

Schlüsseletablierungsprotokolle

Skript zum Vortrag von Thomas Krause im Rahmen des Proseminars „Kryptographische Algorithmen und Protokolle“ an der Humboldt Universität zu Berlin



Autor: Thomas Krause (krause@informatik.hu-berlin.de)
Datum: 06. Februar 2006

Gliederung

1. Einführendes Beispiel: SSH.....	3
2. Mögliche Verfahren.....	4
2.1 Schlüsseltransportprotokolle.....	4
2.1.1 Breitmaulfrosch-Protokoll.....	5
2.1.2 Needham-Schroeder-Protokoll.....	6
2.2 Schlüsselvereinbarungsprotokolle.....	7
2.2.1 Mathematische Hintergründe vom Diffie-Hellmann-Protokoll.....	7
2.2.2 Originales Diffie-Hellmann-Protokoll.....	9
2.2.3 Diffie-Hellmann-Protokoll mit Authentifizierung.....	10
2.2.4 Sicheres authentisches Diffie-Hellmann-Protokoll.....	10
3. Angriffe auf die Protokolle.....	12
3.1 Impersonation.....	12
3.2 Replay-Attacke.....	12
3.3 Man-in-the-middle-Angriff.....	12
4. Literatur.....	13

1. Einführendes Beispiel: SSH

Dieses einführende Beispiel soll den Sinn von Schlüsseletablierungsprotokollen deutlich machen. Angenommen man hat zwei Computer, die über einen unsicheren Weg kommunizieren sollen. Dies ist zum Beispiel bei der Fernwartung, also wenn ein Systemadministrator über seinen eigenen Computer auf einen entfernten Computer zugreift um administrative Arbeiten durchzuführen, der Fall. Auch Versionskontrollsysteme für Softwareentwicklung wie Subversion (SVN), Zugriff auf Dateien der eigenen Website bei einem fremden Server über sicheres FTP (SFTP) sind solche Anwendungsszenarien. Allen diesen Anwendungen ist gemein, dass immer kontinuierlich eine größere Menge an Daten ausgetauscht wird.

Eine mögliche Idee könnte natürlich sein, einfach die Daten mit dem Public-Key des anderen zu verschlüsseln. Das wäre praktisch, weil auch zwischen untereinander unbekanntenen Personen oder Computern Daten ausgetauscht werden könnten. Allerdings ist das Public-Key-Verfahren ein asymmetrisches Verfahren und damit auch sehr ineffektiv, gerade wenn es um größere Datenmengen geht.

Dieses Manko der asymmetrischen Verschlüsselung führt zur anderen Möglichkeit der symmetrischen Verschlüsselung mit einem gemeinsamen Geheimnis. Der Vorteil dieser Lösung besteht in seiner Geschwindigkeit. Sein Nachteil ist natürlich gerade das gemeinsame Geheimnis vorher vertraulich ausgetauscht werden müssen. Da man sich bei den beschriebenen Anwendungen meist nicht physisch treffen kann, ist ein geheimer Schlüsselaustausch oft nicht möglich.

Die Lösung dieser Probleme besteht nun gerade in den Schlüsseletablierungsprotokollen, bei denen ein Schlüssel über unsichere Kanäle sicher ausgetauscht werden kann.

SSH ist ein Protokoll, um mit fremden Rechnern sicher kommunizieren zu können. Es benutzt symmetrische Verschlüsselung mit einem geheim vereinbarten Schlüssel. Da das Internet an sich nur unsichere Übertragungswege bietet, haben sich viele Verfahren basierend auf SSH entwickelt. Ein Beispiel dafür ist gerade SFTP.

2. Mögliche Verfahren

Allgemein kann man in zwei verschiedene Verfahren unterscheiden: Schlüsseltransportprotokolle und Schlüsselvereinbarungsprotokolle.

2.1 Schlüsseltransportprotokolle

Die grundlegende Idee hinter Schlüsseltransportprotokollen ist, dass ein von einem Teilnehmer A generierter symmetrischer Schlüssel über einen unsicheren Kanal an Teilnehmer B übermittelt werden soll. Der Schlüssel soll dabei natürlich geheim bleiben.

Pro Sitzung, also pro zeitlich zusammenhängenden Datenaustausch, soll immer nur ein Schlüssel benutzt werden. Folglich heißt dieser Schlüssel auch Sitzungsschlüssel. Dieses Prinzip ist wichtig, damit wenn ein Sitzungsschlüssel bekannt wird, das wird auch kompromittiert genannt, nicht die Kommunikation der restlichen Sitzungen entschlüsselt werden kann. Dadurch wird die Sicherheit deutlich erhöht.

