

Ein Schlüsselverteilungsprotokoll für kleine geschlossene Peer-to-Peer Systeme

Fuwen Liu, Hartmut König

Brandenburgische Technische Universität Cottbus
Lehrstuhl Rechnernetze und Kommunikationssysteme
PF 10 13 44, 03013 Cottbus
email: {lfw,koenig}@informatik.tu-cottbus.de

Abstract: Vertraulichkeit ist eine der Schlüsselanforderungen für geschäftliche Kommunikation über das Internet. In einer zunehmend mobilen Gesellschaft sind dabei zunehmend spontane Beratungen in Ad hoc-Umgebungen, mitunter mit wechselnden Partnern, erforderlich. Um die Vertraulichkeit der Beratung zu sichern, müssen sich die Partner auf einen gemeinsamen Schlüssel einigen, mit dem sie ihre Kommunikation verschlüsseln. Audio- und Videokonferenzsysteme, die auf einem zentralistischen Ansatz beruhen, bieten dafür praktikable Lösungen an. Dezentrale Lösungen, die dem Peer-to-Peer-Ansatz folgen, bieten hierfür flexiblere Lösungen, die Spontaneität und Flexibilität besser unterstützen. Ein effizienter Schlüsselaustausch stellt für solche Systeme noch eine Herausforderung dar. In diesem Beitrag stellen wir das Schlüsselverteilungsprotokoll VTKD vor, das speziell für den Schlüsselaustausch von kleinen dynamischen Peer-Gruppen mit bis zu 100 Partnern entworfen wurde. Es besteht aus zwei Bestandteilen: einer gegenseitigen Authentifizierung der Partner und einer sicheren Verteilung des Sitzungsschlüssels an die Partner. Das Protokoll nutzt ein virtuelles Token, um den Partner zu bestimmen, der im Fall des Schlüsselwechsels den neuen Schlüssel generiert und verteilt. Wir beschreiben den Protokollverlauf und untersuchen seine Sicherheit. Eine abschließende Leistungsanalyse zeigt, dass VTKD eine geringere Verzögerung für die Schlüsselrenewal benötigt als existierende Schlüsselaustauschprotokolle.

1 Motivation

Moderne gruppenorientierte und kollaborative Applikationen nutzen verstärkt das Peer-to-Peer-Prinzip. Das bietet gegenüber zentralistischen Ansätzen den Vorteil einer größeren Unabhängigkeit von einer möglicherweise teuren Infrastruktur, wie sie z. B. für Audio- und Videokonferenzen mit H.32x-Systemen existiert. Dezentrale Systeme erweisen sich hier flexibler, da es keinen *Single Point of Failure* gibt und sich die Abhängigkeit von einer Infrastruktur reduziert. Dezentrale Lösungen unterstützen insbesondere spontane Treffen und die Mobilität der Partner. Dies ist insbesondere für die geschäftliche Kommunikation über das Internet von Vorteil. Dezentrale Lösungen erfordern jedoch auch entsprechende Mechanismen, um die Vertraulichkeit der geschäftlichen Absprachen abzusichern. Dazu sind insbesondere Verfahren für den Schlüsselaustausch erforderlich, die eine konsistente Erneuerung der Schlüssel bei allen Gesprächspartnern

sichert. Während für zentralistische Ansätze praktikable Lösungen existieren, ist die Entwicklung effizienter und sicherer Protokolle für verteilte Lösungen noch Gegenstand der Forschung.

Eine sichere Kommunikation zwischen einer Gruppe von Geschäftspartnern erfordert, dass nur die aktiven Partner den aktuellen Gruppen- bzw. Sitzungsschlüssel teilen, um die ausgetauschten Daten zu verschlüsseln. Bei einer sich möglicherweise ändernden Gruppenzusammensetzung kann es darüber hinaus gewünscht und gefordert sein, dass Inhalte und Gegenstand der Beratung Partnern, die später beitreten oder diese eher verlassen, nicht zugänglich werden. Wir betrachten im Folgenden diese komplexere Variante einer vertraulichen Beratung. Varianten mit geringeren Vertraulichkeitsanforderungen an eine sich verändernde Gruppenzusammensetzung können daraus abgeleitet werden.

An das Schlüsselmanagement einer solchen Gruppe werden eine Reihe unterschiedlicher Anforderungen gestellt [1], [2]. (1) Jedes Gruppenmitglied hat sicherzustellen, dass niemand anderes außerhalb der Gruppe Zugang zu dem Gruppenschlüssel bekommen kann (*key authentication*). Voraussetzung dafür ist eine gegenseitige Authentifizierung beim Beitritt zur Gruppe, die sicherstellt, dass der eingeladene Partner auch der von der Gruppe erwartete Gesprächspartner ist, und umgekehrt dem Eingeladenen die Gewißheit gibt, dass er auch der Gruppe vertrauen kann. (2) Gruppenmitglieder, die die Sitzung zu irgendeinem Zeitpunkt verlassen, sollen keinen Zugang zu einem später generierten Schlüssel bekommen, um die weitere Kommunikation zu entschlüsseln (*forward confidentiality*). (3) Gruppenmitglieder, die der Sitzung später beitreten, sollen keinen Zugang zu einem älteren Schlüssel erhalten, um die Gespräche vor ihrem Beitritt offenzulegen. (4) Keine mögliche Teilgruppe von Teilnehmern, die die Sitzung verlassen hat, soll in der Lage sein, den aktuellen Schlüssel unter Ausnutzung älterer Schlüssel abzuleiten (*collusion freedom*). Weiterhin ist es wünschenswert, dass die Kompromittierung eines Schlüssels nicht zur Aufdeckung früherer Schlüssel führt (*perfect forward secrecy*) und dass die Aufdeckung von Schlüsseln früherer Sitzungen nicht zur Kompromittierung des aktuellen Schlüssels führen kann (*resistance to known key attacks*). Fast selbstverständlich erscheint die Forderung nach einem effizienten Schlüsselaustauschprotokoll, um die Interferenzzeiten in der Kommunikation für die Schlüsselerneuerung, insbesondere für Realzeit-Anwendungen wie Audio- und Videokonferenzen zu minimieren, da in dem asynchronen Internet Hosts i. d. R. nicht in Lage sind die Schlüssel synchron zu erneuern [3], [4].

An das Schlüsselmanagement einer solchen Gruppe werden eine Reihe unterschiedlicher Anforderungen gestellt [1], [2]. (1) Jedes Gruppenmitglied hat sicherzustellen, dass niemand anderes außerhalb der Gruppe Zugang zu dem Gruppenschlüssel bekommen kann (*key authentication*). Voraussetzung dafür ist eine gegenseitige Authentifizierung beim Beitritt zur Gruppe, die sicherstellt, dass der eingeladene Partner auch der von der Gruppe erwartete Gesprächspartner ist, und umgekehrt dem Eingeladenen die Gewissheit gibt, dass er auch der Gruppe vertrauen kann. (2) Gruppenmitglieder, die die Sitzung zu irgendeinem Zeitpunkt verlassen, sollen keinen Zugang zu einem später generierten Schlüssel bekommen, um die weitere Kommunikation zu entschlüsseln (*forward confidentiality*). (3) Gruppenmitglieder, die der Sitzung später beitreten, sollen keinen Zugang zu einem älteren Schlüssel erhalten, um die Gespräche vor ihrem Beitritt offenzulegen.

(4) Keine mögliche Teilgruppe von Teilnehmern, die die Sitzung verlassen hat, soll in der Lage sein, den aktuellen Schlüssel unter Ausnutzung älterer Schlüssel abzuleiten (*collusion freedom*). Weiterhin ist es wünschenswert, dass die Kompromittierung eines Schlüssels nicht zur Aufdeckung früherer Schlüssel führt (*perfect forward secrecy*) und dass die Aufdeckung von Schlüsseln früherer Sitzungen nicht zur Kompromittierung des aktuellen Schlüssels führen kann (*resistance to known key attacks*). Fast selbstverständlich erscheint die Forderung nach einem effizienten Schlüsselaustauschprotokoll, um die Interferenzzeiten in der Kommunikation für die Schlüsselerneuerung, insbesondere für Realzeit-Anwendungen wie Audio- und Videokonferenzen zu minimieren, da in dem asynchronen Internet Hosts i. d. R. nicht in Lage sind die Schlüssel synchron zu erneuern [3], [4].

