Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Masterarbeit

Datenschutzfreundliche Speicherung unternehmensinterner Überwachungsdaten mittels Pseudonymisierung und kryptographischer Schwellwertschemata

vorgelegt von

Tom Petersen

geb. am 13. Dezember 1990 in Hannover

Matrikelnummer 6359640

Studiengang Informatik

eingereicht am 22. März 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?

Zusammenfassung

In dieser Arbeit wird ein Ansatz zur Speicherung von Überwachungsdaten erarbeitet, der dazu genutzt werden kann, die anomaliebasierte Erkennung von Insiderangriffen datenschutzgerecht zu gestalten. Dabei kommt eine Kombination von Pseudonymisierung und kryptographischem Schwellwertschema zum Einsatz. Dies ermöglicht die Speicherung und Verarbeitung pseudonymisierter Daten, wobei der Pseudonymhalter erst durch Kooperation einer bestimmten Anzahl von Benutzern wieder aufgedeckt werden kann.

Es werden Eigenschaften der Pseudonymisierung insbesondere in Bezug auf notwendige regelmäßige Pseudonymwechsel betrachtet und Grundlagen sowie Erweiterungen kryptographischer Schwellwertschemata für den Anwendungsfall evaluiert. Außerdem werden Lösungen für das durch die Kombination beider Verfahren entstehende Problem der Suche nach bereits bestehenden Pseudonymen betrachtet.

Weiterhin wird ein System entworfen und auch prototypisch implementiert und evaluiert, das in Kombination mit einem SIEM-System die Umsetzbarkeit des Ansatzes zeigt. Hierzu wird aufgrund mangelnder Alternativen eine (eventuell auch in anderen Bereichen nutzbare) kryptographische Bibliothek entwickelt, die das genutzte Schwellwertschema umsetzt.

Insgesamt ermöglicht der Ansatz dieser Arbeit eine Vermittlung zwischen der Notwendigkeit Daten über Angestellte für die Anomalierkennung zu speichern und dem Arbeitnehmerdatenschutz, der die bedingungslose Speicherung und Verarbeitung dieser Daten verbietet.

Inhaltsverzeichnis

1 Einführung

Liest man von erfolgreichen Angriffen auf Unternehmensnetzwerke, so ist die implizite Annahme von externen, unternehmensfremden Angreifern weit verbreitet. Doch häufig sind die Angreifer bereits im Unternehmensnetzwerk selbst ansässig. Es handelt sich um (ehemalige) Mitarbeiter oder Personen mit legitimem Zugriff auf das Netzwerk, wie Geschäftspartnern oder Kunden. Hierbei spricht man von Insider-Angriffen, die ganz unterschiedlich ausfallen können:

- Mitarbeiter, die die Kundendatenbank des Unternehmens kopieren, um diese als Einstellungsgrund für den nächsten Arbeitgeber zu nutzen;
- Mitarbeiter, die aus Verärgerung über ihre bevorstehende Kündigung Projekte durch Datenlöschung sabotieren;
- Mitarbeiter, die einem Konkurrenten Details über das Angebot der eigenen Firma zu einer Ausschreibung liefern, an der der Konkurrent ebenfalls Interesse besitzt;
- ...

Bei dieser Art von Angriffen handelt es sich keineswegs nur um zu vernachlässigende Einzelfälle: In dem *IBM Cyber Security Intelligence Report* von 2015 werden 55 % der Angriffe als aus dem internen Netz stammend angegeben [Bra+15].[1] 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 den Bereichen Datendiebstahl, Industriespionage oder Sabotage durch (ehemalige) Mitarbeiter erfolgten [Bit16].

Auch wenn die genauen Zahlen aufgrund von unterschiedlichen Annahmen und der in diesem Bereich nicht zu vernachlässigenden Dunkelziffer mit Vorsicht zu betrachten sind,[2] so geben sie doch Hinweise darauf, dass Angriffe von Innentätern weit verbreitet sind und ein hohes Schadenspotenzial aufweisen können. Die Erkennung und Verhinderung solcher Angriffe sollte daher ein wichtiger Teil des IT-Sicherheitskonzepts eines Unternehmens sein.

Geht es darum, Insider-Angriffe zu erkennen oder zu verhindern, so sind viele bestehende Lösungen im Netzwerksicherheitsbereich nicht geeignet. Verbreitete Zugriffskontrollmechanismen, Firewalllösungen oder Intrusion-Detection/Prevention-Systeme konzentrieren sich oftmals lediglich auf die Erkennung oder Verhinderung externer Angriffe. Angriffe, die von legitimen Benutzern eines Netzwerks ausgeführt werden, wobei sie erlaubte Handlungen tätigen, sind jedoch auf diese Weise oftmals nicht adäquat zu behandeln.

Ein Ansatz, der diese Art von Angriffen entdecken kann, ist die sogenannte Anomalieerkennung. Dabei werden statistische Verfahren oder Verfahren aus dem Bereich des maschinellen Lernens benutzt, um Verhalten eines Nutzers, das vom erwarteten Verhalten abweicht, zu erkennen. Für

1. Zu beachten ist allerdings, dass nicht nur mit Absicht ausgeführte Angriffe hierunter erfasst wurden, sondern auch unbeabsichtigte wie das versehentliche Veröffentlichen schützenswerter Kundendaten.
2. Insbesondere die Angst vor Imageschäden, die auch in der *Spezialstudie Wirtschaftsschutz* erwähnt wird, könnte ein Grund für das Geheimhalten von Vorfällen sein.

diese Verfahren werden allerdings umfangreiche Daten über die Tätigkeiten und das Verhalten von Mitarbeitern benötigt.

Die Erhebung, Speicherung und Verarbeitung dieser Daten greift jedoch stark in das Recht auf informationelle Selbstbestimmung der Mitarbeiter ein. Aus den erfassten Daten können sich vielfältige Informationen über einen Arbeitnehmer gewinnen lassen: Beispielsweise können Daten aus elektronischen Türschlössern Bewegungsprofile ermöglichen oder Metadaten elektronischer Kommunikation Rückschlüsse auf persönliche Beziehungen erlauben.
Im Jahr 2017 wurde durch das Bundesarbeitsgericht ein Urteil gefällt, das die pauschale Überwachung der Rechnertätigkeit von Mitarbeitern verbietet.[3] Nach Bundesdatenschutzgesetz ist das Verwenden personenbezogener Daten für die Aufdeckung von Straftaten erst bei gegebenem Anfangsverdacht erlaubt. Hierauf bezogen sich die Richter des Bundesarbeitsgerichts in ihrem Urteil. Geklagt hatte ein Arbeitnehmer, dem aufgrund von privater Nutzung eines dienstlichen Computers gekündigt wurde. Sein Arbeitgeber hatte die Nutzung durch einen Keylogger überwacht.

Es gilt also, den Bedarf nach Überwachungsdaten zum Zweck der Anomalieerkennung dem Recht auf informationelle Selbstbestimmung des Arbeitnehmers gegenüberzustellen. Ein Ansatz, der diesen Konflikt beheben kann, ist die Entfernung des direkten Pesonenbezugs der Daten durch die Verwendung von Pseudonymen. Die Anomalieerkennung kann mit den pseudonymisierten Daten normal arbeiten und im Fall des Verdachts auf einen erfolgten Angriff anstelle von Überwachungsdaten, die Mitarbeiternamen im Klartext enthalten, nun pseudonymisierte Daten ausgeben. Nachdem die Anomalie überprüft wurde und ein Insider-Angriff für wahrscheinlich gehalten wird, muss der Halter des Pseudonyms aufgedeckt werden. Die Nutzung eines kryptographischen Schwellwertschemas kann diese Aufdeckung vor Missbrauch durch Einzelne technisch schützen.
Das Schwellwertschema ermöglicht die Aufdeckung eines Pseudonyms erst durch Kooperation mehrerer Beteiligter – denkbar wären beispielsweise je nach Unternehmensform und -größe Personen der Arbeitgeberseite, der Datenschutzbeauftragte oder Mitglieder des Betriebsrats, so dass die Interessen aller Beteiligten gewahrt werden können. Hierbei spricht man auch vom Mehraugenprinzip. Auf der anderen Seite verhindert ein Schwellwertschema auch die Blockadehaltung Einzelner, da für die Aufdeckung nicht alle Beteiligten zustimmen müssen, sondern nur ein im Vorwege bestimmter Anteil.

In dieser Arbeit soll ein System entworfen und prototypisch implementiert werden, das diese Art der datenschutzfreundlichen Speicherung umsetzt.[4] Hierfür sind detaillierte Betrachtungen der beteiligten Komponenten und ihres Zusammenspiels notwendig. Die Umsetzung soll auf Basis eines Security-Information-Event-Management-Systems (SIEM-System) geschehen. Hierbei handelt es sich um Systeme, die dem Sammeln und der Analyse von Ereignissen in Netzwerken dienen, die insbesondere Sicherheitsaspekte betreffen.

Diese Arbeit ist dabei wie folgt aufgebaut: In Kapitel 2 werden Grundlagen für die in dieser Arbeit verwendeten Verfahren dargestellt. Kapitel 3 stellt das zu entwickelnde System auf einer abstrakten Ebene dar. Es werden Anforderungen an ein solches System herausgearbeitet und darauf basierend eine verfahrensunabhängige Architektur entwickelt. Die zu verwendenden Verfahren werden in Kapitel 4 betrachtet, dabei werden mögliche Alternativen dargestellt und bewertet sowie Entscheidungen für den umzusetzenden Prototyp getroffen. Das anschließende 5.

3. BAG Erfurt, Az: 2 AZR 681/16.
4. Die Betrachtung der anschließend auf den gespeicherten Daten ausführbaren Anomalieerkennung wird in dieser Arbeit nicht behandelt.

Kapitel befasst sich mit der Implementation des Prototyp und seiner Evaluation. In Kapitel 6 werden ergänzende Datenschutztechniken und ihre Integration in den Prototyp betrachtet.

Related work

Im Bereich der Erkennung von Insiderangriffen fand bereits einige Forschungsarbeit statt. In [SHS08] bieten die Autoren einen Überblick über Forschungsergebnisse basierend auf unterschiedlichen Verfahren aus der Statistik und aus dem Bereich des maschinellen Lernens sowohl auf Host- als auch auf Netzwerkebene. Hier wird auch die Frage nach Erhalt der Privatsphäre eines Nutzers als Feld weiterer notwendiger Forschung dargestellt:

> „Hence, we also believe that any technologies developed to detect insider attack have to include strong privacy-preserving guarantees to avoid making false claims that could harm the reputation of individuals whenever errors occur. [...] How might a system alert a supervisor of a possible attack without disclosing an employee's true identity unless and until an attack has been validated?“ [SHS08]

Mit dieser Fragestellung beschäftigen sich weitere Veröffentlichungen im Bereich der Intrusion-Detection-Systeme. Oftmals wird – wie in dieser Arbeit auch – Pseudonymisierung als Verfahren zum Erhalt der Privatsphäre genutzt.
In [SFHR97] werden zwei Ansätze zur *Privacy Enhanced Intrusion Detection* vorgestellt. Die Pseudonymisierung wird jeweils bereits im Betriebssystem-Kernel vorgenommen und mithilfe symmetrischer Verschlüsselung erreicht. Auch die Nutzung des Mehraugenprinzips wird bei der Pseudonymaufdeckung bereits erwähnt. Die Autoren nennen hierzu die Aufteilung des symmetrischen Schlüssels auf mehrere Parteien als Ansatz.
In [BK99] stellt der Autor einen Architekturansatz für Intrusion-Detection-Systeme vor, der ebenfalls auf der Nutzung von Pseudonymen beruht. Es werden zwei Ansätze basierend auf Kerberos-Tickets bzw. auf dem Mix-Konzept vorgestellt. Für die Generierung bzw. Aufdeckung eines Pseudonyms wird eine *Trusted Third Party* benötigt.
In [LJ00] wird von den Autoren ein System zur Anomalieerkennung auf Basis von Pseudonymen entwickelt und anhand von Logdaten einer Firewall überprüft. Das Aufdecken von Pseudonymen wird hier als durch organisatorische Maßnahmen zu regelnder Prozess verstanden.
In [BF00] und [BF01] nutzen die Verfasser Shamir's Secret Sharing zur Erzeugung von Pseudonymen. Jeder Share bildet ein Pseudonym. Hierdurch wird sichergestellt, dass ein Pseudonym erst aufgedeckt werden kann, wenn eine einen Schwellwert übertreffende Anzahl von Warnmeldungen zu einem Nutzer im System eingetroffen ist.

Neben den Pseudonym-basierten Lösungen gibt es weitere auf anderen Verfahren basierende Forschungsergebnisse zur datenschutzgerechten Erkennung von Angriffen. In [Par+07] beispielsweise werden von den Autoren die Eigenschaften homomorpher Verschlüsselung zur privatsphäreerhaltenden Angriffserkennung eingesetzt. [Nik+13] verwendet eine Art der Mehrparteienberechnung, um mehrseitigen Datenschutz insbesondere im Hinblick auf Zero-Day-Lücken in einem Intrusion-Detection-System zu garantieren.

In [NKS17] bieten die Autoren einen Überblick über weitere existierende Lösungen im Bereich der *Privacy Enhanced Intrusion Detection*.

2 Grundlagen

Dieses Kapitel widmet sich den Grundlagen der in dieser Arbeit verwendeten Konzepte, Verfahren und Systeme. Die Abschnitte sind getrennt voneinander und auch bei Bedarf im Laufe der Arbeit zu lesen. Mit den Themen vertraute Leser können dieses Kapitel überspringen.

Zu Beginn werden die juristischen Hintergründe des Arbeitnehmerdatenschutzes erläutert, die den rechtlichen Rahmen für das Thema dieser Arbeit bilden. Es folgen Erläuterungen zu den Grundlagen von SIEM-Systemen, in die – wie bereits in der Einleitung erläutert – die prototypische Umsetzung der datenschutzfreundlichen Speicherung erfolgen soll, sowie zu Grundlagen der Pseudonymisierung, kryptographischer Schwellwertschemata und weiterer eingesetzter Verfahren.

2.1 Arbeitnehmerdatenschutz

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 gegeben werden, die für diese Arbeit relevant sind.

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

2.1.1 Das Recht auf informationelle Selbstbestimmung

Im sogenannten Volkszählungsurteil aus dem Jahr 1983 wurde das Recht auf informationelle Selbstbestimmung als Grundrecht anerkannt [2]. Es handelt sich um eine Ausprägung des allgemeinen Persönlichkeitsrechts[3] nach Artikel 2, Absatz 1 in Verbindung mit Artikel 1, Absatz 1 des Grundgesetzes und beschreibt das Recht des Einzelnen, selbst 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

1. In manchen Veröffentlichungen wird der Arbeitnehmerdatenschutz auch als Mitarbeiterdatenschutz, Beschäftigtendatenschutz, Personaldatenschutz oder Betriebsdatenschutz bezeichnet.
2. Bundesverfassungsgericht (BVerfG), Urteil vom 15. Dezember 1983, Az. 1 BvR 209/83, 484/83, 420/83, 362/83, 269/83, 440/83.
3. Das allgemeine Persönlichkeitsrecht beschreibt den Schutz der Persönlichkeit einer Person vor Eingriffen in ihren Lebens- und Freiheitsbereich.

Funktionsbedingung eines auf Handlungs- und Mitwirkungsfähigkeit seiner Bürger begründeten freiheitlichen demokratischen Gemeinwesens“.[4]

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 ebenfalls für 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“.[5]

2.1.2 Datenschutz im Beschäftigungsverhältnis

Eine besondere Situation 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“[6] definiert.

§ 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.“[7]

4. Volkszählungsurteil, Randnummer 172.
5. EU-Grundrechtecharta, Artikel 8, Absatz 1.
6. BDSG, § 3, Absatz 1.
7. BDSG, § 32, Absatz 1.

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 ist im Rahmen dieser Arbeit die Ausrichtung des BDSG auf personenbezogene Daten entscheidend, die – wie bereits definiert – einer bestimmbaren Person zugeordnet sind. Das zu entwerfende System wird durch Pseudonymisierung diesen direkten Personenbezug verhindern und erst durch Kooperation mehrerer Beteiligter im bestätigten Straftatverdacht die De-Pseudonymisierung ermöglichen.[8]

Im Jahr 2018 wird die EU-Verordnung 2016/679, besser bekannt als Datenschutzgrundverordnung, in Kraft treten. 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.

2.2 SIEM-Systeme

SIEM-Systeme dienen dazu, Daten in Netzwerken zu sammeln, um so einen zentralisierten Überblick über das Netzwerk zu erhalten und damit auch Bedrohungen erkennen und verhindern zu können. 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:** Anomalie-Erkennung auf Netzwerkebene durch die Messung von vom Normalzustand abweichendem 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 von Systemen in das Netzwerk.
- **Role based access control:** Zuständigkeitsabhängige Sichten auf Ereignisse.
- **Event Correlation:** siehe unten.

