21.9.2013

**Machbarkeitsstudie**

**S/Chat**

**Verschlüsselter Android Chat**

**Elias Frantar, Wolfram Soyka, Gary Ye**

|  |  |  |  |
| --- | --- | --- | --- |
|  | *Name* | *Datum* | *Unterschrift* |
| *erstellt* | *Elias Frantar* | *17.9.2013* |  |
| *geprüft* |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Version* | *Name* | *QS* | *Datum* | *Kommentar* |
| *0.1* | *Frantar* | *Ye* | *17.9.2013* | *Einführung, Ist-Zustand(Einleitung),Meilensteinplan* |
| *0.2* | *Frantar* | *Ye* | *18.9.2013* | *Projektorganisation, Technologien, Glossar* |
| *0.3* | *Frantar* |  | *21.9.2013* | *Nutzwertanalyse, Soll-Zustand* |
| *0.4* | *Ye* | *Fr.* | *21.9.2013* | *PSP* |
| *0.5* | *Ye* |  | *22.9.2013* | *Produktfunktionen* |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# 

1)    Einleitung

Mit Aufdeckung des NSA-Skandals wird das Thema Datensicherheit zu einem essenziellen Bestandteil der Softwareentwicklung. Aktuell existieren nur sehr wenige Programme, die eine sichere Kommunikation ermöglichen. Von diesen wenigen sind wiederrum nur sehr wenige benutzerfreundlich und werden daher kaum von Privatpersonen eingesetzt.

Deshalb wollen wir in unserem Projekt eine benutzerfreundliche App entwickeln, die es den Usern ermöglicht, sicher Nachrichten zu versenden, ohne jegliches Grundwissen über Verschlüsselungen besitzen zu müssen.

2)    Ist-Zustand

Aktuell existieren nur sehr wenige Programme, die eine sichere Kommunikation ermöglichen. Von diesen wenigen sind wiederrum nur sehr wenige benutzerfreundlich und werden daher kaum von Privatpersonen eingesetzt.

2.1 Marktanalyse:

Es gibt derzeit zwar viele ähnliche Systeme, wie rentalcars oder check24, aber keines ist offen zugänglich oder käuflich erwerbbar und keines erfüllt alle Kriterien und Funktionen der Flitzer AG, deshalb muss ein neues individuelles System erstellt werden.

2.2 Trendanalyse:

Da in diesem Projekt nur eine einfache Website erstellt wird, soll es auch möglich sein das Bestellsystem sowie alle anderen Funktionen problemlos in ein komplexeres Design zu integrieren. Das System kann daher auch an andere Firmen weiterverkauft werden.

3)    Soll-Zustand

**Muss – Ziele:**

1. **Sicherheit**

Die Sicherheit der Nachrichten ist der wichtigste Punkt dieses Projektes, deshalb ist die Implementierung geeigneter Verschlüsselungssysteme (internationale Standards: RSA, AES, SHA3) und Protokolle essenziell.

1. **Bedienbarkeit**

Die App soll benutzerfreundlich und damit leicht bedienbar sein. Wichtig hierbei ist, dass ein User ohne Kenntnisse über Verschlüsselung die App problemlos und ohne Zusatzaufwand bedienen kann.

1. **Zuverlässigkeit**

Das Verschicken der Nachrichten soll zuverlässig funktionieren. D.h. konkret, dass bei guter Internetverbindung rund 95% aller Nachrichten beim ersten Versenden ankommen und 99% beim (automatischen) Wiederholen.

**Kann – Ziele:**

1. **Kompatibilität**

Die App soll von Android-Version 4.x bis 2.3 abwärtskompatibel sein ohne etwaige Funktionalitäten zu verlieren.

1. **Gruppen**

Die App soll nicht nur Nachrichten mit einer Person, sondern auch in Gruppen von mehreren Personen gleichzeitig austauschen können.

**Nicht – Ziele**:

1. **Design**

Ein spezielles modernes Design der Applikation für das extra angefertigte Bedienungselemente notwendig sind.

1. **Funktionen**

Bereitstellen spezielle Zusatzfunktionen, wie das Versenden von Bild oder Videodateien oder das Erstellen eines detaillierten persönlichen Kontaktprofils

4) Produktfunktionen

# Produktfunktionen

## Verschlüsselung

Sämtliche Nachrichten sollen verschlüsselt versendet und abgespeichert werden. Zusätzlich werden sie noch digital signiert. Dazu müssen zahlreiche verschiedene Schlüssel („Sessionkeys“) generiert werden.