In diesem Beitrag stellen wir das Schlüsselaustausch-Protokoll VTKD (*virtual token based key distribution*) vor. VTKD ist ein Schlüsselverteilungsprotokoll, das insbesondere die Schlüsselerneuerung in kleinen geschlossenen Peer-to-Peer Systemen mit bis zu 100 Mitgliedern unterstützen soll. Es unterstützt die oben genannten Bedingungen und weist im Vergleich zu anderen Schlüsselaustauschprotokollen eine bessere Effizienz auf. VTKD ist im Kontext der Videokonferenzforschung an unserem Lehrstuhl entstanden. Es ist Bestandteil der Sicherheitsarchitektur des Peer-to-Peer-Videokonferenzsystems BRAVIS [5], [19] um vertrauliche Videokonferenzen zu unterstützen. Der Begriff der geschlossenen Gruppe bezieht sich dabei auf eine Gruppenzusammensetzung, in der sich alle Teilnehmer kennen. Der Beitritt zur Gruppe erfolgt über eine explizite Einladung, die bei einer vertraulichen Konferenz mit einer Authentifizierung der Partner verbunden ist. Die angestrebte Obergrenze von 100 Teilnehmern ist sehr großzügig ausgelegt. Im Alltag haben Geschäftsberatungen häufig weniger als 15 Teilnehmer. Dieser Beitrag beschreibt den Protokollverlauf von VTKD und diskutiert seine Sicherheitseigenschaften sowie Leistungsparameter. Es ist wie folgt gegliedert. Im 2. Abschnitt geben wir einen kurzen Überblick über den Stand der Entwicklung von verteilten Schlüsselaustauschprotokollen. Abschnitt 3 beschreibt das Prinzip und die Protokollabläufe des Protokolls. Abschnitt 4 diskutiert die Sicherheitseigenschaften und zeigt, wie die obigen Forderungen erfüllt werden. Der 5. Abschnitt bewertet die Leistungsparameter des Protokolls. Einige zusammenfassende Bemerkungen beschließen den Beitrag.

2 Schlüsselaustauschprotokolle für Gruppe

Es werden zwei Arten von Schlüsselaustauschprotokollen für Gruppen unterschieden: die Schlüsselvereinbarungsprotokolle (*key agreement protocols*) und die Schlüsselverteilungsprotokolle (*key distribution protocols*). Die beiden Protokollarten unterscheiden sich in der Art der Schlüsselerneuerung. Schlüsselvereinbarungsprotokolle basieren auf dem klassischen Diffie-Hellmann-Schlüsselaustauschprinzip [6]. Das Grundprinzip besteht darin, dass jedes Gruppenmitglied einen Beitrag zur Schlüsselerzeugung leisten muss. Dazu wird ein Gruppenmitglied ausgewählt, der Zwischenschlüssel generiert, die er an die anderen Mitglieder der Gruppe verteilt. Aus den Zwischenschlüsseln und ihrem Beitrag erzeugen die Gruppenmitglieder dann den Gruppenschlüssel. Bekannte Beispiele für diese Art von Protokollen sind CLIQUES [7] und TGDH [8,9]. Letzteres gilt gegenwärtig als das effizienteste Schlüsselvereinbarungsprotokoll. Im Gegensatz dazu bestimm-

men Schlüsselverteilungsprotokolle dynamisch einen Teilnehmer, der den neuen Schlüssel erzeugt und ihn sicher an die anderen Gruppenmitglieder verteilt. Die meisten Ansätze verwenden einen Schlüsselverteilungsbaum. Sie unterscheiden sich dadurch, wie die Gruppenmitglieder den Schlüssel über diesen Baum erhalten. Beispiele sind DTKM (*distributed tree-based key management scheme*) [10] und der von Rodeh et al. vorgeschlagene Verteilungsbaum [11], einer Erweiterung der zentralisierten logischen Schlüsselhierarchie aus [12].

Im Vergleich beider Protokollarten gelten die Schlüsselverteilungsprotokolle als effizienter, weil sie insgesamt einen geringeren Berechnungs- und Kommunikationsaufwand für die Schlüsselerzeugung und -verteilung benötigen. Für das angestrebte Ziel Echtzeitkommunikation, wie sie Audio- und Videokonferenzen darstellen, wurde von uns der Einsatz eines Schlüsselverteilungsprotokolls priorisiert. Weiterhin sei angemerkt, dass die meisten Schlüsselaustauschprotokolle, so auch die oben zitierten, keine Authentifizierung der Gruppenmitglieder enthalten. Eine solche Erweiterung wurde für CLIQUES in [13] vorgeschlagen. Sie gilt jedoch nicht als sicher [14].

3 Schlüsselverteilungsprotokoll VTKD

VTKD ist ein Schlüsselverteilungsprotokoll, das den Austausch des Gruppenschlüssels in kleinen geschlossenen Peer-Gruppen unterstützen soll, um vertrauliche Kommunikation zu ermöglichen. Wir stellen in diesem Abschnitt die wesentlichen Prinzipien und die wichtigsten Abläufe des Protokolls vor.

3.1. Einbettung in die System-Architektur

VTKD geht von einer 3-Schichtenarchitektur aus mit einer Anwendungsschicht, einer Sicherheitsschicht und einer Kommunikationsschicht. Abbildung 1 zeigt die drei Schichten integriert in die BRAVIS-System-Architektur [5]. Das Schlüsselverteilungsprotokoll ist der Sicherheitsschicht zugeordnet und läuft im Signalisierungsteil von BRAVIS. Mit Hilfe des Gruppenschlüssels können sowohl die Mediendaten als auch die Signalisierungsdaten verschlüsselt werden.

Die Anwendungsschicht braucht hier nicht vollständig spezifiziert werden. Sie enthält die für die jeweilige Anwendung erforderlichen Komponenten. In einer Videokonferenzanwendung sind das u. a. das QoS-Management, die Floorkontrolle, die Audio- und Videomanager und das Whiteboard. Ein wesentlicher Bestandteil ist die Gruppenverwaltung, von der wir im Folgenden annehmen, dass sie ebenfalls in der Anwendungsschicht angesiedelt ist. Das Gruppenmanagement erhält über das Nutzerinterface die Anforderungen zum Beitritt oder Verlassen der Gruppe, die sie über das Gruppenkommunikationsprotokoll an die anderen Teilnehmer weiterleitet. Der Ausfall eines Teilnehmers wird durch das Gruppenkommunikationsprotokoll erkannt und den anderen Teilnehmern angezeigt.

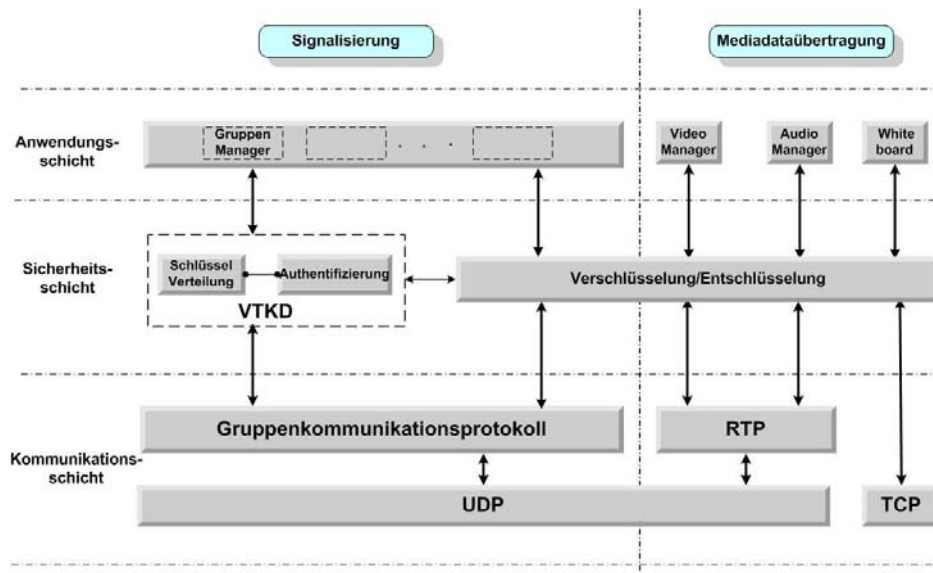


Abbildung 1: Systemarchitektur

Die Sicherheitsschicht enthält die Verschlüsselungsmodule und das Schlüsselverteilungsprotokoll VTKD, das hier im Mittelpunkt der Betrachtung steht. Die Schlüsselerneuerung wird ausgelöst, wenn sich die Gruppenzusammensetzung ändert, d. h. ein neuer Teilnehmer der Gruppe beitrifft (*join*), ein Teilnehmer die Gruppe verlässt (*leave*) oder ein Teilnehmerrechner ausfällt. Der Beitritt zur Gruppe ist in VTKD mit einer gegenseitigen Authentifizierung der Partner verbunden, um sicherzustellen, dass beide Seiten sich vertrauen können.