Welche Art von symmetrischer Verschlüsselung man für das Protokoll einsetzt ist hier nicht weiter von Bedeutung. Die Protokolle kümmern sich nur um den Transport des Schlüssels. Da es aber sehr sichere symmetrische Verschlüsselungsverfahren gibt, kann man sich einfach vorher auf eines dieser Verfahren einigen.

Ein Begriff, der im Folgenden immer wieder genannt werden wird, ist die so genannte „trusted third party“ kurz TTP. Diese vertrauenswürdige dritte Person, wobei „Person“ auch im übertragenen Sinne als Protokollteilnehmer im Allgemeinen gemeint ist, spielt bei den später vorgestellten Protokollen eine teils sehr wichtige Rolle. Ohne diese Instanz könnte keines der vorgestellten Protokolle funktionieren. Das Problem dabei ist, das die beiden Kommunikationspartner der TTP auch wirklich vertrauen können müssen. Dies ist im echten Leben nicht immer der Fall, da zum Beispiel einer staatlichen Institution oder einer privaten Firma nur bedingt Vertrauen zu schenken ist. Wenn man sich aber andererseits ein WLAN vorstellt, ist der Netzbetreiber eine TTP, da sie die Kommunikation auch sonst mithören könnte, da er das Netzwerk stellt. Zwei Teilnehmer A und B im Netz misstrauen sich vielleicht gegenseitig aber nicht dem Netzbetreiber. Bei unterschiedlichen Verfahren ist auch ein unterschiedlicher Grad an Vertrauen gegenüber der TTP vonnöten. Wie man leicht sieht, ist die Vertrauenswürdigkeit der TTP ein kritischer Sicherheitsaspekt.

2.1.1 Breitmaulfrosch-Protokoll

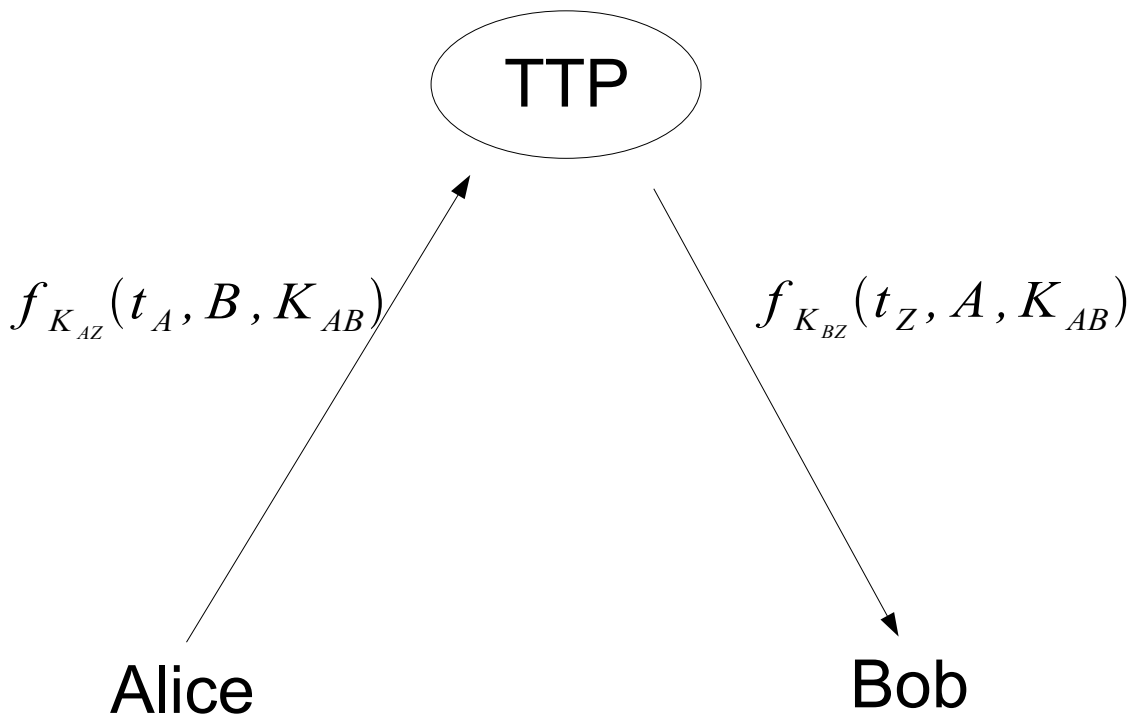


Abbildung 1: Ablauf des Breitmaulfrosch-Protokolls

Dieses Protokoll ist eigentlich das denkbar einfachste. Es werden nur zwei Datenpakete benötigt um den Schlüssel zu übertragen. Erwähnt wurde es zuerst 1989 eher nebenbei in einem Paper von Michael Burrows, Martín Abadi und Roger Needham. Warum es Breitmaulfrosch-Protokoll heißt, haben sie leider nicht erwähnt.