/**LF010**/ **Private / Public Key-pair generieren**

Beim erstmaligen Starten der Applikation wird ein zufälliges Schlüsselpaar generiert. Der öffentliche Teil des Schlüsselpaares wird mit Kommunikationspartnern ausgetauscht. Der private Teil wird zur Entschlüsselung sämtlicher Nachrichten verwendet.

|  |  |  |  |
| --- | --- | --- | --- |
| Funktion | Nutzen | Aufwand | Priorität |
| UseCase:  *Name:* Private / Pubic-Key-pair generieren (LF010)  *Art:* Anwendungsfall  *Kurzbeschreibung:* Beim erstmaligen Starten der Applikation wird ein zufälliges Schlüsselpaar generiert. Der öffentliche Teil des Schlüsselpaares wird mit Kommunikationspartnern ausgetauscht. Der private Teil wird zur Entschlüsselung sämtlicher Nachrichten verwendet.  *Auslöser:* Bei der Hauptinitialisierung, wenn der Benutzer zum ersten Mal die Applikation startet.  *Ergebnis:* Benutzer bekommt einen persönlichen öffentlichen und privaten Schlüssel.  *Akteure:* neuer Benutzer  *Eingehende Informationen:* keine  *Vorbedingungen:* keine  *Nachbedingungen:* keine | gering | mittel | Must Have |

/**LF020**/ **Sessionkeys generieren**

Zum Verschlüsseln jeder Nachricht muss ein eigener symmetrischer Sessionkey zufällig generiert werden.

|  |  |  |  |
| --- | --- | --- | --- |
| Funktion | Nutzen | Aufwand | Priorität |
| UseCase:  *Name:* Sessionkeys generieren (LF020)  *Art:* Anwendungsfall  *Kurzbeschreibung:* Zum Verschlüsseln jeder Nachricht muss ein eigener symmetrischer Sessionkey zufällig generiert werden.  *Auslöser:* Wenn der User eine Nachricht verschicken will.  *Ergebnis:* Die Nachricht bekommt ein Sessionkey  *Akteure:* der Benutzer der eine Nachricht verschickt  *Eingehende Informationen:* keine  *Vorbedingungen:* keine  *Nachbedingungen:* keine | häufig | mittel | Must Have |

/**LF030**/ **Nachrichten signieren**

Jede zu versendende Nachricht muss digital signiert werden, damit der Empfänger ihren Absender eindeutig feststellen kann.

|  |  |  |  |
| --- | --- | --- | --- |
| Funktion | Nutzen | Aufwand | Priorität |
| UseCase:  *Name:* Private / Pubic-Key-pair generieren (LF010)  *Art:* Anwendungsfall  *Kurzbeschreibung:* Jede zu versendende Nachricht muss digital signiert werden, damit der Empfänger ihren Absender eindeutig feststellen kann.  *Auslöser:* Wenn der User eine Nachricht verschicken will.  *Ergebnis:* Die zu versendende Nachricht wird digital signiert.  *Akteure:* der Benutzer der eine Nachricht verschickt  *Eingehende Informationen:* die (???id) von dem Versender  *Vorbedingungen:* keine  *Nachbedingungen:* keine | häufig | mittel | Must Have |

## Userfunktionen

Der User kann folgende Funktionen der App benutzen.

/**LF110**/ **Neuen Schlüssel erstellen**

Ein User kann auf Wunsch ein neues Schlüsselpaar generieren lassen. Es ist zu beachten, dass alle alten Konversationen nach Benutzung dieser Funktion erst nach erneutem Schlüsselaustausch wieder fortgeführt werden können. (wollten wir nicht automatisch dessen Kontakte informieren?)

|  |  |  |  |
| --- | --- | --- | --- |
| Funktion | Nutzen | Aufwand | Priorität |
| UseCase:  *Name:* Neuen Schlüssel erstellen (LF110)  *Art:* Anwendungsfall  *Kurzbeschreibung:* Ein User kann auf Wunsch ein neues Schlüsselpaar generieren lassen. Es ist zu beachten, dass alle alten Konversationen nach Benutzung dieser Funktion erst nach erneutem Schlüsselaustausch wieder fortgeführt werden können.  *Auslöser:* der User wünscht sich ein neues Schlüsselpaar.  *Ergebnis:* der User erhält ein neues, persönliches Schlüsselpaar und alle seine Kontakte aktualisieren automatisch seinen neuen öffentlichen Schlüssel.  *Akteure:* der User, der ein neues Schlüsselpaar anfragt und dessen Kontakte  *Eingehende Informationen:* keine  *Vorbedingungen:* keine  *Nachbedingungen:* keine | selten | mittel | Optional |

