Schlüsselverwaltung für sichere Gruppeninteraktionen über beliebigen Speicheranbietern: ein Überblick

Holger Kühner und Hannes Hartenstein
Karlsruher Institut für Technologie (KIT), Steinbuch Centre for Computing (SCC)
{holger.kuehner | hartenstein}@kit.edu

Abstract: Ein Kernbestandteil vieler Web-Anwendungen ist die Interaktion innerhalb einer geschlossenen Benutzergruppe mit Hilfe eines Speicherdienstes, der nicht von der Benutzergruppe selbst betrieben wird. Eine unverschlüsselte Datenablage bei einem solchen Speicherdienst birgt das Risiko der Kompromittierung der Daten durch den Speicheranbieter selbst oder durch einen Angriff auf den Speicheranbieter. Ansätze für eine verschlüsselte Ablage der Daten erfordern die Verwaltung kryptographischer Schlüssel für die Gruppe mit Hilfe von Group-Key-Management-Protokollen (GKM-Protokollen). Das eingesetzte GKM-Protokoll hat starken Einfluss auf den Ressourcenbedarf und damit auf die Einsetzbarkeit der gesamten Anwendung. Die vorliegende Arbeit ermöglicht eine Vorauswahl passender GKM-Protokolle, indem sie GKM-Protokolle aus verschiedenen Anwendungsgebieten wie Digital Rights Management oder IP-Multicast bezüglich des Ressourcenbedarfs klassifiziert. Anhand exemplarischer Anwendungsszenarien wird demonstriert, wie die vorgestellte Klassifikation zur Vorauswahl von GKM-Protokollen eingesetzt werden kann.

1 Einleitung

Eine grundlegende Charakteristik heute genutzter Web-Anwendungen ist die Interaktion und der damit verbundene Austausch von Daten zwischen verschiedenen Benutzern. Für den Erfolg vieler Anwendungen ist hierbei entscheidend, dass der Empfängerkreis der Daten auf eine geschlossene Benutzergruppe begrenzt werden kann. Zu diesen Anwendungen zählen unter anderem Online Social Networks wie Facebook und LinkedIn, die die Möglichkeit bieten, bestimmte Profilinformationen und Nachrichten nur ausgewählten Benutzern zugänglich zu machen. Weitere Beispiele sind dateibasierte Speicherdienste unterschiedlicher Art: Teilweise sind diese Dienste auf bestimmte Medientypen ausgerichtet, wie Flickr oder Picasa, die sich auf Bilddateien fokussieren und entsprechende Funktionalität anbieten, teilweise bieten diese Dienste auch eine allgemeinere Unterstützung für Dateien aller Art wie beispielsweise Dropbox.

Die Daten liegen bei all diesen Anwendungen üblicherweise auf Systemen, die nicht von den Mitgliedern der zugriffsberechtigten Benutzergruppe selbst betrieben werden. Dem Betreiber des Speicherdienstes ist es somit technisch möglich, die abgelegten Daten einzusehen, zu verändern oder an Benutzer außerhalb der Gruppe weiterzugeben. Das Risiko eines Missbrauchs der Daten kann potentielle Benutzer davon abhalten, sensible Daten beim Speicheranbieter abzulegen. Auf der anderen Seite stellen die unverschlüsselten Da-

ten unter Umständen ein interessantes Ziel für Angriffe von außen dar, so dass der Speicheranbieter kostspielige Schutzmaßnahmen ergreifen muss.

Abhilfe kann ein System schaffen, das auf technischem Weg sicherstellt, dass der Speicheranbieter keinen Zugriff auf die Inhalte der abgelegten Daten erhält. Wir bezeichnen die Nutzung eines solchen Systems im Folgenden als *Sicheres Teilen von Daten* und spezifizieren in Abschnitt 2 die Anforderungen an ein solches System genauer. Eine naheliegende Umsetzungsmöglichkeit für das *Sichere Teilen von Daten* besteht darin, die Daten vor der Ablage beim Speicheranbieter zu verschlüsseln und dafür zu sorgen, dass alle legitimen Konsumenten der Daten die für die Entschlüsselung notwendigen kryptographischen Schlüssel erhalten. Die Erzeugung und Verteilung dieser Schlüssel wird als *Group Key Management (GKM)* bezeichnet. Es existieren zahlreiche Verfahren, die mit Hilfe von GKM das *Sichere Teilen von Daten* in verschiedenen Anwendungsgebieten ermöglichen, und gerade im Kontext der anhaltenden Diskussion zu Sicherheit im Cloud Computing werden solche Verfahren in einigen Forschungsprojekten beleuchtet.

Der Ressourcenbedarf von Ansätzen zum *Sicheren Teilen von Daten*, also zum Beispiel die benötigte Kommunikationsbandbreite, Rechenleistung und Speicherkapazität, werden durch das eingesetzte GKM-Protokoll unter Umständen stark beeinflusst: Wird ein Datum beispielsweise mit den öffentlichen Schlüsseln aller Gruppenmitglieder verschlüsselt, so muss bei Ausscheiden eines Gruppenmitglieds das Datum mit den öffentlichen Schlüsseln aller verbleibenden Gruppenmitglieder neu verschlüsselt werden, was bei entsprechender Größe der Gruppe hohen Rechenaufwand erfordert. Damit ist auch die Einsetzbarkeit des gesamten Ansatzes von dem eingesetzten GKM-Protokoll abhängig.

