Attributbasierter Schlüsselaustausch in virtuellen privaten Netzen

Thomas Koellmer · Michael Rossberg · Guenter Schaefer
Technische Universität Ilmenau
{vorname.nachname}@tu-ilmenau.de

Zusammenfassung

Virtuelle Private Netzwerke (VPN) erlauben eine vertrauliche Datenübertragung über nicht vertrauenswürdige Verbindungen. Dabei werden zusätzlich Integrität, Authentizität und Zugriffskontrolle gesichert. Um auch mit Sabotageangriffen umgehen zu können, werden zukünftige VPN dynamisch auf Veränderungen im Transportnetz reagieren können. Problematisch dabei ist, dass bei solch einem dynamischen Verbindungsaufbau die Identitäten von Knoten eher offenbart werden und so möglicherweise gezielte Angriffe ermöglicht werden. Dieser Artikel stellt daher einen Mechanismus vor, der die Schlüsselaushandlung an Attribute bindet. Dies geschieht dezentral und ohne dass ein Knoten diese Attribute veröffentlichen muss. Die letzten beiden Punkte werden durch den Einsatz von *Bilinear Pairings* auf elliptischen Kurven erreicht.

1 Einleitung und Motivation

Virtuelle private Netzwerke (VPN) ermöglichen eine Datenübertragung innerhalb geschlossener Benutzergruppen und sichern dabei die Vertraulichkeit, die Integrität und die Authentizität der ausgetauschten Daten zu. Ein in diesem Kontext in der Regel nicht abgesichertes Ziel ist die Verfügbarkeit des Systems. Ein *Denial-of-Service-Angriff* (DoS) stellt oft eine einfach durchzuführende und überdies kostengünstige Möglichkeit dar, Konkurrenten und Behörden empfindlich zu treffen (e.g. [Broe10]). Um dem entgegenzuwirken und auf Angriffe reagieren zu können werden zukünftige VPN vermehrt flexible Strukturen und dynamische Verbindungen einsetzen. Dies ermöglicht es, die Abhängigkeit des Netzwerkes von einem angegriffenen Knoten zu verringern, als auch ihn nach dem Angriff wieder in das VPN zu reintegrieren.

Das eigentliche Problem wird dadurch jedoch nicht gelöst: Durch das Abhören (verschlüsselter) Daten und Verbindungsanfragen kann auf eine Netzwerkstruktur und so auch auf potenziell wichtige Knoten geschlossen werden. Hat ein Angreifer einen solchen identifiziert, kann er ihn anschließend mittels DoS-Angriffen stören. Ziel muss es sein, diese Identifikation für den Angreifer zu erschweren indem beobachtbare Datenflüsse innerhalb des VPN reguliert werden: Potenziell wichtige Kernknoten sollen nicht direkt mit weniger wichtigen Knoten kommunizieren dürfen, beispielsweise mit mobilen Außendienstmitarbeitern. Diese haben im Allgemeinen eine höhere Wahrscheinlichkeit der Kompromittierung, und durch die Isolation wird so insgesamt eine höhere Verfügbarkeit realisiert.

Aus diesem Grund schlagen die Autoren von [BrRS09] vor, VPN in verschiedene Sicherheitszonen einzuteilen, und direkte Sicherheitsbeziehungen nur noch in gleichen oder benachbarten Klassen zu ermöglichen. Unter der Bedingung, dass die Sicherheitszone jedes einzelnen Knotens geheim ist, sinken die Möglichkeiten für einen Angreifer, auf die Topologie des VPN zu schließen. Das VPN soll dabei nicht komplett separiert werden, dafür würde die Einrichtung

P. Schartner · J. Taeger (Hrsg.) · D•A•CH Security 2011 · syssec (2011) 1-11.

verschiedener VPN ausreichen, vielmehr können wichtige und unwichtige Knoten weiterhin indirekt kommunizieren, indem andere Knoten eine Weiterleitung von Paketen übernehmen. Im Falle eines DoS-Angriffes würden somit maximal die direkt benachbarten Zonen betroffen sein.

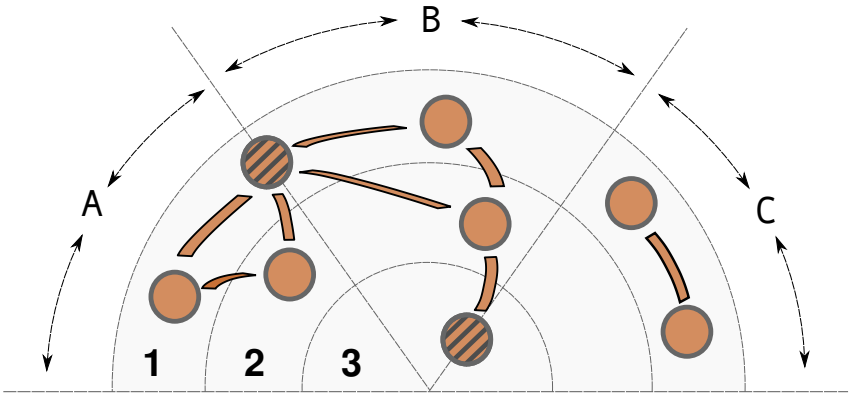


Abb. 1: Kommunikation zwischen Knoten kompatibler Sicherheitszonen

Die etablierten Verfahren von IPsec und SSL (Secure Socket Layer) können die beschriebene Flexibilität nicht leisten, da die Sicherheitsklassifikation eines Knotens nicht einfach in einem Zertifikat veröffentlicht werden kann. Andernfalls könnten Angreifer ohne viel Aufwand systematisch nach sensiblen Systemen suchen, da sie im Falle eines geplanten DoS-Angriffes nicht auf einen ausgehandelten Schlüssel angewiesen sind.