/**LF120**/ **Nachrichten versenden**

Ein User kann eine Nachricht verschlüsselt an einen Kontakt verschicken. Eine Nachricht besitzt eine maximale Länge.

|  |  |  |  |
| --- | --- | --- | --- |
| Funktion | Nutzen | Aufwand | Priorität |
| UseCase:  *Name:* Nachrichten versenden (LF120)  *Art:* Anwendungsfall  *Kurzbeschreibung:* Ein User kann eine Nachricht verschlüsselt an einen Kontakt verschicken. Eine Nachricht besitzt eine maximale Länge.  *Auslöser:* der User verschickt eine Nachricht  *Ergebnis:* der Empfänger bekommt seine Nachricht  *Akteure:* der Sender und der Empfänger  *Eingehende Informationen:* keine  *Vorbedingungen:* die Nachricht überschreitet nicht die vorgegebene maximale Länge  *Nachbedingungen:* keine | häufig | gering | Must Have |

/**LF130**/ **Nachrichten anzeigen**

Beim Öffnen eines Chats mit einem anderen User werden alle alten abgespeicherten Nachrichten automatisch geladen und angezeigt.

|  |  |  |  |
| --- | --- | --- | --- |
| Funktion | Nutzen | Aufwand | Priorität |
| UseCase:  *Name:* Nachrichten anzeigen (LF130)  *Art:* Anwendungsfall  *Kurzbeschreibung:* Beim Öffnen eines Chats mit einem anderen User werden alle alten abgespeicherten Nachrichten automatisch geladen und angezeigt  *Auslöser:* der User will die alte Nachrichten lesen  *Ergebnis:* dem User werden die alten Nachrichten angezeigt  *Akteure:* der User  *Eingehende Informationen:* der User, von dem er die Nachrichten sehen will.  *Vorbedingungen:* keine  *Nachbedingungen:* keine | häufig | gering | Must Have |

/**LF140**/ **Nachrichten löschen**

Ein User kann eine seiner gespeicherten Nachrichten löschen. Die Nachricht wird natürlich nur auf seinem Gerät gelöscht nicht aber auf dem des Kommunikationspartners.

|  |  |  |  |
| --- | --- | --- | --- |
| Funktion | Nutzen | Aufwand | Priorität |
| UseCase:  *Name:* Nachrichten löschen (LF140)  *Art:* Anwendungsfall  *Kurzbeschreibung:* Ein User kann eine seiner gespeicherten Nachrichten löschen. Die Nachricht wird natürlich nur auf seinem Gerät gelöscht nicht aber auf dem des Kommunikationspartners.  *Auslöser:* der User will die Nachrichten löschen  *Ergebnis:* die Nachrichten von einem Versender werden gelöscht  *Akteure:* der User  *Eingehende Informationen:* der User, von dem er die Nachrichten löschen will.  *Vorbedingungen:* Nachrichten müssen vorhanden sein  *Nachbedingungen:* keine | gering | gering | Must Have |

/**LF150**/ **Kontakte hinzufügen**

Ein User kann einen neuen Kontakt durch Eingabe dessen Publickeys seiner Kontaktliste hinzufügen.

|  |  |  |  |
| --- | --- | --- | --- |
| Funktion | Nutzen | Aufwand | Priorität |
| UseCase:  *Name:* Kontakte hinzufügen (LF150)  *Art:* Anwendungsfall  *Kurzbeschreibung:* Ein User kann einen neuen Kontakt durch Eingabe dessen Publickeys seiner Kontaktliste hinzufügen.  *Auslöser:* der User fügt einen neuen Kontakt hinzu  *Ergebnis:* der neue Kontakt wird zu seiner Kontaktliste hinzugefügt  *Akteure:* der User  *Eingehende Informationen:* der neue Kontakt  *Vorbedingungen:* der neue Kontakt darf nicht in der Kontaktlisten vorhanden sein  *Nachbedingungen:* keine | mittel | gering | Must Have |

/**LF140**/ **Kontakte löschen**

Ein User kann einen Kontakt aus seiner Kontaktliste löschen. Dadurch werden auch sämtliche mit diesem Kontakt ausgetauschte Nachrichten gelöscht.

|  |  |  |  |
| --- | --- | --- | --- |
| Funktion | Nutzen | Aufwand | Priorität |
| UseCase:  *Name:* Kontakte löschen (LF140)  *Art:* Anwendungsfall  *Kurzbeschreibung:* Ein User kann einen Kontakt aus seiner Kontaktliste löschen. Dadurch werden auch sämtliche mit diesem Kontakt ausgetauschte Nachrichten gelöscht.  *Auslöser:* der User will einen Kontakt von der Kontaktliste entfernen  *Ergebnis:* der zu löschende Kontakt wird von der Kontaktliste entfernt.  *Akteure:* der User  *Eingehende Informationen:* der zu löschende Kontakt  *Vorbedingungen:* der zu löschende Kontakt muss auf der Kontaktliste des Benutzers vorhanden sein  *Nachbedingungen:* keine | gering | gering | Must Have |