Die Auswahl eines GKM-Protokolls, das aus Sicht des Ressourcenbedarfs für eine bestimmte Anwendung geeignet ist, gestaltet sich schwierig. In vielen Fällen ist der Ressourcenbedarf eines GKM-Protokolls zwar insofern bekannt, als dass asymptotische Abschätzungen bezüglich der Gruppengröße, der Häufigkeit an Gruppenbei- und -austritten oder der Häufigkeit von Schreibzugriffen existieren. Die uns bekannten Übersichten über GKM-Protokolle fokussieren sich allerdings entweder nur auf bestimmte Anwendungsgebiete wie DRM [Lea04] oder IP-Multicast [DMS99], oder die Protokolle werden in diesen Übersichten nicht aus dem Blickwinkel des Ressourcenbedarfs klassifiziert, sondern beispielsweise basierend auf ihrer Architektur [Cha05].

Die Beiträge der vorliegenden Arbeit umfassen:
i) die Spezifikation der Anforderungen, die ein System zum *Sicheren Teilen von Daten* unseres Erachtens erfüllen muss (Abschnitt 2).
ii) eine Klassifikation von GKM-Protokollen, die für das *Sichere Teilen von Daten* eingesetzt werden können, aus Sicht des asymptotischen Verhaltens ihres Ressourcenbedarfs. In diese Klassifikation werden GKM-Protokolle aus verschiedenen Anwendungsdomänen wie beispielsweise IP-Multicast, DRM und verschlüsselte Dateisysteme eingeordnet (Abschnitt 3).
iii) Beispiele, wie die eingeführte Klassifikation konkret genutzt werden kann, um für ein gegebenes Anwendungsszenario eine Vorauswahl von GKM-Protokollen zu treffen, die aus Sicht des Ressourcenbedarfs einsetzbar wären (Abschnitt 4).

2 Anforderungen an ein System zum *Sicheren Teilen von Daten*

Ein System, welches das *Sichere Teilen von Daten* ermöglicht, muss unseres Erachtens folgende Anforderungen erfüllen:
Zugriff auf Daten bei entferntem Speicheranbieter: Der Benutzer greift über ein Netzwerk auf Daten zu, die bei einem Speicheranbieter liegen. Der organisatorische und technische Aufbau des Speicheranbieters ist hierbei nicht von Belang. Der Speicheranbieter kann somit von einer oder mehreren Organisationen oder Privatpersonen betrieben werden und beispielsweise als Cloud-Dienst oder Peer-to-Peer-Netzwerk organisiert sein. Die organisatorische und technische Struktur des Speicheranbieters bleibt dem Benutzer unter Umständen komplett verborgen.
Teilen von Daten: Die beim Speicheranbieter abgelegten Daten können einer beliebigen Anzahl an Benutzern sowohl lesend als auch schreibend zugänglich gemacht werden. Die Gruppe von zugriffsberechtigten Benutzern kann jederzeit geändert werden. Die Autorisierung kann durch den einen bereits zugriffsberechtigten Benutzer, aber auch durch eine vertrauenswürdige Instanz, die selbst nicht zugriffsberechtigt ist, erfolgen.
Vertraulichkeit der Daten gegenüber nicht Zugriffsberechtigten und dem Speicheranbieter: Es wird angenommen, dass der Speicheranbieter nicht garantieren kann, dass die Daten vor Lesezugriff durch Angreifer geschützt sind. Die Angriffe können hierbei sowohl von innerhalb als auch von außerhalb der organisatorischen Grenzen des Speicheranbieters initiiert werden. Die Vertraulichkeit der beim Speicheranbieter abgelegten Daten gegenüber nicht Zugriffsberechtigten einschließlich dem Speicheranbieter selbst muss deshalb über kryptographische Mechanismen sichergestellt werden. Der lesende Zugriff auf ein Datum muss für ausgeschiedene Gruppenmitglieder spätestens dann unterbunden werden, wenn das Datum nach dem Ausscheiden geändert wurde.
Integrität der Daten: Es wird angenommen, dass der Speicheranbieter den schreibenden Zugriff auf die Daten ebenfalls nicht für jeden Angreifer unterbinden kann. Daher muss auch die Integrität der Daten über kryptographische Mechanismen gewährleistet werden.
Keine Datenverarbeitung seitens Speicheranbieter notwendig: Der Speicheranbieter bietet ausschließlich Dienste zur sicheren Ablage und Freigabe von Dateien an. Er nimmt keine Operationen auf den Inhalten der abgelegten Daten vor, weshalb die interne Struktur der Daten komplett vor ihm verborgen bleiben kann.
Unabhängigkeit von Benutzeraktionen: Die gewünschte Aktion eines Benutzers (beispielsweise das Lesen oder Schreiben einer Datei oder eine Änderung der Zugriffsberechtigungen) darf zeitlich nicht von einer Aktion eines anderen Benutzers abhängig sein. Diese Anforderung stellt sicher, dass Benutzer unabhängig voneinander zu beliebigen Zeiten mit dem Speicheranbieter interagieren können. Die Forderung nach zeitlicher Unabhängigkeit von Benutzeraktionen lässt folgende Ausnahmen zu:
i) Unumgänglich ist, dass das Lesen eines Datums erst nach dem Schreiben erfolgen kann.
ii) Der Entzug der Zugriffsrechte auf ein bestimmtes Datum kann unter Umständen erst möglich sein, wenn dieses Datum das nächste Mal geschrieben wird.
Die Forderung nach Unabhängigkeit von Benutzeraktionen impliziert unter anderem:
i) Der Zugriff auf Daten darf kein Quorum von zugriffsberechtigten Benutzern erfordern, die ihre Zustimmung unmittelbar vor dem Zugriff geben müssen.
ii) Da auch das Verbinden mit dem Netzwerk als Benutzeraktion gesehen werden kann,

