IEEE. 2015, S. 322–327.

[DF90] Yvo Desmedt und Yair Frankel. *Threshold cryptosystems*. In: *Advances in Cryptology - CRYPTO' 89 Proceedings*. Hrsg. von Gilles Brassard. Springer, 1990, S. 307–315.

[DJ01] Ivan Damgard und Mads Jurik. *A generalisation, a simplification and some applications of Paillier's probabilistic public-key system*. In: *Public Key Cryptography – Proceedings of the 4th International Workshop on Practice and Theory in Public Key Cryptosystems*. Hrsg. von Kwangjo Kim. Bd. 1992. Springer, 2001, S. 119–136.

[DRS14] Kai-Oliver Detken, Thomas Rossow und Ralf Steuerwald. *SIEM-Ansätze zur Erhöhung der IT-Sicherheit auf Basis von IF-MAP*. In: *Proceedings of the D A CH Security 2014: Bestandsaufnahme, Konzepte, Anwendungen und Perspektiven*. Hrsg. von Peter Schartner und Peter Lipp. syssec, 2014.

[Dö05] Florian Dötzer. *Privacy issues in vehicular ad hoc networks*. In: *International Workshop on Privacy Enhancing Technologies*. Springer. 2005, S. 197–209.

[ElG85] Taher ElGamal. *A public key cryptosystem and a signature scheme based on discrete logarithms*. In: *IEEE transactions on information theory* 31.4 (1985), S. 469–472.

[FPS00] Pierre-Alain Fouque, Guillaume Poupard und Jacques Stern. *Sharing decryption in the context of voting or lotteries*. In: *Financial Cryptography*. Bd. 1962. Springer. 2000, S. 90–104.

[Fra+97] Yair Frankel u. a. *Proactive rsa*. In: *Annual International Cryptology Conference*. Springer. 1997, S. 440–454.

[Gem97] Peter Gemmell. *An introduction to threshold cryptography*. In: *CryptoBytes, a technical newsletter of RSA Laboratories* 2.7 (1997), S. 295–310.

[Gen+96a] Rosario Gennaro u. a. *Robust and efficient sharing of RSA functions*. In: *Advances in Cryptology—CRYPTO'96*. Springer. 1996, S. 157–172.

[Gen+96b] Rosario Gennaro u. a. *Robust threshold DSS signatures*. In: *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 1996, S. 354–371.

[Goh03] Eu-Jin Goh. *Secure indexes*. In: *IACR Cryptology ePrint Archive* 2003 (2003), S. 216.

[KL14] Jonathan Katz und Yehuda Lindell. *Introduction to modern cryptography*. 2. akt. Auflage. CRC press, 2014.

[MVOV96] Alfred J Menezes, Paul C Van Oorschot und Scott A Vanstone. *Handbook of applied cryptography*. CRC press, 1996.

[Ngu05] H.L. Nguyen. *RSA Threshold Cryptography*. 2005. URL: https://www.cs.ox.ac.uk/files/269/Thesis.pdf (besucht am 02. 07. 2017).

[NK11] Mark Nicolett und Kelly M Kavanagh. *Magic quadrant for security information and event management*. In: *Gartner Research Note G00212454* (2011).

[Pet+15] Jonathan Petit u. a. *Pseudonym schemes in vehicular networks: A survey*. In: *IEEE communications surveys & tutorials* 17.1 (2015), S. 228–255.

[PH10] Andreas Pfitzmann und Marit Hansen. *A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management*. 2010. URL: http://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.34.pdf (besucht am 27. 06. 2017).

[PK01] Andreas Pfitzmann und Marit Köhntopp. *Anonymity, unobservability, and pseudonymity — a proposal for terminology*. In: *Designing privacy enhancing technologies*. Hrsg. von Hannes Federrath. Springer, 2001, S. 1–9.

[PWP90] Birgit Pfitzmann, Michael Waidner und Andreas Pfitzmann. *Rechtssicherheit trotz Anonymität in offenen digitalen Systemen*. In: *Datenschutz und Datensicherung DuD* 14.5-6 (1990), S. 243–253.

[Rab98] Tal Rabin. *A simplified approach to threshold and proactive RSA*. In: *Advances in Cryptology—CRYPTO'98*. Springer. 1998, S. 89–104.

[Sha79] Adi Shamir. *How to share a secret*. In: *Communications of the ACM* 22.11 (1979), S. 612–613.

[Sho00] Victor Shoup. *Practical threshold signatures*. In: *Advances in Cryptology—EUROCRYPT 2000*. Springer. 2000, S. 207–220.

[SMK09] Florian Schaub, Zhendong Ma und Frank Kargl. *Privacy requirements in vehicular communication systems*. In: *International Conference on Computational Science and Engineering*. Bd. 3. IEEE. 2009, S. 139–145.

[SS01] Douglas R Stinson und Reto Strobl. *Provably secure distributed Schnorr signatures and a (t, n) threshold scheme for implicit certificates*. In: *ACISP*. Bd. 1. Springer. 2001, S. 417–434.

[SW17] Rolf Schwartmann und Steffen Weiß. *Whitepaper zur Pseudonymisierung der Fokusgruppe Datenschutz*. Juni 2017. URL: https://www.bmi.bund.de/SharedDocs/downloads/DE/veroeffentlichungen/2017/digital-gipfel-one-pager-fokusgruppe.pdf (besucht am 15. 11. 2017).

[SWP00] Dawn Xiaoding Song, David Wagner und Adrian Perrig. *Practical techniques for searches on encrypted data*. In: *Security and Privacy, 2000. S&P 2000. Proceedings. 2000 IEEE Symposium on*. IEEE. 2000, S. 44–55.

[WWC16] Yunling Wang, Jianfeng Wang und Xiaofeng Chen. *Secure searchable encryption: a survey*. In: *Journal of communications and information networks* 1.4 (2016), S. 52–65.