5) Machbarkeit

5.1 Technische Machbarkeit

5.1.1 Technologien:

5.1.1 Verschlüsselungsalgorithmen:

Für dieses Projekt werden kryptografische Algorithmen verschiedenster Art verwendet werden. Konkret werden ein Verfahren für den Schlüsselaustausch, ein symmetrisches Verschlüsselungsverfahren, ein digitales Signaturverfahren und eine kryptografische Hashfunktion benötigt.  
Alle im folgenden Abschnitt behandelte Algorithmen gelten als sicher, d.h. sie werden schon über ein längeren Zeitraum verwendet, von Experten analysiert und es wurden bisher noch keine Sicherheitslücken entdeckt.   
  
*(Alle im folgenden Abschnitt behandelte kryptografische Begriffe können im Glossar gefunden werden)*5.1.1.1 Schlüsselaustauschverfahren:

Grundsätzlich gibt es für den sicheren Schlüsselaustausch zwei verschiedene Möglichkeiten: ein *asymmetrisches Verschlüsselungsverfahren* oder der *Diffie-Hellman-Schlüsselaustausch*.  
Letzterer hat den entscheidenden Nachteil, dass mit jedem einzelnen Kommunikationspartner ein eigener gemeinsamer Schlüssel generiert werden muss. Noch dazu kommt, dass ein asymmetrisches Verfahren gleichzeitig zur Verschlüsselung, *Signierung* und *Authentifizierung* verwendet werden kann. Deshalb wird ein asymmetrisches Verfahren verwendet werden.

Hier gibt es eigentlich auch nur zwei infrage kommende Möglichkeiten: der bewehrte und standardisierte *RSA* oder ein Verfahren auf elliptischen Kurven basierendes Verfahren (*ECC*).  
Erstere wird schon seit den 90iger-Jahren von zahlreichen Kryptoprodukten eingesetzt und hat bisher noch keine (bei fehlerfreier Implementierung) sicherheitskritische Schwäche offenbart. Ein Nachteil dieses Verfahrens ist aber, dass sehr lange Schlüssel benötigt werden (2048bit) um eine hohe Sicherheit zu garantieren. Des Weiteren ist die Schlüsselgenerierung keinesfalls trivial, da hierzu sehr lange voneinander unabhängige zufällige Primzahlen gefunden werden müssen. Im Gegensatz dazu hat ein ECC-Verfahren den entscheidenden Vorteil, dass es vergleichbare Sicherheit mit wesentlich geringerer Schlüssellänge und damit wesentlich höherer Performance ermöglicht. Die Kryptografie auf Basis elliptischer Kurven ist aber relativ neu und wurde bisher kaum eingesetzt und ist daher vergleichsweise weniger analysiert und damit weniger bewehrt als RSA. Außerdem gibt es in diesem Bereich noch kaum international anerkannte Standards. Aus diesem Grund werden wir in unserer Applikation das RSA-Verfahren einsetzen, um ein höheres Vertrauen der User in unsere Applikation zu erlangen.

5.1.1.2 Symmetrisches Verschlüsselungsverfahren:

Grundsätzlich gibt es *Blockchiffren* und *Stromchiffren*, wobei erste viel bewehrter sind und ein Schlüssel (unter richtiger Anwendung) ohne Sicherheitsrisiko mehrmals verwendet werden kann. Deshalb werden wir eindeutig ein symmetrisches Verfahren implementieren.

Hier gibt es sehr viele Möglichkeiten, wobei die bekanntesten und damit bewehrtesten wohl die Finalisten der *AES* (Advanced Encryption Standard)-Wettbewerbs sind. Da der Sieger dieses Wettbewerbs seit 2001 in zahlreichen Kryptoprodukten implementiert wurden und sich auch nach dem Wettbewerb trotz zahlreicher Analyseversuche keine Sicherheitslücken gefunden wurden, werden wir auch auf diesen Algorithmus (AES) setzen, da dieser wahrscheinlich auch bei den Usern auf die höchste Akzeptanz stößt.

5.1.1.3 Digitale Signatur:

Da der sowieso verwendet RSA auch als bewehrtes Signaturverfahren eingesetzt werden kann und die beste Alternative der *DSA* (Digital Signature Standard) einen weiteren Schlüssel pro Kommunikationspartner erfordert, werden wir auch hier RSA verwenden.