darf ein Lösungsansatz nicht davon abhängig sein, dass ein Benutzer oder der Speicheranbieter vor einer gewünschten Aktion einen anderen Benutzer unmittelbar kontaktieren können. Daraus folgt auch: jeder Kommunikationsvorgang wird durch einen Benutzer initiiert und involviert den Speicheranbieter.

3 Klassifikation von Group-Key-Management-Protokollen

Der folgende Abschnitt gibt einen Überblick über Group-Key-Management-Protokolle (GKM-Protokolle), die grundsätzlich einsetzbar sind, um das *Sichere Teilen von Daten* wie in Abschnitt 2 spezifiziert zu ermöglichen.

Die Protokolle werden im weiteren Abschnitt anhand des asymptotischen Verhaltens ihres Ressourcenbedarfs klassifiziert. Ressourcen umfassen im Rahmen unserer Klassifikation Kommunikationsbandbreite, Rechenleistung und Speicherkapazität. Das asymptotische Verhalten des Ressourcenbedarfs wird bei wachsender Gruppengröße, Bei- und Austrittshäufigkeit der Gruppenmitglieder und Häufigkeit von Schreiboperationen betrachtet.

Entscheidend für den Einfluss der Häufigkeit von Bei- und Austritten beziehungsweise von Schreiboperationen ist, ob mit jedem Bei- und Austritt beziehungsweise mit jeder Schreiboperation eine Erneuerung des sogenannten *Gruppenschlüssels* verbunden ist. Aus diesem Grund klassifizieren wir die GKM-Protokolle zunächst anhand des Zeitpunkts der Schlüsselerneuerung. Der Gruppenschlüssel muss jedem Gruppenmitglied zu jeder Zeit bekannt sein, denn mit diesem Schlüssel werden die Nutzdaten verschlüsselt. Ändert sich die Zusammensetzung der Gruppe über die Zeit, so muss der Gruppenschlüssel erneuert werden: Einerseits muss verhindert werden, dass ein aus der Gruppe ausgeschiedener oder ausgeschlossener Benutzer Zugriff auf neue Daten erhält, andererseits soll je nach Anwendungsfall auch unterbunden werden, dass ein neues Gruppenmitglied Daten einsehen kann, die vor seiner Mitgliedschaft geschrieben wurden. Die Erneuerung des Gruppenschlüssels verursacht Aufwand, da ein Gruppenschlüssel erzeugt und über einen sicheren Kanal an die Gruppenmitglieder übertragen werden muss; die Häufigkeit der Schlüsselerneuerung wirkt sich demnach stark auf den Ressourcenbedarf des Protokolls aus.

Die Ver- und Entschlüsselung der Daten mit Hilfe des Gruppenschlüssels sowie die Übertragung und Ablage der verschlüsselten Daten haben ebenfalls großen Einfluss auf den Ressourcenbedarf des Gesamtsystems. Diese Vorgänge sind allerdings nicht Teil des GKM-Protokolls, daher wird deren Ressourcenbedarf im Weiteren nicht berücksichtigt.

Die im weiteren vorgestellten GKM-Protokolle verlangen, dass ein Gruppenmitglied oder ein vertrauenswürdiger Dritter als *Schlüsselverwalter* fungiert. Dieser kennt den öffentlichen Schlüssel jedes Benutzers oder hat mit dem Benutzer ein kryptographisches Geheimnis ausgetauscht, das im Weiteren als persönlicher Schlüssel bezeichnet wird. Der initiale Schlüsselaustausch ist üblicherweise nicht Teil des GKM-Protokolls. Wir beziehen den Ressourcenbedarf auf Seiten des Schlüsselverwalters in unsere Betrachtungen mit ein.

Der Einfluss der Gruppengröße auf den Ressourcenbedarf eines Protokolls wird wesentlich dadurch bestimmt, ob neben dem persönlichen Schlüssel jedes Benutzers und dem

	Keine Hilfsschlüssel für Teilgruppen	**Hilfsschlüssel für Teilgruppen**
Schlüsselerneuerung bei Bei-/Austritt	GKMP [HM97], SMKD [Bal96], Secure Locks [CC89], ACP [ZDB08], ACV [SNPB10]	LKH [WHA99], OFT [SM03], CFKM [Wea99], VersaKey [Wea99], IGKMP [Dea01]
Schlüsselerneuerung bei Schreibzugriff	Chu et al. [Cea02], PGP [Zim95], SNAD [Mea01], EFS [Cro02]	CPRM [Ent03], CS [Nea01], SD [Nea01]
	Benutzer: Speicherbedarf $O(1)$ Rechenzeit $O(S)$ Komm.bandbreite $O(S)$ *Schlüsselverwalter*: Speicherbedarf $O(N)$ Rechenzeit $O(S * N)$ Komm.bandbreite $O(S * N)$	*Benutzer*: Speicherbedarf $O(\log N)$ (für SD [Nea01]: $O(\log^2 N)$) Rechenzeit $O(S * \log N)$ Komm.bandbreite $O(S * \log N)$ *Schlüsselverwalter*: Speicherbedarf $O(N)$ Rechenzeit $O(S * \log N)$ Komm.bandbreite $O(S * \log N)$

Tabelle 1: Klassifikation ausgewählter GKM-Protokolle und asymptotischer Ressourcenbedarf pro Klasse. S bezeichnet die Anzahl an Schlüsselerneuerungen, N bezeichnet die Gruppengröße.

