Fachhochschule Wedel
University of Applied Science

Short Message Authentication Protocols

SMAPs

Seminar IT-Sicherheit
Sommersemester 2018

Eingereicht von:
Jan Trillmich
inf102234

Betreuung durch:
Prof. Dr. Gerd Beuster

21.Juli.2018

1 INHALT

2 PROBLEMATIK

In diesem Paper von Baqer, Bezuidenhoudt, Anderson und Kuhn geht es im Allgemeinen um Mobile Banking in weniger entwickelten Ländern und Gebieten mit schlechter Netzabdeckung (Baqer, Bezuidenhoudt, Anderson, & Kuhn, 2017). Mobile Banking bringt in diesen Gebieten viele Vorteile gegenüber dem Bargeld-System. Zum Beispiel reduziert es die Erpressungs- und Korruptionsraten. Das wird erreicht, indem alle Transaktionen nachvollziehbar werden und niemand mehr Bargeld bei sich trägt. Hinzu kommt, dass alte und hausgebundene Personen so weiter Rente beziehen können ohne dabei auf Zwischenmänner angewiesen zu sein, die das Geld aus der Stadt von der Bank holen und dabei potentiell einen Teil abzweigen. Außerdem können so leichter Rechte auf Subventionen durchgesetzt werden, da Bauern auf sich diese Konten eher leisten können und nicht auf Automaten angewiesen sind die auf dem Land nicht verbreitet sind. Bisher haben viele keine Konten die sie für solche Subventionen angeben können.

Außerdem muss berücksichtigt werden, dass oftmals nur GSM-fähige(2G) Handys verfügbar und die Nutzer zum Teil Analphabeten sind. Somit stehen nur bedingt gute technische Verbindungen zwischen Handys zur Verfügung und übertragene Zeichenketten sollten sich auf Numerische Zeichen beschränken, da diese von Analphabeten leichter weiter zu geben sind als Buchstaben.

Bisher bestehende Services wie M-PESA haben eine Reihe von Nachteilen, wie zum Beispiel Netzzwang und zusätzliche, regelmäßig anfallende Gebühren für SMS. Auch sind sie meistens daran gebunden, dass beide Handelspartner Kunden derselben Bank sind.

3 M-PESA

M-PESA ist der derzeit größte Anbieter von Mobile Banking in Entwicklungsländern. Der Dienst nutzt eine sogenannte SIM-Extension und ist somit auf fast jedem Handy einsetzbar. Der größte Nachteil ist jedoch, dass immer Zugang zum Mobilfunknetz bestehen muss, da der Dienst SMS für jede einzelne Transaktion nutzt. Beworben wird der Dienst unter anderem damit, dass keine regelmäßigen Kontoführungsgebühren anfallen. Jedoch sind die Gebühren pro Transaktion nicht zu verachten, da bis zu 66% pro Zahlung entstehen. Darüber hinaus kommen für jede Transaktion die Kosten für die SMS hinzu, die dafür genutzt werden. Die Nutzung lohnt sich somit offenbar nur, wenn man seltene Zahlungen hat. Ein großer, ursprünglich so nicht vorgesehener Nutzen des Dienstes ist die Verwendung als elektronische Geldbörse. Eigentlich nur für Prepaid Guthaben vorgesehen, erlaubt der Dienst das aufladen und auszahlen von beliebigen Beträgen, natürlich mit entsprechenden Gebühren von bis zu 20%. Auf diese Weise ist das Guthaben von vielen, für die sich Bankkonten aufgrund der Kontoführungsgebühren oder der schlechten Verbreitung von Geldautomaten nicht lohnt, auch auf Reisen sicher. (Safaricom, 2018; GSMA, Hughes, & Lonie, 2007)

4 WAS SIND SMAPS GENAU UND WIE FUNKTIONIEREN SIE?

SMAPs verwenden zur Authentifizierung kurze Ketten von unter fünfzehn Zeichen, sodass im Fall, dass keine technische Möglichkeit zur Übertragung besteht, auch eine mündliche Übertragung möglich ist. Dazu kommt, dass sie mit fehlendem oder schwachem Netz ausgezeichnet umgehen können.

Sie basieren auf dem Austausch von Nonces sowie Challenge-Response und verzichten weitestgehend auf Kommunikation mit dem Provider (Sam) der nur über Internet erreichbar ist.

Über die Entwicklung von Angriffsstrategien wurde die Struktur so weit verbessert, dass sich Angriffe nicht mehr lohnen sofern sie überhaupt praktikabel wären.

4.1 ABLAUF

Zunächst bauen Alice und Bob eine Vertrauenssituation auf und einigen sich auf den zu zahlenden Betrag. Dann wird der Betrag von Alice Konto abgezogen und schließlich Bob gutgeschrieben. Dieser Austausch bezieht sich zunächst auf die lokale elektronische Geldbörse und wird beim nächsten Kontakt zum Internet mit dem Schattenkonto beim Provider Sam synchronisiert (Abbildung 1).