5.1.1.4 Hashfunktionen:

Für das Projekt wird zuerst einmal eine *schlüsselabhängige Hashfunktion* verwendet. Grundsätzlich kann jede Blockchiffre(in unserem Fall AES) in einer speziellen Betriebsart als schlüsselabhängige Hashfunktion eingesetzt werden. Deshalb werden wir auch hier den AES einsetzen, um auf zusätzliche redundante Algorithmen verzichten zu können.

Für eine schlüsselunabhängige Hashfunktion gibt es viele verschiede gute Algorithmen, wie z.B. *SHA2, SHA3, SHA-512*, die sich nicht wesentlich voneinander unterscheiden. Daher kann das Verfahren nach persönlicher Präferenz ausgewählt werden. Wir haben uns für den neuesten, 2012 gewählten Sieger der SHA(Secure Hash Algorithm) Challenge, *Keccak* bzw. SHA3, da wir nur standardisierte und von vielen Experten geprüfte Algorithmen verwenden wollen.

Die genauen Entscheidungskriterien für die einzelnen Verschlüsselungsalgorithmen sind der Nutzwertanalyse zu entnehmen.

5.1.2 Umsetzung:

Es gibt zwar schon unzählige ähnliche Systeme in der Marktwirtschaft, aber keines erfüllt alle Kriterien und Funktionalitäten, deshalb wird ein genau auf die Wünsche der Firma Flitzer AG abgestimmtes, neues Produkt entwickelt.

5.1.2.1 Betriebssystem:

Für das Betriebssystem des Servers gibt es drei verschiedene Varianten, die in Frage kommen würden: Windows-Server, Linux-Server, Unix-Server

Die Vorteile des Windows Servers liegen eindeutig in der einfachen Bedienung und der Erfahrung der Teamitglieder mit diesem System. Der Nachteil wiederum sind aber die relativ hohen Anschaffungskosten.

Der eindeutige Vorteil des Linux Servers ist natürlich, dass er opensource und kostenlos, ist. Der Nachteil aber wiederrum ist die schwierigere Bedienung und die geringere Erfahrung der Teammitglieder mit diesem System. Adi Shamir hat aber bereits einige Erfahrungen mit diesem Betriebssystem.

Der Unix Server wird oft bei professionellen Systemen eingesetzt. Er kommt für dieses Projekt aber eher weniger in Frage in Frage, da er erstens Geld kostet, zweitens das Team kaum Erfahrung damit hat (ausgenommen Adi Shamir mit Linux) und drittens er für unser System nicht notwendig ist, da es nicht mit übermäßig hohen Datenmengen operieren muss.

Die Wahl des Serverbetriebssystems ist aus der Nutzwertanalyse zu ersehen. Die vom Betriebssystem unabhängigen Protokolle werden im folgenden Abschnitt behandelt.

5.1.2.2 Software / Protokolle:

Die Website wird mit Adobe Dreamweaver entwickelt. Der Vorteil gegenüber reinem HTML besteht in der einfacheren und schnelleren Webseitenkreation und der sehr einfach und schnell zu erlernenden Bedienung. Zusätzlich werden weitere Webseitenfunktionen in PHP eingebaut. Der Vorteil gegenüber JavaScript ist, dass sämtliche Berechnungen am Server durchgeführt werden, was bei der relativ großen Datenbank vor allem bei nicht leistungsstarken Endgeräten eine wichtige Rolle für die Performance spielt. Außerdem kann PHP in einem Webbrowser nicht unbeabsichtigt deaktiviert werden. PHP ist aber definitiv schwerer zu erlernen und anzuwenden. Als Verschlüsselungssysteme werden die internationalen Standards AES, RSA, SHA3 eingesetzt. Da diese auch für Dokumente der höchsten Geheimhaltungsstufe verwendet werden, sind sie definitiv für unser Projekt ausreichend sicher. Dafür sind diese Algorithmen natürlich etwas langsamer als ältere weniger starke Verschlüsselungssysteme, wie DES und SHA2. Die Datenbank wird mit PostgreSQL entwickelt. Dieses Programm ist opensource und unser Entwicklungsteam besitzt schon einige Erfahrungen damit. Dieses System ist zwar nicht wie MySQL mit fast allen Geräten kompatibel, aber dies spielt kaum eine Rolle, da die Datenbank nur mit der Website kommunizieren muss.

Genauere Wahl der Systeme ist aus der Nutzwertanalyse zu entnehmen.