Die benutzte symmetrische Verschlüsselungsfunktion wird mit f_X bezeichnet. Dabei ist X der Schlüssel. K_{AB} ist der Sitzungsschlüssel, der zur Kommunikation zwischen Alice und Bob benutzt werden soll. K_{AZ} und K_{BZ} sind die Schlüssel zwischen Alice und der TTP beziehungsweise Bob und der TTP. Die beiden letzten Schlüssel müssen natürlich vorher geheim ausgetauscht worden sein. Der TTP kommt hier eine Art Telefonbuchfunktion zu, da sie alle Teilnehmer bereits vorher kennen muss.

Im ersten Schritt erzeugt Alice den Sitzungsschlüssel. Sie tritt nun mit der TTP in Kontakt und teilt ihr mit, dass sie mit Bob kommunizieren möchte. Den erzeugten Schlüssel K_{AB} verschlüsselt Alice mit K_{AZ} . Nun entschlüsselt die TTP den Sitzungsschlüssel und verschlüsselt ihn zusammen mit der Information, dass Alice Kontakt aufnehmen wollte, mit K_{BZ} . Dieses Chiffre wird an Bob gesendet. Wenn Bob nun die Nachricht entschlüsselt, besitzen Alice und Bob den gemeinsamen Sitzungsschlüssel. Die Zeitstempel t_A und t_Z sollen Replay-Attacken verhindern.

Die Einfachheit des Protokolls bringt natürlich auch seine Probleme mit sich. So ist es fraglich, ob Alice wirklich gute Schlüssel erzeugen kann, da sie mit den begrenzten Ressourcen eines normalen Heimcomputers ausgerüstet ist. Noch drastischer wird dieses Problem, wenn so genannte Smartcards, also Kleinstcomputer im Scheckkartenformat (z. B. die Geldkarte), eingesetzt werden. Dass nur der Schlüssel übertragen wird und keine Authentifizierung vorgesehen ist eröffnet weit reichende Möglichkeiten für Angriffe. Zudem kennt die TTP den Sitzungsschlüssel und könnte damit theoretisch abgehörte Nachrichten entschlüsseln. Ihr muss also ein sehr hohes Vertrauen

entgegenbracht werden.

2.1.2 Needham-Schroeder-Protokoll

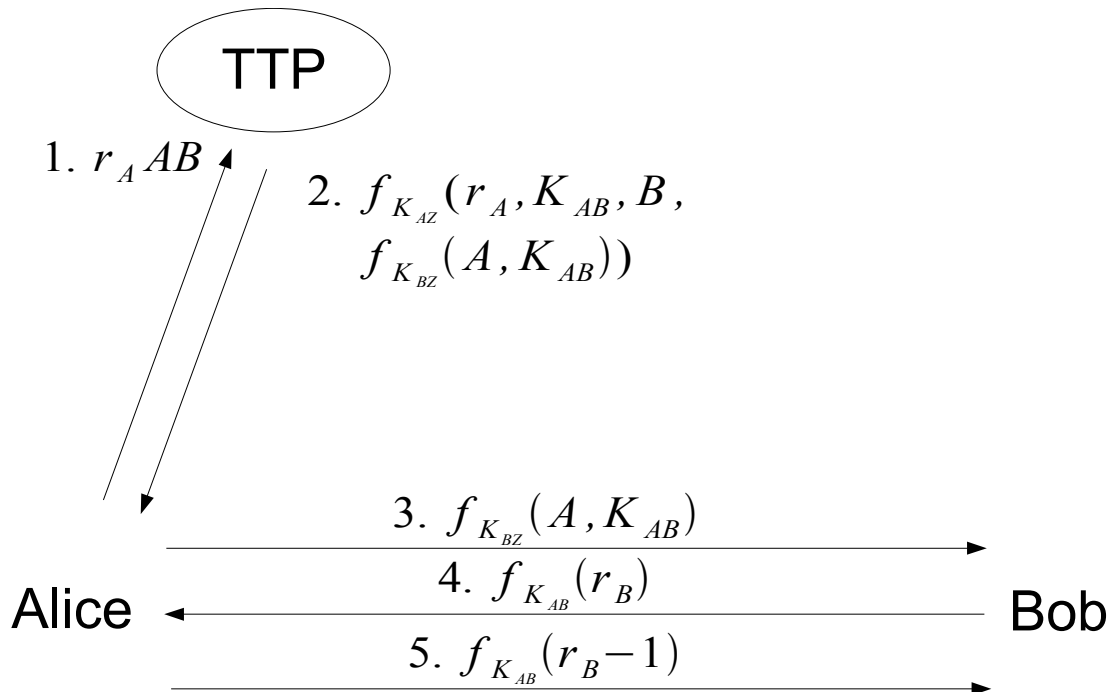


Abbildung 2: Ablauf des Needham-Schroeder-Protokolls

Dieses Protokoll löst einige der beim Breitmaulfrosch-Protokoll genannten Probleme.