8. Eine fundierte rechtliche Bewertung der vorgeschlagenen Lösung kann hier allerdings nicht gegeben werden.

Eine besondere Bedeutung im Kontext dieser Arbeit kommt der Behandlung von sicherheitsrelevanten Ereignissen (Events) zu, die beispielsweise von Intrusion-Detection-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 deren 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 Zeitpunkte, 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 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).

Das Syslog-Protokoll

Das Syslog-Protokoll wird weitverbreitet für die Übertragung von Logdaten in Rechnernetzen genutzt und in dieser Arbeit als Grundlage für den Empfang von Logdaten im SIEM-System genutzt. Daher wird es hier kurz dargestellt. Nachrichten enthalten drei Felder: die *Priority*, die die Schwere des gemeldeten Vorfalls beschreibt, die *Facility*, die die Komponente beschreibt, in der der Vorfall auftrat, und die eigentliche Nachricht in textueller Form. Logdaten werden häufig über UDP übertragen, es gibt jedoch auch Erweiterungen, in denen TCP oder TLS genutzt wreden [Ger09].

2.3 Pseudonymisierung

Ein Pseudonym[9] bezeichnet nach [PH10] einen Identifikator eines Subjekts ungleich seinem echten Bezeichner. Im BDSG wird zusätzlich noch der „Zweck [eines Pseudonyms], die Bestimmung des Betroffenen auszuschließen oder wesentlich zu erschweren“[10] ergänzt. Beispiele für Pseudonyme lassen sich in verschiedensten Bereichen finden: E-Mail-Adressen, Sozialversicherungsnummern oder auch Autoren, die unter einem Pseudonym ihre Schriften veröffentlichen.

Nach [PWP90] lassen sich 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 (auch Pseudonymhalter

9. Ursprünglich aus dem Griechischen stammend: *pseudonumon* - falsch benannt.
10. BDSG, § 3, Absatz 6a.

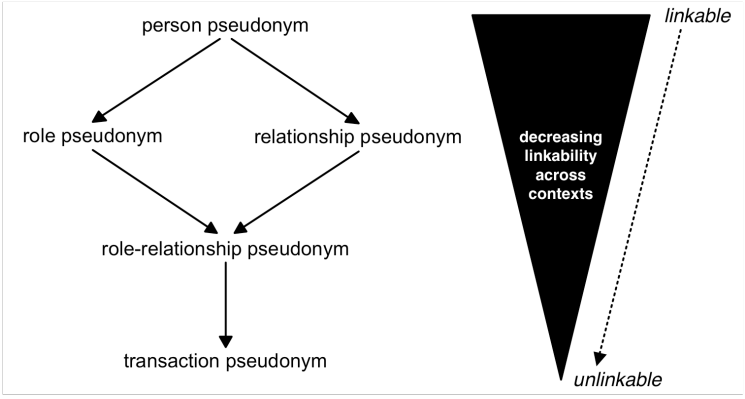

Abbildung 2.1: Pseudonym-Verband entsprechend ihrer Nutzung in verschiedenen Kontexten. Entnommen aus [PH10].

genannt) zu Beginn seiner Verwendung. Dieser Zusammenhang wird 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 öffentlichen Telefonbuch mit der Identität der Person verknüpft ist. Eine Kontonummer zu einem Bankkonto, dessen Inhaber bei der Kontoeröffnung nur der Bank und ihm selbst bekannt ist, bildet ein nicht-öffentliches Pseudonym – die Zuordnung ist nur wenigen Stellen bekannt. Ein anonymes Pseudonym ist zu Beginn nur dem Besitzer bekannt. Ein Beispiel bildet ein selbstgewählter Benutzername in einem Webforum ohne Hinterlegung identifizierender Merkmale wie Klarname oder Adresse.
Die beschriebene Eigenschaft eines Pseudonyms kann sich im Laufe der Zeit ändern, wenn Informationen über den Zusammenhang zwischen Pseudonym und zugehörigem Subjekt veröffentlicht werden.

In enger 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, die sich durch Nutzung des Pseudonyms in verschiedenen Kontexten unterscheiden: Personen-, Rollen-, Beziehungs-, Rollen-Beziehungs- oder Transaktionspseudonym sind mögliche Ausprägungen dieser Eigenschaft. Eine Übersicht über diese Pseudonymarten und deren Zusammenhang zur Verkettbarkeit ist Abbildung 2.1 zu entnehmen.
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 dar, 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.

2.4 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 Systeme. Das einfache Speichern eines Schlüssels an einem einzigen Ort resultiert in einer hohen Verlustwahrscheinlichkeit, da ein einzelner Fehler, wie z. B. 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 $D_1,\ldots,D_n$ (engl. *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].[11] Keine Information meint hier, dass jedes mögliche Geheimnis gleich wahrscheinlich D darstellt und die Kenntnis von weniger Shares als nötig nichts an diesen Wahrscheinlichkeiten ändert. Man spricht hierbei auch von informationstheoretischer Sicherheit.

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.

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

2.4.1 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, wobei n die Anzahl an späteren Share-Besitzern bezeichnet.

Weiterhin wird ein Polynom

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

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

derart gewählt, dass $a_1, \ldots, a_{t-1}$ zufällig gleichverteilt aus der Menge $\mathbb{Z}_p^* = \mathbb{Z}_p \setminus \{0\}$ 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.

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

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

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.

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.

2.4.2 Threshold Decryption

In [Des87] wird das Verfahren der Schwellwertschemata erstmals 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. Hierbei wird 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 einsetzbare 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:

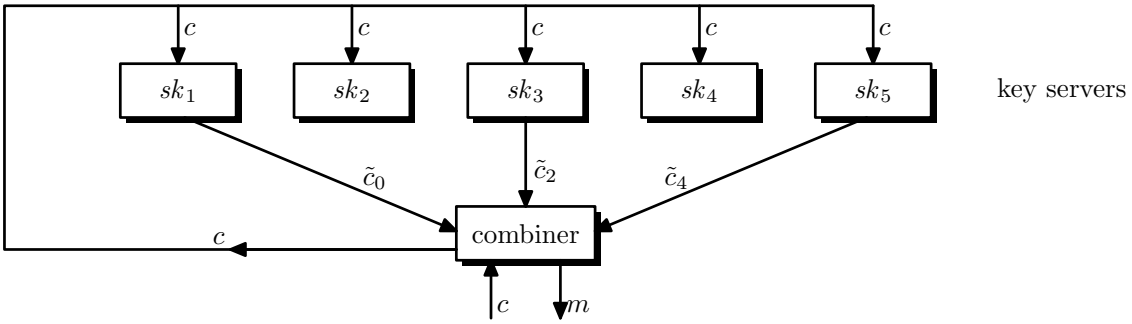

Abbildung 2.2: Übersicht über den Entschlüsselungsvorgang bei der Nutzung eines (3,5)-Schwellwertschemas. Entnommen aus [BS16].

- $G(t,n,r)$ ist der Algorithmus zur Generierung 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*. 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. Der einfließende Zufall r verhindert Wörterbuchangriffe. Näheres dazu ist in Abschnitt 4.4.2 zu finden.
- $D(sk_i,c)$ ist der Algorithmus, der für einen bestimmten *Share* und einen Schlüsseltext c eine partielle Entschlüsselung d'_j liefert. j stellt dabei den Index der partiellen Entschlüsselung in der Gruppe von an der Entschlüsselung beteiligter Shares dar.
- $C(c,d'_1,\ldots,d'_t)$ ist der Algorithmus, der aus dem Schlüsseltext c und aus t durch D generierten partiellen Entschlüsselungen wieder den Klartext m liefert. Dieser Algorithmus wird auch *Combiner* genannt.

Von diesen Algorithmen wird eine weitere 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.

Darüber hinaus 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 Weitere kryptographische Verfahren und Techniken

Dieser Abschnitt stellt die Grundlagen weiterer kryptographischer Verfahren und Techniken vor, die in dieser Arbeit verwendet werden.

2.5.1 Hashfunktionen

Eine Hashfunktion ist eine Funktion, die eine Eingabe variabler Länge auf eine Ausgabe fester Länge (den Hashwert) abbildet.

In der Kryptographie werden meist kryptographisch sichere Hashfunktionen eingesetzt. Bei dieser Art von Hashfunktionen handelt es sich um Einwegfunktionen, d.h. es ist leicht, aus einer Eingabe den Hashwert zu berechnen, jedoch nicht mit vertretbarem Aufwand möglich, zu einem gegebenen Hashwert eine Eingabe zu finden, die auf diesen Wert abgebildet wird.
Zusätzlich müssen die Hashfunktionen kollisionsresistent sein: Für einen gegebenen Wert ist es praktisch nicht möglich einen zweiten Wert zu finden, der den gleichen Hashwert besitzt [Sch06].

2.5.2 Message Authentication Codes

Ein Message Authentication Code (MAC) ist ein symmetrisches Verfahren, das dazu dient, die Authentizität und die Integrität einer Nachricht sicherzustellen. Dazu wird vom Sender aus einem geheimen Schlüssel k und der Nachricht m eine Art Prüfsumme generiert und zusammen mit der Nachricht versendet. Ein Empfänger kann den MAC überprüfen, wenn er im Besitz des gleichen geheimen Schlüssels ist, und kann somit sicher sein, dass die Nachricht nicht verändert wurde [Sch06].

2.5.3 Hybride Kryptosysteme

Als hybrides Kryptosystem wird die Kombination von symmetrischen und asymmetrischen Kryptoverfahren zur Verschlüsselung bzw. Entschlüsselung einer Nachricht bezeichnet. Ein Schlüssel k_{symm} für die Verwendung im symmetrischen Verfahren wird zufällig erzeugt und mithilfe des öffentlichen Schlüssels eines asymmetrischen Verfahrens als c_{public} verschlüsselt. Der zu verschlüsselnde Klartext m wird anschließend mithilfe des symmetrischen Verfahrens und des erzeugten Schlüssels k_{symm} als Chiffretext c_{symm} verschlüsselt.

Zur Entschlüsselung wird c_{public} mit dem geheimen Schlüssel des asymmetrischen Verfahrens entschlüsselt. Der hieraus erhaltene Schlüssel k_{symm} kann nun zur Entschlüsselung von c_{symm} genutzt werden, um m zu erhalten [KL14].

Der Vorteil dieser Lösung besteht darin, dass Vorteile symmetrischer und asymmetrischer Verfahren kombiniert werden: Einerseits sind symmetrische Verfahren im Allgemeinen deutlich

schneller als asymmetrische, andererseits lösen diese jedoch das bei symmetrischen Verfahren bestehende Problem des Schlüsselaustauschs.

2.5.4 Authenticated Encryption Schemes

Symmetrische Kryptosysteme sorgen zunächst einmal nur für den Schutz der Vertraulichkeit einer Nachricht. Wird zusätzlich die Integrität einer Nachricht durch das System geschützt, so spricht man von einem *Authenticated Encryption Scheme*. Hierdurch wird erreicht, dass Änderungen am Schlüsseltext bei der Entschlüsselung erkannt werden und der Vorgang abgebrochen werden kann.

Ein solches System kann durch Berechnung eines MACs zusätzlich zur Verschlüsselung erreicht werden. Alternativ dazu gibt es Schemata, die direkt auf einer Blockchiffre aufbauen [BS16]. Ein Beispiel hierzu ist der GCM-Betriebsmodus, der in Kombination mit AES in vielen verbreiteten Protokollen wie TLS zu finden ist.

2.5.5 Das ElGamal-Kryptosystem

Das ElGamal-Kryptosystem ist ein von Taher ElGamal entwickeltes asymmetrisches Kryptosystem, das zur Verschlüsselung und der Erstellung von Signaturen genutzt werden kann [ElG85]. Im Folgenden wird die Ver- und Entschlüsselung beschrieben, die für das in dieser Arbeit verwendete kryptographische Schwellwertschema relevant ist.

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 gelingt, da

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

gilt. Weitere Details und Beweise zu dem ElGamal-Kryptosystem sind beispielsweise in [KL14] zu finden.

Die Sicherheit des Verfahrens beruht auf dem Diskreten-Logarithmus-Problem. Es beschreibt die Schwierigkeit für einen gegebenen Wert $a = g^x \mod p$ für große Primzahlen p den Exponenten x zu berechnen.

2.6 Searchable Symmetric Encryption

Searchable Symmetric Encryption (SSE) ist ein Konzept, das es ermöglicht, Daten in verschlüsselter Form auf einen Server auszulagern und trotzdem Suchanfragen auf den Daten ausführen zu können. 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.

3 Überblick und Entwurf

Das Ziel dieser Arbeit ist es, ein System zu entwickeln, das mithilfe von Pseudonymisierung die datenschutzgerechte Speicherung von Überwachungsdaten ermöglicht, wobei die Identität eines Pseudonymhalters im Bedarfsfall durch die Kollaboration verschiedener Akteure unter Nutzung eines kryptographischen Schwellwertschemas aufdeckbar sein muss. Um eine Basis für die Erarbeitung von Anforderungen und für einen Systementwurfs zu erhalten, soll nun kurz dargelegt werden, wie die verschiedenen Verfahren ineinander greifen.

Aus Datenquellen wie Firewalls, Zugriffsprotokollen von Dateisystemen oder auch elektronischen Türschlössern werden personenbeziehbare Logdaten[1] an ein SIEM-System gesendet und dort gespeichert.
In diesen Datenfluss wird nun durch ein zu entwickelndes System eingegriffen, das die personenbeziehbaren Informationen des Logdatums[2] durch ein Pseudonym ersetzt. Die Zuordnung zwischen dem gesetzten Pseudonym und der personenbeziehbaren Information wird durch ein kryptographisches Schwellwertschema verschlüsselt und in dem System gespeichert.

Damit die anschließend auf den pseudonymisierten Logdaten stattfindende Anomalieerkennung Aktionen eines Mitarbeiters verknüpfen kann, muss sichergestellt werden, dass für einen speziellen Benutzer das gleiche Pseudonym verwendet wird. Wird nun durch die Anomalieerkennung ein Angriff erkannt, so können berechtigte Benutzer durch kooperative Entschlüsselung der Pseudonymzuordnung den hinter dem Pseudonym stehenden Benutzer wieder aufdecken.

Anschaulich wird der Vorgang in Abbildung 3.1 dargestellt: Die Benutzerin *Eve* agiert in dem Unternehmensnetzwerk. Ihre Aktionen werden protokolliert und Logdaten, die ihren Benutzernamen enthalten, werden versendet. Dieser Benutzername wird durch das Pseudonym *ps01* ersetzt. Die Zuordnung des Pseudonyms wird nun mithilfe eines kryptographischen Schwellwertschemas verschlüsselt und in einer Datenbank abgelegt. Das pseudonymisierte Logdatum wird im SIEM-System gespeichert.
Eingesetzte Anomalieerkennungsverfahren können anschließend auf die Daten des SIEM-Systems zugreifen. Wird ein möglicher Insider-Angriff durch einen Benutzer mit dem Pseudonym *ps01* erkannt, so kann die Zuordnung zu dem ursprünglichen Benutzernamen mithilfe des Schwellwertschemas wieder aufgedeckt werden. Hierzu ist jedoch die Mitarbeit von *Alice*, *Bob* und *Carol* notwendig, die jeweils im Besitz eines Teils des Entschlüsselungsschlüssels sind. Stimmen sie der Aufdeckung zu, so wird *Eve* als Halter des Pseudonyms *ps01* und damit als möglicher Innentäter aufgedeckt.

In diesem Kapitel werden zentrale Anforderungen an ein solches System entwickelt und eine abstrakte Architektur für ein solches entworfen. Anschließend wird darauf aufbauend ein Angreifermodell für das System definiert.

1. Da im Bereich technischer Systeme eher von Logdaten oder Protokolldaten im Gegensatz zu Überwachungsdaten gesprochen wird, wird diese Terminologie hier verwendet. Im Rahmen dieser Arbeit sind die Begriffe jedoch synonym zu verstehen.
2. Um Missverständnisse auszuschließen, sei darauf hingewiesen, dass das Wort Datum in dieser Arbeit als Beschreibung einer Informationseinheit und nicht eines Zeitpunkts verwendet wird.