Die Kommunikationsschicht enthält die Protokolle für die Übertragung der Signallerungs- und Mediendaten. Für die Schlüsselerneuerung ist ausschließlich das Gruppenkommunikationsprotokoll im Signallerungsteil relevant, das, wie weiter unten noch diskutiert wird, eine wichtige Grundlage für das Schlüsselverteilungsprotokoll bildet. In kollaborativen Peer-to-Peer-Anwendungen bildet das Gruppenkommunikationsprotokoll die Grundlage für eine zuverlässige Funktionsweise des Systems oder der Anwendung. Es muss die Gruppendaten in allen Peers aktualisieren und dafür sorgen, dass alle Peers eine konsistente Sicht auf die Gruppe haben, damit sie eigenständig Entscheidungen zu den QoS-Parametereinstellungen, der Zuweisung der Floor und der Erneuerung des Gruppenschlüssels treffen können. Dazu muss das Gruppenkommunikationsprotokoll eine virtuelle Synchronisation zwischen den Gruppenmitgliedern absichern [15]. Virtuelle Synchronisation bedeutet, dass alle Gruppenmitglieder die ausgetauschten Nachrichten zuverlässig in der Reihenfolge erhalten, in der sie gesendet wurden. Das erfordert, dass das Gruppenkommunikationsprotokoll *zuverlässig*, *geordnet* und *atomar* ist, um Datenverluste zu vermeiden, die Übertragungsreihenfolge zu sichern und eine konsistente Aktualisierung der Gruppendaten zu gewährleisten. Virtuelle Synchronisation erfordert, dass das Gruppenkommunikationsprotokoll alle Änderungen der Gruppenzusammensetzung (Beitritt, Verlassen, Ausfall) allen Mitgliedern anzeigt. Es gibt einige Proto-

kolle, die die virtuelle Synchronisation unterstützen wie RMP [16], das Totem Protokoll [17] und GCP [18], das in BRAVIS eingesetzt wird. Ein verteiltes Schlüsselmanagement erfordert eine virtuelle Synchronisation beim benutzten Gruppenkommunikationsprotokoll, d. h., es besteht eine enge Beziehung zwischen diesen beiden Protokollen [9], [11]. Wenn diese Eigenschaft nicht erfüllt ist, kann es aufgrund der unterschiedlichen Sichtweise auf die Gruppe zu einer Konfusion bei der Schlüsselerneuerung kommen, da möglicherweise mehrere Mitglieder für die Schlüsselerneuerung bestimmt werden. Deshalb nehmen wir wie auch andere Schlüsselmanagementprotokolle im Folgenden an, dass die Eigenschaft der virtuellen Synchronisation für das Gruppenkommunikationsprotokoll gegeben ist.

3.2. Prinzip von VTKD

VTKD ist ein verteiltes Schlüsselverteilungsprotokoll, das auf dem Prinzip des Schlüsselaustauschs nach Diffie-Hellman (DH) beruht [6], d. h. es gibt keine zentrale Schlüsselverwaltung. Im Unterschied zum Schlüsselaustausch zwischen zwei Partnern berechnet beim verteilten Ansatz jedes Gruppenmitglied mit jedem Partner einen geheimen Schlüssel nach dem Diffie-Hellmann-Prinzip, der im Weiteren als gemeinsames oder zweiseitiges DH-Geheimnis bezeichnet wird. Diese zweiseitigen Geheimnisse werden bei den Gruppenmitgliedern gespeichert und werden dann für die Verteilung des Gruppenschlüssels genutzt. Bezüglich der Gruppenmitglieder wird angenommen, dass sie dieselben Rechte haben und ihnen das gleiche Vertrauen entgegengebracht wird. Das bedeutet, dass jedes Gruppenmitglied ein neues Mitglied authentifizieren darf und den Gruppenschlüssel erneuern kann. Wir nehmen ferner an, dass ein in die Gruppe aufgenommenes Mitglied vertrauenswürdig ist und nicht aktiv versucht die Beratung zu stören oder den Sitzungsschlüssel an Nichtmitglieder weiterzugeben. Es werden jedoch keine Annahmen über die Vertrauenswürdigkeit der Partner nach dem Verlassen der Gruppe getroffen. Diese Annahmen entsprechen der praktischen Verfahrensweise. Die im Abschnitt 2 referierten Protokolle treffen ähnliche Annahmen.

VTKD ist ein Token-Protokoll. Nur der Tokenhalter hat jeweils das Recht zur Erneuerung des Gruppenschlüssels und zur Authentifizierung beitretender Partner. VTKD verwendet jedoch kein physikalisches Token, das auf einem logischen Kreis der Gruppenmitglieder weitergegeben wird, sondern ein virtuelles Token. Virtuell bedeutet in diesem Fall, dass die Position des virtuellen Tokens und damit des Tokenhalters für jede Schlüsselverteilung neu berechnet wird. Damit wird die explizite Tokenweitergabe mit allen damit verbundenen Problemen wie Tokenverlust und -dopplung vermieden. Die neue Tokenposition PT wird wie folgt berechnet:

$$PT = VK \bmod n. \quad (1)$$

Dabei bezeichnet VK die jeweilige Versionsnummer des Gruppenschlüssels und n die aktuelle Anzahl der Gruppenmitglieder. VK wird bei jeder Erneuerung des Gruppenschlüssels um 1 erhöht. Neben der Bestimmung der Tokenposition wird die Versionsnummer im Protokoll auch dazu genutzt Replay-Attacken zu vermeiden. Darauf gehen

wir weiter unter ein. Die Gewährleistung einer virtuellen Synchronisation durch das verwendete Gruppenkommunikationsprotokoll sichert, dass jedes Gruppenmitglied die aktuelle Gruppengröße und die Schlüsselversion kennt. Damit kann jedes Gruppenmitglied die Position des virtuellen Tokens eindeutig bestimmen.

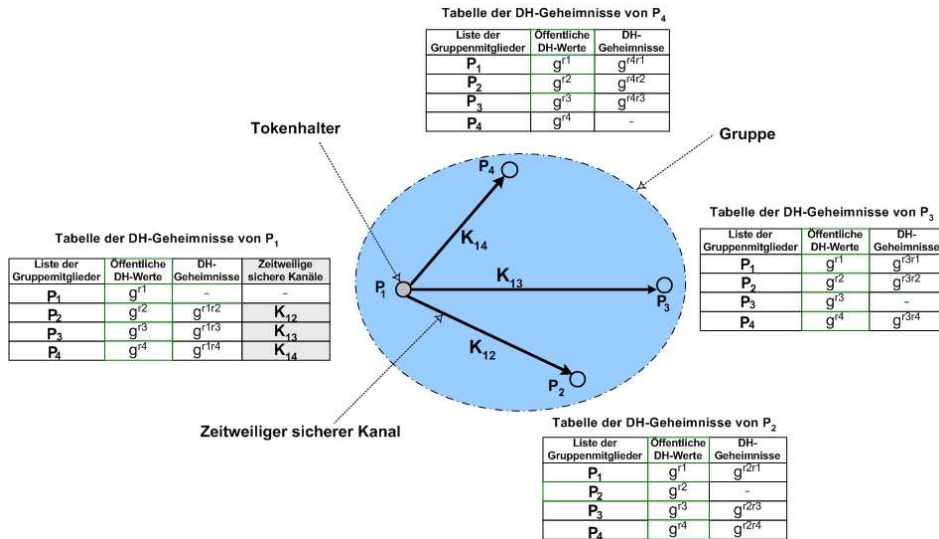


Abb.2: Schlüsselverteilung über zeitweilige sichere Kanäle

Die Schlüsselenerneuerung wird, wie eingangs bereits erwähnt, durch eine Veränderung der Gruppenzusammensetzung ausgelöst. Der jeweilige Tokenhalter generiert einen neuen Schlüssel und beginnt mit der Verteilung an die Gruppenmitglieder. Dazu baut er zeitweilige separate Kanäle zu jedem Gruppenmitglied unter Nutzung des gespeicherten gemeinsamen DH Geheimnisses auf. Abbildung 2 zeigt das Prinzip an einer Gruppe von vier Teilnehmern (P_1, P_2, P_3 und P_4), wobei P_1 der aktuelle Tokenhalter sei. Jedes Mitglied kennt sein Geheimnis mit den anderen Teilnehmern. So speichert P_1 die Geheimnisse $g^{r1r2}, g^{r1r3},$ und g^{r1r4}, P_2 entsprechend die Geheimnisse $g^{r2r1}, g^{r2r3},$ und g^{r2r4} usw. P_1 baut dann unter Verwendung der gemeinsamen Geheimnisse $g^{r1r2}, g^{r1r3}, g^{r1r4}$ geheime Kanäle $K_{12}, K_{13},$ and K_{14} zu $P_2, P_3,$ und P_4 auf, über die er dann den neuen Gruppenschlüssel verteilt. Die separaten geheimen Kanäle sind durch einen geheimen Schlüssel K_{ij} definiert, der zwischen den beiden Gruppenmitgliedern berechnet wird. Dabei wird folgendes Berechnungsschema benutzt:

$$K_{j-e} = H(g^{rirj}, g^{rij} | N_i | ID_i | ID_j | 0) \quad (2)$$

$$K_{j-a} = H(g^{rirj}, g^{rij} | N_i | ID_i | ID_j | 1) \quad (j=1, 2, \dots, n \text{ and } j \neq i) \quad (3)$$

Es wird ein Schlüsselpaar berechnet. K_{ij-c} wird für die Verschlüsselung der Nachricht genutzt, während K_{ij-a} zur Prüfung der Authentizität der Nachrichten dient. Die Erzeugung der Schlüssel erfolgt mit Hilfe einer kryptographischen Hash-Funktion $H(k,M)$ berechnet, wobei k ein Schlüssel und M die Nachricht bezeichnet. In unserem Fall wird HMAC [20] genutzt. In die Berechnung gehen das gemeinsame Geheimnis zwischen dem Tokenhalter und dem Gruppenmitglied, ihre Identitäten ID und eine Zufallszahl N ein, die der Tokenhalter an das Gruppenmitglied schickt (siehe Abschnitt 3.3). Das Symbol „|“ bedeutet dabei Verkettung.

Die entscheidende Vorbedingung für VTKD ist, dass jedes Gruppenmitglied jeweils die gemeinsamen DH Geheimnisse mit den anderen Gruppenmitgliedern entsprechend der aktuellen Gruppenzusammensetzung besitzt. Diese Vorbedingung ist einfach zu erfüllen. Wenn ein Gruppenmitglied die Gruppe verlässt, löschen die verbleibenden Mitglieder jeweils das zugehörige Geheimnis in ihren Tabellen. Der umgekehrte Fall ist komplizierter, da das neue Gruppenmitglied wie auch die alten Mitglieder jeweils nicht den öffentlichen DH-Wert der Gegenseite kennen. Deshalb sendet der Tokenhalter während der Authentifizierungsphase alle öffentlichen DH-Werte der Gruppe an das neue Mitglied. Umgekehrt übergibt das neue Mitglied seinen öffentlichen DH-Wert an den Tokenhalter, der es an die Gruppenmitglieder weiterleitet. Jedes Gruppenmitglied berechnet dann das gemeinsame Geheimnis mit dem neuen Mitglied. Damit ist die Vorbedingung wieder erfüllt und jedes Mitglied könnte die Schlüsselerneuerung und -verteilung ausführen, falls ihm das Token zugewiesen wird.

3.3 Join-Prozedur

Der Beitritt zur Gruppe wird über die Join-Prozedur realisiert. Sie umfasst zwei Schritte: (1) eine Authentifizierungsphase und (2) die durch die Veränderung der Gruppenzusammensetzung erforderliche Erneuerung des Sitzungsschlüssels (siehe Abbildung 3). Es werden fünf Nachrichten bzw. Kommunikationsrunden benötigt: vier für die Authentifizierung und eine für die Schlüsselerneuerung.

Authentifizierung

Für die gegenseitige Authentifizierung zwischen dem Tokenhalter und dem Eingeladenen kann jedes Protokoll für die Instanzen-Authentifizierung eingesetzt, z. B. die X.509 Authentifizierungsprozedur [21], IKE [22], JFK [23] u. a. Für unsere Anwendung haben wir uns für den Einsatz des IKEv2 (*Internet Key Exchange Protocol*) [24] entschieden, das im Gegensatz zu den meisten zuvor genannten Protokollen, die Identität der Partner schützt und weniger Kommunikationsrunden benötigt. Es waren einige Modifikationen an den ausgetauschten Nachrichten erforderlich, um das Protokoll für die Gruppenkommunikation zu adaptieren.

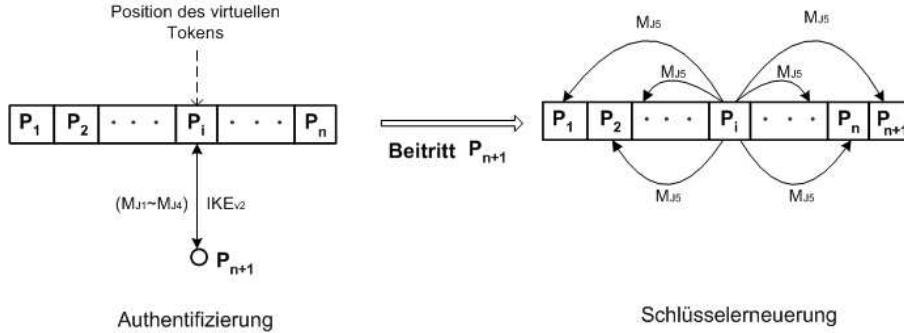


Abbildung 3: Beitritt eines neuen Mitglieds

IKEv2 unterstützt zwei Formen der Authentifizierung: digitale Signaturen und vorvereinbarte gemeinsame Geheimnisse. Wir nutzen hier digitale Signaturen, die besser geeignet für Peer-to-Peer-Anwendungen sind als gemeinsame Geheimnisse, die Client/Server-Architekturen unterstützen. Bei digitalen Signaturen hängt die erfolgreiche Authentifizierung von der Authentizität des öffentlichen Schlüssels ab. Dies wird überwiegend durch die Verwendung von Zertifikaten überprüft. Das X.509-Zertifikat [25] wird hierfür gegenwärtig am meisten genutzt. Es empfiehlt die Anwendung der RSA-Signatur. Dazu müssen beide Partner im Besitz eines RSA-Schlüsselpaars des sein. Selbstverständlich ist der öffentliche Schlüssel von CA (*Certificate Authority*) für beide Partner notwendig, um die Zertifikate, die von CA signiert sind, überprüfen zu können.

Abbildung 3 zeigt den Beitritt eines neuen Partners P_{n+1} zu einer aus n Teilnehmern bestehenden Beispielgruppe $P_1 \dots P_n$. Wir nehmen an, dass Teilnehmer P_i der aktuelle Tokenhalter sei, der entsprechend Formel (1) bestimmt wurde. Für die gegenseitige Authentifizierung zwischen P_i und P_{n+1} werden die folgenden vier Nachrichten ausgetauscht, wobei *HDR* den Nachrichtenkopf bezeichnet:

$$M_{J1}(P_i \rightarrow P_{n+1}): HDR, a^{r_i}, SA_i, NA_i$$

$$M_{J2}(P_{n+1} \rightarrow P_i): HDR, a^{r_{n+1}}, SA_{n+1}, NA_{n+1}$$

$$M_{J3}(P_i \rightarrow P_{n+1}): HDR, SK\{ID_i, CERT_i, SIG_i, ID_1, ID_2 \dots ID_n, g^{r_1}, g^{r_2} \dots g^{r_n}\}$$

$$M_{J4}(P_{n+1} \rightarrow P_i): HDR, SK\{ID_{n+1}, CERT_{n+1}, SIG_{n+1}, g^{r_{n+1}}\}$$

Dabei bedeuten *CERT* das Zertifikat des öffentlichen RSA-Schlüssels und *SIG* die digitale Signatur. $SK\{M\}$ bedeutet, dass Nachricht *M* unter dem Verschlüsselungsschlüssel SK_e verschlüsselt und unter dem Authentisierungsschlüssel SK_a authentifiziert werden. g^f ist der öffentliche DH-Wert zur Erzeugung des zeitweiligen sicheren Kanals *K*. a^f ist der öffentliche DH-Wert zur Erzeugung des Sitzungsschlüssels *SK*. Wie die digitale Signatur *SIG* und der Sitzungsschlüssel *SK* generiert werden ist ausführlich im Standard IKEv2 [24] dargestellt.