Soweit nicht anders beschrieben, haben die Bezeichner die gleiche Bedeutung wie bei der Beschreibung des Breitmaulfrosch-Protokolls.

Zuerst teilt Alice der TTP mit, dass sie mit Bob kommunizieren möchte. r_A ist dabei eine von Alice gewählte Zufallszahl, die öffentlich ist. Die TTP erzeugt einen Sitzungsschlüssel und chiffriert mit dem Schlüssel zwischen der TTP und Alice mehrere Informationen:

- r_A um sich gegenüber von A zu authentifizieren. Nur die TTP und Alice können die Zufallszahl korrekt mit K_{AZ} ver- oder entschlüsseln.
- Den Sitzungsschlüssel selbst.
- Die Information, dass mit Bob kommuniziert werden soll.
- Ein Tupel, das vorher mit dem Schlüssel zwischen der TTP und Bob verschlüsselt wurde. Dieses Tupel enthält wiederum die Information, dass Alice Kontakt aufnehmen möchte und den Sitzungsschlüssel. Dieses Tupel wird doppelt verschlüsselt, damit Alice es nicht entschlüsseln kann.

Im nächsten Schritt entschlüsselt Alice alles und verschickt das noch verschlüsselte Tupel weiter an Bob. Bob kann nun den Sitzungsschlüssel entschlüsseln. Damit sich Alice und Bob gegenseitig authentifizieren können, führen sie am Ende noch einen so genannten Handshake aus. Dabei wird von Bob eine Zufallszahl r_B gewählt und verschlüsselt an Alice geschickt. Alice muss nun diese Zahl korrekt entschlüsseln können und sendet $r_B - 1$ mit dem Sitzungsschlüssel verschlüsselt zurück an Bob. Damit können Alice und Bob sicher sein, dass der jeweils andere den

Sitzungsschlüssel kennt. Diese Methode ist auch als Challenge-Response bekannt, bei der man sich von der Kenntnis eines Geheimnisses durch die Stellung von Aufgaben und deren korrekte Beantwortung des anderen überzeugt.

Das Needham-Schroeder-Protokoll ist die Grundlage der Kerberos-Protokolle für Netzwerke.

Allerdings treten auch hier Probleme auf. So kann sich Alice zwar über die Aktualität des Sitzungsschlüssels durch die Einmalzufallszahl überzeugen, Bob kann das jedoch nicht und fällt eventuell auf einen alten und kompromittierten Sitzungsschlüssel herein. Eine Lösung des Problems besteht darin, aus dem zuerst von der TTP erstellten und dann über Alice an Bob gesendeten Tupels ein Tripel zu machen. Dieses enthält zusätzlich einen Zeitstempel. Wie auch im vorherigen Protokoll kennt die TTP den Sitzungsschlüssel. Allerdings ist die TTP im Gegensatz zu Alice in der Lage, gute Schlüssel zu erzeugen.

2.2 Schlüsselvereinbarungsprotokolle

Bei den Schlüsseltransportprotokollen musste der TTP ein hohes Vertrauen entgegenbracht werden, da sie die Sitzungsschlüssel kannte. Bei den Schlüsselvereinbarungsprotokollen soll die TTP nur noch als Zertifizierungsinstanz dienen. Der gemeinsame Sitzungsschlüssel soll durch die beiden Kommunikationspartner in einem Dialog erzeugt werden. Trotzdem soll ein Angreifer durch Abhören der unsicheren Kommunikationswege nicht in den Besitz des Schlüssels kommen können. Das klingt natürlich erstmal ziemlich abenteuerlich, ist aber möglich. Dabei spielt das Diffie-Hellmann-Protokoll eine zentrale Rolle.

2.2.1 Mathematische Hintergründe vom Diffie-Hellmann-Protokoll

Für jede Primzahl p existiert ein so genannter Generator g . Dieser hat die Eigenschaft, dass er alle ganzzahlige Zahlen von 0 bis $p-1$ erzeugen kann. Dabei gilt

$$\{g^i \bmod p \mid i=0, \dots, p-2\} = \{1, \dots, p-1\}$$

Wenn z. B. $p=7$ dann ist $g=3$ ein passender Generator.

$$3^0 \bmod 7 = 1 \bmod 7 = 1$$

$$3^1 \bmod 7 = 3 \bmod 7 = 3$$

$$3^2 \bmod 7 = 9 \bmod 7 = 2$$

$$3^3 \bmod 7 = 27 \bmod 7 = 6$$

$$3^4 \bmod 7 = 81 \bmod 7 = 4$$

$$3^5 \bmod 7 = 243 \bmod 7 = 5$$

Das Interessante an der diskreten Exponentialfunktion $g^x \bmod p$ ist, dass sie eine Einwegfunktion darstellt. Sie ist also in die eine Richtung leicht berechenbar, ihre Umkehrfunktion dagegen nur durch Ausprobieren zu lösen. Die Umkehrfunktion heißt in diesem Fall diskreter Logarithmus. Einwegfunktionen sind in kryptographischen Anwendungen wegen ihrer Eigenschaften sehr beliebt.