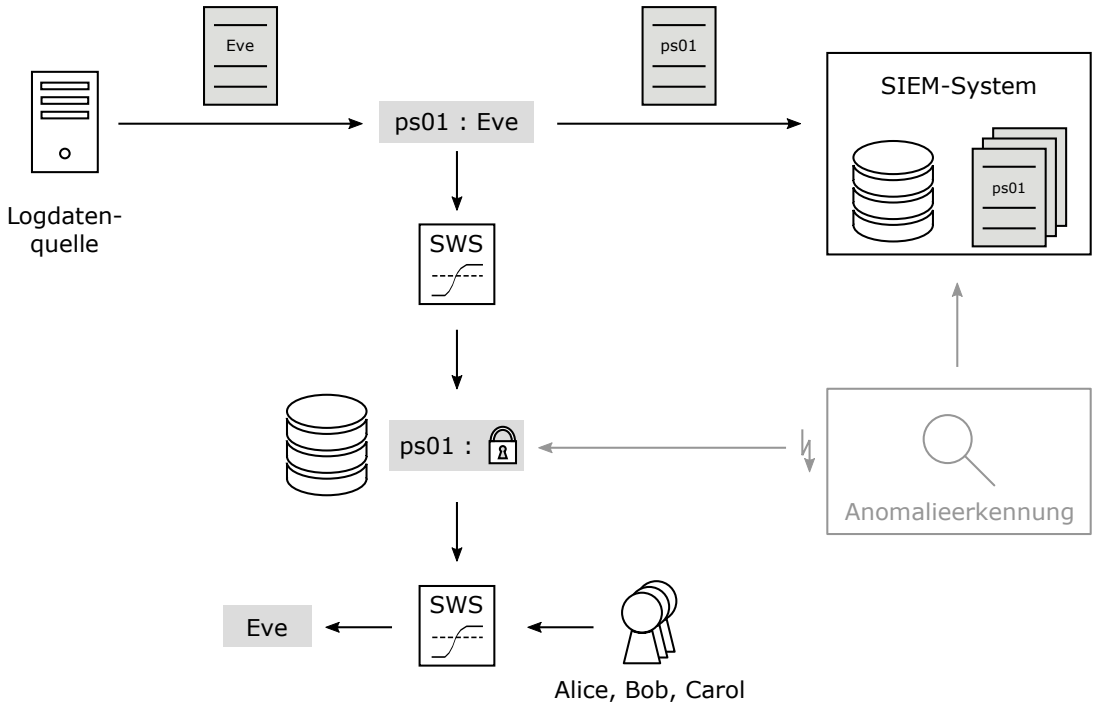

Abbildung 3.1: Übersicht zu dem angestrebten Verfahren.

3.1 Anforderungen

Neben den primären Anforderungen, die sich direkt aus der Funktionsbeschreibung des Systems und dem Zusammenspiel der enthaltenen Verfahren ergeben, sollte das System noch weitere Eigenschaften wie beispielsweise die Erweiterbarkeit um zusätzliche Datenschutztechniken erfüllen. All diese Anforderungen sollen im folgenden Abschnitt aufgestellt und näher erläutert werden.

3.1.1 Integration in das SIEM-System

Für den Eingriff in den Datenfluss der Logdaten zwischen ihrer Quelle und dem verwendeten SIEM-System muss eine geeignete Stelle gefunden werden. Hierzu müssen Auswirkungen des Eingriffs betrachtet sowie die Vor- bzw. Nachteile der verschiedenen Möglichkeiten gegeneinander abgewogen werden.

3.1.2 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 Mehrfachverwendung von Pseudonymen kommen.

Weiterhin muss es beim Pseudonymisieren von Logeinträgen eine Möglichkeit 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 stets mit dem gleichen Pseudonym versehen werden, um mithilfe der Verknüpfung von Einträgen Anomalieerkennungsverfahren sinnvoll einsetzen zu können. Auf diese Anforderung wird in Abschnitt 4.4 noch genauer eingegangen.

Außerdem muss es eine Möglichkeit geben, die Parameter der Pseudonymisierung, wie den Zeitraum ihrer Verwendung, konfigurierbar zu machen (siehe Abschnitt 4.2).

3.1.3 Einsatz eines kryptographischen Schwellwertschemas

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, in Abschnitt 2.4 beschriebenen Parameter t und n und auch die beteiligten Share-Besitzer müssen in dem System initial konfigurierbar 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 Stellen sicher gespeichert und abrufbar sein. Für diese Phase gibt es zwei Möglichkeiten:

- **Zentrale Generierung von öffentlichem Schlüssel und Shares**: Eine vertrauenswürdige Komponente generiert ein Schlüsselpaar und zerlegt den geheimen Schlüssel in die einzelnen Shares, die anschließend verteilt werden können.
- **Verteilte Schlüsselgenerierung**: Hierbei generieren die einzelnen Share-Besitzer jeweils ihre eigenen Shares. Durch verteilte Berechnungen kann hieraus der gemeinsame öffentliche Schlüssel erzeugt werden. Der geheime Schlüssel liegt auf diese Weise niemals an einer Stelle vor und ein vertrauenswürdiger Dritter ist nicht notwendig. Aus diesem Grund ist diese Lösung zu bevorzugen.

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 Akteure koordinieren. Anschließend muss eine Komponente die Rolle des *Combiners* übernehmen, so dass anschließend der den Pseudonymhalter beschreibende, entschlüsselte Datensatz im System vorliegt.

3.1.4 Benutzerinteraktion

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

Das Konfigurieren des Systems zur Integration verschiedener Datenquellen muss einem berechtigten Nutzer zugänglich gemacht werden. Ebenso sollte es für die – in der Aufgabenstellung geforderte – Erweiterbarkeit um 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.

Einem Administrator des Systems sollte es für die Benutzung eines kryptographischen Schwellwertschemas 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 am Prozess des Aufdeckens mithilfe des Schwellwertschemas zu beteiligen.

3.1.5 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 dem verwendeten SIEM-System weiterhin zu ermöglichen.

3.1.6 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 6). Das zu entwickelnde System 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.

3.1.7 Performanz

Das System sollte es, eingesetzt in einem Unternehmensnetzwerk, ermöglichen eine ausreichende Menge von Logdaten in einer bestimmten Zeitspanne behandeln zu können.

3.1.8 Übersicht

Ein System, wie es in dieser Arbeit angestrebt wird, sollte also folgende Eigenschaften aufweisen:

- Geeignete Stelle zum Eingriff in den Datenfluss zwischen Logdatenquelle und SIEM-System,
- parameterabhängige Generierung eindeutiger, aber in gewissem Rahmen verknüpfbarer Pseudonyme,
- sicherer, verteilter Einsatz eines anpassbaren kryptographischen Schwellwertschemas – vorzugsweise mit verteilter Schlüsselgenerierung,
- geeignete Benutzerinteraktion mit dem System an notwendigen Stellen,

- Erweiterbarkeit um unbekannte Datenquellen,
- Erweiterbarkeit um weitere Datenschutztechniken,
- Performanz.

3.2 Systementwurf

In diesem Abschnitt wird basierend auf den Anforderungen aus Abschnitt 3.1 eine von den eingesetzten Verfahren unabhängige Architektur für das System entworfen. Der erste Abschnitt beschäftigt sich mit der Frage, an welcher Stelle in den Datenfluss zwischen Quelle der Logdaten und SIEM-System eingriffen werden kann. Im anschließenden Abschnitt wird hierauf basierend die Systemarchitektur erstellt.

3.2.1 Eingriff in den Datenfluss des SIEM-Systems

Für den Eingriff zur Pseudonymisierung der Logdaten bieten sich verschiedene Stellen im Datenfluss eines SIEM-Systems an. Im Folgenden werden diese Möglichkeiten dargestellt und bezogen auf die jeweils resultierenden Eigenschaften einer Möglichkeit bewertet:

- **Veränderung des SIEM-Systems:** Muss das eingesetzte SIEM-System für die Umsetzung der Lösung angepasst werden? Dies wäre im Hinblick auf zukünftige Updates, die das SIEM-System durch seinen Entwickler erfährt, nicht wünschenswert, da jedes dieser Updates dafür sorgen könnte, dass die umgesetzte Lösung angepasst werden muss. Weiterhin würde dieser Ansatz ein SIEM-System erfordern, das entweder quelloffen vorliegt und verändert werden darf oder das die gewünschte Verhaltensänderung zumindest durch Erweiterungen zulässt.
- **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 des SIEM-Systems nicht beeinflusst werden kann, wäre es wünschenswert, dass die Logdaten das SIEM-System bereits in pseudonymisierter Form erreichen.
  Die Relevanz dieser Eigenschaft lässt sich am Beispiel des später in dieser Arbeit genutzten SIEM-Systems OSSIM erkennen: Wird das Syslog-Protokoll genutzt, um Logdaten in OSSIM aufzunehmen, so werden die Einträge erst in einer Logdatei abgelegt 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 oder verändert werden.
- **Mehrfaches Parsen von Logdaten:** Durch das SIEM-System werden die Logdaten - wie in Abschnitt 2.2 beschrieben - geparst und normalisiert. Aus Performancegründen ist eine Lösung zu bevorzugen, die diesen Vorgang oder Teile davon nicht mehrfach voraussetzt.
- **Abhängigkeit von Besonderheiten des SIEM-Systems:** Einige SIEM-Systeme bieten die Möglichkeit der verteilten Installation oder andere spezifische Eigenschaften. Eine Lösung, die unabhängig von dem verwendeten SIEM-System funktioniert, ist zu bevorzugen, da sie universell einsetzbar ist.

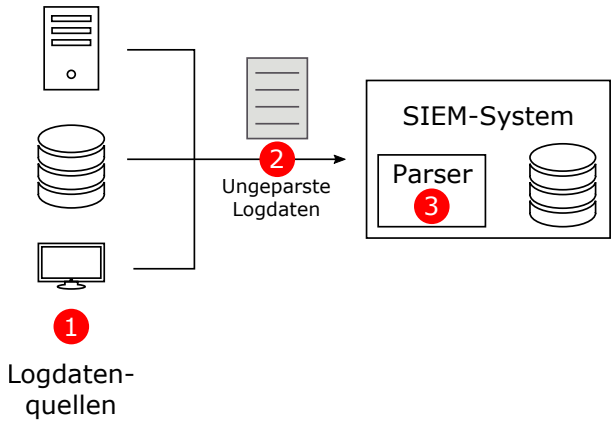

Abbildung 3.2: Mögliche Eingriffspunkte in den Datenfluss eines SIEM-Systems.

Eine Übersicht über die verschiedenen Stellen des Eingriffs bietet Abbildung 3.2. Die dort mit Ziffern gekennzeichneten Möglichkeiten sind:

1. **In der Quelle der Logdaten**: Bei diesem Ansatz werden die Daten bereits verädert, bevor sie die Datenquelle verlassen. Dieser Ansatz sorgt dafür, dass die Daten bereits pseudonymisiert auf der Übertragungsstrecke und im SIEM-System vorliegen. Es ist kein mehrfaches Parsen der Daten notwendig und der Ansatz ist unabhängig vom verwendeten SIEM-System zu realisieren. Auf der anderen Seite macht der Ansatz die Veränderung generell jeder Datenquelle notwendig. Dies kann bei Datenquellen, die auf ähnlichen gut erweiterbaren Plattformen beruhen, relativ einfach umzusetzen sein. Beispielsweise könnte die im nächsten Ansatz vorgestellte Proxy-Komponente lokal auf der Datenquelle eingesetzt werden. Schwierigkeiten würde dieser Ansatz hingegen bei Datenquellen bereiten, die beispielsweise aus Gründen abgespeckter zugrundeliegender Betriebssysteme oder wegen geringer Rechenleistung nur schwer erweiterbar sind. Außerdem würde der Ansatz in vielen Fällen die Kooperation des Herstellers voraussetzen, wenn es sich um nicht quelloffene „Box“-Lösungen handelt.

2. **Proxy-basierter Ansatz**: Dieser Ansatz pseudonymisiert die Daten vor dem ersten Kontakt mit dem SIEM-System, indem Datenquellen ihre Logdaten an einen Proxy senden, der die Daten pseudonymisiert und erst anschließend an das SIEM-System weiterreicht. Hierdurch wird erreicht, dass die Daten zu keiner Zeit nicht-pseudonymisiert im SIEM-System vorliegen. Der Ansatz ist unabhängig von den Datenquellen und dem SIEM-System und erfordert somit keinen direkte Eingriffe (abgesehen von geringen Konfigurationsanpassungen). Ein Nachteil dieser Lösung ist, dass sie das Parsen und Neuzusammensetzen der Logdaten im Proxy zusätzlich zu deren anschließender Behandlung im SIEM-System erfordert. Außerdem müssen für verschiedene Arten der Logdatenübermittlung (Protokolle wie Syslog oder SNMP) unterschiedliche Proxys entwickelt werden.

3. **Patchen des SIEM-Systems**: Die dritte Möglichkeit ist das Verändern des SIEM-Systems selbst. Hierzu wird in die Logdaten parsende Komponente eingegriffen, um vor, während oder nach diesem Vorgang die Logdaten zu pseudonymisieren. Dieser Ansatz erfordert kein mehrfaches Bearbeiten von Logdaten wie im proxybasierten Ansatz. Auf der anderen Seite ist er abhängig vom eingesetzten SIEM-System und erfordert seine Veränderung. Zusätzlich liegen die Daten erst einmal in nicht veränderter Form im SIEM-System vor, was die in Abschnitt 3.1.1 erwähnten Nachteile mit sich bringt.

Aus datenschutztechnischer Sicht ist eine frühestmögliche Pseudonymisierung zu bevorzugen, wie sie auch in [SW17] empfohlen wird: „Die Pseudonymisierung ist im Verarbeitungsprozess so früh wie möglich durchzuführen.“ Daher wäre eine Pseudonymisierung bereits in der Datenquelle der Optimalfall. Demgegenüber stehen jedoch die erwähnten Umsetzbarkeits-Nachteile des ersten Ansatzes, da hierzu jede mögliche Quelle von Logdaten verändert werden müsste. Eine erst im SIEM-System stattfindende Pseudonymisierung bringt jedoch die beschriebenen Risiken des Vorliegens pseudonymisierter Daten im Originalformat mit sich.

Dies begründet die Entscheidung für den proxybasierten Ansatz. Dass die Lösung außerdem noch keine Anpassungen an dem SIEM-System selbst erfordert, wiegt den Nachteil des zusätzlichen Parsens und Wiederzusammensetzens der Lognachricht bei Weitem auf.

3.2.2 Architektur

Ausgehend von diesen Überlegungen wird ein System entworfen, das die Anforderungen aus Abschnitt 3.1 erfüllt und proxybasiert in den Datenfluss eingreift.

Bei dem Entwurf handelt es sich um ein verteiltes System, bei dem die Verarbeitung der Logdaten und die Speicherung der Pseudonymzuordnung an unterschiedlichen Stellen geschieht. Hierfür sprechen verschiedene Gründe. Die Kompromittierung der speichernden Komponente schützt die erstellten Pseudonyme vor Aufdeckung durch die Verschlüsselung der Datensätze mit einem kryptographischen Schwellwertschema. Die Kompromittierung der verarbeitenden Komponente lässt zwar eine Verknüpfung neu erstellter Pseudonyme mit eintreffenden Daten zu, sorgt aber nicht für eine Aufdeckung bereits erstellter Pseudonyme, da diese in der anderen Komponente vorliegen.

Weiterhin sorgt dieser Ansatz auch für eine zusätzliche Erweiterbarkeit des Systems. Eine speichernde Komponente kann so als Datenspeicher für mehrere verarbeitende Komponenten agieren, was beispielsweise die Erweiterung um zusätzliche Protokolle (vgl. Abschnitt 5.1) oder Pseudonyme über verschiedene Datenarten (vgl. Abschnitt 4.4.6) ermöglicht.

Einen Überblick über den Entwurf bietet Abbildung 3.3. Die verschiedenen Komponenten des Systems werden im Folgenden näher beschrieben.

Die erste Komponente ist der **Log-Proxy**, der die Daten entgegennimmt, verändert und anschließend an das SIEM-System 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 3.1.6 ermöglicht. Der Proxy leistet die Behandlung von Logdaten aus verschiedenen Quellen wie in Abschnitt 3.1.5 beschrieben. Das für diese Art des Dateneingriffs erforderliche Parsen und Wiederzusammensetzen der Daten muss hier datenquellenabhängig zu konfigurieren sein.

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

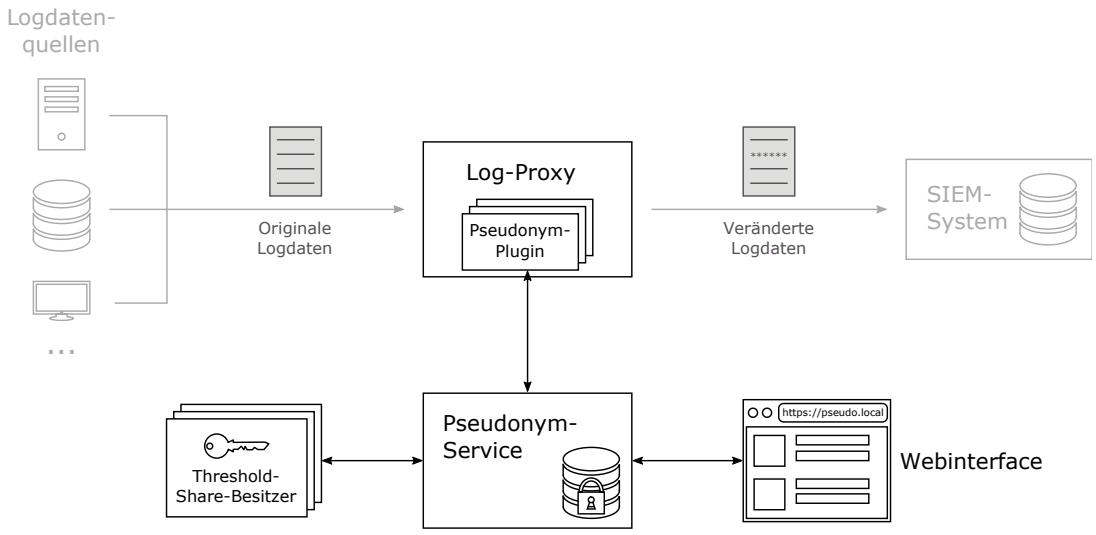

Abbildung 3.3: Ein Überblick über die entworfene Architektur.