Die Nachrichten M_{J1} und M_{J2} erfüllen zwei Funktionen. Sie dienen zum einen dazu, die Sicherheitsassoziation SA auszuhandeln. Die Sicherheitsassoziation SA spezifiziert kryptographische Parameter, die in den Nachrichten M_{J3} und M_{J4} genutzt werden. Des Weiteren werden mit den Nachrichten M_{J1} und M_{J2} die öffentlichen DH-Werte a^r und die Zufallszahlen NA beider Partner ausgetauscht. Diese werden zur Erzeugung des Sitzungsschlüssels SK verwendet, der zum Schützen der folgenden Nachrichten M_{J3} and M_{J4} eingesetzt wird.

Die Nachrichten M_{J3} und M_{J4} dienen in IKEv2 der wechselseitigen Authentifizierung der Partner und dem Aushandeln der Sicherheitsassoziation, die für die weitere Kommunikation zwischen den beiden Partnern genutzt wird. Die Authentifizierung der Partner erfolgt durch die gegenseitige Verifikation der Signaturen SIG. Dazu signieren beide Peers die Verkettung ihrer ersten Nachricht mit der Zufallszahl des Partners mit ihrem privaten RSA Schlüssel. Das schließt zugleich mögliche Man-in-the-Middle-Attacken aus, da der Angreifer die Signaturen nicht ändern kann, ohne die privaten RSA-Schlüssel beider Partner zu kennen. Das Prinzip der Aushandlung der Sicherheitsassoziation kann nicht direkt auf VTKD übertragen werden, da es sich in IKEv2 um eine zweiseitige Beziehung handelt. M_{J3} und M_{J4} tauschen in VTKD zusätzlich Gruppeninformationen aus. Die Nachricht M_{J3} transportiert die Identitäten aller Gruppenmitglieder (ID_1, ID_2, \dots, ID_n) und deren zugehörigen öffentliche DH-Werte ($g^{r1}, g^{r2}, \dots, g^{rn}$). Der eingeladene Partner gibt mit M_{J4} seine Identität ID_{n+1} und seinen öffentlichen DH-Wert g^{m+1} zurück

Wenn die Authentifizierung nicht erfolgreich ist, informiert der Tokenhalter mit der Nachricht M_{Jf} :

$$M_{Jf}(P_i \rightarrow P_1, P_2 \dots P_n) : HDR, GK_{old} \{ID_{n+1}\}$$

die Gruppe. Der Beitrittsprozess wird abgebrochen. Die Gruppe kann weiterhin den gleichen Sitzungsschlüssel verwenden.

Erneuerung des Gruppenschlüssels

Bei einer erfolgreichen Authentifizierung erneuert P_i den Gruppenschlüssel. Der neue Schlüssel GK_{new} wird zufällig generiert und ist unabhängig von den vorangegangenen. Der Tokenhalter sendet den neuen Schlüssel mit der Multicast-Nachricht M_{J5} an die erweiterte Gruppe. Für den Austausch von M_{J5} werden die im Abschnitt 3.2 erläuterten sicheren Kanäle genutzt. M_{J5} hat folgende Struktur:

$$M_{J5}(P_i \rightarrow P_1, P_2 \dots P_{n+1}) : HDR, GK_{old} \{ID_i, N_i\}, K_{il} \{VK, GK_{new}\} \\ \dots, K_{in} \{VK, GK_{new}\}, SK \{GK_{new}, VK, GSA, ID_i\}, GK_{new} \{g^{m+1}, ID_{n+1}\}$$

Die Nachricht hat vier Teile, die unterschiedlichen Zielen dienen. Der erste Teil enthält die Identität des Tokenhalters ID_i und eine Zufallszahl N . Er wird mit dem alten Gruppenschlüssel verschlüsselt. Beide werden genutzt, um den sicheren Kanal entsprechend Formel (2) und (3) aufzubauen. Der zweite Teil enthält den neuen Gruppenschlüssel GK_{new} und die aktuelle Versionsnummer VK . Beide Elemente werden mit Hilfe des je-

weiligen Schlüssels des geheimen Kanals K_{ij} ($j=1, 2, \dots, n$ mit $j \neq i$) für jedes Gruppenmitglied separat verschlüsselt. Beim Empfang berechnet das Gruppenmitglied unter Zuhilfenahme der Angaben aus Teil 1 die aktuellen Schlüssel seines Kanals und entschlüsselt seinen Teil. Der dritte Teil übermittelt den Gruppenschlüssel und dessen Versionsnummer an das neue Mitglied. Weiterhin werden die Sicherheitsassoziation der Gruppe GSA sowie die ID des Tokenhalters übermittelt, damit das neue Mitglied den Tokenhalter als Absender erkennt. Dieser Teil wird mit dem während der Authentifizierungsphase vereinbarten Sitzungsschlüssel SK verschlüsselt. Der vierte Teil, der mit dem neuen Gruppenschlüssel verschlüsselt ist, enthält die Identität des neuen Gruppenmitglieds ID_{n+1} und dessen öffentlichen DH-Wert g^{m+1} . Nach der Entschlüsselung des vierten Teils und der Berechnung des zweiseitigen Geheimnisses mit dem neuen Partner verfügen alle Gruppenmitglieder wieder über die gleichen Informationen, d. h. sie sind bei einer Zuweisung des virtuellen Tokens in der Lage, den Gruppenschlüssel, wie beschrieben, zu erneuern.

3.4 Leave-Prozedur

Wenn ein Gruppenmitglied die Gruppe verlässt, informiert das Gruppenmanagement die verbleibenden Gruppenmitglieder darüber. Die Gruppenmitglieder bestimmen nach Formel (1) den neuen Tokenhalter und dieser startet die Erneuerung des Gruppenschlüssels. Abbildung 4 zeigt ein Beispiel.

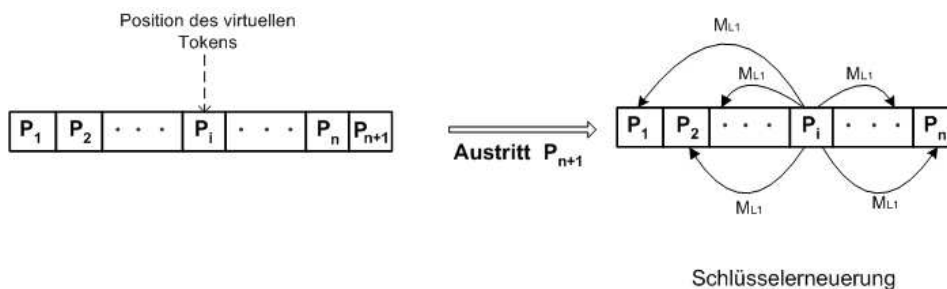


Abbildung 4: Austritt eines Gruppenmitglieds

Sie enthält zuerst wieder die Identität des Tokenhalters und eine Zufallszahl für die Erzeugung der Schlüssel K_{ij} ($j=1, 2, \dots, n$ mit $j \neq i$) für die separaten geheimen Kanäle nach (2) und (3). Beide sind mit dem alten Gruppenschlüssel verschlüsselt. Der neue Gruppenschlüssel GK_{new} und die aktuelle Schlüsselversion VK werden mit dem Schlüssel des jeweiligen Kanals verschlüsselt. Der ausscheidende Teilnehmer kann nicht in den Besitz des neuen Gruppenschlüssels gelangen, da er nicht in der Lage ist, die geheimen Kanäle ohne Kenntnis der zweiseitigen Geheimnisse $g^{r_{i1}}, g^{r_{i2}}, \dots, g^{r_{im}}$ zwischen dem Tokenhalter und den anderen Mitgliedern abzuleiten. Wenn die verbleibenden Gruppenmitglieder die Nachricht M_{L1} erhalten, können sie sie, wie das oben für die Nachricht M_{J5} beschrieben wurde, entschlüsseln.

3.5 Ausfall von Teilnehmern

In VTKD ist die Position des virtuellen Tokens jederzeit bekannt. Auch Änderungen der Position bei einer Veränderung der Gruppenzusammensetzung können genau bestimmt werden. Als Sonderfall muss jetzt noch der Ausfall eines Hosts einschließlich der des Tokenhalters betrachtet werden. Diese Änderung der Gruppenzusammensetzung wird der Sicherheitsschicht nicht vom Gruppenmanagement sondern durch das Gruppenmanagementprotokoll signalisiert, das den Ausfall erkennt. Vom Ablauf entspricht der Ausfall eines Hosts dem Austritt eines Gruppenmitglieds, d. h., nach Anzeige des Ausfalls verhalten sich die Gruppenmitglieder wie bei dem zuvor beschriebenen Austritt.