5.2)   WIRTSCHAFTLICHE MACHBARKEIT

5.2.1 Personalaufwand:

Der Umfang der Software ist relativ groß, denn es müssen eine einfache Website und eine Datenbank erstellt werden. Außerdem muss ein Server konfiguriert und mit zahlreichen Verschlüsselungssystemen gesichert werden. Dies wird einiges an Durchführungszeit benötigen.

Die Gesamtstundenanzahl beträgt ungefähr 200 Personenstunden. Diese werden gleichmäßig auf alle vier Projektmitglieder aufgeteilt, um möglichst effizientes Arbeiten zu gewährleisten.

5.2.1 Investitionsaufwand:

Die Investitionsaufwände sind relativ gering, da der Großteil der verwendeten Softwarepakete opensource und damit kostenlos ist. Die einzigen Anschaffungskosten sind der Kauf des Adobe – Dreamweavers, der ungefähr 100 Euro kostet. Das Projekt wird in Wien in Räumlichkeiten der Flitzer AG mit Firmencomputern durchgeführt. Daher fallen hier keine zusätzlichen Kosten an.

5.2.3 Nutzen:

Das System wird ausschließlich der Firma Flitzer AG gehören. Die Website ist für alle Internetbenutzer zugänglich und das Bestellsystem kann von allen Personen mit Führerschein, die das Fahrzeug in Österreich abholen, in Anspruch genommen werden.

5.2.4 Risikoanalyse:

Bei diesem Projekt sind Risiken nicht auszuschließen. Ein Risikofaktor ist der krankheitsbedingte Ausfall eines Teammitglieds. In diesem Fall werden die anderen Teamitglieder den Teil des Abwesenden übernehmen. Dabei würde es höchstwahrscheinlich zu einer Verzögerung des Projektabschlusses kommen. Krankheiten sind absolut zufällig und können nicht wirklich verhindert werden, aber da das Projekt im Winter stattfindet wird natürlich für einen beheizten Arbeitsraum gesorgt.

Ein weiteres nicht vorhersehbares Problem ist ein Virus auf den Arbeitsgeräten. Dies kann verheerende Folgen für das Projekt haben. Deshalb werden die Daten täglich nach Arbeitsschluss extern und zusätzlich auf einer Cloud gesichert, um die Auswirkungen eines Virus möglichst gering zu halten. Sollten die aktuellen Daten verloren gehen, werden die zuletzt gesicherten wiederhergestellt und weiterverwendet.

Eine andere Projekterschwerung könnten Netzwerkprobleme verursachen. Es wird daher hauptsächlich (in den Bereichen, in denen es möglich ist) mit LAN anstatt WLAN gearbeitet. Sollten trotzdem Probleme auftreten werden, die LAN – Kabel ausgetauscht. Bei einem Netzwerkproblem, bei dem LAN keine Alternative darstellt, muss der Netzwerkadministrator das Problem beheben und das Projekt muss bis zu diesem Zeitpunkt angehalten werden.

Eine weitere Fehlerquelle können die Server darstellen. Deshalb werden zusätzliche Server zur Sicherheit reserviert, sodass im Eintrittsfall auf einen alternativen Server gewechselt werden kann. Sollten auch diese ausfallen, muss das Projekt bis zu Reparatur dieser durch den Serverinhaber pausiert werden. Dieser Fall ist aber sehr unwahrscheinlich

5.3) Persönliche Machbarkeit

Der schwierigste Teil, das Sichern des Systems durch kryptographische Algorithmen, stellt kein Problem dar, denn dafür sind in unserem Team drei absolute Experten zuständig. Das Wissen über die Webseiten- und Datenbankkreation ist auch vorhanden. Das Team hat zwar nicht sehr viele Erfahrungen in diesen zwei Bereichen, aber die Aufgabe ist nicht zu komplex, daher wird dies kein Hindernis darstellen.

5.4) Nutzwertanalyse

5.4.1 Schlüsselaustauschverfahren:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Schlüsselaustauschverfahren** | | RSA | | Diffie-Hellman | | ECC | |
| Kriterium | % | Rang | G \* R | Rang | G \* R | Rang | G \* R |
| Sicherheit | 20 | 1 | 20 | 1 | 20 | 1 | 20 |
| Performance | 10 | 2 | 20 | 2 | 20 | 1 | 10 |
| Schlüssellänge | 10 | 3 | 30 | 2 | 20 | 1 | 10 |
| Bewährtheit | 40 | 1 | 40 | 2 | 80 | 3 | 120 |
| Schlüsselmanagement | 20 | 1 | 20 | 3 | 60 | 1 | 20 |
| Gesamt | 100 | **1** | 130 | 3 | 200 | 2 | 180 |