Der **Pseudonym-Service** erfüllt zwei Aufgaben: Speichern und Verwalten der Pseudonyme sowie die Integration des kryptographischen Schwellwertschemas. Initial muss die Schlüsselgenerierung des Schwellwertschemas durch den Service geleistet werden. Dies kann wie bereits im vorhergehenden Abschnitt beschrieben zentral oder verteilt geschehen.

Während des Betriebs können 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 (dieses Problem wird in Abschnitt 4.4 genauer dargestellt).
Über ein Webinterface kann ein berechtigter 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. Weiterhin kann über dieses Webinterface auch die initiale Konfiguration des Systems in Bezug auf Eigenschaften der Pseudonymisierung und des kryptographischen Schwellwertschemas vorgenommen werden.

Benutzer, die über das Webinterface oder den Webservice mit dem Pseudonym-Service agieren möchten, werden durch geeignete Maßnahmen authentifiziert und ihre Autorisierung wird überprüft.

Benutzer, die für die Bewertung von Anfragen zur Aufdeckung eines Pseudonyms zuständig sind, erhalten die Möglichkeit zur Interaktion mit dem System über eine **Client-Anwendung**, für die der Pseudonym-Service ebenfalls als Webservice agiert. Diese Anwendung verwaltet den *Share* des Benutzers für das kryptographische Schwellwertschema. Sie kann, nachdem der Benutzer der Aufdeckung eines Pseudonyms zugestimmt hat, die partielle Entschlüsselung eines Datensatzes berechnen und diese an den Pseudonym-Service senden.

Für alle Übertragungsstrecken wird angenommen, dass ein Angreifer keinen Zugriff auf Kommunikationsinhalte erhält oder diese verändern kann. Dies ist durch Transportverschlüsselung mittels des weitverbreiteten TLS-Protokolls zu erreichen, muss jedoch bei der Verwendung des Systems beachtet werden.

3.3 Angreifermodell

Das Sicherheitsziel des Systems lässt sich folgendermaßen definieren: Das Pseudonym eines Nutzers erlaubt (ohne Anwendung von Hintergrundwissen) keinen Rückschluss auf die Identität eines Nutzers. Erst die Kooperation berechtigter Akteure ermöglicht das Aufdecken eines Pseudonyms.
Deswegen soll sich auch das nachfolgend aufgestellte Angreifermodell auf dieses Ziel fokussieren. Andere Angriffsarten, wie beispielsweise Angriffe auf die Verfügbarkeit des Systems, werden dementsprechend nicht betrachtet.

Ein Angreifermodell beschreibt die maximale Stärke eines Angreifers in Bezug auf verschiedene Faktoren, gegen die ein System abgesichert ist. Enthalten sind die Rolle eines Angreifers, seine Verbreitung im System, aktives/passives und beobachtendes/veränderndes Verhalten und die Rechenkapazität, die der Angreifer zum Überwinden der eingesetzten Schutzmaßnahmen aufbringen kann. [BFP14]

Bezogen auf die verfügbare Rechenleistung des Angreifers sollen verbreitete und nach heutigem Wissensstand für sicher befundene kryptographische Algorithmen als nicht mit vertretbarem Aufwand zu brechen angesehen werden. Es handelt sich um die Annahme von komplexitätstheoretischer Sicherheit. So wird also bespielsweise das in Abschnitt 2.5.5 erwähnte Diskrete-Logarithmus-Problem für ausreichend große Primzahlen als praktisch nicht zu brechen betrachtet.

Im Bezug auf die Verbreitung eines Angreifers 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 Schutz ihrer Integrität über das Netzwerk. Hierdurch könnte ein Angreifer bereits vor dem Eintreffen der Daten im Proxy passiv alle Daten mitlesen und die anschließend stattfindende Pseudonymisierung würde nichts an dem gewonnen Wissen ändern können. Die Unterbindung solcher Anrgiffsmöglichkeiten ist indes nicht Inhalt dieser Arbeit, in der es um die datenschutzfreundliche **Speicherung** von Überwachungsdaten geht. Daher werden bei der Verbreitung eines Angreifers die Datenquellen und Übertragungswege zum Log-Proxy nicht betrachtet.

Mit dieser Einschränkung ergeben sich verschiedene Rollen und darauf basierend Verbreitungen, die ein Angreifer annehmen kann:

Außenstehender : Als Außenstehender wird hier jeder Akteur verstanden, der keinen legitimen Zugriff auf Teile des Systems besitzt. Gemeint sind also genauso Mitarbeiter in einem Unternehmensnetzwerk, in dem das System genutzt wird, wie externe Angreifer. Sind die Zugriffsmechanismen im Pseudonym-Service korrekt umgesetzt, bleibt Außenstehenden nur das passive Beobachten von Nachrichten. Durch den Einsatz der in Abschnitt 3.2.2 erwähnten notwendigen Transportverschlüsselung erfahren passive Angreifer jedoch keine brauchbaren Informationen. Diese Transportverschlüsselung verhindert auch das Verändern der Nachrichten auf der Übertragungsstrecke durch aktive Angreifer.

Benutzer mit SIEM-Zugriff : Ein Benutzer, der Zugriff auf das SIEM-System besitzt, sieht nur die pseudonymisiert gespeicherten Logdaten. Hieraus erfährt er erst einmal nichts über den Benutzer hinter dem Pseudonym. Durch die Anwendung von Hintergrundwissen und die Verknüpfung von Datenbankeinträgen kann er bestimmte Pseudonyme eventuell aufdecken – ein Angriff, der nicht zu verhindern ist, durch regelmäßige Pseudonymwechsel jedoch zumindest in seiner Reichweite beschränkt werden kann.

Benutzer mit Recht auf Pseudonymaufdeckung : Legitimierte Benutzer können Anfragen zur Aufdeckung eines Pseudonyms stellen. Durch den Einsatz des kryptographischen Schwellwertschemas führt jedoch erst die Kooperation einer ausreichenden Zahl von Share-Besitzern zur wirklichen Aufdeckung. Vorher erfährt der Benutzer nichts über den Pseudonymhalter.

Share-Besitzer : Besitzer eines Shares erhalten für aufzudeckende Pseudonymzuordnungen die verschlüsselten Daten und berechnen aus ihrem Share und dem Schlüsseltext eine partielle Entschlüsselung. Ausgehend von den Eigenschaften des kryptographischen Schwellwertschemas erfahren sie hieraus jedoch nichts über das verschlüsselte Datum, solange nicht eine Mindestanzahl t an partiellen Entschlüsselungen vorliegt. Erst eine Kollaboration von mindestens t als aktive Angreifer handelnden Share-Besitzern kann so unberechtigt Pseudonyme aufdecken, wenn sie zusätzlich Zugriff auf die Datenbank des Pseudonym-Service erlangen.
Eine Kollaboration von mindestens $n-t+1$ böswilligen Share-Besitzern könnte auch dazu führen, dass die Aufdeckung eines Pseudonyms durch das Senden fehlerhafter partieller Entschlüsselungen fehlschlägt. Die sinnvollen Aufteilung der Shares und damit die Modellierung von verteiltem Vertrauen spielt also in dem System eine wichtige Rolle. Zusätzlich könnten im Falle der verteilten Schlüsselgenerierung böswillige Share-Besitzer beispielsweise durch das Senden falscher Daten versuchen, die Generierung der Schlüssel zu stören und damit das anschließende Aufdecken von Pseudonymen zu verhindern. Dieser Angriff muss bei der Schlüsselgenerierung durch das verwendete Schema verhindert werden.

Administrator mit Proxy-Zugriff : Der Zugriff auf den Proxy, an dem die Logdaten im Klartext eintreffen und pseudonymisiert werden, erlaubt die Zuordnung von Pseudonymen zu Benutzern. Hier muss der Zugriff nach der Initialisierung eingeschränkt werden und soweit wie möglich sichergestellt werden, dass auch zugriffsberechtigte Benutzer den Zugriff nicht ausnutzen (z.B. durch das Mehraugenprinzip geschützt oder zumindest durch Protokollierung der Handlungen).

Administrator mit Pseudonym-Service-Zugriff : Der Administrator des Pseudonym-Service hat Zugriff auf die Datenbank der verschlüsselten Pseudonymzuordnungen. Durch den Einsatz des Schwellwertschemas und die Verschlüsselung der Logdaten-Pseudonym-Zuordnung bereits im Proxy erfährt er (kein Hintergrundwissen vorausgesetzt) nichts über die Pseudonymhalter. Besondere Bedeutung kommt der Schlüsselgenerierung im Pseudonym-Service im Falle der zentralen Schlüsselgenerierung zu. Gelingt es dem Administrator während der Schlüsselgenerierung in den Besitz des temporär erzeugten geheimen Schlüssels oder von mindestens t Shares zu kommen, so kann er jederzeit in der Datenbank abgelegte Pseudonymzuordnungen entschlüsseln, ohne dass andere Benutzer dies mitbekommen. Aus diesem Grund ist die verteilte Schlüsselgenerierung unbedingt zu bevorzugen.

4 Auswahl von Verfahren und Systemen

In diesem Kapitel wird der aktuelle Stand der Forschung und Entwicklung bezogen auf die in dieser Arbeit verwendeten Klassen von Systemen bzw. Verfahren betrachtet. Ausgehend von den Anforderungen an das zu entwickelnde System werden passende Lösungen ausgewählt und in für folgende Kapitel notwendiger Detailtiefe beschrieben.

Der erste Abschnitt befasst sich mit zur Zeit verfügbaren SIEM-Systemen und beschreibt Eigenschaften des ausgewählten Systems näher, die für die Integration des in dieser Arbeit zu entwickelnden Prototyp relevant sind.

Im zweiten Abschnitt werden Eigenschaften von Pseudonymen und Anforderungen an die Verwendung von Pseudonymisierung herausgearbeitet, die bei der Entwicklung eines Systems beachtet werden müssen.

Der dritte Abschnitt stellt den Forschungsstand im Bereich der kryptographischen Schwellwertschemata dar und beschreibt das ausgewählte Verfahren im Detail.

Der letzte Abschnitt befasst sich mit verschiedenen Ansätzen zur Lösung des in Abschnitt 3.2.2 erwähnten Problems beim Suchen und Wiederverwenden bereits vergebener Pseudonyme für ein eintreffendes Datum. Basierend auf Vor- und Nachteilen der verschiedenen Lösungen wird eine Auswahl für ein zu implementierendes Verfahren getroffen.

4.1 SIEM-Systeme

Zur Zeit gibt es eine vielfältige Auswahl an SIEM-Systemen auf dem Markt: Splunk[1], QRadar von IBM[2] oder ArcSight von Micro Focus[3] sind nur einige Beispiele aus diesem Bereich. Neben den in Abschnitt 2.2 beschriebenen grundlegenden Funktionen eines SIEM-Systems, die von allen Kandidaten in unterschiedlichem Maße bereitgestellt werden, unterscheiden sie sich insbesondere in darüber hinausgehenden Techniken: Beispielhafte Möglichkeiten sind hier die Nutzung von Machine Learning zur Erkennung ungewöhnlichen Verhaltens oder die Automatisierung von Handlungen im Bedrohungsfall.

In diesem Bereich ist die Auswahl an quelloffener Software jedoch sehr gering. Eine Ausnahme 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 2.2 beschriebenen Funktionen bereitstellt. AlienVault bietet zusätzlich 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

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. https://www.alienvault.com/products/ossim

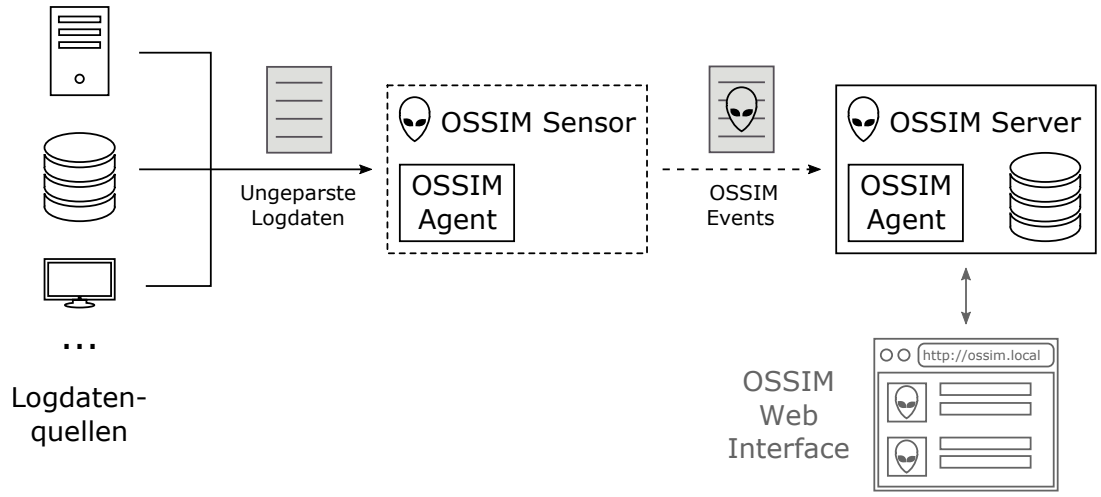

Abbildung 4.1: High-Level-Übersicht über die OSSIM-Architektur und den Datenfluss.

die in USM fließt, profitiert jedoch auch OSSIM, beispielsweise durch die Aktualisierung von Plugins für die Einbindung von aktuellen Netzwerkgeräten.

Die Entscheidung des in dieser Arbeit verwendeten SIEM-Systems fiel im Wesentlichen aus zwei Gründen auf OSSIM: Zum einen ist die Quelloffenheit gerade im Sicherheitsbereich generell zu bevorzugen, da die Funktionalität von Komponenten jederzeit und durch jedermann überprüfbar ist. Zum anderen bietet dies auch die Möglichkeit, Komponenten des SIEM-Systems direkt zu verändern, falls es für diese Arbeit notwendig ist.

4.1.1 AlienVault OSSIM

Zunächst soll eine Übersicht über die für diese Arbeit relevanten Komponenten von OSSIM und deren Zusammenspiel gegeben werden. Diese ist auch in Abbildung 4.1 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 von 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.

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 stattfinden 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.

4.1.2 Parsen von Logdaten in OSSIM

Von besonderer Bedeutung für diese Arbeit ist die Verarbeitung von Logdaten. OSSIM ermöglicht es, Logdaten aus unterschiedlichen Quellen entgegenzunehmen bzw. aktiv selbst 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 *Simple Network Management Protocol* (SNMP),
- Entgegennehmen von Daten über proprietäre Protokoll wie *Security Device Event Exchange* (SDEE) oder *Windows Management Instrumentation* (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 selbst entwickelte Plugins für verschiedene Datenquellen zu aktivieren. Für eintreffende Logdaten überprüft der Agent anhand von regulären Ausdrücken, ob ein Plugin für das entsprechende Datum zuständig ist. Ist ein solches Plugin gefunden, so wird ein neues OSSIM-Event angelegt und anhand der Angaben im Plugin werden die entsprechenden vorgegebenen Felder des Events gesetzt. Hierbei kann es sich beispielsweise um 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.

4.2 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 von das Subjekt 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 auch eine aufsteigende Nummerierung als Pseudonym könnte durch den hierdurch genauer spezifizierten Erstellungszeitpunkt Rückschlüsse auf das Subjekt hinter dem Pseudonym ermöglichen.

Pseudonymisierung sagt etwas über die Verwendung eines Verfahrens aus, jedoch nichts über die daraus entstehenden Auswirkungen auf die Identifizierbarkeit eines Subjekts oder auf 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.

4.2.1 Pseudonymisierung in der Praxis

Um die Auswirkungen obiger Eigenschaften einordnen zu können, wird an dieser Stelle die Pseudonymisierung in zwei Systemen betrachtet und die Relevanz der eben genannten Eigenschaften verdeutlicht: Pseudonyme in Mobilfunknetzen und in der Fahrzeug-zu-Fahrzeug-Kommunikation.

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 die längerfristige Ortung der Teilnehmer und damit Bewegungsprofile verhindern soll.

In [Ara+14] beschreiben die Autoren Schwächen der Implementierungen von Mobilfunkstandards in Netzen bei der (Neu-)Vergabe einer TMSI. Bestimmte Eigenschaften für die Unverkettbarkeit von Pseudonymen und damit für die Privatsphäre der Nutzer werden in vielen Netzen aufgrund einiger Schwächen nicht erreicht:

- Pseudonyme werden zu selten geändert,
- Pseudonyme werden nicht abhängig von der Häufigkeit ihrer Nutzung geändert,
- Pseudonyme werden in verschiedenen Funkbereichen beibehalten,
- die Neuvergabe ist für Replay-Angriffe anfällig.