4 Sicherheitsanalyse von VTKD

Dieser Abschnitt diskutiert, wie die in der Einleitung formulierten Sicherheitsanforderungen von VTKD eingehalten werden.

Key Authentication: Der Zugang zum Gruppenschlüssel von außerhalb wird zum einen dadurch verhindert, dass jedes neue Gruppenmitglied vor dem Beitritt auf seine Identität überprüft wird. Nur wenn diese Überprüfung erfolgreich ist, erhält er den Gruppenschlüssel. Umgekehrt überprüft der Beitretende anhand der mit Nachricht M_{J_3} übermittelten Signatur, dass die übermittelten Identitäten und öffentlichen DH-Werte wirklich der beizutretenden Gruppe zuzuordnen sind. Die Schlüsselerneuerungsprozedur stellt durch die Nutzung geheimer Kanäle, die aus den zweiseitigen Geheimnissen der authentifizierten Mitglieder abgeleitet werden, sicher, dass der neue Schlüssel nur an die aktuelle Gruppe ausgeliefert werden kann.

Forward confidentiality: Die weiterführende Vertraulichkeit wird durch die Leave-Prozedur gesichert. Ein ausscheidendes Gruppenmitglied kann keinen Zugriff auf den neuen Gruppenschlüssel erlangen, da er aufgrund der fehlenden Kenntnis der zweiseitigen Geheimnisse und der neugenerierten Zufallszahl keinen Zugriff auf die geheimen Kanäle zwischen Tokenhalter und verbleibenden Mitgliedern erlangen kann, über die der neue Schlüssel verteilt wird.

Backward confidentiality: Rückwirkende Vertraulichkeit wird dadurch erreicht, dass der alte Gruppenschlüssel dem beitretenden Partner mit der Nachricht M_{J_5} nicht ausgeliefert wird. Die Teile von M_{J_5} , die mit dem neuen Schlüssel entschlüsselt werden können, enthalten den alten Schlüssel nicht.

Collusion freedom: Ein geheimes Einverständnis zwischen Partnern zur Aufdeckung des aktuellen Schlüssels wird dadurch vermieden, dass jeder neugenerierte Schlüssel nicht in Beziehung zu den vorangegangenen Schlüsseln steht, so dass diese Partner ihre alten Schlüssel nicht für eine Aufdeckung nutzen können.

Perfect Forward Secrecy: Eine dauerhafte Geheimhaltung einer abgeschlossenen Sitzung ist dann nicht gewährleistet, wenn ein langfristiges Credential kompromittiert wird oder es einem aktiven Angreifer gelingt, ältere Schlüssel aufzudecken. VTKD besitzt nur ein langfristiges Credential, den privaten RSA-Schlüssel, der während der Authentifizie-

rungsphase benutzt wird. Das RSA-Schlüsselpaar wird jedoch niemals für die Verschlüsselung des Gruppenschlüssels genutzt, so dass der Besitzer mit einem kompromittierten RSA-Schlüssel nicht an den Gruppenschlüssel gelangen kann. Der zweite Fall ist nur relevant, wenn das Treffen aus mehreren Sitzungen besteht, für die verschiedene Sitzungsschlüssel verwendet werden. Wir betrachten hier als Beispiel ein solches Treffen, das aus vier Sitzungen bestehen. Der Gruppenschlüssel wird für jede Sitzung erneuert. Damit wird jede Sitzung durch seine Sitzungsschlüssel und die zugehörigen Schlüsselmaterialien, wie sie in der folgenden Tabelle angegeben sind charakterisiert.

Schlüsselmaterialien	Session1	Session2	Session3	Session4
Gruppenschlüssel (GK)	GK ₁	GK ₂	GK ₃	GK ₄
Zeitweiliger geheimer Schlüssel K _{ij} zwischen P _i und P _j	K _{ij1}	K _{ij2}	K _{ij3}	K _{ij4}
Zufallszahl (N _i)	N _{i1}	N _{i2}	N _{i3}	N _{i4}
Gemein. Geheimnis von P _i u. P _j	g ^{ri·rj}	g ^{ri·rj}	g ^{ri·rj}	g ^{ri·rj}
Geheimer DH Wert von P _i	r _i	r _i	r _i	r _i

Resistance to known key attacks: Resistenz gegenüber Attacken mit bekannten Schlüssel bedeutet, dass ein aufgedeckter älterer Schlüssel nicht zur Kompromittierung des aktuellen Sitzungsschlüssels verwendet werden kann. Hier sind wieder zwei Fälle zu unterscheiden [2]. Der erste Fall betrachtet den passiven Angreifer, der die Kommunikation aufzeichnet und später analysiert. Wir nehmen an, dass der passive Angreifer die vorangegangenen Schlüssel und die Zufallszahl N für die Erzeugung des zeitweiligen geheimen Schlüssels K_{ij} zwischen zwei Partnern kennt. Das reicht aber nicht. Um den Schlüssel zu erzeugen, benötigt er das gemeinsame DH-Geheimnis der Partner. Diese werden jedoch niemals über die Verbindung übertragen. Der andere Fall betrifft den aktiven Angreifer, der versucht, die Daten auf der Verbindung zu ändern. Wir betrachten hier die Situation, dass der Tokenhalter P_i den Gruppenschlüssel durch Aussenden der Nachrichten M_{j5} oder M_{L1} erneuert. Weiterhin nehmen wir an, dass der aktive Angreifer irgendwie in den Besitz des alten Gruppenschlüssel GK_{old} gelangt ist, so dass er M_{j5} oder M_{L1} abfangen könnte und mit Hilfe von GK_{old} die Nachricht wie folgt ändern könnte:

P _i (Tokenhalter)	Aktiver Angreifer	Empfänger P ₁
HDR, GK _{old} {ID _i , N _i }, K _{i1} {VK, GK _{new} }...	HDR, GK _{old} {ID _i ', N _i '}, → K _{i1} {VK, GK _{new} }...	HDR, GK _{old} {ID _i ', N _i '}, → K _{i1} {VK, GK _{new} }...

Der Angreifer ersetzt ID_i und N_i durch eine andere Identität ID_i' und Zufallszahl N_i' . Die gefälschte Nachricht würde jedoch bei den Empfängern zur Generierung anderer zeitweiliger geheimer Schlüssel K_{ij}' führen, was jedoch durch das Authentifizieren des

Nachrichtenteils $\{VK, GK_{\text{new}}\}$ mittels inkorrektem Schlüssel K_{ij} aufgedeckt wird. Wenn also ein Angreifer mit einem älteren Schlüssel Teile der Nachricht verfälscht, dann erkennen die Gruppenmitglieder den Angriff.

5 Bewertung des Leistungsverhaltens

Zur Bewertung des Leistungsverhaltens von VTKD vergleichen wir es mit dem von Rodeh und anderen vorgeschlagenen Schlüsselverteilungsprotokoll [11] und dem effizientesten Schlüsselvereinbarungsprotokoll TKDH [8,9]. Für den Vergleich verwenden wir den Benchmark für kryptographische Algorithmen aus [26], der im Anhang zusammengefasst ist. Der Vergleich untergliedert sich in den Authentifizierungs- und den Schlüsselerneuerungsteil.

Authentifizierung

Authentifizierung ist nur in VTKD enthalten, nicht aber in den beiden Vergleichsprotokollen. Weshalb hier nur der Aufwand für VTKD angegeben werden kann. Der Aufwand für die Authentifizierung ergibt sich aus den Berechnungskosten für die vier Nachrichten $M_{j1} \sim M_{j4}$ und die zusätzliche Berechnung der zweiseitigen DH-Geheimnisse. Nach dem IKEv2-Standard umfassen die Berechnungskosten den Aufwand für die Berechnung von zwei RSA Signaturen, der Überprüfung von zwei Signaturen, vier symmetrischen kryptographischen Operationen und vier Hash-Abbildungen. Der zusätzliche Aufwand für die Berechnung der n DH-Geheimnisse geht dabei noch wesentlich stärker ein. Für Gruppen bis zu 100 Teilnehmern, wie sie für VTKD vorgesehen sind, ergibt sich mit dem Benchmark nach [26] eine Berechnungsdauer von 386 ms, was akzeptabel wäre. Genaugenommen kann diese Berechnung aber *“offline“* ausgeführt werden, da die Geheimnisse nicht während der Authentifizierungsphase benötigt werden, sondern erst bei der Schlüsselerneuerung für die geheimen Kanäle.