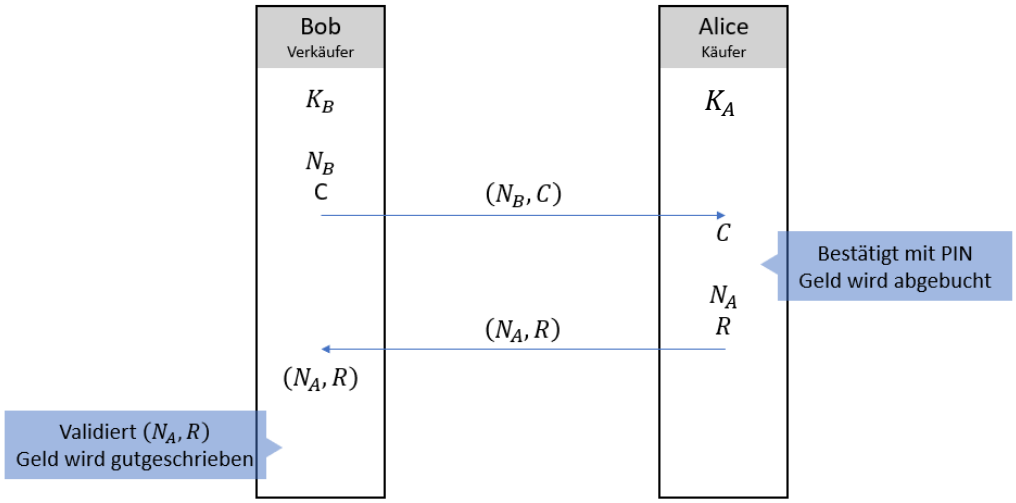

Abbildung 1: Grober Ablauf des Protokolls. K_B steht hier für den symmetrischen Schlüssel von Bob und Sam der der sicheren Kommunikation dieser beiden Parteien dient. Bob verfügt darüber, spielt hier aber keine Rolle. K_A ist Alice' Version von K_B. N steht für die von Bob generierte Nonce und C für die daraus aufgestellte Challenge. Nonce und Challenge werden an Alice übertragen die Nonce N_A und Response R an Bob zurückschickt.

4.1.1 Ausführlich

In seiner einfachsten Ausführung sieht das Protokoll vor, dass zunächst die Telefonnummern ausgetauscht werden, anhand derer sich beide identifizieren. Dann generiert der Händler Bob eine Nonce und eine Mac und zeigt sie dem Kunden Alice. Die Mac wird dabei über eine Hashfunktion berechnet, die die Telefonnummern von Bob und Alice sowie den Betrag auf den sich beide geeinigt haben und die Nonce als Bestandteile hat, dann mit dem gemeinsamen Key verschlüsselt wird und schließlich über eine Modulo-Funktion auf drei Zeichen gekürzt wird und damit so kurz das sie ohne weiteres fehlerfrei auch mündlich weitergegeben werden kann. Daraufhin nimmt Alice die Nonce und Mac, gibt diese ein und lässt ihre SIM validieren. Dazu berechnet diese ihrerseits mit derselben Funktion die Mac und vergleicht die Ergebnisse. Stimmen ihr Ergebnis und Bobs überein, glaubt Alice Bob, dass er den selben Preis erwartet und autorisiert die Zahlung mit ihrem Pin. Nun wird das Geld

von ihrem Konto abgebucht und Alice generiert ihrerseits eine Nonce und eine Mac-Response die Bob bei sich eingibt, verifiziert und in der Folge das Geld gutgeschrieben bekommt. Alice‘ Response-Mac ist vier Zeichen lang. Sie berücksichtigt neben den Telefonnummern und der Nonces auch noch Bobs Challenge um auf diese Weise Bobs SIM glaubhaft zu versichern, dass dies die Antwort auf genau diese ursprünglich Challenge ist und die Antwort nicht von einer dritten Partei kommt.

Die Verzögerung in der Kommunikation mit der Bank / dem Provider Sam wird über den Bug im Needham-Schroeder Protokoll (siehe Abschnitt 4.2) ermöglicht, der in der ursprünglichen Fassung keine Zeitbindung enthält wodurch eine beliebige Verzögerung der Antworten ermöglicht wird.

Beide Parteien könne die Transaktionen der Bank mitteilen, da beide über alle nötigen Transaktions-Logs verfügen. Auf diese Weise reicht es, wenn einer von beiden regelmäßigen Internet Zugang erhält. Außerdem wird auf diese Weise sichergestellt das keiner betrügt, da anhand der Informationen beider Parteien nachgewiesen werden kann, wer manipuliert hat.

Dieser Ansatz zeigt zunächst, dass es möglich ist, über kurze Zeichenketten eine Authentifizierung zu ermöglichen, jedoch ist diese auf diese einfache Weise noch einer ganzen Reihe Angriffsmöglichkeiten ausgesetzt. Diese werde ich im Folgenden ansprechen und entsprechende Gegenmaßnahmen erläutern.

4.2 NEEDHAM-SCHROEDER PROTOKOLL

Das Needham-Schroeder Protokoll dient der Akquise eines Shared Keys für Bob und Alice von Sam ohne Kommunikation zwischen Sam und Bob. Es wurde 1978 von R. Needham und M.D. Schroeder am MIT entwickelt.

Das Protokoll sieht vor das zunächst Alice an Sam eine Anfrage stellt, die eine Identifikation von Alice und Bob enthält sowie eine Nonce. Sam erstellt daraufhin den gewünschten Schlüssel und schickt diesen zurück an Alice. Diese Nachricht ist nun mit dem gemeinsamen Schlüssel von Alice und Sam verschlüsselt und enthält neben der Nonce, Bobs Identifikation und dem Schlüssel außerdem einen Block den Alice selbst nicht lesen kann, da er mit dem gemeinsamen Schlüssel von Sam und Bob entschlüsselt ist. Dieser enthält seinerseits Alice‘ Identifikation und den neuen Schlüssel.