Anlehnend an die Multi Level Security (MLS) aus dem Betriebssystemumfeld, ist eine Sicherheitszone ein Tupel aus einem Sicherheitslevel und einem Compartment. Ausschließlich Knoten innerhalb des selben Compartments und in der gleichen oder benachbarten Schicht sind in der Lage, einen gemeinsamen Schlüssel auszuhandeln. Zonen, zwischen denen Verbindungen erlaubt sein sollen, werden im Folgenden auch als *kompatibel* bezeichnet. Hier werden auch die Unterschiede zum strengen MLS-Begriff deutlich: Die Relation auf erlaubten Verbindungen ist symmetrisch und intransitiv. Außerdem entfällt die Trennung zwischen Subjekten und Objekten. Abbildung 1 zeigt ein Beispiel mit jeweils drei Sicherheitsstufen und Compartments, eingezeichnet sind die erlaubten Verbindungen. Die gestreiften Knoten sind als Mitglied zweier Sicherheitszonen jeweils in zwei Compartments vertreten. Die Möglichkeit, einem Knoten mehrere Sicherheitsklassen zuzuordnen ist zwingend notwendig, sollen mittels des oben vorgestellten Systems beliebige Netzwerktopologien nachgebildet und mit einer praktikablen Anzahl von Compartments und Schichten abgebildet werden können.

Ergebnis dieser Arbeit ist ein Authentisierungsverfahren, beziehungsweise Schlüsselaustauschprotokoll, welches ausschließlich dann einen gemeinsamen Schlüssel aushandelt, falls die Sicherheitszonen der zwei beteiligten Knoten kompatibel sind. Ist dem nicht der Fall, lernen beide Knoten nichts (außer der Inkompatibilität) über die Zonenzugehörigkeit des Gegenübers. Insbesondere erfahren sie auch nicht, ob sie überhaupt zum selben VPN gehören. Dies wird erreicht, ohne bei jeder Authentisierung eine Trusted Third Party (TTP) einzuschalten. Da der Schlüsselaustausch abhängig von einem gegenseitig nicht veröffentlichtem Geheimnis der Teilnehmer gemacht wird, gehört der vorgestellte Mechanismus zur Gruppe der Zero-Knowledge-Protokolle.

Der Artikel ist im Folgenden in drei Abschnitte gegliedert: Zunächst wird auf grundlegende

Anforderungen eines geheimen Schlüsselaustausches eingegangen, bevor anschließend das vorgeschlagene Verfahren hergeleitet wird. Den Abschluss bilden eine kurze Zusammenfassung sowie ein Ausblick.

2 Anforderungen an den Schlüsselaustausch

Ein dem Szenario entsprechendes Protokoll soll drei Hauptanforderungen erfüllen:

1. Das sichere Ableiten eines paarweisen Sitzungsschlüssels
2. Sind die Sicherheitszonen der beteiligten Knoten nicht kompatibel, schlägt der Schlüsselaustausch fehl, ohne dass ein Teilnehmer seine Sicherheitszone preisgibt.
3. Der Schlüsselaustausch erfolgt aufgrund der Verfügbarkeitsanforderungen ohne eine (vertrauenswürdige) dritte Partei.

Zusätzlich soll es praktisch möglich sein, ihn an die Stelle eines herkömmlichen Diffie-Hellman Schlüsselaustauschs [DiHe76] zu setzen und die Anzahl ausgetauschter Nachrichten möglichst klein zu halten.

Die dritte Anforderung bezüglich der Involvierung einer zentralen dritten Instanz bezieht sich lediglich auf die Zeit während des Handshakes. Globale Parameter, wie z.B. Details über die Schlüssellängen und der benutzen algebraischen Gruppen müssen – wie beim Diffie-Hellmann Schlüsselaustausch auch – bereits vorher festgelegt werden.

Ferner soll es möglich sein, dass ein Knoten Mitglied mehrerer Sicherheitszonen ist um somit trotz der Einschränkungen durch das Klassenkonzept beliebige Netzwerktopologien nachzubilden. Dies sollte aber aus Gründen der Effizienz nicht dazu führen, dass die Länge des Schlüsselaustauschs proportional steigt.