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.

VANets

Ein anderer Bereich, in dem man 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 Erkennung von Bremsvorgängen naher Fahrzeuge oder für 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 Erstellen von Bewegungsprofilen verhindern lassen.

Unter anderem in [Dö05] und in [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. vom Angreifermodell sind und außerdem gegenüber den Anforderungen

5. Es werden beispielweise Lösungen vorgestellt, die abhängig von Geschwindigkeitsänderungen, von einer gewissen Anzahl anderer Fahrzeuge oder von besonderen Verkehrssituationen, wie Kreuzungen, Pseudonymänderungen vornehmen. Das Ziel ist hier immer, die Möglichkeit der Pseudonymverkettung bzw. der Bewegungsprofilerstellung durch die äußere Situation der Pseudonymänderung zu erschweren.

von Sicherheitsanwendungen[6] abgewogen werden müssen.
Bei der Nutzung von Pseudonymen in VANets handelt es sich natürlich um eine Anwendung mit anders gelagerten Prioritäten verglichen mit dem Kontext dieser Arbeit. Dennoch wird deutlich, dass die Strategie zum Pseudonymwechsel stark von der Anwendungssituation abhängig ist.

Im hier vorliegenden Anwendungsfall werden zum einen Besonderheiten der Datenquelle, wie die Häufigkeit von auftretenden Überwachungsdaten, und zum anderen Anforderungen, die die auf den Daten beruhende Anomalieerkennung an die Verknüpfbarkeit von Ereignissen stellt, zu beachten sein.

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

4.2.2 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 sein.

Wie auch in den Beispielen deutlich wurde, müssen Pseudonyme abhängig von dem Anwendungsszenario in bestimmten Fällen für einen Benutzer gewechselt werden. Im 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. Es handelt sich also um eine schwächere Form der Transaktionspseudonyme, bei der ein Pseudonym je nach Pseudonymwechselstrategie nur für eine bestimmte Anzahl an Ereignissen verwendet wird.

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

- Die notwendigen Pseudonymwechsel 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 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.

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.

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, auf Datendiebstahl schließen lassen könnte.

- Weiterhin muss die Pseudonymwechselstrategie auch von der später auf den pseudonymisierten Überwachungsdaten auszuführenden automatisierten Anomalieerkennung abhängen. Je nachdem, welche Verfahren auf Daten aus welchen Datenquellen eingesetzt werden sollen, könnte hier unterschiedliche Verknüpfbarkeit der Daten erforderlich sein. Hieraus ergibt sich ein Konflikt zwischen den Anforderungen der Anomalieerkennung und dem Arbeitnehmerdatenschutz, denn Verknüpfbarkeit von mehr Daten kann zu genaueren Benutzerprofilen führen und damit eher Rückschlüsse auf den Halter eines Pseudonyms ermöglichen.

Aus diesen Gründen wird eine parameterabhängige Pseudonymwechselstrategie implementiert, die sowohl zeit- als auch nutzungsabhängige Wechsel ermöglicht. Wie lange bzw. häufig ein Pseudonym verwendet wird, kann so in Anwendungen mit konkreten Rahmenbedingungen beurteilt und verwendet 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]

Zusätzlich könnten in speziellen Anwendungsbereichen eventuell weitere parameterabhängige Wechselstrategien sinnvoll sein. Als Beispiel sei auf die bereits erwähnten Pseudonymwechsel bei einem Ortswechsel in Mobilfunknetzen verwiesen. Diese aufbauenden Strategien müssen jedoch im konkreten Fall individuell betrachtet werden.

Weiterhin wird angestrebt, für die Pseudonyme bzw. ihre Aufdeckung die erwähnte Perfect Forward Privacy zu ermöglichen. Die konkrete Umsetzung dieser Eigenschaft und die Folgen daraus wird in Abschnitt 5.2.4 näher eingegangen.

4.3 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.

4.3.1 Übersicht

Eine Vielzahl von Veröffentlichungen behandeln 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].

4.3.2 ElGamal-basiertes Schwellwertschema

Ein Verfahren zur *Threshold Decryption*, das auf auf einer geschickten Kombination von Shamir's Secret Sharing (Abschnitt 2.4.1) mit dem ElGamal-Kryptosystem (Abschnitt 2.5.5) basiert, veröffentlichten die Autoren in [DF90]. Aufbereitete Darstellungen lassen sich in [KL14] und [BS16] finden.

Es ist eines der ersten veröffentlichten Schwellwertschemata und erfuhr dadurch viel Beachtung; entsprechend existieren viele aufbauende Arbeiten, die Verbesserungen vorschlagen. Durch die zugrundeliegende Mathematik bietet das Schema gegenüber RSA-basierten Verfahren einfachere Umsetzbarkeit (auch von Erweiterungen wie dezentraler Schlüsselgenerierung).[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.

Der Rest dieses Abschnitts stellt das Verfahren nun entsprechend den in Abschnitt 2.4.2 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 werden an die Teilnehmer verteilt. Anschließend

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

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 $(*)$ der Gleichung folgt direkt aus dem zugrundeliegenden Secret-Sharing-Schema und ist in dieser Form bereits in Abschnitt 2.4.1 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 $|C| \geq t$.

4.3.3 Verteilte Schlüsselgenerierung

Ein Nachteil dieses Verfahrens in der Phase der Schlüsselgenerierung ist, dass für die Generierung des geheimen Schlüssels und der daraus resultierenden Shares eine zentrale und vertrauenswürdige Instanz notwendig ist. Diese Problematik wurde bereits in Abschnitt 3.1.3 dargestellt und die Auswirkungen wurden in Abschnitt 3.3 betrachtet.

In [Ped91] wurde vom Autor eine Möglichkeit der verteilten Schlüsselgenerierung für das dargestellte Verfahren vorgeschlagen, die von den Autoren in [Gen+99] noch verbessert wurde.

Das Verfahren besteht aus zwei Phasen: In der ersten Phase wird von allen potentiellen Share-Besitzern ein *Verifiable Secret Sharing Scheme*[11] (VSS) nach Pedersen ausgeführt, das dafür sorgt, dass anschließend alle ehrlichen Beteiligten jeweils im Besitz eines Shares sind, die zusammengenommen den geheimen Schlüssel x bilden (der jedoch weder irgendwo vorliegt noch im Laufe des Verfahrens vorlag). In der zweiten Phase wird ein VSS nach Feldman dazu genutzt, den gemeinsamen öffentlichen Schlüssel $y = g^x$ auf eine Weise zu berechnen, die wiederum dafür sorgt, dass der geheime Schlüssel nirgendwo vorliegen muss.
Auf diese Weise wird die vertrauenswürdige Instanz vermieden und es ist trotzdem sichergestellt, dass die ehrlichen Beteiligten im Besitz von Shares sind, die die Verwendung des kryptographischen Schwellwertschemas so ermöglichen, wie im letzten Abschnitt für das Verfahren mit zentraler Schlüsselgenerierung vorgestellt.

4.3.4 ECC-ElGamal

Eine andere Verbesserung für das Verfahren ist die Verwendung von *Elliptic Curve Cryptography*. Hier werden die Berechnungen des ElGamal-Verfahrens nicht mehr in der beschriebenen Untergruppe von $\mathbb{Z}_p^*$, sondern als Operationen auf elliptischen Kurven über endlichen Körpern ausgeführt [Kob87].

Der Vorteil der Verwendung liegt darin, dass im Vergleich zum ursprünglichen Verfahren eine deutlich geringere Schlüssellänge für vergleichbare Sicherheit benötigt wird.[12] Durch diese kürzeren Schlüssel werden auch Berechnungszeit und Speicherverbrauch trotz komplexerer Berechnungen eingespart.

4.3.5 Komplexe Zugriffsstrukturen

Das bisher beschriebene Verfahren ermöglicht genau eine Art von Schema zur Entschlüsselung: Mindestens t Benutzer, die im Besitz von mindestens t von n Shares sind, können eine verschlüsselte Nachrichten entschlüsseln. Hierbei spricht man auch von der Zugriffstruktur des Verfahrens.

11. Verifiable Secret Sharing Schemes sind Secret Sharing Schemes, die es den Share-Besitzern erlauben zu überprüfen, ob ihre Shares konsistent sind, d.h. ob es möglich ist, aus den Shares ein gemeinsames Geheimnis wiederherzustellen. Bei dem in Abschnitt 2.4.1 vorgestellten Secret Sharing nach Shamir ist dies beispielsweise nicht der Fall. Ein bösartiger Erzeuger von Shares könnte für jeden Beteiligten ein anderes Geheimnis benutzen, sodass bei der Rekonstruktion abhängig von beteiligten Share-Besitzern unterschiedliche Geheimnisse erhalten werden.
12. Das BSI gibt eine Schlüssellänge von etwa 250 Bit für ECC-Verfahren an, die eine vergleichbare Sicherheit zu 2000-Bit-Schlüsseln für Verfahren wie RSA oder auf dem Diskreten-Logarithmus-Problem beruhenden Verfahren bietet [Inf18].

Durch die Vergabe unterschiedlich vieler Shares an verschiedene Benutzer lassen sich verschiedene Zugriffsstrukturen ermöglichen. In [ISN89] erbringen die Autoren den Beweis, dass durch dieses Vorgehen beliebige Zugriffsstrukturen ermöglicht werden können – auf Kosten einer (strukturabhängig) relativ großen Zahl von Shares.

Auf diese Weise kann die zur Entschlüsselung notwendige Benutzergruppe abhängig von der Organisationsstruktur eines Unternehmens beliebig modelliert werden.

4.3.6 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[13] oder Helios[14], die jedoch alle eng 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 beschriebene Schwellwertschema notwendigerweise in Teilen selbstständig implementiert.

4.4 Identifizierung existierender Pseudonyme

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 4.2 ist dieses Problem anhand eines Beispiels dargestellt. Zu einem Zeitpunkt liegen in der Pseudonymtabelle Pseudonyme für zwei Benutzer *User A* und *User B* vor. Dass 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.

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

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

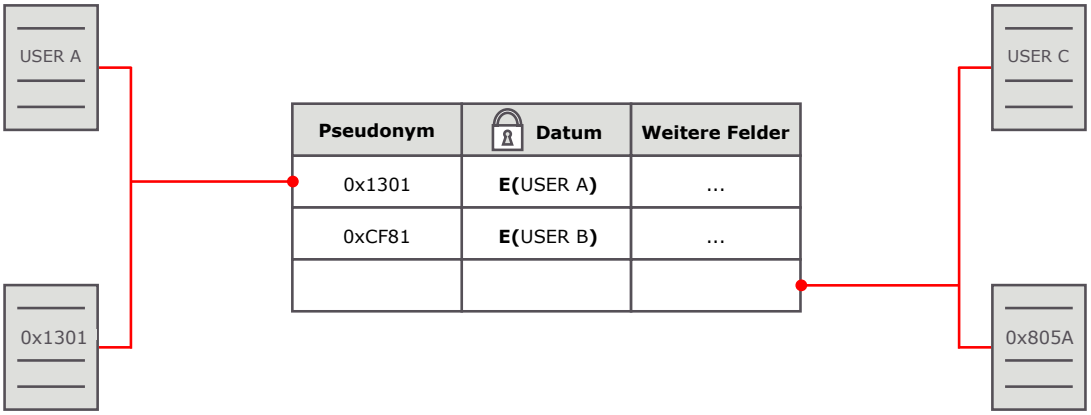

Abbildung 4.2: Erhalt von Pseudonymen aus der Zuordnungstabelle: Links für den Fall eines bereits bekannten Datums (hier Benutzer), rechts für ein unbekanntes Datum.

4.4.1 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 Basisannahmen, die der Sicherheit des Systems zugrundeliegen.

4.4.2 Deterministische Verschlüsselung

Die zweite, ebenfalls 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 handelt, bei dem bei der Verschlüsselung ein Zufallswert einfließt – folglich ein nicht-deterministisches Verfahren.

Dieser Nicht-Determinismus ist notwendig, da ansonsten zur Aufdeckung eines Pseudonyms auch ein Wörterbuchangriff 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 einen solchen Angriff relativ effizient machen. Aus diesem Grund muss ein nicht-deterministisches Verschlüsselungsverfahren genutzt werden und damit ist die Überprüfung auf Schlüsseltextgleichheit nicht möglich.

4.4.3 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 Wörterbuchangriff: Ein Angreifer könnte mithilfe der bekannten Hashfunktion die Hashwerte aller möglichen Werte berechnen und mit dem Hashwert des aufzudeckenden Pseudonyms vergleichen.

4.4.4 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 wird auch in [Goh04] erwähnt.

Ein Nachteil bezogen auf das für diese Arbeit zu entwickelnde System ist jedoch die notwendige Generierung von Pseudonymen und die Speicherung der Zuordnungstabelle an der Stelle, an der neue Daten eintreffen. Hierdurch wird die Verwendung eines leichtgewichtigen Log-Proxys, wie es in der Architektur vorhergesehen ist (siehe Abschnitt 3.2), verhindert. Außerdem würde eine Kompromittierung dieser Komponente direkt zur Aufdeckung des Pseudonymzusammenhanges führen. Zusätzlich könnte ein verteilter Ansatz, bei dem mehrere Log-Proxys eine gemeinsame Datenbank und gleiche Pseudonyme für ein Datum nutzen, nicht umgesetzt werden.

4.4.5 Message Authentication Codes

Aufbauend auf der Hash-basierten Lösung lässt sich jedoch auch eine nicht für einen Wörterbuchangriff anfällige Lösung entwickeln. Dazu wird der verwendete Hash durch einen schlüsselabhängigen MAC (siehe Abschnitt 2.5.2) ersetzt. Beim Speichern eines neuen Eintrags wird dazu unter Zuhilfenahme 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 Wörterbuchangriff 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, denn er ermöglicht einer Komponente die Zuordnung eines Datums zu einem Pseudonym, ohne 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 Kombination mit Hintergrundwissen könnte so möglicherweise in bestimmten Fällen der Inhaber eines Pseudonyms herausgefunden werden.

Ein zusätzliches, zumindest in der Theorie bestehendes Problem ist, dass verschiedene Daten den gleichen MAC erzeugen könnten. In der Praxis ist die Wahrscheinlichkeit hierfür jedoch zu vernachlässigen: Bei einer MAC-Länge von 256 Bit tritt nach dem Geburtstagsparadoxon selbst bei 2^{80} MAC-Berechnungen eine Kollision nur mit einer Wahrscheinlichkeit von 2^{-80} auf [BS16].

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

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 zwischen den 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 [Goh04] 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, worin 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.

5 Implementierung

In diesem Kapitel wird die Implementation des Prototyp aufbauend auf dem in Kapitel 3 erstellten Entwurf und auf den in Kapitel 4 erarbeiteten konkreten Verfahren dargestellt. Neben der Einbindung in das SIEM-System und der Pseudonymisierung wird insbesondere die für die Nutzung eines kryptographischen Schwellwertschemas entwickelte Bibliothek im Detail erörtert. Anschließend erfolgt eine Evaluation des entwickelten Prototyp.

5.1 Einbindung in OSSIM

Wie in Abschnitt 4.1 beschrieben, erlaubt OSSIM eine verteilte Installation am Empfang von Logdaten beteiligter Komponenten durch die Einführung optionaler Sensoren, die den Empfang und das Parsen von Logdaten übernehmen. Dadurch bieten sich gegenüber den in Abschnitt 3.2.1 beschriebenen Möglichkeiten zum Eingriff in den Datenfluss eines SIEM-Systems zwei weitere OSSIM-spezifische Möglichkeiten, die folgend als mögliche Alternativen beschrieben werden.
Eine Übersicht über den entstehenden Datenfluss bietet Abbildung 5.1. Die Ziffern beschreiben die Stelle des Eingriffs, wie in der Abbildung gekennzeichnet:

4. **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 der Logdaten 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 – ein Nachteil, der bereits in Abschnitt 4.1 ausführlich diskutiert wurde. Zusätzlich erfordert diese Lösung die verteilte Installation von OSSIM-Sensor und -Server, schließt also die All-In-One-Installation aus.

5. **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.

Diese beiden Möglichkeiten leiden also unter denselben Nachteilen wie der Ansatz einer Veränderung des SIEM-Servers und sind zusätzlich nur bei der Nutzung von OSSIM einsatzbar. Deshalb fiel die Entscheidung für den zu entwickelnden Prototypen auf die bereits in Abschnitt 3.2.1 präferierte Lösung eines Proxys zwischen Datenquelle und SIEM-System.

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.