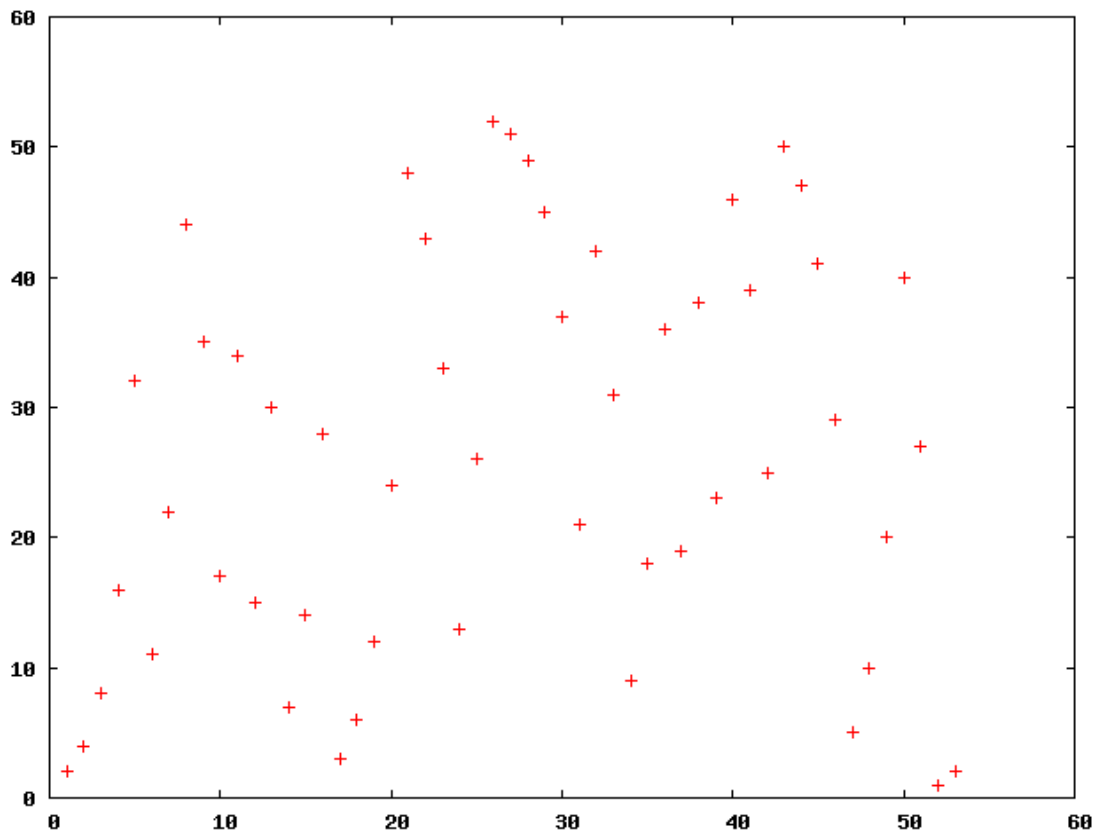


Abbildung 3: Eine diskrete Exponentialfunktion mit $p=53$ und $g=2$ ($f(x) = 2^x \bmod 53$)

Abbildung 3 zeigt exemplarisch eine diskrete Exponentialfunktion. Die Umkehrfunktion würde sich aus vertauschen der x- und y-Achse ergeben. Offensichtlich sind die Werte der Funktion und vor allem auch der Umkehrfunktion sehr unregelmäßig verteilt. Die Frage ist nun, ob es effizient möglich ist aus den bekannten Größen $g^x \bmod p$, p und g den gesuchten Wert x zu bestimmen. In einzelnen Spezialfällen ist das möglich, bei guter Auswahl der Eingabezahlen gibt es nach bisherigem Wissensstand keine effiziente Möglichkeit.