3 Beschreibung des Schlüsselaustauschverfahrens

Die Grundlage des hier vorgestellten Verfahrens bietet der *Secret Handshake* von Sorniotti und Molva [SoMo09]. Dieser leistet zunächst folgendes: Jeder Teilnehmer bekommt von einer TTP ein sogenanntes *Credential*, welches den Knoten als Besitzer einer Eigenschaft ausweist. Auf der anderen Seite existieren sogenannte *Matching References*, welche genau auf ein Credential passen und es verifizieren können. Für eine gegenseitige Überprüfung müssen beide Knoten jeweils ein Credential versenden beziehungsweise empfangen. Eigentliches Ziel des Secret Handshakes ist jedoch nicht die alleinige Feststellung, dass zwei Knoten eine Eigenschaft teilen (*Secret Matchmaking*), sondern darauf aufbauend einen gemeinsamen Schlüssel zu errechnen. Die Autoren schlagen auch eine Lösung vor, die genau dann einen gemeinsamen Schlüssel zwischen den Teilnehmern etabliert, wenn die Attribute übereinstimmen. Allerdings geht dabei die Information verloren, ob das Matching erfolgreich war, und die Teilnehmer müssen anschließend testen, ob die errechneten Schlüssel identisch sind.

Die mathematischen Details können hier aus Platzgründen nur stark verkürzt wiedergegeben werden: Die Zero Knowledge-Eigenschaften des Secret Handshakes werden mithilfe so genannter *Bilinear Pairings* erreicht. Es handelt sich hierbei um eine Konstruktion auf elliptischen Kurven, die durch die Identity Based Encryption von Boneh und Franklin ([BoFr01]) als kryptographisches Werkzeug etabliert wurde.

Mit der Bilinearität kann folgende Eigenschaft der Abbildung $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_3$ abgeleitet

werden ($P \in \mathbb{G}_1, Q \in \mathbb{G}_2; a, b \in \mathbb{F}$):

$$\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$$

Damit gilt:

$$\hat{e}(aP, bQ) = \hat{e}(abP, Q) = \hat{e}(P, abQ) = \hat{e}(aP, Q)^b = \hat{e}(P, bQ)^a = \hat{e}(P, Q)^{ab}.$$

Zusätzlich können die Inverse der Koeffizienten verwendet werden:

$$\hat{e}(aP, bQ)^{a^{-1}} = \hat{e}(P, bQ) = \hat{e}(abP, Q)^{a^{-1}} = \ldots = \hat{e}(P, Q)^b$$

Für Details zur eigentlichen Berechnung des Pairings sei auf [Joux09] verwiesen, insbesondere auf *Millers* Algorithmus, der die effiziente Berechnung des Pairings erst möglich macht. Trotzdem ist die Auswertung eines Pairings im Vergleich zu „normalen“ Operationen auf elliptischen Kurven sehr aufwendig. Die kryptographische Relevanz entsteht durch die Vermutung, dass das Bilinear Diffie Hellman Problem (BDH) nicht effizient zu lösen ist: Selbst mit gegebenen P, aP, bP, cP lässt sich nicht leicht auf $\hat{e}(P, P)^{abc}$ schließen oder im Entscheidungsfall (Bilinear Decisional Diffie Hellman Problem, BDDH) auch nur mit einer Wahrscheinlichkeit größer $0{,}5$ entscheiden, ob die Werte derart zusammenhängen.

Zusätzlich zur Ausnutzung der Bilinearität spielt der diskrete Logarithmus auf elliptischen Kurven eine wichtige Rolle zur Realisierung des Schlüsselaustauschs. Das *Elliptic Curve Discrete Logarithm Problem* (ECDLP) ist wie folgt definiert: Seien P und Q zwei Punkte auf einer elliptischen Kurve, dann besteht das ECDLP darin, eine Zahl n zu finden, sodass $Q = nP$ gilt. Im Vergleich zum „klassischen“ diskreten Logarithmus auf einem finiten Feld ist dies zwar eine Multiplikation, die grundlegende Addition von Punkten auf einer elliptischen Kurve ist allerdings derart komplex, dass dies keine Rückschlüsse auf die zu erwartende Schwere des Problems zulässt. Für ausgewählte Kurven und Bitlängen ist das ECDLP als praktisch sicher anzusehen.

Der im VPN eingesetzte Secret Handshake benutzt nun die Eigenschaften dieser Pairings, um das Folgende zu erreichen: Zu jedem zu beweisenden Wert (*Credential*) gibt es ein zugehöriges Token (*Matching Reference*). Das Credential bleibt hierbei immer Geheimnis des besitzenden Knotens. Nur eine durch den diskreten Logarithmus „geschützte“ Version wird an einen anderen Knoten geschickt. Dieser ist nun in der Lage, mit seiner Matching Reference die mögliche Kompatibilität zu überprüfen. Jede Sicherheitszone besteht aus einer von der TTP *im Voraus* festgelegten Konkatenation zweier Zufallszahlen, die als „Koordinate“ von Compartment und Layer die Sicherheitszone repräsentieren. Ein einzelner Knoten bekommt beispielsweise das Credential für seine Klassenzugehörigkeit, sowie drei Matching References: das der eigenen und der beiden benachbarten Stufen. Dies funktioniert, da es nicht möglich ist, aus der Matching Reference auf das eigentliche Credential zu schließen (diskreter Logarithmus). Durch geschickte Nutzung des Pairings kann aber festgestellt werden, ob Credential und Matching Reference zusammenpassen.