Erneuerung des Gruppenschlüssels

Ein breit akzeptiertes Kriterium für die Einschätzung der Effizienz von Schlüsselaustauschprotokollen ist die Zeit zwischen dem Auslösen einer Schlüsselerneuerung und der Verfügbarkeit des neuen Schlüssels bei allen Teilnehmern. Diese Verzögerung wird hauptsächlich durch die Kommunikations- und Berechnungsaufwand bestimmt. Wir vergleichen diese beiden Aspekte für die drei betrachteten Protokolle.

Kommunikationsaufwand

Der Kommunikationsaufwand bezieht sich auf die Zahl der Kommunikationsrunden und die Größe der Nachrichten. Sie sind für die drei Protokolle in Tabelle 1 zusammengefasst. Für VTKD und TGDH ist der Kommunikationsaufwand gering, da für das Beitreten und Verlassen jeweils nur eine Multicast-Nachricht ausgesendet wird. Rodeh's Protokoll benötigt mehrere Kommunikationsrunden für das Verlassen der Gruppe. Bezüglich der Nachrichtengröße ist das Protokoll von Rodeh im Vorteil. Es versendet nur kleine Nachrichten, während die Nachrichtengrößen bei VTKD für 100 Mitglieder 4 Kbyte und für TGDH sogar 25 Kbyte betragen. Jedoch ist das kein allzu großes Problem, da 25 Kbyte problemlos in einem UDP-Paket untergebracht werden können.

Tabelle 1: Kommunikationsaufwand für die Schlüsselerneuerung

Proto- koll	Opera- tion	Kommunikationsaufwand				
		Kommunika- tionsrunden	Multi- cast	Größe der Multi- cast-Nachricht	Uni- cast	Größe der Uni- cast-Nachricht
Rodeh	Beitritt	2	2	$\log_2 n$ ¹⁾ symmetri- sche Schlüssel ²⁾	1	1 symmetrischer Schlüssel
	Austritt	$\log_2 n$	$\log_2 n$	$\log_2 n$ symmetrische Schlüssel	$\log_2 n$	1 symmetrischer Schlüssel
TGDH	Beitritt ⁴⁾	1	1	2n asymmetrische Schlüssel ³⁾	1	1 asymmetrischer Schlüssel
	Austritt	1	1	$\log_2 n$ asymmetrische Schlüssel	-	-
VTKD	Beitritt	1	1	n symmetrische Schlüssel + 1 a- symmetrischer Schlüssel	-	-
	Austritt	1	1	n symmetrische Schlüssel	-	-

Legend: 1) n ist die Zahl der Gruppenmitglieder.

2) Die typische Größe eines symmetrischen Schlüssels ist 128 bit=16 bytes.

3) Die typische Größe eines asymmetrischen Schlüssels ist 1024 bit=128 bytes.

4) Der Beitritt in TGDH erfordert zwei Kommunikationsrunden, aber nur eine Kommunikationsrunde dient der Schlüsselerneuerung.

Berechnungsaufwand

Tabelle 2 enthält den Berechnungsaufwand der Protokolle durch Angabe der unterschiedlichen kryptographischen Operationen, die sie verwenden. Der Vergleich zeigt,

dass die Protokolle asymmetrische und symmetrische Operationen unterschiedlich einsetzen. Während TGDH auf der einen Seite intensiv asymmetrische Berechnungen durchführt, nutzt VKTD vorrangig symmetrische Operationen. Rodeh's Protokoll liegt dazwischen. Da bekanntlich asymmetrische kryptographische Berechnungen wesentlich langsamer sind als symmetrische, ist der gesamte Berechnungsaufwand von VKTD geringer als der der anderen Protokolle.

Tabelle 2: Berechnungsaufwand für die Schlüsselerneuerung

Protokoll	Operation	Mitglieder	Berechnungsaufwand				
			DH-Geheimnisse ⁴⁾	RSA-Signatur ⁵⁾	RSA-Verifikation ⁵⁾	Hash und sym. Verschlüsselung	Hash und sym. Entschlüsselung
Rodeh	Join	Leiter des Baums	1	-	-	2	-
		Neues Mitglied	1	-	-	-	1
		Teilnehmer	-	-	-	-	1
	Leave	Leiter des Baums	$\log_2 n^1)$	-	-	$\log_2 n$	-
		Leiter des Unterbaums	1	-	-	-	1
		Teilnehmer	-	-	-	-	1
TGDH ²⁾	Beitritt	Sponsor	$2\log_2 n$	1	-	-	-
		Neues Mitglied	$2\log_2 n$	-	1	-	-
		Teilnehmer	$1 \dots 2\log_2 n$	-	1	-	-
	Aus-	Sponsor	$2\log_2 n$	1	-	-	-
		Teilnehmer	$1 \dots$		1	-	-

	tritt		$2\log_2 n$				
VTKD ³)	Bei- tritt	Tokenbesitzer	-	-	-	n+3	-
		Neues Mitglied	-	-	-	-	2
		Teilnehmer	1	-	-	-	3
	Aus- tritt	Tokenbesitzer	-	-	-	n+2	-
		Teilnehmer	-	-	-	-	2

Legend: 1) n ist die Zahl der Gruppenmitglieder.

2) Wir betrachten den besten Fall von einem ausgeglichenen Schlüsselbaum für TGDH hier. Der schlechteste Fall erfordert n DH- Geheimnisberechnungen.

3) Hier wird der Berechnungsaufwand für die Gruppenschlüsselerneuerung dargestellt, der bei der Erstellung der Nachrichten M_{J_5} und M_{L_1} in VTKD auftritt.

4) Eine DH- Geheimnisberechnung bedeutet eine Exponentialberechnung.

5) RSA-Signatur in TGDH wird für die Nachrichtenauthentisierung genutzt [9].

Verzögerung der Schlüsselerneuerung

Mit der Tabellen 1 und 2 kann nun die Verzögerung der Schlüsselerneuerung D_{gkr} wie folgt bestimmt werden:

$$D_{gkr} = D_{cs} + D_{gc} + D_{cr} \quad (4)$$

wobei D_{cs} und D_{cr} die kryptographische Berechnungsverzögerung des Senders bzw. Empfängers und D_{gc} die Kommunikationsverzögerung bezeichnen. Für den Vergleich nehmen wir weiter an, dass alle Protokolle über dem gleichen Gruppenkommunikationsprotokoll laufen, das eine Verzögerung von 20 ms für jede Kommunikationsrunde erzeugt. Das ist eine typische Verzögerung in mittelgroßen Netzen, wie dem G-WiN, in dem wir eine maximale Umlaufzeit von 40 ms zu beliebigen Knoten gemessen haben. Die Berechnungsverzögerung für den Sender und Empfänger wurde wieder unter Verwendung des kryptographischen Benchmarks aus [26] bestimmt. Die sich insgesamt ergebenden Verzögerungen für die Schlüsselerneuerung für den Beitritt zur Gruppe und das Verlassen sind in Abbildung 5 dargestellt. VTKD erweist sich dabei effizienter als die anderen beiden Protokolle. Der Grund dafür liegt darin begründet, dass VTKD weniger Kommunikationsrunden benötigt als die anderen beiden und überwiegend symmetrische Verschlüsselungsoperationen nutzt.