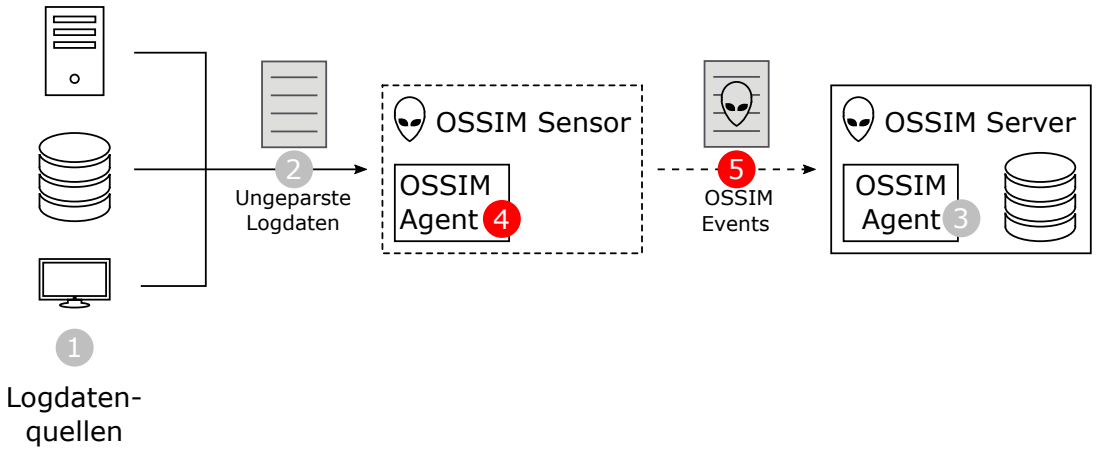

Abbildung 5.1: Mögliche Eingriffspunkte in den OSSIM-Datenfluss.

Weiterhin wird der am häufigsten[2] genutzte Weg des Datenerhalts in OSSIM verwendet: das Entgegennehmen der Daten über das Syslog-Protokoll (siehe Abschnitt 2.2). Lösungen für weitere Wege (siehe Abschnitt 4.1.2) lassen sich bei Bedarf jedoch vergleichbar implementieren.

Der entwickelte Log-Proxy nimmt Logdaten über das Syslog-Protokoll entgegen. Verschiedene Datenquellen und Eventformate können mithilfe von Konfigurationsdateien angelegt werden, wie im nächsten Abschnitt beschrieben. Für einzelne Felder eines Logdatums können dabei Plugins angegeben werden, die die Behandlung des Feldes übernehmen.
Der Proxy-Server kann durch einfach zu entwickelnde Plugins leicht um weitere Datenschutztechniken erweitert werden. Dies wird im übernächsten Abschnitt beschrieben. Nach der Behandlung eines Logdatums wird es anschließend über das Syslog-Protokoll an den OSSIM-Server oder -Sensor geschickt.

5.1.1 Konfiguration von Datenquellen

Die Behandlung von verschiedenen Datenquellen wird durch Konfigurationsdateien ermöglicht:

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

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 enthalten sein können, so unterstützt dies doch die Annahme des Syslog-Protokolls als am häufigsten genutzten Weg und damit als geeignet für diese Arbeit.

Eine Konfigurationsdatei kann aus mehreren Bereichen bestehen. Der `general`-Bereich enthält dabei allgemeine Angaben über das Plugin. Um unterschiedliche Lognachrichten eines Gerätes bündeln zu können, kann eine Konfigurationsdatei weiterhin 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, das die Gruppe verarbeiten soll.
Durch diese Konfigurationsdateien können Nachrichten unbekannter Formate aus neuen Datenquellen leicht in das bestehende System eingebunden werden.