Der ursprüngliche Schlüsselaustausch soll nun genauer beschrieben werden, folgende Notation wird dabei genutzt:

Geheime Parameter	
r, s, t, v	$\in \mathbb{Z}_q$ – Geheimnisse der TTP
x_u	$\in \mathbb{Z}_q$ – geheime Nutzeridentifikation
$r_{1_U}, r_{2_U}, r_{3_U}$	Nonces des Knotens U
n	Nonce für einen Schlüsselaustausch
Öffentliche Parameter	
$E(\mathbb{F}_q)$	Elliptische Kurve
P	Generator von $\mathbb{G}_1$
$\tilde{P}$	rP
S	sP
T	tP
V	vrP
$\mathbb{G}_1$	Additive Gruppe über E
$\mathbb{G}_2$	Multiplikative Gruppe über $\mathbb{F}_{q^2}$
$\hat{e}_\ell : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$	Bilineares Pairing
$\mathcal{P}$	Eigenschaftsmenge
$H : \mathcal{P} \rightarrow \mathbb{G}_1$	Kryptographische Hash-Funktion

Die TTP generiert nun für alle Benutzer u des Systems Matching References über die Eigenschaft $p \in \mathcal{P}$ mit folgendem Aufbau:

$$match_{u,p} = t^{-1}r(cred_p + x_u P)$$

Das Credential ist durch die Multiplikation mit r geschützt und kann so nicht aus einer Matching Reference abgeleitet werden. Es hat folgende Form:

$$cred_p = vH(p)$$

$H(p)$ ist eine kryptographische Hash-Funktion, die von einem bestimmten Attribut, in diesem Fall der Klassenzugehörigkeit, auf einen Punkt in $\mathbb{G}_1$ abbildet. Diese Werte werden nur in Verbindung mit v von der TTP veröffentlicht, können also auch nur von ihr generiert werden. Allerdings kann ein Benutzer überprüfen, ob die Werte die er von der TTP bekommen hat wirklich seinen Eigenschaften entsprechen:

$$\hat{e}(cred_p, \tilde{P}) \stackrel{?}{=} \hat{e}(H(p), V)$$

In ähnlicher Art und Weise kann die Gültigkeit der empfangenen Matching References von jedem Benutzer überprüft werden:

$$\hat{e}(match_{u,p}, T) \stackrel{?}{=} \hat{e}(H(p), V) \cdot \hat{e}(X_u, S)$$

X_u ist hierbei die Benutzeridentifikation, die jeder Nutzer zusätzlich von der TTP empfangen hat: $X_u = x_u s^{-1} rP$.

Nachrichtenaustausch

Der grundlegende Austausch zwischen zwei Knoten wird in [SoMo09] derart vorgeschlagen:

$\mathcal{B} \longrightarrow \mathcal{A} \quad \langle N_1 \mid N_2 \rangle = \langle nP \mid n\tilde{P} \rangle$
$\mathcal{A} \longrightarrow \mathcal{B} \quad \langle r_1 cred_{p_A} \mid r_2 N_2 \mid r_1 r_2 S \mid r_1 r_2 T \rangle$

$\mathcal{B}$ ist nun im Besitz des (geschützen) Credentials von $\mathcal{A}$. Für den Fall, dass das Credential und ein Matching Reference von $\mathcal{A}$ übereinstimmen, gilt folgende Gleichheit:

$$\hat{e}(r_1 cred_{p_A}, r_2 N_2)^{n^{-1}} \cdot \hat{e}(r_1 r_2 S, X_B) \stackrel{?}{=} \hat{e}(match_{B,p_B}, r_1 r_2 T)$$

Falls diese Bedingung erfüllt ist, beziehen sich Credential und Matching Reference auf die gleiche Eigenschaft p. Dies liegt in der Bilinearität des Pairings begründet:

$$\begin{aligned}
\hat{e}(r_1 cred_{p_A}, r_2 N_2)^{n^{-1}} \cdot \hat{e}(r_1 r_2 S, X_B) &= \hat{e}(r_1 vH(p), r_2 n\tilde{P})^{n^{-1}} \cdot \hat{e}(r_1 r_2 sP, x_B s^{-1} rP) \\
&= \hat{e}(r_1 vH(p), r_2 rP) \cdot \hat{e}(r_1 r_2 P, x_B rP) \\
&= \hat{e}(rr_1 r_2 vH(p), P) \cdot \hat{e}(rr_1 r_2 P, x_B P) \qquad (*)
\end{aligned}$$

$$\begin{aligned}
\hat{e}(match_{B,p_B}, r_1 r_2 T) &= \hat{e}(t^{-1} r(cred_p + x_B P), r_1 r_2 tP) \\
&= \hat{e}(r(vH(p) + x_B P), r_1 r_2 P) \\
&= \hat{e}(vH(p) + x_B P, rr_1 r_2 P) \\
&= \hat{e}(vH(p), rr_1 r_2 P) \cdot \hat{e}(x_B P, rr_1 r_2 P) \\
&= \hat{e}(rr_1 r_2 vH(p), P) \cdot \hat{e}(rr_1 r_2 P, x_B P) \qquad (**)
\end{aligned}$$

Dies bedeutet, dass $(*)$ und $(**)$ genau für den Fall gleich sind bei dem Credential und Matching Reference zusammenpassen. Dies erlaubt es nun auf ein Matching zu überprüfen. Um gleichzeitig einen Schlüssel zu errechnen wird das übersandte Credential mit einem zusätzlichen Geheimnis ergänzt: $r_1 cred_p$ wird zu $r_1(cred_p + r_3 P)$.