Alice entschlüsselt im folgenden Schritt nun den ihr zugänglichen Teil und schickt den für Bob bestimmten Block an diesen weiter.

Bob kann diesen entschlüsseln und schickt eine neue Nonce verschlüsselt mit dem neuen gemeinsamen Schlüssel an Alice. Diese verändert die entschlüsselte Nonce in dem sie diese um Eins erhöht und schickt sie erneut verschlüsselt an Bob zurück. Nun sind beide Parteien sicher, dass der neue gemeinsame Schlüssel gilt und der Partner vertrauenswürdig ist. (HU-Berlin, 2003)

4.2.1 Der ‚Bug‘

Der Bug beschreibt eine Replay-Attacke durch die ein Angreifer Eve Bob zwingen kann einen alten Key, den der Eve bereits geknackt hat, wieder zu verwenden. Da in der Nachricht von Alice an Bob kein Zeitstempel und auch keine von Bob ausgesuchte Nonce enthalten ist, kann Bob nicht verifizieren, dass diese Nachricht neu ist (Lowe, An Attack on the Needham-Schroeder Public-Key Authentication Protocol, 1995). Wird die von Lowe beschriebene Ergänzung des Protokolls zur Lösung des Bugs integriert, die vorsieht einen Zeitstempel in diese Nachricht zu implementieren (Lowe, Breaking and Fixing the Needham-Schroeder Public-Key Protocol using FDR, 1996), kann, zumindest solange das Protokoll eine kurze Lebensdauer der Nachricht vorsieht, ein Zeitversetzter Abschluss nicht sicher erfolgen.

5 ENTWICKLUNG

5.1 ÜBERSICHT ÜBER ANGRIFFE

In seiner einfachsten Ausführung ist das Protokoll für eine ganze Reihe von Angriffen anfällig. Die meisten davon gehen vom Händler Bob aus, der sich auf diesem Weg mehr Geld gutschreiben lässt als die Kunden eigentlich zahlen, es gibt jedoch auch Angriffe, die eine Reihe von Alices, Bobs und sogar Sam einschließen kann.

Der erste Angriff geht von Bob aus und hat zum Ziel mehr Geld zu bekommen als mit Alice ausgemacht. Hier verliert Alice jedoch kein zusätzliches Geld, sondern Bob generiert theoretisch neues Geld da er sich nur mehr gutschreiben lässt während Alice nur der ausgemachte Betrag abgebucht wird.
Dieser Angriffe macht sich zunutze, dass die kurzen Zeichenketten schnell Kollisionen verursachen. Dabei sucht Bob nach einer Kollision zwischen den originalen Daten der Nonce, dem Preis und einem veränderten (höheren) Preis. So kann er Alice mit einer korrekten Challenge und Nonce den Anschein eines korrekten Ablaufs vermitteln, bekommt aber selber mehr Geld gutgeschrieben da er seinem eigenen Handy einen höheren Preis gegeben hat. Diese Methode kostet Alice kein Geld da nur der abgemachte Betrag abgebucht wird. Diese Kollisionsberechnungen gehen auch auf langsameren Smartphones sehr schnell und es ist kaum eine Verzögerung zu bemerken.

Der nächste Angriff geht ebenfalls von Bob aus und benötigt überhaupt keine Alice. Hier wird versucht die korrekten Antworten von Alice zu erraten. Dieses Vorgehen ist bei so kurzen Zeichenketten durchaus schnell erfolgreich und ermöglicht immerhin eine Erfolgswahrscheinlichkeit von 10^{-4}, also Erfolg in einem von 1000 Versuchen.

Ein weiterer Angriff sieht vor, die Antworten von Alice zu sammeln und eine Liste anzulegen mit korrekten Antworten auf bestimmte Nonces (Abbildung 2). Auf diese Weise kann Bob ohne Alice‘ zutun Geld von ihr an sich überweisen. Auch hier bekommt Bob das Geld gutgeschrieben ohne das Alice etwas verliert. Die Erfolgswahrscheinlichkeit liegt hier sogar bei $n \cdot 10^{-3}$ wobei n für die Anzahl der aufgezeichneten Antworten steht. Hier sind also n mal 1% der Versuche erfolgreich.

Im Verlauf der Lösungssuche werden aufgrund dieser Ansätze weitere Angriffe möglich werden, die dann an den entsprechenden Stellen direkt angesprochen werden. Dazu gehört das Klonen von Sessions und Fake-Transaktionen.

5.2 LÖSUNGSANSÄTZE

An dieser Stelle nun werden auf die verschiedenen Angriffe die möglichen Gegenmaßnahmen untersucht. Alle angesprochenen Angriffe werden nacheinander abgearbeitet und gelöst.

5.2.1 Response raten

Ein möglicher Angriff basiert darauf nach Erzeugung der Challenge die von Alice erwartete Response einfach zu erraten. So wird keine Alice benötigt und Bob kann sich einfach Geld gutschreiben.

Da das erraten von richtigen Responses vergleichsweise einfach ist, wird ein Zähler eingeführt, der fehlerhafte Zahlungen in Folge zählt und nach einer bestimmten Anzahl sperrt. Dieser Wiederholungszähler sollte pro Handelspartner gelten da er ansonsten auch zum absichtlichen sperren durch ‚Denial of Service‘-Attacke genutzt werden könnte.