5.1.2 Umsetzung von Datenschutztechniken durch Plugins

Die Erweiterbarkeit um neue Datenschutztechniken wird durch leicht erweiterbare Plugins ermöglicht. Ein Plugin muss lediglich die Methode `handle_data` implementieren, die die originalen Daten und alle in der Konfiguration angegebenen Parameter erhält.
Ein einfaches Plugin, das die Daten durch einen 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
```

5.2 Umsetzung der Pseudonymisierung

Zum Pseudonymisieren von Daten wurde ein Plugin für den Proxy entwickelt, wie in Abschnitt 5.1 beschrieben. 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 hingenommen werden. Die Speicherung dieser Zuordnung erfolgt jedoch nur im 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.

5.2.1 Setup-Phase

Vor Verwendung der Pseudonymisierung müssen die Parameter zur Pseudonymgenerierung (vgl. Abschnitt 4.2) dem System bekannt gemacht werden. Diese Parameter können in dem Pseudonym-Service mittels einer Konfigurationsoberfläche durch einen berechtigten Benutzer

gesetzt werden. Die für den Proxy relevanten Parameter können anschließend über eine Schnittstelle abgefragt werden. Vorerst handelt es sich hierbei lediglich um das maximale Zeitintervall, in dem Pseudonyme genutzt werden dürfen.

5.2.2 Proxy

Beim Start des Proxies wird zuerst das beschriebene Zeitintervall erhalten. Anschließend können eintreffende Logdaten verarbeitet werden. Das eintreffende Datum wird verschlüsselt (siehe dazu Abschnitt 5.3) und anschließend zusammen mit einem über das Datum generierten MAC, der – wie in Abschnitt 4.4 beschrieben – für die Überprüfung auf bereits bestehende Pseudonyme genutzt wird, an den Pseudonym-Service gesendet. Das ursprüngliche Datum wird nun durch das gelieferte Pseudonym (siehe nächsten Abschnitt) ersetzt und das so veränderte Logdatum wird an das SIEM-System weitergeleitet.

Der Schlüssel, der für die Generierung des MACs verwendet wird, wird abhängig von dem erhaltenen Parameter nach einer bestimmten Zeitspanne neu generiert. Durch diesen Schlüsselwechsel wird erreicht, dass 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 eventuell dann 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 4.2 beschrieben Rückschlüsse auf das ursprüngliche Datum des Pseudonyms liefern könnte. Daher wurde dieser Ansatz nicht weiter verfolgt.

5.2.3 Service

Auf der Service-Seite wird anhand des empfangenen MACs durch Vergleich mit in der Datenbank vorliegenden MACs überprüft, ob bereits ein Pseudonym für das eintreffende Datum vergeben wurde, das noch nicht zu häufig verwendet wurde. Hierzu wird der in der Konfiguration gesetzte Parameter zur maximalen Nutzungshäufigkeit von Pseudonymen verwendet. Liegt kein solches Pseudonym vor, so wird ein noch nicht verwendetes, zufälliges Pseudonym erstellt und zusammen mit dem MAC und dem verschlüsselten Datum in der Datenbank gespeichert. Anderenfalls wird das bereits vergebene Pseudonym zurückgeliefert.

5.2.4 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. Auf diese Weise 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 4.2 beschriebenen Prinzip der *Perfect Forward Privacy* widerspricht.

Dieses Problem könnte durch eine Hashwert-Berechnung für den eintreffenden MAC auf der Service-Seite verhindert werden, die einen Pseudonym-abhängigen Zufallswert (eine Art Salt) einbezieht. Hierdurch enthalten Datenbankeinträge, die innerhalb eines Zeitintervalls zu dem gleichen Datum gehören und daher den gleichen MAC besitzen, durch den einfließenden Zufallswert unterschiedliche Hashwerte. Durch die Einweg-Eigenschaft der Hashfunktion wäre die Verkettbarkeit verschiedener Pseudonyme verhindert. Jedoch erfordert dieser Ansatz eine Hash-Berechnung pro Datenbankeintrag für jede Anfrage und ist daher aus Performance-Sicht kritisch zu betrachten. Aus diesem Grund wird diese Möglichkeit vorerst nicht implementiert. Das bestehende Problem ist jedoch für ein konkretes Anwendungsszenario und bei der Wahl der Parameter – insbesondere des Zeitintervalls – zu beachten.

5.3 Implementierung und Integration des Schwellwertschemas

In diesem Abschnitt wird zuerst die entwickelte Bibliothek zur Nutzung eines kryptographischen Schwellwertschemas dargestellt und anschließend ihre Nutzung in den unterschiedlichen Systemkomponenten beschrieben.

5.3.1 Kryptographische Bibliothek

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

- **Parametergenerierung:** Diese Funktion dient dem Erhalt der benötigten sicheren Primzahl bzw. des Generator (siehe Abschnitt 4.3.2). Hierbei kann zwischen Neugenerierung und Verwendung vorberechneter Parameter verschiedener Schlüsselstärken entschieden werden. Näheres dazu ist im Unterabschnitt *Parametergenerierung* zu finden.
- **Schlüssel- und Sharegenerierung:** Durch diese Funktion wird ein zufälliger geheimer Schlüssel erzeugt und daraus der öffentliche Schlüssel sowie die einzelnen *Shares* abgeleitet.
- **Verschlüsselung einer Nachricht:** Diese Funktion verschlüsselt mithilfe des öffentlichen Schlüssels eine Nachricht (siehe auch Unterabschnitt *Hybride Kryptographie*) und wird im Proxy verwendet.
- **Berechnung einer partiellen Entchlüsselung:** Mithilfe eines *Shares* wird die zugehörige partielle Entschlüsselung einer verschlüsselten Nachricht berechnet. Diese Funktion wird in den Threshold-Clients genutzt.
- **Kombinieren von partiellen Entschlüsselungen:** Aus einer ausreichenden Anzahl von partiellen Entschlüsselungen kann die Nachricht wiederhergestellt bzw. entschlüsselt werden (siehe wiederum Unterabschnitt *Hybride Kryptographie*).

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 Unterabschnitt *Hybride Kryptographie*).
- **PartialDecryption:** Enthält die zu einer partiellen Entschlüsselung gehörenden Werte, die zum Entschlüsseln der vollständigen Nachricht genutzt werden.

An dieser Stelle wird auf zwei Kernelemente des implementierten Verfahrens eingegangen.

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 Bereich der vorgegebenen Schlüsselstärke zufällig gewählt, bis $2q+1$ ebenfalls eine Primzahl ist. Zur Überprüfung der Primalität der entsprechenden Zahlen wird der Miller-Rabin-Test[3] 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

$$(g^q \equiv 1 \mod p) \text{ sowie } (g^2 \not\equiv 1 \mod p)$$

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 das Secret Sharing lediglich für die Aufteilung des geheimen Schlüssels in Shares sorgt, aber an den grundlegenden Eigenschaften der Ver- und Entschlüsselung im ElGamal-Schema nichts ändert.
Daher werden 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 als hinreichend sicher für die nächsten Jahre angegeben[4].

3. Der Miller-Rabin-Primzahltest ist ein Algorithmus, der basierend auf probabilistischen Annahmen die Primalität einer Zahl überprüft. Durch die Nutzung einer variablen Rundenzahl kann dies mit beliebig hoher Wahrscheinlichkeit geschehen.
4. Für einen Vergleich verschiedener Empfehlungen siehe: https://www.keylength.com

Hybride Kryptographie

Die Verschlüsselung bzw. Entschlüsselung wurde in Form eines hybriden Verschlüsselungsverfahrens umgesetzt (siehe Abschnitt 2.5.3): 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, k_{symm} zu verschlüsseln. Ein Schlüsseltext besteht daher aus drei Teilen:

- $v = g^k$ – der erste Teil der ElGamal-Verschlüsselung des symmetrischen Schlüssels (siehe Abschnitte 2.5.5 und 4.3.2);
- $c_{tc} = k_{symm} \cdot g^{ak}$ – der zweite Teil der ElGamal-Verschlüsselung des symmetrischen Schlüssels (siehe Abschnitte 2.5.5 und 4.3.2);
- $c_{symm} = E^{symm}_{k_{symm}}(m)$ – 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 neben dem im Grundlagenkapitel erwähnten Geschwindigkeitsvorteil weitere Vorteile:

Das Vorgehen ermöglicht es, 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, das neben der Verschlüsselung auch die Validität der Daten überprüft - ein sogenanntes *Authenticated Encryption* (AE) Schema (siehe Abschnitt 2.5.4).
In der Implementierung wird die Standardfunktion zur symmetrischen Verschlüsselung aus der Kryptographie-Bibliothek NaCl[5] genutzt – ein AE-Schema auf Basis der Algorithmen Salsa20 und Poly1305. Diese verwendete Bibliothek ist weit verbreitet und von erfahrenen Kryptographen entwickelt und überprüft. Dies erhöht das Vertrauen in eine sichere Umsetzung der Verfahren.

Eine direkte Implementierung des bis hierhin beschriebenen hybriden 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 des Nachrichtenraums $r \in \mathbb{Z}_q^*$ gewählt und der symmetrische Schlüssel mithilfe einer kryptographisch sicheren Hashfunktion

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

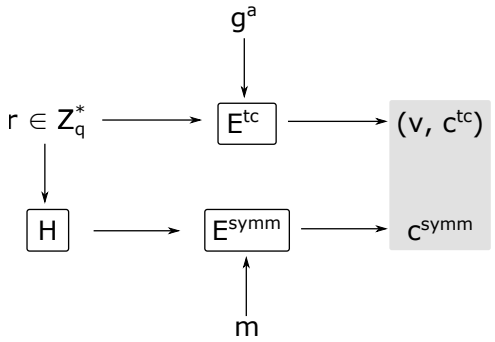

Abbildung 5.2: Übersicht zu dem umgesetzten hybriden Verschlüsselungsschema.

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 nach [ABR99] 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-Schema geschützt ist. Einen Überblick über das letztlich umgesetzte Schema, das auch diesen Angriff verhindert, bietet Abbildung 5.2.

5.3.2 Service und Setup-Verfahren

Um die Parameter für das kryptographische Schwellwertschema setzen zu können, wurde die Konfigurationsoberfläche erweitert. Hier müssen nun Angaben zur Schlüsselstärke, zu beteiligten Teilnehmern und zur erforderlichen Anzahl an Teilnehmern für die Entschlüsselung eines Datenbankeintrages gemacht werden.

Anschließend können berechtigte Benutzer die Aufdeckung eines Pseudonyms über das Webinterface anfragen. Danach 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.

5.3.3 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 zu 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 Teilnehmer diese annehmen oder ablehnen. Die Annahme führt zur Berechnung einer partiellen Entschlüsselung mithilfe der Bibliotheksfunktion, die anschließend an den Pseudonym-Service gesendet wird.

5.3.4 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 in Abschnitt 5.2 beschriebenen weiteren Daten an den Service gesendet.

5.4 Evaluation

Grundsätzlich konnten die in Abschnitt 3.1 aufgestellten Anforderungen in der Implementierung des Prototyp erfüllt werden. Mit der Entwicklung eines Log-Proxies wurde eine geeignete Stelle für den Eingriff in den Logdatenfluss gewählt. Die zeit-und nutzungsabhängige Generierung eindeutiger Pseudonyme wurde ebenso umgesetzt wie die Implementierung des ausgewählten Schwellwertschemas – vorerst allerdings nur mit zentraler Schlüsselgenerierung.
Die benötigten Parameter können während des Setup-Vorgangs über ein Webinterface gesetzt werden. Dieses ermöglicht auch die Erstellung von Anfragen zur Aufdeckung eines Pseudonymhalters. Besitzer eines Shares können diese Anfragen bearbeiten oder ablehnen. Die derzeitige Umsetzung als Konsolenanwendung kann allerdings als eher unkomfortabel angesehen werden. Der entwickelte Log-Proxy ist um weitere Datenschutztechniken und Datenquellen einfach erweiterbar.

In dem Prototyp bleiben allerdings auch noch einige Angriffsmöglichkeiten offen, die im nächsten Abschnitt dargestellt werden. Betrachtungen zur Performanz der entwickelten Lösung werden in dem darauf folgenden Abschnitt angestellt.

5.4.1 Angriffsmöglichkeiten

Zentrale Schlüsselgenerierung

Ein bereits in Abschnitt 3.1.3 betonter Punkt ist die Präferenz von verteilter gegenüber der zentralen Schlüsselgenerierung. In dem entwickelten Prototyp wurde jedoch aus Zeitgründen bisher nur die zentrale Schlüsselgenerierung umgesetzt. Dies ermöglicht einem Angreifer mit (legitimen oder nicht legitimen) Zugriff auf den Pseudonym-Service während der Schlüsselgenerierung den in Abschnitt 3.3 beschriebenen Angriff zur beliebigen Entschlüsselung von Pseudonymzuordnungen. Abhilfe würde ein Schema wie das in Abschnitt 4.3.3 beschriebene schaffen.

	512 Bit	1024 Bit	2048 Bit
μ	25	509	5394
σ	28	297	6719

Tabelle 5.1: Mittelwert (μ) und Standardabweichung (σ) in Sekunden für die Dauer der Schlüsselgenerierung bei unterschiedlichen Schlüsselstärken.

Nicht überprüfte kryptographische Funktionen

Die Bibliotheksfunktionen des kryptographischen Schwellwertschemas wurden nach bestem Wissen und Gewissen umgesetzt. Trotzdem wurden sie bisher nicht von erfahrenen Kryptographen überprüft und können daher eine Vielzahl von Schwächen und Angriffsmöglichkeiten aufweisen. Die Nutzung einer quelloffenen und vielfach überprüften Bibliothek wäre wünschenswert, aber wie in Abschnitt 4.3.6 beschrieben, wurde eine solche Bibliothek nicht gefunden.

Schlüsseltexte offenbaren Klartextlänge

Eine Eigenschaft des entwickelten hybriden Kryptoschemas kann dazu führen, dass trotz der Verschlüsselung auf den Halter eines Pseudonyms geschlossen werden kann. Bei dem verwendeten symmetrischen Verschlüsselungsalgorithmus Salsa20 handelt es sich um eine Stromchiffre. Hierdurch hat der Schlüsseltext, der in der Datenbank für ein Pseudonym gespeichert wird, genau die gleiche Länge wie der Klartext. Unterscheiden sich die Klartexte (beispielsweise Benutzernamen) in ihrer Länge, so kann durch Vergleich dieser Längen auf den Pseudonymhalter geschlossen werden. Dieser Angriff erfordert Zugriff auf den Inhalt der Datenbank des Pseudonym-Service.

Dieses Problem lässt sich durch Padding beheben, das alle Klartexte auf die gleiche Länge bringt. Die konkrete Umsetzung ist allerdings abhängig von dem Wertebereich der eintreffenden Daten.

Anwendung von Hintergrundwissen

In [LJ99] wird das Problem der Identifikation eines Pseudonymhalters durch die Anwendung von Hintergrundwissen, wie z. B. die Kenntnis über normales Nutzerverhalten, dargestellt. Dies ist auch in dem in dieser Arbeit entwickelten System der Fall, wie bereits in Abschnitt 3.3 erwähnt.

Dieser Angriff lässt sich aus Sicht des Autors nicht vollständig verhindern, da Beobachtungen in der realen Welt nicht zu vermeiden sind. Regelmäßige, parameterabhängige Pseudonymwechsel, wie sie implementiert wurden, sollten diesen Angriff jedoch in seinen Auswirkungen beschränken.

5.4.2 Performanz

Zu Beginn der Setup-Phase können vorberechnete Parameter gewählt werden oder diese Parameter neu generiert werden. Die (nicht optimierte) Umsetzung des in Abschnitt 5.3.1 beschriebenen Algorithmus in Python erfordert lange Berechnungsdauern, deren Mittelwerte und Standardabweichungen in Tabelle 5.1 zu finden sind. Die hohe Standardabweichung folgt aus einer geringen

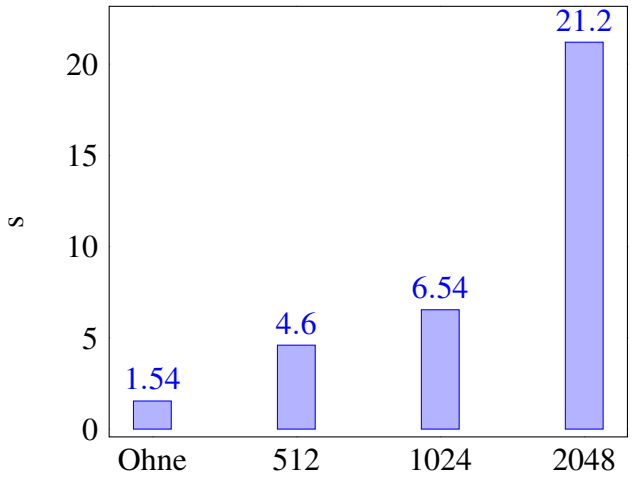

Abbildung 5.3: Dauer der Bearbeitung von 100 Lognachrichten in Sekunden.

Messanzahl,[6] aber insbesondere auch aus dem Einfließen von Zufallszahlen und der zufälligen Verteilung der Primzahlen.

Es zeigt sich, dass die Zeiten für angemessene Schlüsselstärken um 2000 Bit für die interaktive Nutzung während der Setup-Phase des Systems nicht mehr geeignet sind. Diese ließen sich durch eine optimierte Implementierung sicherlich deutlich reduzieren. Wie in Abschnitt 5.3.1 bereits beschrieben sollte jedoch die Nutzung vorberechneter Parameter kein Sicherheitsrisiko darstellen.

Die Schlüsselgenerierung muss nur einmalig während der Setup-Phase ausgeführt werden. Entscheidender für die Performanz des Systems im Betrieb sind die Betrachtungen zur Dauer der Logdatenbehandlung. Hierfür wurden Messungen an einem einfachen Aufbau aus zwei Rechnern vorgenommen, von denen der eine den Log-Proxy, der andere den Pseudonym-Service inklusive dessen Datenbank beherbergte. Gemessen wurde die Zeit des Empfangs von Logdaten in praxisnaher Größe (etwa 0.5 KB), die ohne Pseudonymisierung bzw. mit Pseudonymisierung in unterschiedlichen Schlüsselstärken durch den Log-Proxy verändert wurden. Die gemessenen Zeiten sind Abbildung 5.3 zu entnehmen.

Abbildung 5.4 zeigt zusätzlich die Zeitanteile, die im für die Pseudonymisierung zuständigen Plugin des Log-Proxy gemessen wurden, wie sie die jeweiligen Teilfunktionen bei den unterschiedlichen Schlüssellängen benötigten:

- **MAC** – Zeitanteil für die Berechnung des MACs, der für die Suche nach existierenden Pseudonymen genutzt wird,
- **Schwellwert** – Zeitanteil für die Verschlüsselung durch das kryptographische Schwellwertschema,
- **Pseudonym** – Zeitanteil für die Kommunikation mit dem Pseudonym-Service und die dort stattfindende Suche nach existierenden Pseudonymen.

Aus diesen Messungen ergibt sich, dass der größte Anteil an Bearbeitungszeit einer Lognachricht in der Abfrage des Pseudonym-Service nach einem existierenden Pseudonym und der Kommunikation zwischen dem Log-Proxy und dem Pseudonym-Service liegt.

6. Wegen der langen Messzeiten musste die Anzahl von Messungen sehr beschränkt werden.

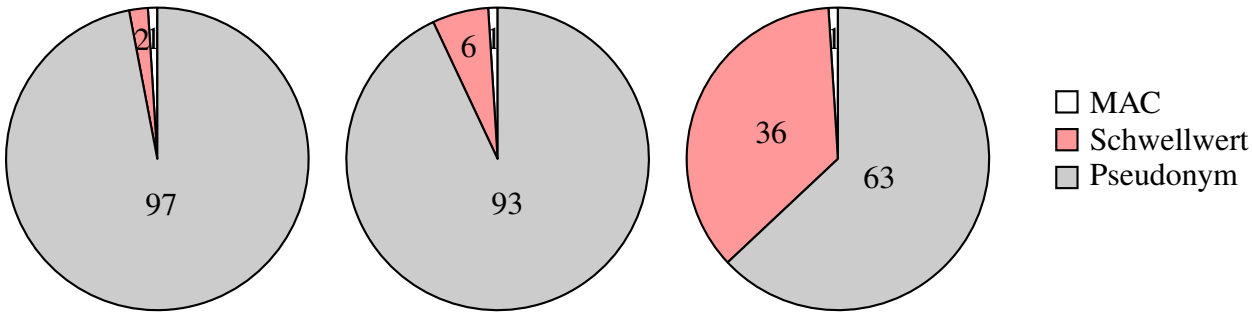

Abbildung 5.4: Zeitlicher Anteil der verschiedenen Berechnungen im Pseudonymisierungs-Plugin bei Schlüssellängen von 512 Bit (links), 1024 Bit (Mitte) und 2048 Bit (rechts).

Sollte sich die benötigte Zeit für die Bearbeitung von Logdaten im produktiven Einsatz als zu lang herausstellen, könnten die Komponenten zusammengeführt werden – unter Verzicht auf die in Abschnitt 3.2.2 beschriebenen Vorteile bezogen auf die Sicherheit des Pseudonymzusammenhangs durch ein verteiltes System.
Mit steigenden Schlüssellängen steigt jedoch auch der Zeitanteil, den die Berechnungen des kryptographischen Schwellwertschemas benötigen, erheblich. Die Veränderung ist hier alleinig auf den asymmetrischen Teil des umgesetzten hybriden Schemas (siehe Abschnitt 5.3.1) zurückzuführen, da sich der symmetrische Teil nicht verändert. Um die Performanz der Verschlüsselungsfunktion zu erhöhen, bietet sich die Nutzung von *Elliptic Curve Cryptography* an (siehe Abschnitt 4.3.4). Eine zusätzliche Möglichkeit ist auch die Implementierung dieser zeitkritischen Funktion durch eine maschinennähere Sprache als Python (wie beispielsweise C).

Für die Berechnung der partiellen Entschlüsselungen und der kompletten Entschlüsselung eines Schlüsseltextes (und damit für die Aufdeckung eines Pseudonyms) wurde auf Performanzmessungen verzichtet. Diese Aktionen sind im umgesetzten System nicht zeitkritisch. Zusätzlich werden sie nur in seltenen Fällen – nämlich bei der Aufdeckung eines Pseudonyms – eingesetzt.

Insgesamt kam es bei den Messungen zu einigen Problemen, insbesondere wenn die Nachrichtenlast und damit die Rechenlast stieg:

- seltene Komplettausfälle des Systems aufgrund des Datenbanktreibers für die verwendete PostgreSQL-Datenbank,
- Nicht-Behandlung von Nachrichten durch das unzuverlässige UDP-Protokoll.

Bessere (Mess-)Ergebnisse sollte das Deployment der Komponenten auf dafür ausgelegter Infrastruktur mit Standardlösungen für Python (Apache mit mod_wsgi, uWSGI, ...) und entsprechender Rechenleistung bringen. Ebenso sollte die Nutzung von TCP-Verbindungen für das verwendete syslog-Protokoll mehr Zuverlässigkeit gewährleisten.[7]

7. RFC 3195 (Reliable Delivery for syslog) beschreibt diese Erweiterung. Ob dies in der Praxis durch verbreitete Logdaten produzierende Geräte genutzt wird, bedarf allerdings weiterer Recherche.

6 Ergänzende und alternative Datenschutztechniken

Der in dieser Arbeit verfolgte Ansatz der Pseudonymisierung unter Einsatz kryptographischer Schwellwertschemata ist nicht für alle Arten von Logdaten sinnvoll. Sollen beispielsweise Zeitstempel verändert werden, um keine direkten Rückschlüsse auf eine Person durch Kombination mit typischen Verhaltensmustern zu ermöglichen, auf der anderen Seite jedoch zumindest grobe Erkenntnisse aus dem Zeitstempel für die Anomalieerkennung genutzt werden, so kann eine auf Pseudonymisierung basierende Lösung dies nicht leisten.

Daher werden in diesem Abschnitt ergänzende Techniken zum Erhalt der Privatspäre eines Arbeitnehmers bei der Speicherung von Logdaten dargestellt und erläutert, wie diese in den entwickelten Prototyp eingebunden werden können. Natürlich können auch mehrere kombinierte Techniken für einzelne Logdaten, die aus mehreren Feldern bestehen, sinnvoll sein. Mit Zeitstempeln versehene Logdaten eines Türschließsystems, die die Benutzerkennungen der Mitarbeiter enthalten, könnten so beispielsweise durch Pseudonymisierung der Benutzerkennungen und Verrauschung der Zeitstempel geschützt werden.

Einen Ansatz für ein Framework zur stufenweisen Anonymisierung von Logdaten, in dem viele der im Folgenden beschriebenen Techniken Anwendung finden, beschreiben die Autoren in [SLL06]. In der Veröffentlichung sind insbesondere Beispiele für die Behandlung bestimmter Datentypen gegeben, die über den Rahmen dieses Kapitel hinausgehen.