Berechnet man nun

$$\frac{\hat{e}(r_1(cred_{p_A} + r_3 P), r_2 N_2)^{n^{-1}} \cdot \hat{e}(r_1 r_2 S, X_B)}{\hat{e}(match_{B,p_B}, r_1 r_2 T)},$$

ergibt sich in ähnlicher Rechnung wie oben:

$$\frac{\hat{e}(r_1(cred_{p_A} + r_3 P), r_2 N_2)^{n^{-1}} \cdot \hat{e}(r_1 r_2 S, X_B)}{\hat{e}(match_{B,p_B}, r_1 r_2 T)}$$

$$= \frac{\hat{e}(rr_1 r_2 vH(p), P) \cdot \hat{e}(r_1 r_2 r_3 rP, P) \cdot \hat{e}(rr_1 r_2 P, x_B P)}{\hat{e}(rr_1 r_2 vH(p), P) \cdot \hat{e}(rr_1 r_2 P, x_B P)}$$

$$= \hat{e}(r_1 r_2 r_3 rP, P) = \hat{e}(P, P)^{r_1 r_2 r_3 r}.$$

Damit bekommt $\mathcal{A}$ den Wert von $\hat{e}(r_{1_B} r_{2_B} r_{3_B} rP, P)$ und $\mathcal{B}$ den von $\hat{e}(r_{1_A} r_{2_A} r_{3_A} rP, P)$. Somit müssen beide Teilnehmer diesen Wert nur mit ihren eigenen Geheimnissen potenzieren und kommen so im Falle passender Attribute zu dem gleichen Schlüssel:

$$\begin{aligned} K = \hat{e}(r_{1_B} r_{2_B} r_{3_B} rP, P)^{r_{1_A} r_{2_A} r_{3_A}} &= \hat{e}(r_{1_A} r_{2_A} r_{3_A} rP, P)^{r_{1_B} r_{2_B} r_{3_B}} \\ &= \hat{e}(r_{1_B} r_{2_B} r_{3_B} r_{1_A} r_{2_A} r_{3_A} rP, P). \end{aligned}$$

Hierfür muss der initiale Nachrichtenaustausch allerdings in beiden Richtungen erfolgen, damit beide Knoten die Möglichkeit haben ein Credential zu überprüfen und den Schlüssel zu errechnen. Der folgende Abschnitt stellt die in diesem Artikel vorgenommen Änderungen an dem eben beschriebenen Secret Handshake vor, der ihn für das Szenario der Sicherheitszonen anpasst. Abschnitt 3.2 beschreibt im Anschluss das entstandene Protokoll für den attributbasierten Schlüsselaustausch.

3.1 Adaption für Sicherheitszonen

Der Einsatz in dem anfangs motivierten VPN-Szenario erscheint geradlinig, allerdings gibt es bei Umsetzung des Konzepts von Schichten und Compartments einige Hürden, die überwunden werden müssen. Das Hauptproblem ist, dass der Secret Handshake immer nur auf einen Wert gleichzeitig prüfen kann, zusammen mit der Eigenschaft, dass im Falle der Berechnung eines Schlüssels die Knoten nicht entscheiden können, ob sie einen gemeinsamen Schlüssel besitzen ohne dies in Challenge-Response-Manier zu testen. Werden für die Mitgliedschaft in verschieden Sicherheitszonen verschiedene Credentials übertragen, ist es allerdings möglich, dass lediglich eine oder gar keine passt. Das heißt, jedes Credential muss mit jeder vorhandenen Matching Reference abgeglichen werden, und zusätzlich zu diesem Aufwand nach jeder Matching-Operation mit dem Aufwand einer RTT (Round Trip Time) auf Gleichheit des Schlüssels geprüft werden. Somit skaliert der Ansatz nicht über eine wachsende Anzahl von Credentials pro Knoten.

Im Folgenden werden nun eine Reihe von Möglichkeiten präsentiert, aus der oben geschilderten Variante ein praktikables Protokoll mit einer festen Anzahl auszutauschender Nachrichten zu entwickeln.

3.1.1 Vorgeschaltetes Matching

Da es nicht möglich ist, gleichzeitig festzustellen, ob das Matching erfolgreich war und einen Schlüssel zu berechnen werden beide Vorgänge getrennt. Die Knoten bereiten pro Sicherheitszone zwei Credentials vor: Eines zum Ausführen des Matchings und eines um einen Schlüssel zu berechnen. Der empfangende Knoten testet erst, welches der vorgeschlagenen Matching-Credentials kompatibel ist und verwendet anschließend das korrespondierende Credential zum Generieren des Schlüssels.