Dieser Ansatz ist jedoch sehr einfach auszuhebeln, in dem nach der entsprechenden Anzahl fehlversuchen eine valide Zahlung durchgeführt wird. Das ist insbesondere dann einfach, wenn der Betrüger der lokale Provider ist und somit Zugriff auf eine ganze Reihe von Kunden-SIMs hat, die er alle nacheinander computergesteuert testen kann. An dieser Stelle hilft dann auch die weiter unten eingeführte Session-Chain.

5.2.2 Veränderung des Preises

Wenn Bob im Verlauf des Bezahlvorgangs den Preis manipuliert, so merkt Alice davon zunächst nichts. Um Bob zum vereinbarten Preis zu zwingen, kann bei der Berechnung der Response der Parameter, der bisher die Challenge bekommen hat, direkt durch den Preis ersetzt werden. Da die Response auch mit der manipulierten Challenge zum selben Ergebnis kommt, würde die SIM von Bob keinen Unterschied feststellen können. Nur durch die direkte Inkorporation des Preises an dieser Stelle kann das gewährleistet werden.

Zum besseren Verständnis folgend ein Beispiel:
Vereinbarter Preis $X = 1{,}50€$ mit daraus berechneter Challenge $C = 45832$
Bob gibt diese Challenge an Alice weiter, sucht selbst jedoch mit einem Preis $X = 2000€$ nach einer weiteren Lösung die ebenfalls die Challenge $C = 45832$ ergibt und lässt seine eigene SIM mit dieser Lösung und dem angepassten Preis weiterarbeiten. Antwortet Alice nun mit einer Response die nur auf der Challenge basiert, kann die SIM von Bob das nicht erkennen, da im fall des manipulierten Preises die selbe Challenge erzeugt wurde. Nur wenn Alice anstatt der Challenge den abgemachten Preis in der Berechnung der Response verwendet kommt es zu einem unterschied in der Response den Bobs SIM erkennen kann.

5.2.3 Transaktions-Response speichern

Der Möglichkeit, dass Responses vergangener Transaktionen einfach gespeichert und dann später als Nachschlagewerk genutzt werden, kann am effektivsten durch die Einführung von zusätzlicher Entropie und einer Session-Chain begegnet werden.

Der Angriff funktioniert, weil bisher die Antwort auf eine bestimmte Challenge – Nonce Kombination immer dieselbe Response ergibt. Es gibt keine zusätzlichen zufälligen Elemente die jedes Mal ein anderes Ergebnis garantieren. Dem entsprechend werden die Challenge und Response so umgestaltet, dass zusätzliche Informationen integriert werden können und ein Teil nur vom Provider Sam verifiziert werden kann. Auf diese Weise kann der Provider Manipulationen aufdecken. Außerdem wird eine Session-Chain etabliert, die pro Handelspartner eine Session pflegt. Jedes Mal, wenn mit diesem gehandelt wird, wird die entsprechende Session genutzt und Counter und Chain-Status werden abgeglichen, bei Differenzen dieser Werte wird eine Manipulation schnell offenbart (Abbildung 3). Als weitere Schwierigkeit kann auch eine Variation der Länge von Challenge und Response pro Transaktion angewendet werden.

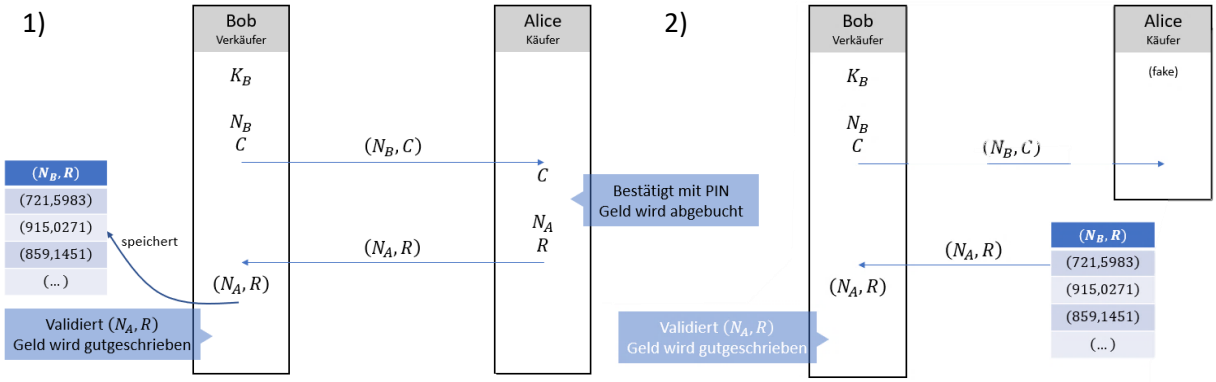

Abbildung 2: Speicherung der Antworten von Alice. Der Angriff erfolgt in zwei Schritten.