Gruppenschlüssel noch weitere Hilfsschlüssel zum Einsatz kommen. Diese Unterscheidung dient als zweites Kriterium für unsere Klassifikation. Jeder Hilfsschlüssel wird üblicherweise an einen Teil der Gruppenmitglieder weitergegeben, so dass der jeweilige Hilfsschlüssel für die Kommunikation mit dieser Teilgruppe verwendet werden kann. Anzahl und Aufteilung der Hilfsschlüssel variieren hierbei von Protokoll zu Protokoll.

Im Folgenden nicht betrachtet werden die sog. *Group-Key-Agreement*-Protokolle: Diese erfordern eine Mitwirkung aller Gruppenmitglieder, um einen neuen Gruppenschlüssel zu erzeugen, was der Forderung nach *Unabhängigkeit von Benutzeraktionen* widerspricht. Ebenso nicht betrachtet werden GKM-Protokolle, die eine Erneuerung des Schlüssels ausschließlich nach Ablauf eines bestimmten Zeitintervalles vornehmen: Diese Protokolle erlauben keinen sofortigen Ausschluss eines Benutzers aus der Gruppe, was jedoch entsprechend der Anforderung *Teilen von Daten* verlangt wird.

Einen Überblick über die vorgestellten GKM-Protokolle, deren Einteilung in Klassen und die jeweiligen asymptotischen Aufwände bietet Tabelle 1.

Schlüsselerneuerung bei Bei- oder Austritt ohne Hilfsschlüssel: Bei dieser Klasse von GKM-Protokollen kommen üblicherweise ein oder mehrere dedizierte Schlüsselverwalter zum Einsatz, die nicht zwingend Gruppenmitglieder sind. Der Ressourcenbedarf dieser Protokolle ist unabhängig von der Häufigkeit von Schreiboperationen. Im Allgemeinen kann der Schlüssel bei einem Beitritt zur Gruppe mit geringem Aufwand erneuert werden, da im einfachsten Fall der neue Gruppenschlüssel zum einen mit dem persönlichen Schlüssel des beitretenden Benutzers verschlüsselt und an diesen übermittelt, zum anderen mit dem alten Gruppenschlüssel verschlüsselt und in der restlichen Gruppe verteilt wird.

Zunächst werden GKM-Protokolle erläutert, bei denen jeder Benutzer initial nur einen per-

sönlichen Schlüssel, jedoch keine Hilfsschlüssel erhält. Dies bedeutet, dass der Speicherbedarf für die Schlüssel auf Nutzerseite konstant ist, jedoch ein vergleichsweise großer Rechenaufwand und Kommunikationsoverhead beim Schlüsselverwalter entsteht.

Zu den Protokollen ohne Verwendung von Hilfsschlüsseln zählt *GKMP* [HM97], bei dem bei einem Betritt der neue Gruppenschlüssel mit dem persönlichen Schlüssel des Beitretenden und einem in der Gruppe bekannten *Key Encryption Key (KEK)* verschlüsselt und dann an den Beitretenden beziehungsweise die bestehende Gruppe übertragen wird. Die Trennung in GKMP zwischen dem KEK und dem eigentlichen Gruppenschlüssel, mit dem die Nutzdaten verschlüsselt werden, verhindert, dass bei einer Kompromittierung des Gruppenschlüssels alle im Folgenden verteilten Gruppenschlüssel auch unmittelbar kompromittiert wären. Bei Austritt eines Gruppenmitglieds werden ein neuer Gruppenschlüssel und KEK erzeugt, mit dem persönlichen Schlüssel jedes verbleibenden Benutzers verschlüsselt und an diese übermittelt. Der Rechen- und Kommunikationsaufwand für den Schlüsselverwalter wächst hierdurch linear mit der Gruppengröße an.

SMKD [Bal96] ist eine dezentrale Variante von GKMP, bei der unterstützende Schlüsselverwalter definiert werden können, die dafür verantwortlich sind, den neuen Gruppenschlüssel an einen Teil der Gruppe weiterzugeben.

Ebenfalls ausschließlich mit persönlichen kryptographischen Geheimnissen arbeiten die als *broadcast secret schemes* bekannten Protokolle: Hier wird für die Schlüsselerneuerung ein vom Schlüsselverwalter generierter Wert - im Folgenden *Broadcast* genannt - an alle Gruppenmitglieder gesendet. Aus dem Broadcast kann jedes Gruppenmitglied mit Hilfe des persönlichen Geheimnisses den Gruppenschlüssel ableiten. Eines der ältesten broadcast secret schemes ist *Secure Locks* [CC89]. Secure Locks basiert auf dem Chinesischen Restsatz. Ein weiteres broadcast secret scheme, *Access Control Polynomial* [ZDB08], basiert auf Polynomen. Bei *Access Control Vector* [SNPB10] stellt der Broadcast einen Vektor dar, aus dem durch Bilden des Skalarproduktes mit einem nur dem Benutzer bekannten Vektor der Gruppenschlüssel errechnet wird.