3.1.2 Simultanes Matching

Besitzer mehrerer Matching References müssen diese in allen Kombinationen mit den empfangenen Credentials testen. Durch Vorberechnungen kann erreicht werden, dass die Anzahl der benötigten Pairing-Auswertungen möglichst niedrig wird, und so diese aufwendige Operation innerhalb des Handshakes nicht unnötig ausgeführt wird. Losgelöst von dem eigentlichen Ab-

lauf des Protokolls, muss ein Teilnehmer folgenden Quotienten errechnen, um zu überprüfen ob ein Credential mit einer Reference übereinstimmt:

$$1 \stackrel{?}{=} \frac{\hat{e}(r_1 cred, r_2 N_2)^{n^{-1}} \cdot \hat{e}(r_1 r_2 S, X_U)}{\hat{e}(r_1 r_2 T, match)} \left(= \frac{\hat{e}(r_1 r_2 T, match)}{\hat{e}(r_1 r_2 T, match)} \right)$$

Der Fokus liegt hierbei bei der Betrachtung der drei einzelnen Pairing-Auswertungen: die erste im Zähler enthält das gerade zu überprüfende Credential, das im Nenner die entsprechende Matching Reference. Werden alle n ankommenden Credentials mit allen m Matching References überprüft, sind $3mn$ Pairing-Auswertungen zu berechnen. Für jede Matching Reference kann jedoch folgender Block vorberechnet werden:

$$\frac{\hat{e}(r_1 r_2 S, X_U)}{\hat{e}(r_1 r_2 T, match)}$$

Es verbleiben somit nur noch $(n+2)m$ Pairing-Auswertungen. Da ein Knoten immer alle seine m Matching References durchprobieren muss, wäre eine Möglichkeit sie zusammenzufassen sinnvoll. Unter Ausnutzung der Bilinearität ist eine weitere Aggregation möglich:

$$\hat{e}(r_1 r_2 T, \sum_{i=1}^{m} match_i) = \prod_{i=1}^{m} \hat{e}(r_1 r_2 T, match_i)$$

Setzt man nun anstatt einer „normalen“ Reference deren Summe ein, ergibt sich für ein erfolgreiches Matching einer der folgenden m Werte ($match_j$ bezeichnet die kompatible Reference):

$$\begin{aligned} \frac{\hat{e}(r_1 cred, r_2 N_2)^{n^{-1}} \cdot \hat{e}(r_1 r_2 S, X_U)}{\hat{e}(r_1 r_2 T, \sum_{i=1}^{m} match_i)} &= \frac{\hat{e}(r_1 r_2 T, match_j)}{\hat{e}(r_1 r_2 T, \sum_{i=1}^{m} match_i)}, 1 \leq j \leq m \\ &= \hat{e}(r_1 r_2 T, \sum_{i=1}^{m} match_i - match_j)^{-1} \end{aligned}$$

Statt den Quotienten auf den Wert 1 zu testen, gleicht man ihn mit den vorberechneten m Werten ab. Diese hängen nur von den Matching References eines Knotens ab, können also bei jedem ankommenden Credential verwendet und vorberechnet werden. Dies senkt die Anzahl der benötigten Pairing-Berechnungen von $3nm$ auf $m + 2 + n$, der quadratische Term wurde also durch einen linearen ersetzt, eine Erhöhung der Zonenzugehörigkeiten ändert die Größenordnung des Berechnungsaufwands nicht mehr.

3.2 Der abgeleitete Schlüsselaustausch

Das Resultat ist ein 4-Wege-Handshake, der die folgende Struktur aufweist:

1. $\mathcal{A}$ sendet eine Verbindungsanfrage zu $\mathcal{B}$, zusammen mit zwei Zufallspunkten, die $\mathcal{B}$ in seiner Antwort einbauen wird:

   $\mathcal{A} \longrightarrow \mathcal{B} \quad \langle n_{\mathcal{A}} P \mid n_{\mathcal{A}} \tilde{P} \rangle$
2. $\mathcal{B}$ bereitet ebenfalls Zufallswerte vor, sendet jedoch zusätzlich bereits zwei Listen von Credentials: Die erste zur Überprüfung der Matching References, die zweite – in der gleichen Reihenfolge – mit Credentials zum Erzeugen eines gemeinsamen Schlüssels. In diesem Fall ist $r_3 P$ die Komponente, welche die beiden Arten von Credentials unterscheidet

und als Information in den Schlüssel eingeht. Falls Matching Reference und Credential zusammenpassen, ergibt sich für Knoten $\mathcal{A}$ der Wert $\hat{e}(r_{1_\mathcal{B}} r_{2_\mathcal{B}} r_{3_\mathcal{B}} rP, P)$. Dabei ist r ein privater Parameter der TTP, welcher in $\tilde{P} = rP$ übertragen wird, $r_{1-3_\mathcal{U}}$ Geheimnisse des Nutzers $\mathcal{U}$. Der Schlüssel bei $\mathcal{A}$ ergibt sich zu $\hat{e}(r_{1_\mathcal{B}} r_{2_\mathcal{B}} r_{3_\mathcal{B}} rP, P)^{r_{1_\mathcal{A}} r_{2_\mathcal{A}} r_{3_\mathcal{A}}}$, bei $\mathcal{B}$ zu $\hat{e}(r_{1_\mathcal{A}} r_{2_\mathcal{A}} r_{3_\mathcal{A}} rP, P)^{r_{1_\mathcal{B}} r_{2_\mathcal{B}} r_{3_\mathcal{B}}}$, welche aufgrund der Bilinearität des Pairings identisch sind.