5.2.3.1 Session-Chain

Zwei Handelspartner, die noch nie miteinander gehandelt haben, initialisieren zunächst eine Session. Dazu wird der Counter auf null gesetzt und die Chain mit einem Startwert von $S_0 = H(0, T_0)$, wobei $T_0 = (A, B)$ ist, initialisiert. Beide sollten nun den selben Wert für S_0 haben. S hat hier eine Länge von 256 Bit. Dann gleichen beide den Betrag, auf den sie sich geeinigt haben, mit ihren jeweils zutreffenden Limits ab. Diese Limits ermöglichen es einzelne Transaktionen zu limitieren um beispielsweise nicht erpresst zu werden. Sie werden mit dem Provider (Sam) abgestimmt. Bob erzeugt dann eine Challenge n_c die aus drei Teilen besteht (1.). Einen Teil den Alice und Sam verifizieren können, einen den nur Sam verifizieren kann und einen Entropieteil, der sicherstellt das sich die Challenge niemals wiederholt, wie zum Beispiel einen Zeitstempel. Diese Challenge ist um die fünf Zeichen lang. Zur Sicherstellung, dass die einzelnen Abschnitte nur von den entsprechenden Parteien verifiziert werden können, für die sie gedacht sind, sind diese jeweils mit den entsprechenden Keys verschlüsselt. So ist der erste Abschnitt der Challenge mit dem gemeinsamen Schlüssel von Alice und Bob verschlüsselt und der zweite Abschnitt mit dem von Bob und Sam. Alice wird so aus diesem Abschnitt ausgeschlossen.

$$C_{i+1} = E^{n_c}_{h_{K_{AB}}(S_i)}\left[\left(Mac_{K_{AB}}(i, S_i, X)\ mod\ 10^{n_{c,i}}\right) \parallel \left(Mac_{K_B}(i, S_i, X, F_B) mod 10^{n_{c,2}}\right) \parallel N_B\right] \quad 1.$$

Alice generiert, nachdem sie diese Challenge verifiziert hat, eine Response n_r die ihrerseits zwischen 6 und zwölf Zeichen lang ist. Sie besteht ebenfalls aus drei Teilen, die von Bob und von Bob und Sam verifiziert werden und Wiederholung vermeiden. Auch hier ist die Verwendung der unterschiedlichen Schlüssel dafür verantwortlich das einen Teil der Response nur Sam verifizieren kann.

Bevor Alice diese Response nun Bob übermittelt, aktualisiert sie ihre lokalen Session-Daten. Sie zählt den Counter i hoch und aktualisiert den Status auf $S_i = H(i, T_i)$ wobei T_i nun die Transaktionsdaten enthält: $T_i = (S_{i-1}, X, C_i, R_i)$.
Außerdem wird Ihrem Konto (M_A) der Wert X abgezogen: $M_A = M_A - X$.

Nun verifiziert Bob seinerseits die Response n_r und bekommt im Erfolgsfall den abgemachten Betrag gutgeschrieben. Auch er passt seine Session an und sollte somit dieselben Werte haben wie Alice. Ein abgleich der Werte ist hier optional, da die transaktion mit den selben werten gestartet wurde und der komplette ablauf unter aufsicht beider parteien stattfand. Außerdem wird vor der nächsten Transaktion ein erneute abgleich vorgenommen.
Der Provider Sam kann nun beim nächsten Abgleich der Logs die jeweils für ihn verschlüsselten Teile abgleichen und die Transaktionen entweder akzeptieren oder revidieren. Um entsprechende

Rückzahlungen zu bekommen muss aber der jeweils andere Transaktionspartner ebenfalls ein Update von Sam erhalten.

Nun könnte Bob eine Session klonen und die Antwort von Alice mit beiden Sessions für sich nutzen. Dagegen hilft sowohl der Zähler, der fehlerhafte Zahlungen zählt, als auch die Synchronisation und Validierung mit Sam.

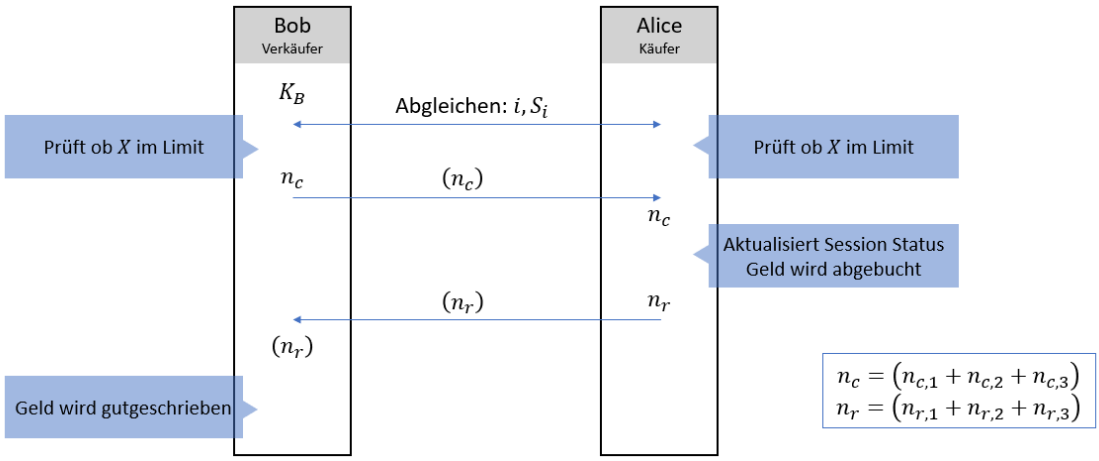

Abbildung 3: Verwendung einer Session Chain und aufgeteilten Challenges und Responses

6 KEYS

Die Kommunikation der beiden Parteien erfolgt symmetrisch. Es muss also beiden ein gemeinsamer Schlüssel bekannt sein. Die Einigung auf einen gemeinsamen Schlüssel stellt hier also die Herausforderung dar. Die Autoren dieses Papers (Baqer, Bezuidenhoudt, Anderson, & Kuhn, 2017) sind zu dem Schluss gekommen das eine Broadcast Encryption den besten Weg darstellt entsprechend sichere Schlüssel zur Verfügung zu stellen. Dabei werden Gruppenschlüssel verwendet, die jeweils die Kommunikation zwischen Gruppenmitgliedern sichern. Dazu werden alle Nutzer in 100 Key-Groups eingeteilt, die jeweils die Schlüssel zu allen anderen Gruppen, sowie zur eigenen Gruppe, enthalten. Zur eigenen Gruppe ist ein Schlüssel notwendig, wenn beide Handelspartner derselben Gruppe angehören.