2.2.2 Originales Diffie-Hellmann-Protokoll

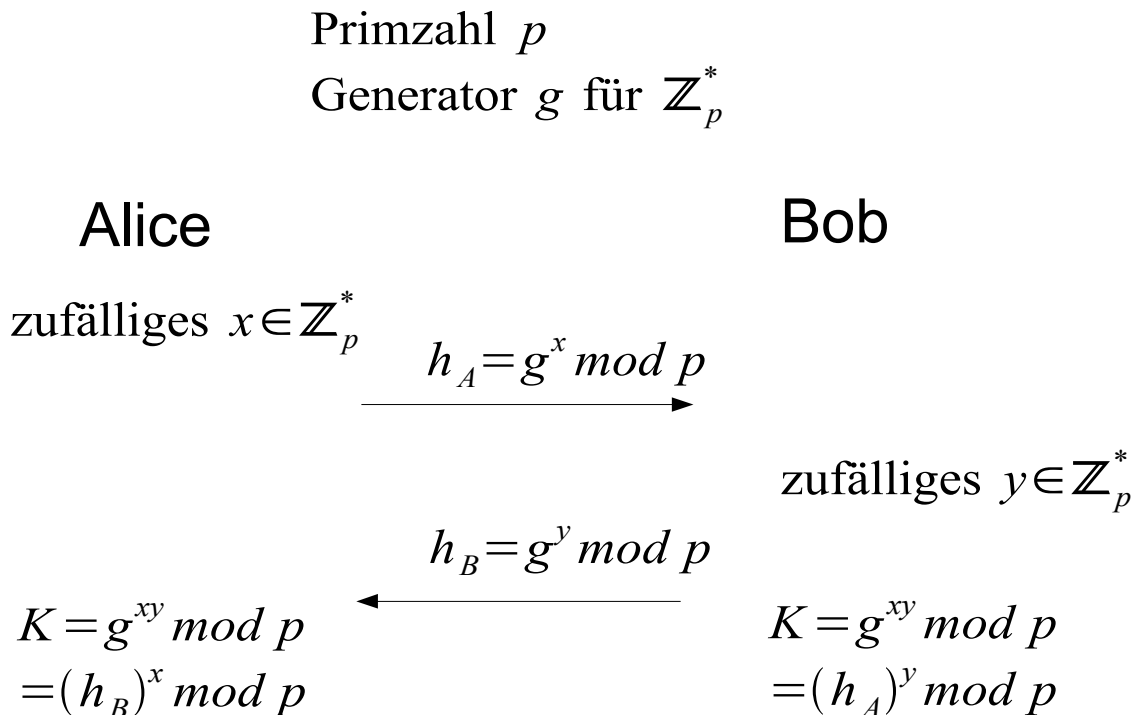


Abbildung 4: Ablauf des originalen Diffie-Hellmann-Protokolls

Dieses Protokoll wurde 1979 vorgeschlagen. In der gleichen Arbeit wurde das Public-Key-System vorgeschlagen, allerdings noch ohne einen konkreten Algorithmus. Dafür wurde dieses Schlüsselvereinbarungsprotokoll beschrieben.

In diesem Protokoll sind die Primzahl p und der dazugehörige Generator g öffentlich bekannt. Diese müssen vorher von den Teilnehmern vereinbart werden. Zuerst wählt Alice eine Zufallszahl x . Das Ergebnis der diskreten Exponentialfunktion $h_A = g^x \bmod p$ wird an Bob gesendet. Nun wählt Bob seinerseits eine Zufallszahl y und sendet $h_B = g^y \bmod p$ an Alice. Zusammen mit der jeweiligen selbst gewählten Zahl können Alice und Bob den gemeinsamen Sitzungsschlüssel $K = g^{xy} \bmod p$ berechnen. Obwohl p , g und die versendeten Nachrichten bekannt sind bzw. durch passives Abhören in Erfahrung gebracht werden können ist das Verfahren gegenüber passiven Angriffen sicher. Weder x noch y können aufgrund des Einwegcharakters aus h_A oder h_B berechnet werden.

Folgend eine Beispielrechnung.

$$p=52 \quad g=2$$

Alice und Bob wählen die Zufallszahlen $x=6$ und $y=3$.

Dann werden diese Nachrichten verschickt.

$$\text{An Bob: } h_A = 2^6 \bmod 53 = 64 \bmod 53 = 11$$

$$\text{An Alice: } h_B = 2^3 \bmod 53 = 8 \bmod 53 = 8$$

Danach berechnen Alice und Bob jeweils den gemeinsamen Schlüssel.

$$\text{Alice: } K = 8^6 \bmod 53 = 262144 \bmod 53 = 262144 - (4946 * 53) = 6$$

$$\text{Bob: } K = 11^3 \bmod 53 = 1331 \bmod 53 = 1331 - (25 * 53) = 6$$

Noch mal zur Kontrolle:

$$K = g^{xy} \bmod 53 = 2^{18} \bmod 53 = 262144 \bmod 53 = 6$$

Allerdings weist das originale Diffie-Hellmann-Protokoll auch Schwächen auf, die in den nächsten Kapiteln gelöst werden sollen.