$$\begin{aligned}\mathcal{B} \longrightarrow \mathcal{A} \quad \langle \;\; & n_\mathcal{B} P \mid n_\mathcal{B} \tilde{P} \\ & \mid r_1 cred_{1_\mathcal{B}} \mid r_1 cred_{2_\mathcal{B}} \mid \cdots \mid r_1 cred_{\rho_\mathcal{B}} \\ & \mid r_1(cred_{1_\mathcal{B}} + r_3 P) \mid \ldots \mid r_1(cred_{\rho_\mathcal{B}} + r_3 P) \\ & \mid r_2 N_2 \mid r_1 r_2 S \mid r_1 r_2 T \\ \rangle\end{aligned}$$

3. Falls $\mathcal{A}$ eine Übereinstimmung feststellt, sendet es die beiden entsprechenden Credentials an $\mathcal{B}$. Hierbei reicht es, nur ein Credential zu senden und somit Bandbreite zu sparen. Für den Fall, dass kein Matching gelang, sendet $\mathcal{A}$ ein gefälschtes Credential: eine Zufallszahl. Das Übersenden von $\hat{e}(r_{1_\mathcal{B}} r_{2_\mathcal{B}} r_{3_\mathcal{B}} r n_\mathcal{B} P, P)$ gibt $\mathcal{B}$ die Möglichkeit schnell zu überprüfen, ob $\mathcal{A}$ den Schlüssel errechnet hat: $\mathcal{A}$ ist nur nach erfolgreichem Matching in Besitz dieser Werte, $\mathcal{B}$ kann diesen Ausdruck vorberechnen.

$$\begin{aligned}\mathcal{A} \longrightarrow \mathcal{B} \quad \langle \;\; & \hat{e}(r_{1_\mathcal{B}} r_{2_\mathcal{B}} r_{3_\mathcal{B}} r n_\mathcal{B} P, P) \\ & \mid r_1 cred_{1_\mathcal{B}} \\ & \mid r_1(cred_{1_\mathcal{B}} + r_3 P) \\ & \mid r_2 N_2 \mid r_1 r_2 S \mid r_1 r_2 T \\ \rangle\end{aligned}$$

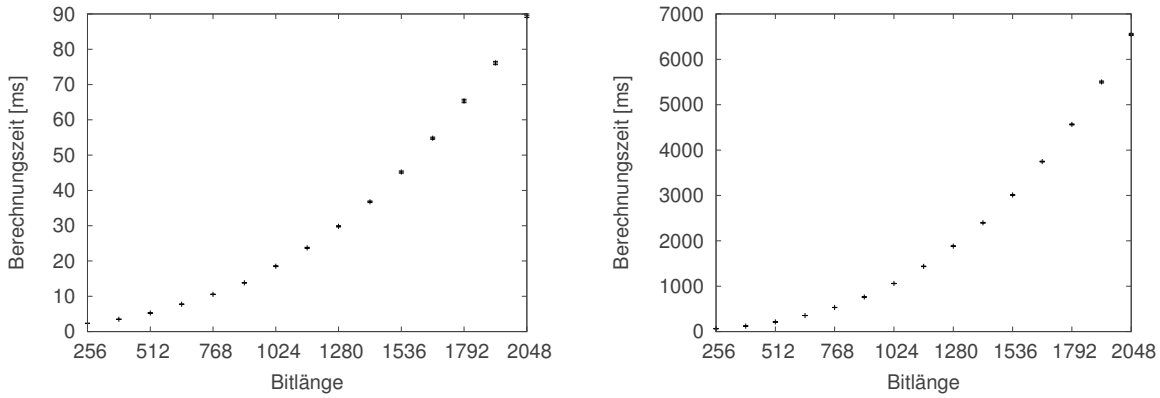
4. Mit der vierten Nachricht schließt $\mathcal{B}$ den Austausch ab. Dies ist im Falle eines positiven Ausganges ebenfalls der Beweis für $\mathcal{A}$, dass das Matching erfolgreich war.

$$\mathcal{B} \longrightarrow \mathcal{A} \quad \langle \;\; \hat{e}(r_{1_\mathcal{A}} r_{2_\mathcal{A}} r_{3_\mathcal{A}} r n_\mathcal{A} P, P) \;\; \rangle$$

3.3 Sicherheitsbetrachtungen

Das vorgestellte Protokoll überträgt den Handshake von Sorniotti und Molva auf ein Szenario, in dem auch auf mehrere Attribute abgetestet werden muss und sieht deshalb entsprechende Optimierungen vor. Trotzdem gehen die von ihnen bewiesenen Sicherheitseigenschaften (Nichtveröffentlichung des Attributs, Nichtfälschbarkeit der Credentials und Matching References) nicht verloren. Die in Abschnitt 2 vorgestellten Hauptanforderungen werden von dem entwickelten Protokoll erfüllt: ein Sitzungsschlüssel wird wie oben beschrieben abgeleitet, die Sicherheitszonen der einzelnen Teilnehmer werden nur durch einen diskreten Logarithmus geschützt übertragen. Ferner wird eine TTP nur zur initialen Vergabe der Credentials und Matching References benötigt, sowie um die globalen Systemparameter festzulegen. Ist dies erfolgt, läuft der Schlüsselaustausch ausschließlich zwischen den beiden Teilnehmern ab. Entstanden ist ein 4-Wege-Handshake, der so an die Stelle eines anderen Schlüsselaustauschs (z.B. Diffie-Hellmann) gesetzt werden kann. Dies wäre zum Beispiel in einer Implementierung eines neuartigen Schlüsselaustausches für IPsec möglich.