Die Anzahl der Schlüssel entspricht der Anzahl der Kanten eines vollständigen Graphen plus der Anzahl der Knoten[1]. Dementsprechend lässt sich die Formel der oberen Dreiecksmatrix (2.) zunutze machen um die genaue Anzahl der Schlüssel zu ermitteln.

$$n = \frac{d(d+1)}{2} \qquad 2.$$

Bei $d = 100$ Gruppen sind also insgesamt 5050 Schlüssel notwendig um zwischen allen Gruppen und zur eigenen Gruppe eine Verbindung sichern zu können. Dabei muss jede einzelne Gruppe 100 dieser Schlüssel erhalten.

In welche Gruppe eine SIM eingeteilt wird berechnet sich aus dem private Key des SIM und der Telefonnummer. Der 128bit Schlüssel K_G den Sam jeder SIM mitgibt dient hier zunächst der

[1] Knoten entsprechen hier den Gruppen und Kanten den symmetrischen Schlüsseln

Zuordnung der einzelnen SIMs in die verschiedenen Gruppen und dient später dazu den gemeinsamen Schlüssel von Alice und Bob zu ermitteln.

Wenn nun eine Transaktion gestartet wird, wird der entsprechende Key anhand des K_G des anderen Teilnehmers ermittelt. Dieser Key ist nur in den beiden Gruppen verfügbar in denen die beiden sind. Sollte nun ein Dritter versuchen die Transaktion zu manipulieren, muss er in einer der beiden Gruppen sein, also auch beide private Keys kennen. Der nun nötige aufwand und die letztendlich geringen Beträge auf den Konten machen es jedoch uninteressant diese Informationen herauszufinden.

Außerdem kann von Bob ein gemeinsamer Key für nur diese Transaktionsteilnehmer angefordert werden, dafür kann das Needham-Schroeder Protokoll genutzt werden.

6.1 Übersicht über alle im Protokoll genutzten Schlüssel

Jede SIM enthält zunächst einmal eine eindeutige Telefonnummer. Diese wird zur Identifizierung ausgetauscht. Daneben enthalten die SIMs einen Symmetrischen Schlüssel für die Kommunikation mit dem Provider Sam K_{AS}. Außerdem hat jede SIM einen Schlüssel K_G der der Gruppenzuordnung dient und ebenfalls zur Auswahl des gemeinsamen Schlüssels genutzt wird. Und schließlich enthält jede SIM alle Group-Keys der Gruppe der sie angehört.

Außerdem können weitere symmetrische Schlüssel hinzugefügt werden die ausschließlich der Kommunikation zweier Parteien dient. Weitere Erläuterungen dazu folgen im Abschnitt 7 (Delay)

Der Provider Sam verwahrt nur die symmetrischen Schlüssel zu jeder ausgestellten SIM.

7 Delay

Wie funktioniert die Weitergabe von Daten an den Provider unabhängig vom Mobilfunknetz? Dazu muss zunächst einmal die genaue Situation geklärt werden. In viele Orten auf der Welt gibt es kaum oder garkeinen Mobilfunk. Also auch kein Internet. Um dieses Problem zu lösen und in entlegenen Gebieten Internetnutzung zu ermöglichen wurden eine Reihe von Techniken entwickelt, wie beispielsweise die „Data-mules“. Das ist eine Technik, nach der in einem Dorf ein lokales Netzwerk aufgebaut wird, das Email und ähnliches ermöglicht. Jedoch werden geschriebene Emails und andere dort erstellten Inhalte zunächst auf dem lokalen Server gespeichert. Dann kommt der „Data-mule“, ein Bus oder ähnliches, der die Daten beim Vorbeifahren einsammelt und beim nächsten Internetanschluss dann hochlädt und im Gegenzug Updates herunterlädt und ins Dorf bringt. Dies geschieht im Optimalfall automatisch. (Hasson, Pentland, & Fletcher, 2004)

Diese Technik lässt sich auch für das Mobile-Banking anwenden. Der Data-mule sammelt die Transaktionslogs von Alice ein und synchronisiert diese mit Sam. Wenn diese Logs mit dem privaten Key von Alice verschlüsselt sind, kann kein Dritter die Daten manipulieren und für Sam steht die Authentizität von Alice als Urheber fest und die Transaktionen werden mit dem Schattenkonto synchronisiert.