2.2.3 Diffie-Hellmann-Protokoll mit Authentifizierung

Die originale Version vom Diffie-Hellmann-Protokoll ist anfällig für Man-in-the-middle-Angriffe, der im Gegensatz zum passiven Abhören einen aktiven Angriff darstellt. Eine mögliche Lösung wäre, feste Werte für h_A und h_B zu verwenden, die von einer TTP authentifiziert werden könnten. Dann würde allerdings immer der gleiche Schlüssel erstellt, wenn zwei bestimmte Teilnehmer kommunizieren. Dieser wäre dann kein echter Sitzungsschlüssel mehr. Daher wird bevorzugt mit Signaturen und Zertifikaten gearbeitet. Solche Überlegungen führen zum sicheren authentischen Diffie-Hellmann-Protokoll.

2.2.4 Sicheres authentisches Diffie-Hellmann-Protokoll

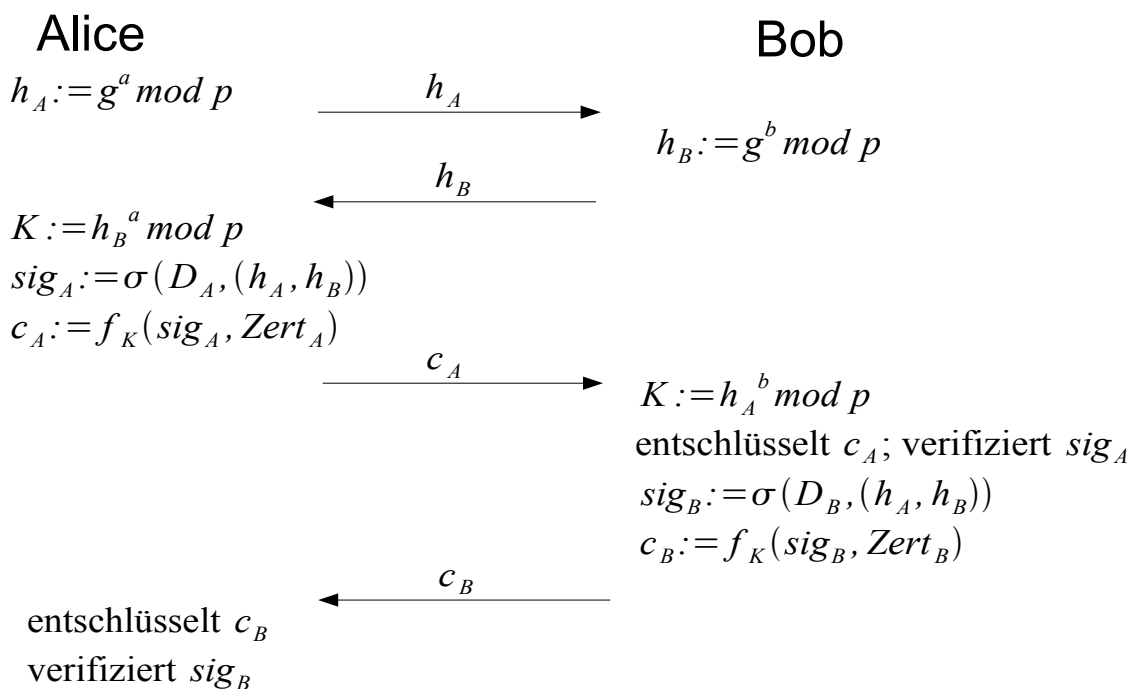


Abbildung 5: Ablauf des sicheren authentischen Diffie-Hellmann-Protokolls

Hier wird erst das originale Diffie-Hellmann-Protokoll ausgeführt. Danach benutzt Alice ein Public-Key-Verfahren, um h_A und h_B mit seinem privaten Schlüssel zu signieren. Damit beweist Alice das die vorherige Nachricht wirklich von Alice stammt. Diese Signatur zusammen mit einem Zertifikat verschlüsselt Alice mit dem gemeinsamen Schlüssel K und sendet dieses Chifftrat an Bob. Diese Verschlüsselung ist ein Handshake, um zu zeigen, dass Alice im Besitz des gemeinsamen Schlüssels ist. Ansonsten könnte ein Angreifer durch Impersonation Bob glaubhaft machen, der Sitzungsschlüssel K sei gemeinsames Geheimnis zwischen dem Angreifer und Bob.

Thomas Krause: Schlüsseletablierungsprotokolle

Nach dem Versenden entschlüsselt und verifiziert die Nachricht von Alice. Nun macht Bob genau dasselbe wie Alice. Daraufhin kann auch Alice Bobs Nachricht entschlüsseln und verifizieren.

Man könnte auf die Idee kommen, dass Bob c_B schon zusammen mit h_B versendet. Dies wäre aber ungünstig, weil die Berechnung der Signaturen viel Zeit kostet. Damit könnte ein Angreifer sehr viele Anfragen zur Eröffnung eines Diffie-Hellmann-Protokolls starten und damit Bob für andere Anfragen un erreichbar zu machen (DOS-Angriff). Bei der eingesetzten Variante muss der Angreifer zuerst diesen Aufwand betreiben und ein Verfügbarkeitsangriff wird dabei sehr erschwert.