5.4.2 Symmetrisches Verschlüsselungsverfahren:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Symmetrisches Kryptoverfahren** | | AES | | Twofish | | RC6 | |
| Kriterium | % | Rang | G \* R | Rang | G \* R | Rang | G \* R |
| Sicherheit | 30 | 1 | 30 | 1 | 30 | 1 | 30 |
| Performance | 30 | 1 | 30 | 3 | 90 | 2 | 60 |
| Bewährtheit | 40 | 1 | 40 | 2 | 80 | 2 | 80 |
| Gesamt | 100 | **1** | 70 | 3 | 200 | 2 | 170 |

5.4.3 Hashfunktion:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Hashfunktion** | | SHA-3 | | SHA-512 | | SHA-2 | |
| Kriterium | % | Rang | G \* R | Rang | G \* R | Rang | G \* R |
| Sicherheit | 20 | 1 | 20 | 1 | 20 | 1 | 20 |
| Performance | 30 | 1 | 30 | 3 | 90 | 2 | 60 |
| Userakzeptanz | 25 | 1 | 25 | 2 | 50 | 2 | 50 |
| Bewährtheit | 25 | 3 | 75 | 1 | 25 | 1 | 25 |
| Gesamt | 100 | **1** | 150 | 3 | 185 | 2 | 155 |

6)    Projektorganisation

Die Projektauftraggeber und Ansprechpartner dieses Projekts sind Dr. Walter Rafeiner-Magor und Erhard List.

**Gary Ye:**

|  |  |
| --- | --- |
| *Position:* | Projektleiter |
| *Spezialgebiet:* | Algorithmen |
| *Aufgabenbereich:* | Programmieren und Konfigurieren der Kommunikation über das Internet, Implementieren der Datenspeicherung, Projektleitung |

**Elias Frantar**:

|  |  |
| --- | --- |
| *Position:* | Teammitglied |
| *Spezialgebiet:* | Kryptographie |
| *Aufgabenbereich:* | Planung und Implementierung der kryptografischen Systeme, Android Entwicklung |

**Wolfram Soyka**:

|  |  |
| --- | --- |
| *Position:* | Teammitglied |
| *Spezialgebiet:* | Android |
| *Aufgabenbereich:* | Android Entwicklung, Unterstützung in allen Aufgabenbereichen |

Erhard List

Walter Rafeiner-Magor

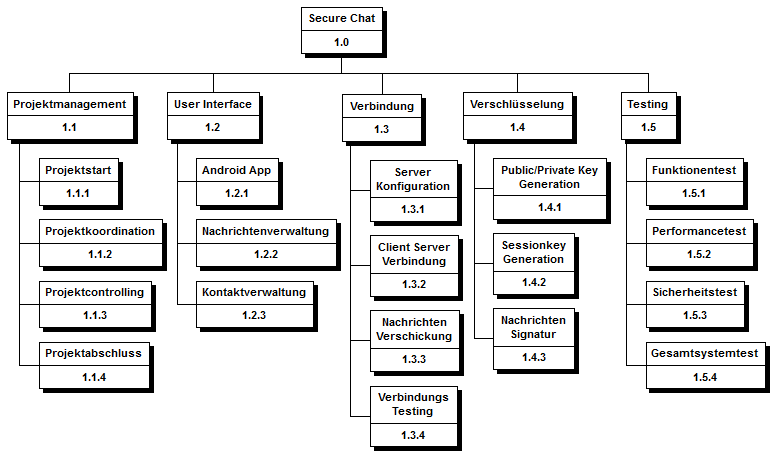
Gary Ye

Elias Frantar

Wolfram Sokya

7)    Projektplanung:

7.1) Projektstrukturplan



7.2) Meilensteinplan

|  |  |  |
| --- | --- | --- |
| **Meilenstein** | **Deliverable** | **Datum** |
| Start | Alle Hefte mit enthaltenen Informationen | 01.09.2013 |
| Android Skelett fertiggestellt | Eine App, die alle grundlegenden GUI-Elemente enthält, aber weder ein gutes Design noch Funktionen bereitstellt | 07.10.2013 |
| Versenden von Nachrichten über das Internet funktioniert | Es können Textnachrichten zwischen zwei Geräten über das Internet (unverschlüsselt versendet werden) | 21.10.2013 |
| Implementierung der Verschlüsselungssysteme abgeschlossen | Die Nachrichten können nun verschlüsselt versendet werden | 15.11.2013 |
| Implementierung der Datenspeicherung abgeschlossen | Die ausgetauschten Nachrichten können nun abgespeichert und wieder geladen werden. | 06.12.2012 |
| Gesamtsystemtest abgeschlossen | Fertige App | 15.12.2013 |

8)    Management Summary:

Die Firma Flitzer AG möchte ein Online – Bestellsystem, welches dem Kunden erlaubt 24 Stunden täglich Fahrzeuge zu mieten, einführen. Außerdem soll dieses System die Organisation automatisieren und die Papierorganisation komplett wegoptimieren.