Eine Übersicht über weitere in sehr speziellen Einsatzgebieten zu nutzende Datenschutztechniken wie Bloomfilter oder der Einsatz homomorpher Verschlüsselung sind in [NKS17] zu finden.

6.1 Unterdrückung

Als ergänzende Maßnahme für Datenfelder, die für die Anomalieerkennung nicht benötigt werden, aber Rückschlüsse auf den Benutzer zulassen, kommt die Unterdrückung in Frage. Hierbei wird der Wert des Feldes schlicht entfernt oder durch eine Konstante ersetzt.

Ein Beispiel im Kontext dieser Arbeit ist die Unterdrückung von IP-Adressen eines durch einen Arbeitnehmer benutzten Rechners für Logdaten, in denen auch der Benutzername enthalten ist. Beim Einsatz von Pseudonymisierung könnte die Zuordnung von Benutzer zu Pseudonym erleichtert werden, wenn der Arbeitnehmer dem physischen Gerät zu einem Zeitpunkt zugeordnet werden kann und die IP-Adresse des Geräts noch in den Logdaten enthalten ist.

6.2 Generalisierung

Bei der Generalisierung wird der Feldinhalt durch einen Wert ersetzt, der das gleiche Konzept beschreibt, jedoch allgemeiner ist. Durch mehrfache Verallgemeinerung entstehen sogenannte Generalisierungshierarchien, wobei die Stufe der höchsten Generalisierung hier gleichbedeutend

mit der Unterdrückung ist, da jeder Wert durch den konstanten Wert der höchsten Generalisierungsstufe ersetzt wird.

Ein Beispiel im Unternehmenskontext dieser Arbeit ist die Generalisierung eines Mitarbeiters zu seiner Arbeitsgruppe oder Abteilung – eine Information, die für die Anomalieerkennung ausreichend sein könnte, wenn es beispielsweise um Zugriffe auf Ressourcen geht, die für bestimmte Abteilungen üblich, für andere jedoch ungewöhnlich sind. Dieses könnte zusätzlich zur Pseudonymisierung ausgeführt werden, um der Anomalieerkennung zusätzliche Daten zur Verfügung zu stellen, ohne die Identität eines Nutzers direkt offenzulegen.

6.3 Verrauschen

Diese Maßnahme verändert den Wert eines Datenfeldes, indem diesem Feld Werte aus einer Wahrscheinlichkeitsverteilung hinzugefügt werden (statistisches Rauschen). Hierdurch lassen sich die Rückschlüsse auf einen Nutzer aus einem einzelnen Datensatz verringern, aber die Gesamtverteilung bleibt erhalten bzw. lässt sich leicht berechnen. So können zumindest Abweichungen von Durchschnittswerten zur Anomalieerkennung genutzt werden. Es wird jedoch eine ausreichend große Datenmenge benötigt. Alternativ lassen sich zumindest Aussagen über den Bereich treffen, in dem ein Wert sich befinden muss. Dies könnte beispielsweise beim Verrauschen von Ereigniszeitstempeln sinnvoll sein, bei dem zwar nicht auf den konkreten Zeitpunkt geschlossen werden kann, aber zumindest Aussagen darüber getroffen werden können, ob das Ereignis in einem üblichen Intervall, wie in den normalen Bürostunden, auftrat.

Die Maßnahme ist jedoch nur für bestimmte Felder bzw. Datenarten sinnvoll einsatzbar. Gegenbeispiele sind unter anderem Freitextfelder, wie Benutzernamen, oder Felder für Aufzählungstypen, wie Raumnummern, die sich mit Rauschen nicht sinnvoll verändern lassen.

6.4 Nutzung von Hashverfahren

Neben der zufälligen Generierung von Pseudonymen, wie es in dieser Arbeit genutzt wird, ist auch die Nutzung von Hashwerten als Pseudonym für Daten denkbar. Dies würde die Verknüpfbarkeit von Logdaten ermöglichen, da für gleiche Daten der gleiche Hashwert berechnet wird. Durch den geschickten Einsatz von zusätzlichen zeitabhängig wechselnden Eingaben für das Hashverfahren (sogenannte *Salts*) ließe sich auch die nötige Verknüpfbarkeit für die Anomalieerkennung gegenüber dem Schutz der Privatsphäre eines Benutzers abstimmen.
Auf der anderen Seite wäre der Einsatz von Hashverfahren bei einem kleinen Wertebereich für Eingaben wie Benutzernamen anfällig für Wörterbuchangriffe (vgl. Abschnitt 4.4). Außerdem wären auch Rückschlüsse auf den Pseudonymhalter nicht ohne zusätzlichen Aufwand möglich.

Für Datenfelder, bei denen nur die Verknüpfbarkeit, jedoch nicht der ursprüngliche Wert, für die Anomalieerkennung entscheidend ist und deren Wertebereich ausreichend groß ist, können Hashverfahren sinnvoll sein.

6.5 Vorgehen zur Integration

Die Integration der vorgestellten Datenschutztechniken in den entwickelten Prototyp stellt kein Problem dar. Die jeweiligen Techniken können, wie in Abschnitt 5.1.2 beschrieben, als Plugins entwickelt werden. Hierbei entsteht je nach Datenschutztechnik unterschiedlicher Aufwand.

Für die Generalisierung und das Verrauschen müssen beispielsweise eingabedatenabhängige Plugins entstehen. Die Generalisierung von Zeitstempeln (Generalisierung auf Minute, Stunde, ... oder selbst gewählte Zeitabschnitte) unterscheidet sich beispielsweise stark von der Generalisierung der Umgebung eines Mitarbeiters (unternehmensspezifische Generalisierung auf Arbeitsgruppe, Abteilung, ...). Die Unterdrückung oder Nutzung von Hashverfahren kann hingegen unabhängig von den Eingaben entwickelt werden.

Für Datenquellen können diese Plugins nun einzeln oder in Kombination, wie in Abschnitt 5.1.1 beschrieben, genutzt werden. Durch diesen Ansatz lässt sich für jede Datenquelle abhängig von verwendeten Anomalieerkennungsverfahren eine gute Abwägung zwischen Nutzbarkeit der Daten und Schutz der Privatsphäre eines Arbeitnehmers schaffen.

7 Fazit

In dieser Arbeit wurde das Zusammenspiel von Pseudonymisierung und kryptographischen Schwellwertschemata als Lösung für die datenschutzgerechte Speicherung von Logdaten untersucht. Als zentrale Anforderung an die Pseudonymisierung wurde die für unterschiedliche Einsatzkontexte per Parameter anpassbare Pseudonymwechselstrategie ausgemacht. Generell einsetzbare Parameter stellen die Nutzungshäufigkeit und das Zeitintervall der maximalen Nutzungsdauer dar. Es wurde das Feld der kryptographischen Schwellwertschemata untersucht und ein Schema auf Basis von Shamirs Secret Sharing und dem ElGamal-Verfahren ausgewählt sowie Verbesserungen in Form von *Elliptic Curve Cryptography*, dezentraler Schlüsselgenerierung und komplexen Zugriffsstrukturen dargestellt. Das Problem der Identifikation bestehender Pseudonyme für verschlüsselte Daten wurde als zentrales Problem bei der Kombination beider Verfahren ausgemacht und verschiedene Lösungsansätze diskutiert. Ein Ansatz basierend auf der Nutzung von MACs als einfache Form deterministischer Verschlüsselung stellte sich für das entwickelte System als am geeignetsten heraus. Zusätzlich wurden verschiedene Ansätze für den Eingriff in den Logdatenfluss zwischen Quelle und SIEM-System betrachtet und ein Proxy-basierter Ansatz als sinnvolle Balance von frühestmöglicher Pseudonymisierung und praktischer Einsetzbarkeit ausgemacht. Aus den beschriebenen Verfahren wurde ein verteiltes System entworfen, das in den Logdatenfluss eingreift, um Logdaten zu pseudonymisieren. Anschließend gibt es berechtigten Benutzern im Fall des Verdachts auf einen Insider-Angriff die Möglichkeit, die Aufdeckung von Pseudonymen zu beantragen bzw. über einen solchen Antrag zu entscheiden.

Basierend auf diesen theoretischen Betrachtungen wurde ein Prototyp des entworfenen Systems implementiert, der Logdaten über das Syslog-Protokoll entgegennimmt. Neben der Entwicklung der verschiedenen Komponenten des verteilten Systems erwies sich inbesondere die Entwicklung des kryptographischen Schwellwertschemas als zentrale Herausforderung. Entgegen den Erwartungen vor Erstellung der Arbeit scheint es keine quelloffene und überprüfte Bibliothek zu geben, die die benötigten Funktionen bereitstellt – eine Lücke, die die in dieser Arbeit entwickelte Bibliothek füllen kann. Dies erfordert jedoch eine ausgiebige Überprüfung durch erfahrene Kryptographen.

In einem weiteren Kapitel wurden ergänzende Datenschutztechniken zu dem verfolgten Ansatz dargestellt, da die Pseudonymisierung nicht für alle Arten von Daten sinnvoll einsatzbar ist. Zusätzlich wurde ihre Einbindung in den entwickelten Prototypen dargestellt.

Der in dieser Arbeit gewählte Ansatz zur datenschutzfreundlichen Speicherung von Logdaten ist gut geeignet, um (eingeschränkte) Verknüpfbarkeit dieser Daten zur Anomalieerkennung und durch das Mehraugenprinzip geschützte Aufdeckbarkeit eines möglichen Innentäters im Verdachtsfall zu ermöglichen. Im Gegensatz zu bisherigen Lösungen, die oftmals eine vertrauensvolle Partei für die Generierung und Aufdeckung von Pseudonymen voraussetzen, kann durch diesen Ansatz verteiltes Vertrauen mathematisch-technisch modelliert werden. Durch die parameterabhängige Erstellung von Pseudonymen kann abhängig von unterschiedlichen Anforderungen des Einsatzkontexts gut zwischen den Anforderungen der Anomalieerkennung und dem Recht auf informationelle Selbstbestimmung des Arbeitnehmers abgewogen werden.

Ob dies auch den rechtlichen Anforderungen des BDSG genügt, bleibt allerdings noch zu beurteilen.

Der entwickelte Prototyp zeigt auch die praktische Einsetzbarkeit des Verfahrens. Wie in dem Evaluationsabschnitt bereits beschrieben, sind vor dem Produktiveinsatz des Systems gerade im Hinblick auf die Performanz noch einige Optimierungen vorzunehmen.

Insgesamt kann der Ansatz dieser Arbeit die Speicherung der zur Anomalie-basierten Erkennung von Insider-Angriffen erforderlichen Überwachungsdaten und die Privatsphäre eines Arbeitnehmers zusammenbringen. Verfahren der Anomalieerkennung können relativ unabhängig von dieser Datenspeicherung entwickelt werden. Lediglich die Wahl der Parameter für die Pseudonymisierung und damit die Verknüpfbarkeit verschiedener Logdaten muss für einzelne Verfahren getroffen werden. Damit ist ein Schritt zur datenschutzgerechten Erkennung und Verhinderung von Insider-Angriffen getan, auf dem kommende Arbeiten aufbauen können.

Hierfür bietet das Themenfeld ausgehend von dem erreichten Stand dieser Arbeit noch einiges Potential für aufbauende Entwicklungs- und Forschungsarbeit. Der entwickelte Prototyp lässt sich noch an einigen Stellen optimieren:

- Implementierung der verteilten Schlüsselgenerierung (siehe Abschnitt 4.3.3),
- Überprüfung der entwickelten Bibliothek für die Nutzung kryptographischer Schwellwertschemata durch erfahrene Kryptographen,
- Performanzsteigerung durch die Nutzung von elliptischen Kurven (siehe Abschnitt 4.3.4),
- Ermöglichung von komplexen Zugriffsstrukturen, die über die 1:1-Zuweisung von Shares an Beteiligte hinausgehen (siehe Abschnitt 4.3.5).

Neben diesen Implementierungsarbeiten bieten sich jedoch auch einige Fragestellungen für weiterführende Forschungen an:

- Welche Auswirkungen hat die Wahl der Parameter für Nutzungshäufigkeit und zeitliche Begrenzung der verwendeten Pseudonyme für bestehende oder zu entwickelnde Anomalieerkennungsverfahren? Wie lässt sich also am besten zwischen dem Schutz der Privatsphäre eines Arbeitnehmers und erfolgreicher Anomalieerkennung vermitteln?
- Gibt es weitere (eventuell kontextabhängige) Parameter, die für die Pseudonymisierung genutzt werden sollten?
- Wie könnte ein System umgesetzt werden, das verteilte Datenverarbeitung – also beispielsweise auch die Pseudonymisierung direkt in der Datenquelle – ermöglicht? Insbesondere das Suchproblem nach bereits verwendeten Pseudonymen aus Abschnitt 4.4 muss hier betrachtet werden.

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).

[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. Bd. 3680. Lecture Notes in Computer Science. Springer, 2006, S. 226–243.

[BBO07] Mihir Bellare, Alexandra Boldyreva und Adam O'Neill. *Deterministic and efficiently searchable encryption*. In: Hrsg. von Alfred Menezes. Bd. 4622. Lecture Notes in Computer Science. Springer, 2007, S. 535–552.

[BF00] Joachim Biskup und Ulrich Flegel. *Threshold-based identity recovery for privacy enhanced applications*. In: *Proceedings of the 7th ACM Conference on Computer and Communications Security*. Hrsg. von Pierangela Samarati. ACM, 2000, S. 71–79.

[BF01] Joachim Biskup und Ulrich Flegel. *On pseudonymization of audit data for intrusion detection*. In: *Designing Privacy Enhancing Technologies*. Hrsg. von Hannes Federrath. Bd. 2009. Lecture Notes in Computer Science. 2001, S. 161–180.

[BFP14] Ulrike Baumann, Elke Franz und Andreas Pfitzmann. *Kryptographische Systeme*. Springer, 2014.

[Bit16] Bitkom. *Spezialstudie Wirtschaftsschutz*. 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: *Advances in Cryptology - ASIACRYPT 2000*. Hrsg. von Tatsuaki Okamoto. Bd. 1976. Lecture Notes in Computer Science. Springer, 2000, S. 30–43.

[BK99] Roland Büschkes und Dogan Kesdogan. *Privacy enhanced intrusion detection*. In: *Proceedings of the Conference on Multilateral Security in Communications*. Hrsg. von Kai Rannenberg und Günter Müller. Addison-Wesley, 1999, S. 187–207.

[Bla79] George Robert Blakley. *Safeguarding cryptographic keys*. In: *Proceedings of the AFIPS 1979 National Computer Conference*. Hrsg. von Richar E. Merwin. Bd. 48. AFIPS Press, 1979, S. 313–317.

[Bon+04] Dan Boneh u. a. *Public key encryption with keyword search*. In: *Advances in Cryptology - EUROCRYPT 2004*. Hrsg. von Christian Cachin und Jan L. Camenisch. Bd. 3027. Lecture Notes in Computer Science. Springer, 2004, S. 506–522.

[Bra+15] Nicholas Bradley u. a. *IBM 2015 Cyber Security Intelligence Index*. 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*. 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: *Applied Cryptography and Network Security*. Hrsg. von John Ioannidis, Angelos Keromytis und Moti Yung. Bd. 3531. Lecture Notes in Computer Science. Springer, 2005, S. 442–455.

[Des87] Yvo Desmedt. *Society and Group Oriented Cryptography: a New Concept*. In: *Advances in Cryptology - CRYPTO '87*. Hrsg. von Carl Pomerance. Bd. 293. Lecture Notes in Computer Science. Springer, 1987, S. 120–127.

[Des93] Yvo Desmedt. *Threshold cryptosystems*. In: *Advances in Cryptology - AUSCRYPT '92*. Hrsg. von Jennifer Seberry und Yuliang Zheng. Bd. 718. Lecture Notes in Computer Science. Springer, 1993, S. 1–14.

[Des97] Yvo Desmedt. *Some recent research aspects of threshold cryptography*. In: *Information Security*. Hrsg. von Eiji Okamoto, George Davida und Masahiro Mambo. Bd. 1396. Lecture Notes in Computer Science. 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: *Proceedings of the 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*. Hrsg. von Gilles Brassard. Bd. 435. Lecture Notes in Computer Science. 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*. Hrsg. von Kwangjo Kim. Bd. 1992. Lecture Notes in Computer Science. 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: *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: *Privacy Enhancing Technologies*. Hrsg. von George Danezis und David Martin. Bd. 3856. Lecture Notes in Computer Science. 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*. Hrsg. von Yair Frankel. Bd. 1962. Lecture Notes in Computer Science. Springer, 2000, S. 90–104.

[Fra+97] Yair Frankel u. a. *Proactive RSA*. In: *Advances in Cryptology - CRYPTO '97*. Hrsg. von Burton S. Kaliski. Bd. 1294. Lecture Notes in Computer Science. Springer, 1997, S. 440–454.

[Gem97] Peter Gemmell. *An introduction to threshold cryptography*. In: *CryptoBytes* 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*. Hrsg. von Neil Koblitz. Bd. 1109. Lecture Notes in Computer Science. Springer, 1996, S. 157–172.

[Gen+96b] Rosario Gennaro u. a. *Robust threshold DSS signatures*. In: *Advances in Cryptology - EUROCRYPT '96*. Hrsg. von Ueli Maurer. Bd. 1070. Lecture Notes in Computer Science. Springer, 1996, S. 354–371.

[Gen+99] Rosario Gennaro u. a. *Secure distributed key generation for discrete-log based cryptosystems*. In: *Advances in Cryptology - EUROCRYPT '99*. Hrsg. von Jaques Stern. Bd. 1592. Lecture Notes in Computer Science. Springer, 1999, S. 295–310.

[Ger09] Rainer Gerhards. *The syslog protocol*. RFC 5424. 2009.

[Goh04] Eu-Jin Goh. *Secure indexes*. 2004. URL: https://crypto.stanford.edu/~eujin/papers/secureindex/secureindex.pdf (besucht am 23. 11. 2017).

[Inf18] Bundesamt für Sicherheit in der Informationstechnik. *Kryptographische Verfahren: Empfehlungen und Schlüssellängen (BSI TR-02102-1)*. 2018. URL: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR02102/BSI-TR-02102.pdf (besucht am 06. 03. 2018).

[ISN89] Mitsuru Ito, Akira Saito und Takao Nishizeki. *Secret sharing scheme realizing general access structure*. In: *Electronics and Communications in Japan (Part III: Fundamental Electronic Science)* 72.9 (1989), S. 56–64.

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

[Kob87] Neal Koblitz. *Elliptic curve cryptosystems*. In: *Mathematics of computation* 48.177 (1987), S. 203–209.

[LJ00] Emilie Lundin und Erland Jonsson. *Anomaly-based intrusion detection: privacy concerns and other problems*. In: *Computer networks* 34.4 (2000), S. 623–640.

[LJ99] Emilie Lundin und Erland Jonsson. *Privacy vs. Intrusion Detection Analysis*. 1999. URL: http://www.raid-symposium.org/raid99/PAPERS/Lundin.pdf (besucht am 30. 01. 2018).

[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).

[Nik+13] Salman Niksefat u. a. *ZIDS: a privacy-preserving intrusion detection system using secure two-party computation protocols*. In: *The Computer Journal* 57.4 (2013), S. 494–509.

[NK11] Mark Nicolett und Kelly M Kavanagh. *Magic quadrant for security information and event management*. In: (2011). URL: https://www.gartner.com/doc/1679814/magic-quadrant-security-information-event (besucht am 04. 10. 2017).

[NKS17] Salman Niksefat, Parisa Kaghazgaran und Babak Sadeghiyan. *Privacy issues in intrusion detection systems: A taxonomy, survey and future directions*. In: *Computer Science Review* 25 (2017), S. 69–78.

[Par+07] Hyun-A Park u. a. *PPIDS: privacy preserving intrusion detection system*. In: *Intelligence and Security Informatics*. Hrsg. von Christopher C. Yang u. a. Bd. 4430. Lecture Notes in Computer Science. Springer, 2007, S. 269–274.

[Ped91] Torben Pedersen. *A threshold cryptosystem without a trusted party*. In: *Advances in Cryptology - EUROCRYPT '91*. Hrsg. von Donald W. Davies. Bd. 547. Lecture Notes in Computer Science. Springer, 1991, S. 522–526.

[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. Bd. 2009. Lecture Notes in Computer Science. 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*. Hrsg. von Hugo Krawczyk. Bd. 1462. Lecture Notes in Computer Science. Springer, 1998, S. 89–104.

[Sch06] Bruce Schneier. *Angewandte Kryptographie - Der Klassiker. Protokolle, Algorithmen und Sourcecode in C*. Pearson Studium, 2006.

[SFHR97] Michael Sobirey, Simone Fischer-Hübner und Kai Rannenberg. *Pseudonymous audit for privacy enhanced intrusion detection*. In: *Information Security in Research and Business*. IFIP — The International Federation for Information Processing. Springer, 1997, S. 151–163.

[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*. Hrsg. von Bar Preneel. Bd. 1807. Lecture Notes in Computer Science. Springer, 2000, S. 207–220.

[SHS08] Malek Ben Salem, Shlomo Hershkop und Salvatore J Stolfo. *A survey of insider attack detection research*. In: *Insider Attack and Cyber Security*. Hrsg. von Salvatore J. Stolfo u. a. Bd. 39. Advances in Information Security. Springer, 2008, S. 69–90.

[SLL06] Adam J Slagell, Kiran Lakkaraju und Katherine Luo. *FLAIM: A Multi-level Anonymization Framework for Computer and Network Logs*. In: *Proceedings of the 20th Large Installation System Administration Conference*. 2006.

[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: *Information Security and Privacy*. Hrsg. von Vijay Varadharajan und Yi Mu. Bd. 2119. Lecture Notes in Computer Science. 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: *Proceedings of the 2000 IEEE Symposium on Security and Privacy*. 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.

Eidesstattliche Versicherung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit im Masterstudiengang Informatik selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel – insbesondere keine im Quellenverzeichnis nicht benannten Internet-Quellen – benutzt habe. Alle Stellen, die wörtlich oder sinngemäß aus Veröffentlichungen entnommen wurden, sind als solche kenntlich gemacht. Ich versichere weiterhin, dass ich die Arbeit vorher nicht in einem anderen Prüfungsverfahren eingereicht habe und die eingereichte schriftliche Fassung der auf dem elektronischen Speichermedium entspricht.

Ich stimme der Einstellung der Arbeit in die Bibliothek des Fachbereichs Informatik zu.

Hamburg, der 22. März 2018

Tom Petersen