Broadcast secret schemes bieten im Allgemeinen den Vorteil, dass Außenstehende aus dem Broadcast keine Schlüsse dahingehend ziehen können, welche Benutzer Mitglied der Gruppe sind. Sie verbessern den Ressourcenbedarf gegenüber GKMP oder SMKD jedoch nicht: Der Rechenaufwand für die Erzeugung des Broadcasts wie auch der Kommunikationsaufwand für die Übermittlung steigen linear mit der Größe der Gruppe an.

Schlüsselerneuerung bei Bei- oder Austritt mit Hilfsschlüsseln: Im Folgenden werden GKM-Protokolle vorgestellt, bei denen jeder Benutzer neben dem persönlichen Schlüssel und dem Gruppenschlüssel weitere Hilfsschlüssel erhält, die er sich mit anderen Gruppenmitgliedern teilt. Die durch einen Hilfsschlüssel aufgespannte Teilgruppe kann mit nur einer Verschlüsselungsoperation mit einem neuen Gruppenschlüssel versorgt werden, indem dieser mit dem Hilfsschlüssel verschlüsselt wird. Der Einsatz von Hilfsschlüsseln verringert folglich die Rechenlast und den Kommunikationsbedarf, die dem Schlüsselverwalter entstehen. Dem gegenüber steht ein erhöhter Ressourcenbedarf auf Benutzerseite, da der Benutzer nun zusätzlich die Hilfsschlüssel vorhalten muss. GKM-Protokolle, die diesem Ansatz folgen, werden in der Literatur als *subset cover schemes* bezeichnet.

Die Aufteilung der Hilfsschlüssel auf die Gruppenmitglieder wird in einigen GKM-Proto-

kollen durch eine Baumstruktur bestimmt, in der die Gruppenmitglieder beziehungsweise deren persönliche Schlüssel die Blätter darstellen, so zum Beispiel in *Logical Key Hierarchy (LKH)* [WHA99]. Der Gruppenschlüssel ist in LKH die Wurzel des Baumes, innere Knoten stellen Hilfsschlüssel dar. Jedes Gruppenmitglied erhält alle Hilfsschlüssel, die auf dem Pfad von dem ihm zugeordneten Blattknoten zur Wurzel liegen. Tritt ein Gruppenmitglied aus, wird der neue Gruppenschlüssel mit Hilfsschlüsseln verschlüsselt, die der ausgetretene Benutzer nicht kennt, und an die verbleibende Gruppe verteilt; die Zahl der einzusetzenden Hilfsschlüssel hängt logarithmisch von der Gruppengröße ab. Die Hilfsschlüssel, die der ausgetretene Benutzer kennt, werden anschließend erneuert.

In *One-Way Function Tree (OFT)* [SM03] werden die Hilfsschlüssel nicht vom Gruppenverwalter zufällig erzeugt, stattdessen ergibt sich der einem Knoten zugeordnete Hilfsschlüssel durch die Anwendung von Hashfunktionen auf die Schlüssel aller Kindknoten. Jeder Benutzer kann somit den aktuellen Gruppenschlüssel eigenständig berechnen, sofern ihm bestimmte Hilfsinformationen zur Verfügung stehen. Die Größe dieser Hilfsinformationen wächst logarithmisch mit der Gruppengröße.

IGKMP [Dea01] ordnet die Gruppenmitglieder ebenfalls in einer Baumstruktur an, die inneren Knoten dienen hier allerdings nicht mehr nur der Zuordnung von Hilfsschlüsseln, sondern sie stellen unterstützende Schlüsselverwalter dar, die mit den Gruppenmitgliedern in ihrem Teilbaum über den Hilfsschlüssel gesichert kommunizieren.

Wie Hilfsschlüssel ohne hierarchische Anordnung der Gruppenmitglieder eingesetzt werden können, zeigt *Centralized Flat Key Management (CFKM)* [Wea99]: In CFKM werden Hilfsschlüssel eingesetzt, deren Anzahl mit dem Logarithmus der Gruppengröße wächst. Jedes Gruppenmitglied erhält die Hälfte der Hilfsschlüssel, wobei sichergestellt ist, dass verschiedene Gruppenmitglieder unterschiedliche Mengen an Hilfsschlüsseln halten. Scheidet ein Mitglied aus der Gruppe aus, so wird der neue Gruppenschlüssel mit allen Hilfsschlüsseln verschlüsselt, die das ehemalige Gruppenmitglied nicht kennt. Anschließend werden die ihm bekannten Hilfsschlüssel mit Hilfe des neuen Gruppenschlüssels ersetzt. CFKM bildet einen Teil von *VersaKey* [Wea99], das als Framework konzipiert ist, welches einen Wechsel zwischen verschiedenen GKM-Protokollen erlaubt. VersaKey unterstützt neben einer zentralen und einer dezentralen Variante von CFKM ein hierarchische GKM-Protokoll. Der Ressourcenbedarf von CFKM und VersaKey wächst auf Benutzerseite logarithmisch mit der Größe der Gruppe.