Eine Besonderheit des Handshakes ist, dass er sowohl auf die Sicherheit des diskreten Logarithmus auf elliptischen Kurven (ECDLP) als auch auf die Berechnung des Pairings vertraut. Dieses kann man aber dazu nutzen, den diskreten Logarithmus auf den zugrundeliegenden end-



(a) Einzelne Pairingberechnung

(b) Kompletter Handshake

Abb. 2: Berechnungszeiten

lichen Körper übertragen (MOV-Reduktion). Somit ist die Existenz eines Pairings ein Angriff auf das ECDLP, womit sich hier ein Widerspruch auftut. Dieser wird dadurch gelöst, dass man auf einen Vorteil von ECC verzichtet: die kurzen Schlüssellängen, wodurch das zugrundeliegende Feld so groß ist, dass dort auch der herkömmliche diskrete Logarithmus sicher ist. Dies geht allerdings zu Lasten der Performance und erhöht die Nachrichtenlänge. Die Suche und Standardisierung von Kurven, auf denen zum einen das Pairing effizient berechenbar ist und zum anderen die MOV-Reduktion auch bei kleinen Schlüssellängen keine sicherheitskritischen Auswirkungen hat ist nach wie vor ein offenes Problem.

3.4 Implementierung

Neben der reinen algebraischen Betrachtung wurde eine Proof-of-Concept-Implementierung entworfen, in der die vorgestellten Bearbeitungsschritte durchgeführt werden. Als Softwarebibliothek für elliptische Kurven wurde MIRACL 5.4 (Multiprecision Integer and Rational Arithmetic C/C++) genutzt, welche neben den grundlegenden Funktionen für elliptische Kurven auch eine Implementierung von Millers Algorithmus zur Auswertung der bilinearen Pairings besitzt.

Um den Größenaufwand der Berechnungen darzustellen wurde die Laufzeit einiger Protokollprimitiven in Abhängigkeit der benutzten Bitrate gemessen (Testsystem: Intel Core 2 Quad mit 2.66GHz, Testprogramm nutzt nur einen Thread).

Abbildung 2(a) zeigt den Zeitaufwand für die Berechnung eines einzelnen Pairings der Form $\hat{e}(P,Q)^a$, Abbildung 2(b) die zusammengefasste Rechendauer einer kompletten Schlüsselaushandlung ohne Netzwerkverzögerungen. Obwohl der Aufwand mit steigender Bitlänge deutlich wächst, liegt er für für eine einzelne Pairingberechnung stets im Millisekundenbereich. Auch der Aufwand für den kompletten Handshake bleibt in praktikablen Grenzen, vor allem wenn man bedenkt, dass dies lediglich ein Testsystem ist, Optimierungen in Software und Algorithmus also möglich scheinen.

4 Zusammenfassung und Ausblick

Ziel der Arbeit war die Entwicklung eines kryptographischen Protokolls, das es erlaubt, die Schlüsselaushandlung an den Abgleich gewisser Attribute (in diesem Fall Sicherheitszonen) zu binden, ohne das Attribut selbst zu übertragen. Ausgangspunkt für die Lösung war der Se-

cret Handshake von Sorniotti und Molva, welcher derart erweitert wurde, dass eine effiziente Überprüfung mehrerer Zonenzugehörigkeiten möglich ist. Das entstandene Protokoll weist eine ähnliche Struktur wie der etablierte Diffie-Hellman-Schlüsselaustausch auf, kann diesen in bestehenden Systemen also ersetzen.

Je nach verwendeter elliptischer Kurve können die Sicherheit und Performanz der kryptographischen Operationen stark variieren. Zukünftige Forschungen im Bereich der pairing-basierten Kryptographie müssen sich vor allem auf das optimale Abwägen dieser beiden Ziele konzentrieren. Nur so können neuartige Sicherheitsprotokolle, wie das vorgestellte, die Möglichkeiten der Bilinear-Pairings in Produktivsystemen effizient und sicher nutzen.

Literatur

[BoFr01] D. Boneh, M. Franklin: Identity-based encryption from the Weil pairing. *In: Proceedings of Advances in Cryptology – CRYPTO*, Springer (2001), 213–229.

[Broe10] M. Broersma: Botnet price for hourly hire on par with cost of two pints (2010), .

[BrRS09] M. Brinkmeier, M. Rossberg, G. Schaefer: Towards a Denial-of-Service Resilient Design of Complex IPsec Overlays. *In: Proceedings of International Conference on Communications (ICC)* (2009).

[DiHe76] W. Diffie, M. Hellman: New Directions in Cryptography. In: *IEEE Transactions on Information Theory*, 22, 6 (1976), 644–654.

[Joux09] A. Joux: Algorithmic Cryptanalysis. Chapman & Hall (2009).

[SoMo09] A. Sorniotti, R. Molva: A Provably Secure Secret Handshake with Dynamic Controlled Matching. *In: Proc. of 24th International Information Security Conference (IFIP SEC)*, Springer (2009), 330–341.