Es soll also eine Website mit Online – Bestellsystem entwickelt werden, die eine Datenbank bei Bestellung automatisch aktualisiert. Es gibt zwar bereits sehr viele ähnliche Systeme, aber keines entspricht allen gewünschten Kriterien oder enthält alle Funktionen. Deshalb wird ein speziell abgestimmtes Produkt entwickelt.

Als Serverbetriebssystem wird Linux verwendet, da es erstens kostenlos ist und zweiten Adi Shamir, der die Serverkonfiguration übernehmen wird schon einige Erfahrungen damit hat.

Die Website wird mit Adobe Dreamweaver und PHP erstellt, da die Alternative Javascript in einigen Webbrowsern versehentlich deaktiviert werden kann. Außerdem werden alle Berechnungen bereits am Webserver durchgeführt, was schnelle Antwortzeiten auf Smartphones und weniger leistungsstarken PCs gewährleistet.

Die Umsetzung ist sowohl technisch, wirtschaftlich als auch persönlich definitiv machbar. Das Team und die Arbeitsmittel sind ausreichend. Etwaige Wissenslücken können sehr schnell und ohne großen Zeitverlust geschlossen werden. Das Know – How ist vor allem im komplexesten und schwierigsten Part, nämlich der Sicherheit des Systems, ausreichend vorhanden. Zur Verwirklichung des Projekts muss nur noch ein Programm um ungefähr 100 Euro angeschafft werden.

Das Bestellsystem wird (ausgenommen der Mitarbeiterkosten) ungefähr 100 Euro kosten. Außerdem muss noch die monatliche Miete der Website – Domäne von ungefähr 10 Euro beachtet werden.

Das komplette Projekt wird ca. 3, 5 Monate dauern. Genaue Termine sind in der Meilensteinplanung ersichtlich.

Alles in allem ist das Projekt definitiv machbar.

9)    Glossar:

* **Asymmetrische Verschlüsselung:** Ver- und Entschlüsselungen geschehen durch unterschiedliche Schlüssel (bei symmetrischen Verfahren durch denselben Schlüssel)
* **Public/Private Key-pair:** Ein asymmetrisches Verschlüsselungsverfahren benötigt immer einen öffentlichen Schlüssel zur Verschlüsselung und einen privaten Schlüssel zur Entschlüsselung von Nachrichten. Der öffentliche Schlüssel ist allen Sendern bekannt, der private aber nur dem Empfänger.
* **Session Key:** Jede Nachricht wird aus Performancegründen nur mit einem symmetrischen Sessionkey verschlüsselt, der anschließend asymmetrisch verschlüsselt wird.
* **Digitale Signatur:** Mithilfe von asymmetrischen Verschlüsselungsverfahren können Nachrichten digital signiert werden
* **Stromchiffre:** Eine Chiffrenart, die beliebig große Daten verschlüsseln kann. Dabei wird jedes Bit der Daten einzeln verschlüsselt. Ein Schlüssel kann nur einmal verwendet werden
* **Blockchiffre:** Eine Chiffrenart, die einen Datenblock bestimmter Größe verschlüsselt. Ein Schlüssel kann mehrmals verwendet werden.
* **Diffie-Hellman-Schlüsselaustausch:** Ein auf dem diskreten Logarithmus basierendes Verfahren zur geheimen Generierung eines gemeinsamen Schlüssel
* **RSA (Rivest Shamir Adelmann):** Ein asymmetrisches Verschlüsselungsverfahren dessen Sicherheit auf dem Faktorisierungsproblem beruht, mit einem privaten und einem öffentlichen Schlüssel
* **ECC (Elliptic Curve Cryptography):** Ein public-key-Kryptographieverfahren auf Basis elliptischer Kurven
* **AES (oder Rijndael):** Der Sieger des Blockchiffrenstandardisierungswettbewerbs für den *Advanced Encryption Standard*
* **SHA3 (Keccak):** Der Sieger des Einweghashfunktionsstandardisierungswettbewerbs für den *Secure Hash Algorithm 3*
* **SHA2, SHA512:** andere sichere kryptografische Einweghashfunktionen