Schlüsselerneuerung bei Schreibzugriff ohne Hilfsschlüssel: Im Rest dieses Abschnittes werden GKM-Protokolle erläutert, bei denen der Schlüssel bei jedem Schreibzugriff erneuert wird. Die Schlüsselerneuerung muss bei dieser Klasse von Protokollen vom Schreibenden initiiert werden. In der Regel ist bei dieser Klasse von Protokollen jeder Schreibende gleichzeitig Schlüsselverwalter, alternativ kann eine Kommunikation zwischen Schreibendem und dediziertem Schlüsselverwalter unmittelbar vor dem Sendevorgang stattfinden. Der Ressourcenbedarf ist bei dieser Klasse von GKM-Protokollen unabhängig von der Häufigkeit von Bei- und Austritten von Gruppenmitgliedern. Das zu schreibende Datum stellt je nach Anwendungsfall des GKM-Protokolls beispielsweise eine Datei, eine Nachricht oder einen inhaltlich abgeschlossenen Teil eines Datenstroms dar.

Zunächst werden GKM-Protokolle vorgestellt, die keine Hilfsschlüssel einsetzen. Der Res-

sourcenbedarf auf Benutzerseite ist bei diesen Protokollen unabhängig von der Gruppengröße, der Ressourcenbedarf auf Seite des Schlüsselverwalters wächst linear mit der Gruppengröße. Das von *Chu et al.* [Cea02] vorgestellte Protokoll erlaubt jedem Gruppenmitglied das Schreiben von Daten. Hierzu erzeugt der Schreibende einen zufälligen Gruppenschlüssel, verschlüsselt diesen mit seinem persönlichen Schlüssel und sendet ihn an den Schlüsselverwalter. Der Schlüsselverwalter wiederum entschlüsselt den empfangenen Gruppenschlüssel, verschlüsselt ihn mit dem persönlichen Schlüssel jedes Gruppenmitglieds und sendet ihn an die einzelnen Gruppenmitglieder.

Während im Protokoll von Chu et al. die Rollen von Schreibendem und Schlüsselerneuerer üblicherweise von verschiedenen Parteien eingenommen werden, fungiert in *PGP* [Zim95] jeder Schreibende gleichzeitig als Schlüsselverwalter. Da der paarweise Austausch kryptographischer Geheimnisse zwischen allen Schreibenden und allen Empfängern über einen sicheren Kanal bei steigender Gruppengröße nicht mehr praktikabel ist, wird asymmetrische Kryptographie eingesetzt. Der Schreibende verschlüsselt die Daten mit einem zufällig gewählten Gruppenschlüssel, verschlüsselt den Gruppenschlüssel mit dem öffentlichen Schlüssel jedes Gruppenmitglieds und hängt ihn an die verschlüsselten Daten an.

Auch einige verschlüsselte Dateisysteme wie *SNAD* [Mea01] oder *EFS* [Cro02] basieren darauf, dass der Schreibende die öffentlichen Schlüssel jedes Gruppenmitglieds kennt. Die eigentlichen Nutzdaten werden in diesen Systemen mit einem symmetrischen Schlüssel verschlüsselt, der wiederum mit den öffentlichen Schlüsseln aller Gruppenmitglieder verschlüsselt und in den Metadaten der jeweiligen Datei hinterlegt wird.

Schlüsselerneuerung bei Schreibzugriff mit Hilfsschlüsseln: Den Ansatz, Gruppenmitglieder und Hilfsschlüssel in einer Baumstruktur anzuordnen, greifen *Complete Subtree* [Nea01] und *Subset Difference* [Nea01] auf. Complete Subtree begreift analog zu LKH die Gruppenmitglieder als Blätter des Baumes und die Hilfsschlüssel als innere Knoten, unterscheidet sich allerdings von LKH darin, dass die Hilfsschlüssel nicht erneuert werden. Während in Complete Subtree jeder Teilbaum über einen gemeinsamen Schlüssel adressiert werden kann, gilt dies in *Subset Difference* für jede Teilmenge von Gruppenmitgliedern, die in der Form „Teilbaum X exklusive Teilbaum Y“ ausdrückbar ist. Hierfür müssen die Gruppenmitglieder im Vergleich zu Complete Subtree mehr Hilfsschlüssel vorhalten, diese können allerdings zum Teil voneinander abgeleitet werden.

Ohne eine hierarchische Anordnung der Gruppenmitglieder arbeitet *Content Protection for Recordable Media (CPRM)* [Ent03]. CPRM basiert auf einer Matrix von Hilfsschlüsseln, von denen jedes Gruppenmitglied einen Hilfsschlüssel pro Spalte der Matrix kennt. Verschiedenen Gruppenmitgliedern sind unterschiedliche Mengen an Hilfsschlüssel bekannt. Bei einem Austritt eines Gruppenmitglieds werden die ihm bekannten Hilfsschlüssel als ungültig markiert und im Weiteren nicht mehr verwendet.