Wenn Alice neben dem Gruppenkey nun einen privaten Key für ihre Transaktionen mit Bob wünscht, würden sich im Normalfall beide über Internet bei Sam authentifizieren und einen entsprechenden Key beantragen. Um das nun ohne Netz Zugang zu ermöglichen könnte laut den Autoren ein Bug im Needham-Schroeder-Protokoll (Abschnitt 4.2.1) ausgenutzt werden, wodurch es bei der Kommunikation keinerlei Zeitliche Bindungen gibt. Das Protokoll erlaubt es auch, dass nur einer der

beiden mit Sam in Kontakt treten muss. Wenn Alice also diejenige mit Internet Zugang ist, reicht es, wenn sie sich mit Sam austauscht und Bob glaubt ihr schließlich dennoch, dass der Key von Sam erstellt wurde. Dazu im Folgenden nochmal der genaue Ablauf des Protokolls und wie der Bug eine Zeitverzögerung erlaubt.

Zunächst generiert Alice eine Nonce n_A und packt diese zusammen mit ihrer Telefonnummer und der von Bob in eine Nachricht, die sie mit ihrem privaten Schlüssel K_{AS} verschlüsselt. Diese übermittelt sie an Sam. Der kann diese entschlüsseln und erzeugt eine neue Nachricht mit dem gewünschten Schlüssel K_{AB} und der zuvor von Alice generierten Nonce, die für Alice sicherstellt, dass die Nachricht von Sam kommt, da nur er diese Nonce kennen kann. Außerdem enthält diese Nachricht einen mit K_{BS} verschlüsselten Teil für Bob. Diesen nimmt Alice nun mit und gibt ihn an Bob weiter.
Im Dorf angekommen gibt Alice die Nachricht von Sam an Bob weiter, der diese entschlüsselt und nun ebenfalls über den Schlüssel K_{AB} verfügt. Zur abschließenden Validierung erzeugt nun Bob eine Nonce n_B die er mit dem neuen Key verschlüsselt und an Alice schickt. Die entschlüsselt die Nachricht, verändert die Nonce $(n_B - 1)$ und schickt die veränderte Nachricht verschlüsselt mit K_{AB} zurück an Bob. (Abbildung 4) (HU-Berlin-Informatik, 2003)

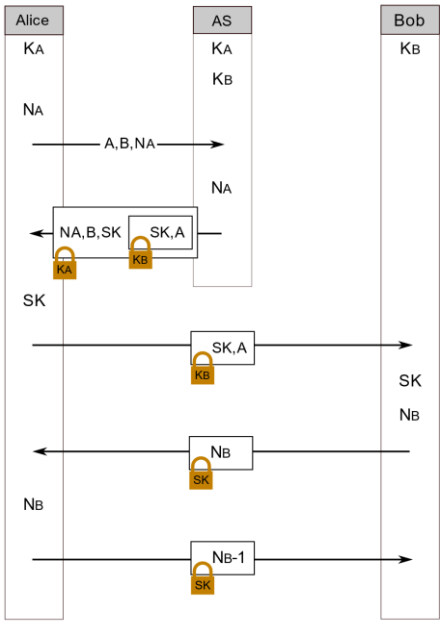

Abbildung 4: Needham-Schröder Protokoll von M.F. Schönitzer (Schönitzer, 2017)

Der in Abschnitt 4.2.1 beschriebene Bug erlaubt es einem Angreifer Bob zu zwingen einen alten Key, den der Angreifer geknackt hat, wieder zu verwenden. Wird die im Paper (Lowe, Breaking and Fixing the Needham-Schroeder Public-Key Protocol using FDR, 1996) beschriebene Lösung implementiert, so wird die Möglichkeit der Zeitversetzten Vollendung des Protokolls erschwert, da die Schritte des Protokolls innerhalb der Lebensdauer des Zeitstempels, beziehungsweise vor Ablauf dessen abgeschlossen werden müssen.

8 Zusammenfassung aller Mechanismen

Wenn nun alle Entwicklungsschritte kombiniert werden, kann folgender Ablauf daraus geschlossen werden, der nun gegen alle von den Autoren bedachten Angriffe geschützt ist.

Der Ablauf beginnt mit dem Provider Sam, der alle SIM-Karten in Key-Groups aufteilt und jeder Gruppe 100 von 5050 Keys[2] anhand des, jeder SIM eigenen private Key K_G (128bit), zuordnet. Daneben enthält jede SIM ebenfalls noch einen private Key K_{AS} (AS $\widehat{=}$ Alice-Sam, Alice steht hier stellvertretend für den jeweiligen SIM-Inhaber) der nur dieser SIM und Sam bekannt ist, um Daten mit ihm zu synchronisieren.

Wollen nun Alice und Bob ins Geschäft kommen und haben jeweils eine SIM von Sam, so tauschen sie zunächst ihre Telefonnummern A und B aus. Im nächsten Schritt werden auch die private Group-

[2] Das Zustandekommen dieser Anzahl wird in Abschnitt 6 näher erklärt.

Keys K_G[3] ausgetauscht um den gemeinsamen Schlüssel K_{AB} (3.3.) zu ermitteln. Dieser Austausch ist bei der ersten Transaktion einer Session einmalig nötig, sofern nicht anders möglich, sogar mündlich. Dieser ‚private' Schlüssel ist dadurch nicht 100% geheim, er dient lediglich der gemeinsamen Schlüsselfindung und wird nur zwischen Handelspartnern ausgetauscht.

$$K_{AB} = \begin{cases} h_{K_{g_A,g_B}}(A,B)\ wenn\ g_A \leq g_B \\ h_{K_{g_B,g_A}}(A,B)\ wenn\ g_A > g_B \end{cases} \quad 3.$$

Wenn dies ihre erste Transaktion ist, wird eine neue Session etabliert, mit dem Counter $i = 0$ und dem Status $S_0 = H(0, T_0)$[4].
Jetzt kann Bob mit dem Key K_{AB} verschlüsselt, die Challenge n_C (~ 5 Zeichen) an Alice übermitteln, in der neben dem aktuellen Session-Status, auch der ausgehandelte Betrag, sowie Zusatzinfo (wie ein Zeitstempel) F_B, die an Sam übermittelt wird, stecken.