Bisher sind keine Probleme oder Schwächen dieses Protokolls bekannt. Es hat alle formalen Methoden zu Protokollsicherheit bestanden. Damit kann es zurzeit als sicher angesehen werden.

Das sichere authentische Diffie-Hellmann-Protokoll wird zum Beispiel in IPSec v6 und beim als einführendes Beispiel genannten SSH eingesetzt.

3. Angriffe auf die Protokolle

Man unterscheidet die Angriffe auf die Protokolle im Allgemeinen in passive und aktive Angriffe.

Bei passiven Angriffen kann der Angreifer, der in der Literatur oft Eve genannt wird, nur die Kommunikation abhören, aber nicht selbst als Teilnehmer am Protokoll haben. Der Angreifer kann bei einem aktiven Angriff hingegen Informationen und kann Nachrichten verändern oder löschen. Er ist praktisch gesehen ein Teilnehmer beim Protokoll.

Viele Protokolle sind nicht anfällig für passive Angriffe, können aber sehr leicht durch aktive Angriffe ausgehebelt werden. Es ist sehr schwer Schwachstellen gegenüber aktiven Angriffen in den Protokollen zu finden, da es sehr viele verschiedene Möglichkeiten zum Eingriff gibt.

3.1 Impersonation

Bei der Impersonation schafft Eve, also der Angreifer, es sich gegenüber Bob als Alice auszuweisen. Dazu kann er versuchen Passwörter, Zertifikate oder Signaturen abzufangen und bei einem Angriff wieder zu verwenden. Solche Angriffe können durch Zeitstempel erschwert werden.

Ein Beispiel. Eine Bank erhält einen Transaktionsauftrag von Conrad, Geld an Alice zu überweisen. Die vorherige Kommunikation zwischen Alice und der Bank wurde vom Angreifer Eve abgehört. Nun kann Eve die abgefangene Signatur benutzen, um sich gegenüber der Bank als Alice auszugeben. Dadurch kann Eve unerlaubterweise das Geld, das für Alice bestimmt war, erhalten.

Für diesen Angriff ist zum Beispiel das Diffie-Hellmann-Protokoll ohne Handshake anfällig.

3.2 Replay-Attacke

Bei der Replay-Attacke wird eine belauschte Kommunikation vom Angreifer gespeichert und später wieder in der richtigen Reihenfolge „abgespielt“. Dieser Angriff scheint auf den ersten Blick nicht allzu gefährlich zu sein. Das folgende Beispiel soll das Problem daher etwas deutlicher machen.

Alice überweist über eine Bank einen bestimmten Geldbetrag an einen unseriösen Online-Shop mit Geschäftssitz auf den Caymaninseln. Der Betreiber des Online-Shops kann nun die Kommunikation zwischen Alice und der Bank belauschen. Später spielt der die Konversation in der Rolle von Alice gegenüber der Bank ab und kann sich so beliebig oft den Geldbetrag zum Schaden von Alice überweisen lassen.

3.3 Man-in-the-middle-Angriff

Dieser Angriff ist einer der Machtvollsten. Dabei geht der Angreifer zwischen die Kommunikation von Alice und Bob. Gegenüber Alice gibt sich der Angreifer als Bob aus und gegenüber Bob als Alice. Damit kann er das Protokoll zwischen Alice und Bob fast beliebig manipulieren.

Insbesondere das originale Diffie-Hellmann-Protokoll ist dafür anfällig. Um diesen Angriff zu verhindern, müssen solche Manipulationen erkannt werden können.

4. Literatur

- **Kryptografie in Theorie und Praxis : mathematische Grundlagen für elektronisches Geld, Internetsicherheit und Mobilfunk** / Albrecht Beutelspacher ; Heike B. Neumann ; Thomas Schwarzpaul. - 1. Aufl. . - Wiesbaden : Vieweg, 2005. - XIV, 319 S. . - ISBN: 3-528-03168-9
- **Practical cryptography** / Niels Ferguson ; Bruce Schneier. - Indianapolis, Ind. : Wiley, 2003. - XX, 410 S. . - ISBN: 0-471-22894-X. - ISBN: 0-471-22357-3
- Micheal Burrows, Martín Abadi, Roger Needham: **A Logic of Authentication** 1989, 1990
- **SSH, the Secure Shell : the definitive guide** / Daniel J. Barrett ; Richard E. Silverman, and Robert G. Byrnes. - 2. ed. . - Beijing [u.a.] : O'Reilly, 2005 (2005). - XVIII, 645 S. . - ISBN: 0-596-00895-3