4 Exemplarische Anwendung der Klassifikation

Die in Abschnitt 3 vorgestellte Klassifizierung von GKM-Protokollen kann dazu verwendet werden, für ein gegebenes Anwendungsszenario eine Vorauswahl an GKM-Protokollen

Szenario	Charakteristik	Empfohlene Klasse von GKM-Protokollen
Software-entwicklung	kleine Gruppe, wenige Bei- und Austritte, viele Schreibzugriffe	Schlüsselerneuerung bei Bei- oder Austritt, keine Hilfsschlüssel
Wissenschaftliche Kollaboration	mittelgroße Gruppe, wenige Bei- und Austritte, viele Schreibzugriffe	Schlüsselerneuerung bei Bei- oder Austritt, Einsatz von Hilfsschlüsseln
Verteilung kostenpflichtiger Inhalte	große Gruppe, viele Bei- und Austritte, wenige Schreibzugriffe, genügend Ressourcen auf Benutzerseite	Schlüsselerneuerung bei Schreibzugriff, Einsatz von Hilfsschlüsseln
Verteilung kostenpflichtiger Inhalte	große Gruppe, viele Bei- und Austritte, wenige Schreibzugriffe, knappe Ressourcen auf Benutzerseite	Schlüsselerneuerung bei Schreibzugriff, keine Hilfsschlüssel

Tabelle 2: Exemplarische Anwendungsszenarien für Sicheres Teilen von Daten und empfohlene Klasse von GKM-Protokollen

zu treffen, die in diesem Szenario hinsichtlich ihres Ressourcenbedarfs einsetzbar wären. Dies wird im folgenden Abschnitt anhand vier exemplarischer Anwendungsszenarien des *Sicheren Teilens von Daten* demonstriert (siehe Tabelle 2).

Das erste Szenario umfasst ein zehnköpfiges Team von Softwareentwicklern mit relativ konstanter Besetzung, die für die Verwaltung ihres vertraulichen Quellcodes auf einen Drittanbieter wie github oder SourceForge zurückgreifen möchten. Die Entwickler ändern den Quellcode mehrmals täglich. Angesichts der geringen Anzahl an Bei- oder Austritten, aber hohen Anzahl an Schreibzugriffen empfiehlt sich der Einsatz eines GKM-Protokolls, das die Schlüsselerneuerung bei Bei- oder Austritt vornimmt. Durch die geringe Gruppengröße ist ein GKM-Protokoll, das Hilfsschlüssel verwendet, nicht notwendig, selbst wenn die Rolle des Schlüsselverwalters durch einen der Entwickler selbst eingenommen wird.

Das zweite Szenario stellt eine Kooperation innerhalb eines wissenschaftlichen Projekts dar. Hierbei verfassen 400 Wissenschaftler die Projektberichte gemeinschaftlich, beispielsweise mittels eduPad. Die über die Zeit vergleichsweise konstante Zusammensetzung der Gruppe legt den Einsatz eines GKM-Protokolls nahe, das die Schlüsselerneuerung bei Bei- oder Austritt durchführt. Da die Gruppe im Vergleich zum ersten Szenario größer ist, kann der Einsatz von Hilfsschlüsseln die Ressourcen des Schlüsselverwalters schonen.

Im dritten Szenario möchte ein Benutzer über einen Speicherdienst kostenpflichtige Inhalte anbieten. Andere Benutzer können diese Inhalte zu jeder Zeit abonnieren und wieder abbestellen. Die Inhalte werden über breitbandige Netze und leistungsstarke Geräte wie Laptops oder Desktop-Rechner konsumiert. Neue Inhalte werden relativ selten zur Verfügung gestellt. Die Gruppe der Abonnenten umfasst eine große Anzahl an Benutzern, die Zusammensetzung der Gruppe ändert sich aufgrund neu abgeschlossener beziehungsweise gekündigter Abonnements häufig. In diesem Szenario sollte ein GKM-Protokoll eingesetzt werden, das den Schlüssel vor jedem Schreibzugriff erneuert. Da von Benutzerseite in diesem Fall genügend Kommunikationsbandbreite, Rechenleistung und Speicherkapazität bereitgestellt werden kann, bietet sich der Einsatz von Hilfsschlüsseln an.

Das vierte Szenario unterscheidet sich vom dritten Szenario lediglich darin, dass die Inhalte vorwiegend über mobile Geräte konsumiert werden. In diesem Fall sollte angesichts knapper Ressourcen auf Benutzerseite auf Hilfsschlüssel verzichtet werden.

Die asymptotischen Aufwände, die in Abschnitt 3 für Benutzer und Schlüsselverwalter beschrieben wurden, beziehen sich immer auf eine einzelne Gruppe. Natürlich kann aber im praktischen Einsatz ein Schlüsselverwalter mehrere Gruppen verwalten, und ein Benutzer kann mehreren Gruppen angehören. Ob der Ressourcenbedarf für das GKM proportional zur Anzahl der Gruppen steigt, hängt nicht vom eingesetzten GKM-Protokoll ab, sondern davon, wie das GKM-Protokoll in einer gegebenen Anwendung eingesetzt wird. Die Untersuchung dieser Zusammenhänge kann Ausgangspunkt für weitere Forschungen sein.

Weiterhin bleibt in der vorliegenden Arbeit der Ressourcenbedarf auf Seiten des Speicheranbieters unberücksichtigt. Da der der Datenaustausch zwischen Benutzern über den Speicheranbieter erfolgt, bietet es sich an, erneuerte Gruppen- und Hilfsschlüssel ebenfalls über den Speicheranbieter auszutauschen. Dem Speicheranbieter entsteht dadurch Speicher- und Kommunikationslast. Um diesen Ressourcenbedarf asymptotisch abzuschätzen, müssten alle in Abschnitt 3 beschriebenen GKM-Protokolle, die von einer synchronen Kommunikation zwischen den Benutzern ausgehen, in eine asynchrone Variante transformiert werden, was nicht Gegenstand dieser Arbeit ist. Außerdem steht auf Seiten des Speicheranbieters der erhöhte Bedarf an Ressourcen durch GKM Einsparungen gegenüber, die sich dadurch ergeben, dass der Speicheranbieter die Zugriffskontrolle nicht selbst vornimmt und somit unter Umständen auf eine eigene Zugriffsverwaltung verzichten kann.