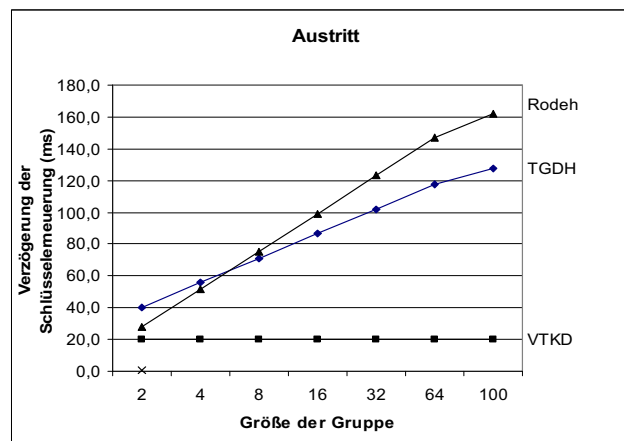
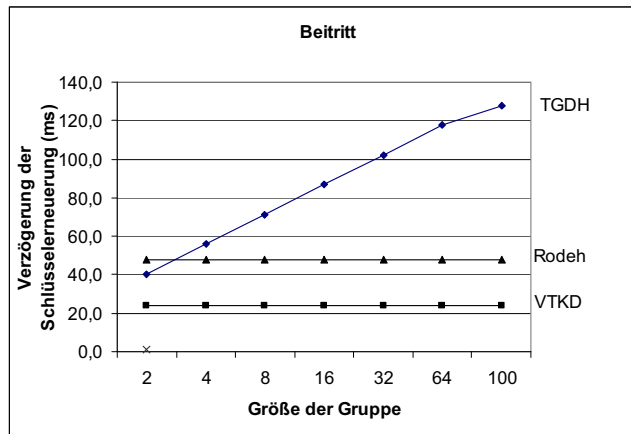


Abbildung 5: Verzögerung der Schlüsselerneuerung

6 Schlussbemerkungen

Die effiziente und sichere Schlüsselerneuerung bildet die Grundlage für die vertrauliche Kommunikation in geschlossenen dynamischen Gruppen. Solche Einsatzfälle sind insbesondere im geschäftlichen Bereich gegeben, wo zunehmend Verhandlungen und Beratungen auch über das Internet abgewickelt werden. Für zentralistische Ansätze mit einem Gruppenserver existieren praktikable Ansätze. Mit der steigenden Nutzung der Mobilkommunikation sind zunehmend Lösungen gefragt, die auf einen Gruppenserver verzichten und eine Peer-to-Peer-Kommunikation der Partner unterstützen. Damit werden vor allem auch Ad hoc-Meetings unterstützt. Die eingesetzten Verfahren müssen effizient sein, da neben der Schlüsselverteilung und Audio/Videoverschlüsselung mit der

Komprimierung und Dekomprimierung der Mediendaten auch andere zeit- und ressourcenintensive Prozesse auf den Endsystemen laufen. Die Skalierung des Protokolls ist für solche Anwendungen weniger das Problem sondern die Effizienz und Sicherheit des Verfahrens.

In dem vorliegenden Beitrag haben wir ein neues Schlüsselverteilungsprotokoll vorgestellt, das ein einfaches und intuitives Prinzip nutzt. VTKD erweist sich für den angestrebten Einsatz für kleine dynamische Peer-Gruppen effizienter als existierende Protokolle. Die spezifizierte Obergrenze von 100 Teilnehmern ist sehr großzügig ausgelegt. In der Regel sind für vertrauliche Treffen viel weniger Teilnehmer üblich. Grundlage für das vorgestellte Prinzip ist die Verwendung eines Gruppenkommunikationsprotokolls mit virtueller Synchronisation. Solche Protokolle existieren in der Zwischenzeit. VTKD wird gegenwärtig in das kommerziell verfügbare Peer-to-Peer-Videokonferenzsystem BRAVIS [19] zur Unterstützung vertraulicher Konferenzen integriert.

Literaturverzeichnis

- [1] J. Snoeyink, S. Suir, and G. Vorghese: A Lower Bound for Multicast Key Distribution. IEEE INFOCOM 2001, pp. 422-431.
- [2] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone: Handbook of applied cryptography, CRC Press series on discrete mathematics and its applications. CRC Press, 1997.
- [3] P. McDaniel, A. Prakash, and P. Honeyman: Antigone: A Flexible Framework for Secure Group Communication“, CITI Technical Report 99-2, University of Michigan, September 8, 1999.
- [4] P. S. Krus: A survey of multicast security issues and architectures. 21st National Information Systems Security Conference (NISSC), Oct. 1998. <http://csrc.nist.gov/nissc/1998/proceedings/paperF10.pdf>.
- [5] The BRAVIS video conference system (research variant). <http://www.bravis.tu-cottbus.de>
- [6] E. Rescorla: Diffie-Hellman Key Agreement Method. RFC 2631, June 1999.
- [7] M. Steiner, G. Tsudik, and M. Waidner: CLIQUES: A new approach to group key agreement. IEEE International Conference on Distributed Computing Systems, 1998, pp. 380-397.
- [8] Y. Kim, A. Perrig, and G. Tsudik: Simple and fault-tolerant key agreement for dynamic collaborative groups. In S. Jajodia (ed.): 7th ACM Conference on Computer and Communications Security, Athens, Greece, Nov. 2000, ACM Press, pp. 235–244.
- [9] Y. Kim, A. Perrig, and G. Tsudik: Tree-based Group Key Agreement. <http://www-users.cs.umn.edu/~kyd/paper/kpt02.pdf>.
- [10] L. Dondeti, S. Mukherjee, and A. Samal: Disec: A distributed framework for scalable secure many-to-many communication. In: Proceedings of the Fifth IEEE Symposium on Computers and Communications (ISCC 2000), July 2000
- [11] O. Rodeh, K. P. Birman, D. Dolev: Optimized Group Rekey for Group Communication Systems. In Symposium Network and Distributed System Security (NDSS), San Diego, California, Feb. 2000, pp. 39-48.
- [12] C. Wong, M. Gouda, and S. Lam: Secure group communication using key graphs. IEEE/ACM Transactions on Networking 8 (2000)1: 16-30.
- [13] G. Ateniese, M. Steiner, and G. Tsudik: New Multiparty Authentication Services and Key Agreement Protocols. IEEE Journal Selected Areas in Communication 18 (2000) 4.

- [14] O. Pereira and J. J. Quisquater: A Security Analysis of Cliques Protocol Suites. Proceedings of the 16th International Conference on Information Security: Trusted information: the new decade challenge. Paris, France. 2001, pp. 151-166.
- [15] G. V. Chockler, I. Keidar, and R. Vitenberg: Group communication specifications: A comprehensive study. ACM Computing Surveys, 4 (December 2001), 427-469.
- [16] B. Whetten, T. Montgomery, and S. Kaplan: A High Performance Totally Ordered Multicast Protocol. In Theory and Practice in Distributed Systems, International Workshop, LNCS 938, Springer, 1994. pp. 33-57.
- [17] D. A. Agarwal: Totem: A Reliable Ordered Delivery Protocol for Interconnected Local Area Networks, Ph.D Thesis, University of Santa Barbara, Dec 1994.
- [18] E. C. Popovici, R. Mahlo, M. Zuehlke, and H. Koenig: Consistency Support for a Decentralized Management in Closed Multiparty Conferences Using SIP. In Proc. of the 11th IEEE International Conference on Networks (ICON 2003), Sydney, Australia, IEEE Press, 2003, pp. 295 - 300
- [19] BRAVIS Videokonferenzen (Produkt-Variante). <http://www.bravis.eu>
- [20] H. Krawczyk, M. Bellare, and R. Canetti: HMAC: Keyed-Hashing for Message Authentication, RFC 2104, February 1997.
- [21] ITU-T Recommendation X.509 | ISO/IEC 9594-8: Public Key and Attribute Certificate Frameworks.
- [22] D. Harkins and D. Carrel: The Internet Key Exchange (IKE), RFC 2409, Nov. 1998.
- [23] W. Aiello, S. M Bellovin, M. Blaze R. Canetti, J. Ioannidis, A. D. Keromytis, and O. Reingold: Just Fast Keying (JFK). draft-ietf-ipsec-jfk-04.txt. July 2002.
- [24] C. Kaufman, Internet Key Exchange (IKEv2) Protocol, draft-ietf-ipsec-ikev2-07.txt, April 2003.
- [25] R. Housley, W. Ford, W. Polk, and D. Solo: Internet X.509 Public Key Infrastructure Certificate and CRL Profile. RFC 2459, Jan.1999.
- [26] Crypto++ 5.2.1 Benchmarks. <http://www.eskimo.com/~weidai/benchmarks.html>

Anhang

Verwendeter Benchmark für die kryptographischen Operationen

Die folgende Tabelle enthält die Geschwindigkeiten für die Ausführung der kryptographischen Operationen nach [26], die zur Bestimmung des Berechnungsaufwands im Abschnitt 5 verwendet wurden. Diese wurden auf einen Pentium 4-Prozessor mit 2.1 GHz unter Windows XP gewonnen.

Algorithmen	Symmetrische kryptographische Operationen		Asymmetrische kryptographische Operationen		
	SHA-1	AES (128-bit Schlüssel)	RSA 1024 Signatur	RSA 1024 Verifikation	DH 1024 Geheimnisberechnung
Geschwindigkeit	68 Mbyte/s	61 Mbyte/s	4.75 ms/Operation	0.18 ms/Operation	3.86 ms/Operation