Wie in Abschnitt 5.2.3.1 beschrieben, berechnet Alice nun ihrerseits anhand des ihr bekannten Keys K_{AB} und der übergebenen Daten ebenfalls die Challenge und vergleicht das Ergebnis. Dabei kann sie aber nur den ersten Teil ($n_{c,1}$) verifizieren, da der Rest aufgrund des dort verwendeten Keys K_{BS} abweicht. Im nächsten Schritt berechnet Alice ihrerseits eine Response n_r (~ 12 Zeichen) in der neben dem Preis auch die Challenge enthalten ist, und aktualisiert anschließend ihren Session-Status. Nun kann die Response an Bob gegeben werden, der seinerseits den ersten Teil verifiziert und das Geld gutgeschrieben bekommt. Auch er aktualisiert nun entsprechend seine Session. Bekommt einer der beiden nun in Zukunft Mobilfunkempfang werden die Transaktionsdaten an Sam übermittelt. Dieser kann nun die jeweiligen Transaktionen nachstellen und so die für ihn bestimmten Teile verifizieren.

9 VEREINFACHUNG DER ÜBERTRAGUNG

Sofern es die technische Ausstattung von Bob und Alice zulassen, kann für den Austausch der Daten ein schnellerer und sicherer Kanal genutzt werden. Beispielsweise bieten sich grafische Darstellungen an, die auch Analphabeten die Gleichheit von Challenges/Responses klar ersichtlich machen und mit einer Kamera eingelesen werden können. Aber auch bekannte Kanäle wie Bluetooth, Infrarot oder NFC würden sich anbieten. Auf diesem Wege könnten manuelle Eingaben vollkommen eliminiert werden.

[3] 128bit ≙ 40 Zeichen – könnte jedoch auch kürzer ausgelegt werden (Baqer, Bezuidenhoudt, Anderson, & Kuhn, 2017, S. 11).
[4] Nähere Erklärung dazu in Abschnitt 5.2.3.1.

10 FAZIT

Die hier beschriebenen Techniken erlauben es, sicher Geld auszutauschen, ohne dabei auf eine Authentifizierung über das Internet angewiesen zu sein, sollte neben den hier beschriebenen Angriffsszenarien weitere bestehen die erfolgreich sind, ist davon bisher nichts bekannt. Ebenfalls erlaubt die Technik durch die lokal gepflegten elektronischen Geldbörsen den Kontakt mit dem Provider auf ein Minimum zu beschränken, was die Mobilfunknetzpflicht verschwinden lässt, die andere Anbieter oder Protokolle enthalten. Die Fähigkeit mit Data-mules zu arbeiten erlaubt es sogar Hausgebundenen Personen weiterhin beispielsweise Rente zu beziehen ohne dabei auf andere, möglicherweise nicht vertrauenswürdige Personen angewiesen zu sein.

Auf der anderen Seite erlaubt die Pflege des Schattenkontos beim Provider möglichen Betrug aufzudecken und zu verhindern.

11 LITERATURVERZEICHNIS

Baqer, K., Bezuidenhoudt, J., Anderson, R., & Kuhn, M. (27. 08 2017). *cl.cam.ac.uk*. Von https://www.cl.cam.ac.uk/~kabhb2/papers/SPW24.pdf abgerufen

GSMA, Hughes, N., & Lonie, S. (2007). *gsma.com*. Von https://www.gsma.com/mobilefordevelopment/wp-content/uploads/2012/06/innovationsarticleonmpesa_0_d_14.pdf abgerufen

Hasson, A., Pentland, A. S., & Fletcher, R. (2004). DakNet: Rethinking Connectivity in Developing Nations. *Computer, 37*, S. 78-83. doi:10.1109

HU-Berlin. (02. 02 2003). Abgerufen am 04. 06 2018 von https://www2.informatik.hu-berlin.de/Forschung_Lehre/algorithmenII/Lehre/WS2002-2003/Ana_kryp_Prot/12APA/html/node5.html

HU-Berlin-Informatik. (02. 02 2003). Von https://www2.informatik.hu-berlin.de/Forschung_Lehre/algorithmenII/Lehre/WS2002-2003/Ana_kryp_Prot/12APA/html/node5.html abgerufen

Lowe, G. (1995). An Attack on the Needham-Schroeder Public-Key Authentication Protocol. *INFORMATION PROCESSING LETTERS, 56*, S. 131-133.

Lowe, G. (1996). *Breaking and Fixing the Needham-Schroeder Public-Key Protocol using FDR.* Springer-Verlag.

Safaricom. (21. 05 2018). *safaricom.co.ke*. Von https://www.safaricom.co.ke/personal/m-pesa/getting-started/using-m-pesa abgerufen

Schönitzer, M. F. (26. Januar 2017). *wikimedia.org.* Abgerufen am 21. Juli 2018 von wikimedia: https://commons.wikimedia.org/wiki/File:Symetric_Needham-Schroeder-Protocol_%E2%80%93_linear.svg