5 Zusammenfassung und Ausblick

In dieser Arbeit haben wir Group Key Management-Protokolle, die aus unterschiedlichen Anwendungskontexten stammen, hinsichtlich ihres Ressourcenbedarfs klassifiziert. Damit ist es möglich, für konkrete Anwendungsszenarien des *Sicheren Teilens von Daten* eine Vorauswahl an GKM-Protokollen benennen zu können, die hinsichtlich ihres Ressourcenbedarfs geeignet sein könnten. Zu diesem Zweck haben wir zunächst spezifiziert, welche Anforderungen das *Sichere Teilen von Daten* unseres Erachtens erfüllen muss. Die GKM-Protokolle, die diese Anforderungen grundsätzlich erfüllen können, wurden anhand des Zeitpunkts der Erneuerung des Gruppenschlüssels und anhand des Einsatzes von Hilfsschlüsseln klassifiziert. Die resultierenden vier Klassen umfassen GKM-Protokolle, deren asymptotischer Ressourcenbedarf hinsichtlich Gruppengröße, Bei- und Austrittshäufigkeit der Gruppenmitglieder und der Häufigkeit von Schreiboperationen jeweils identisch ist. Unseres Wissens wurde bisher noch keine Klassifikation von GKM-Protokollen publiziert, deren Klassen direkt auf den Ressourcenbedarf der Protokolle schließen lassen. Zum Abschluss der Arbeit wird exemplarisch gezeigt, wie anhand der Klassifikation für konkrete Anwendungsszenarien potentiell geeignete und ungeeignete GKM-Protokolle unterschieden werden können. Wie sich der Ressourcenbedarf bei einer steigenden Anzahl von Gruppen im Gesamtsystem verhält, ist in weiteren Arbeiten zu untersuchen.

Literatur

[Bal96] A. Ballardie. RFC 1949: Scalable Multicast Key Distribution, Mai 1996.

[CC89] Guang-Huei Chiou und Wen-Tsuen Chen. Secure broadcasting using the secure lock. *IEEE Transactions on Software Engineering*, 15(8):929–934, 1989.

[Cea02] Hao-hua Chu et al. A secure multicast protocol with copyright protection. *ACM SIGCOMM Computer Communication Review*, 32(2):42–60, April 2002.

[Cha05] Yacine Challal. Group key management protocols: A novel taxonomy. *International Journal of Information Technology*, 2(2):105–118, 2005.

[Cro02] D. Cross. *Encrypting File System in Windows XP and Windows Server 2003*. Microsoft Corporation, 2002. http://technet.microsoft.com/en-us/library/bb457065.aspx.

[Dea01] B. DeCleene et al. Secure group communications for wireless networks. In *Proceedings of MILCOM 2001*, Jgg. 1, Seiten 113–117. IEEE, 2001.

[DMS99] Lakshminath R. Dondeti, Sarit Mukherjee und Ashok Samal. Survey and Comparison of Secure Group Communication Protocols. 1999.

[Ent03] 4C Entity. Content protection for recordable media: Introduction and cryptographic elements, 2003. http://4centity.com.

[HM97] H. Harney und C. Muckenhirn. RFC 2093: Group Key Management Protocol (GKMP) Specification. Juli 1997.

[Lea04] Jeffrey Lotspiech et al. Anonymous Trust: Digital Rights Management Using Broadcast Encryption. *Proceedings of the IEEE*, 92(6):898–909, Juni 2004.

[Mea01] Ethan Miller et al. Strong Security for Distributed File Systems. In *Conference Proceedings of the 2001 IEEE International Performance, Computing, and Communications Conference (Cat. No.01CH37210)*, Seiten 34–40. IEEE, 2001.

[Nea01] Dalit Naor et al. Revocation and Tracing Schemes for Stateless Receivers. In Joe Kilian, Hrsg., *Advances in Cryptology - CRYPTO 2005*, Seiten 41–62. 2001.

[SM03] A.T. Sherman und D.A. McGrew. Key establishment in large dynamic groups using one-way function trees. *IEEE Transactions on Software Engineering*, Mai 2003.

[SNPB10] Ning Shang, Mohamed Nabeel, Federica Paci und Elisa Bertino. A privacy-preserving approach to policy-based content dissemination. In *2010 IEEE 26th International Conference on Data Engineering (ICDE 2010)*, Seiten 944–955. IEEE, 2010.

[Wea99] Marcel Waldvogel et al. The VersaKey framework: versatile group key management. *IEEE Journal on Selected Areas in Communications*, 17(9):1614–1631, 1999.

[WHA99] D. Wallner, E. Harder und R. Agee. RFC 2627: Key Management for Multicast: Issues and Architectures, 1999.

[ZDB08] X. Zou, Y.-S. Dai und E. Bertino. A Practical and Flexible Key Management Mechanism For Trusted Collaborative Computing. In *2008 IEEE INFOCOM - The 27th Conference on Computer Communications*, Seiten 538–546. IEEE, April 2008.

[Zim95] Philip R. Zimmermann. *The official PGP user's guide*. MIT Press, 1995.