Zentrum für sichere Informationstechnologie – Austria
Secure Information Technology Center – Austria

A-SIT

Seidlgasse 22 / 9, 1030 Wien
Tel.: (+43 1) 503 19 63–0
Fax: (+43 1) 503 19 63–66

Inffeldgasse 16a, 8010 Graz
Tel.: (+43 316) 873-5514
Fax: (+43 316) 873-5520

http://www.a-sit.at
E-Mail: office@a-sit.at

DVR: 1035461
ZVR: 948166612
UID: ATU60778947

SICHERE ECHT-ZEIT TEXTKOLLABORATION

Version 1.0 vom 03.10.2017
Florian Reimair – florian.reimair@iaik.tugraz.at

Google Docs und Office 365 sind Fixpunkte, wenn es um Cloud-gestützte Datenverarbeitung geht. Der Hype um das zeitgleiche Editieren eines einzigen Dokuments wird lediglich durch Bedenken bezüglich der Wahrung der Vertraulichkeit von Inhalten gebremst. In diesem Projekt wurde ein Überblick über aktuelle Ansätze zu sicherer Echt-Zeit Textkollaboration aus Wissenschaft und Wirtschaft gewonnen und daraus eine Liste von bisher offen gebliebenen Herausforderungen abgeleitet. Auf Basis dieser Information wird eine Architektur vorgeschlagen, die diese offen-gebliebenen Herausforderungen adressiert. Alles in allem hilft der präsentierte Demonstrator, Echt-Zeit Textkollaboration besser zu verstehen und die Diskussion über offen gebliebene Probleme zeigt die nächsten Schritte hin zu einer vollständig sicheren Lösung an.

Inhaltsverzeichnis

1. Einleitung

Google Docs und Office 365 haben sich zu Fixpunkten in der Cloud-gestützten Datenverarbeitung entwickelt. Andere Lösungen ergänzen das Angebot von Echt-Zeit Dokumentenkollaboration. Diese Lösungen funktionieren mittlerweile sehr verlässlich und werden mehr und mehr im professionellen Umfeld verwendet. Gebremst wird der Hype um das zeitgleiche Editieren von Dokumenten im Webbrowser lediglich von Bedenken zur Wahrung der Vertraulichkeit von Inhalten. Eine kryptografisch saubere Lösung gibt es noch nicht und dem professionellen Umfeld ist ein vertragliches Versprechen der Anbieter, dass sie den datenschutzbezogenen Ansprüchen der Kunden nachkommen, oft zu wenig, zumal es immer wieder zu erfolgreichen und schwerwiegenden Angriffen auf große Serviceanbieter kommt.

In diesem Projekt wurde ein Überblick über aktuelle Ansätze zu Echt-Zeit Textkollaboration mit dem Fokus auf Datensicherheit gewonnen. Es zeigen sich aber nach wie vor Unzulänglichkeiten in aktuellen Lösungen sowohl aus der Wissenschaft als auch der Industrie.
Auf Basis dieser Erkenntnisse wird in diesem Projekt eine Architektur vorgeschlagen, die viele der Unzulänglichkeiten adressiert. Eine ausführliche Diskussion des Vorschlags zeigt Vorteile, Nachteile und offene Probleme, die im Rahmen dieses Projekts nicht gelöst werden konnten.
Abschließend wird der im Projekt entstandene Demonstrator beschrieben. Der Demonstrator bedient sich des Cryptographic Service Interoperability Layers CrySIL [1] (ehemals Arbeitstitel Skytrust) um Schlüsselverwaltung, Teilung der Belange und Komfort zu gewinnen.
Die Ergebnisse des Projekts helfen, die Herausforderungen von Echt-Zeit-Textkollaboration besser zu verstehen, damit vollständigere Lösungen anzustreben und womöglich den letzten großen Schritt zu einer vollständig sicheren Kollaborationslösung zu gehen.

2. Sichere Echt-Zeit Textkollaboration

Echt-Zeit Textkollaboration hat eine unvermutet lange Vergangenheit. Bereits im Jahr 1968 im Zuge der im Nachhinein als „Mother of All Demos“ bezeichneten Präsentation hat Douglas Engelbart diese Möglichkeit aufgezeigt. EtherPad[1], eine in Scala implementierte Dokumentenkollaborationslösung, wurde 2008 veröffentlicht und nur ein Jahr später von Google gekauft und der Allgemeinheit zur Verfügung gestellt. Andere Lösungen bieten Google Docs oder auch Microsofts Office 365. Gemeinsam haben diese Lösungen alle, dass sie den Fokus sehr stark auf die Bedienbarkeit und Funktionalität legen. Vertraulichkeit und Sicherheit von Daten spielt oft eine untergeordnete Rolle. Damit sind diese Ansätze zwar interessant für zum Beispiel Privatpersonen, denen die Vertraulichkeit ihrer Daten nicht so wichtig ist, sind aber den aktuellen Anforderungen der Wirtschaft nur teilweise gewachsen. In der Forschung und auch vereinzelt in der Wirtschaft gibt’s es aber sehr wohl Ansätze, wie Kollaborationslösungen mit mehr Datensicherheit und Vertraulichkeit verwirklicht werden könnten oder wie bestehende Lösungen durch zusätzlichen Aufwand sicherer gemacht werden können.
Das zugrundeliegende Konzept einer Echt-Zeit-Textkollaboration funktioniert meist mit Hilfe von Änderungspaketen. Diese Änderungspakete gleichen einem Patch. Sie enthalten alle Information die benötigt werden, um einen Datensatz um die Daten, die im Paket enthalten sind, zu erweitern. Als Beispiel enthält ein Datensatz für eine Textkollaborationslösung den einzufügenden Text und seine Position im Dokument. Wenn also mehrere Teilnehmer und Teilnehmerinnen vom gleichen Dokument ausgehen und das Änderungspaket einpflegen, können alle Teilnehmer und Teilnehmerinnen unabhängig voneinander das Dokument so ergänzen, dass die Dokumente aller Teilnehmer und Teilnehmerinnen wieder ident sind.
Wenn also ein Teilnehmer oder eine Teilnehmerin ausreichend Inhalte produziert hat, wird ein Änderungspaket geschnürt. Dieses Änderungspaket wird an ein Service gesendet. Dieses Service überprüft, ob das vorliegende Änderungspaket mit einem anderen eben eingetroffenen Änderungspaket im Konflikt liegt. Gibt es einen Konflikt, d.h. ändern zum Beispiel zwei Teilnehmer oder Teilnehmerinnen zur selben Zeit den selben Bereich eines Dokuments aber die Inhalte unterscheiden sich, versucht das Service, diesen Konflikt zu beheben. Jetzt verteilt das Service eine Zahl von bei Bedarf angepassten Änderungspaketen, die der jeweilige Teilnehmer oder die jeweilige Teilnehmerin wieder konfliktlos in seinen oder ihren lokalen Datensatz einpflegen kann. Persistiert wird ein Dokument meist vom Service selbst mit dem Nebeneffekt, dass neuen Teilnehmern oder Teilnehmerinnen gleich der gesamte Datensatz gesendet werden kann.
Mittlerweile haben sich mehr und mehr Anwendungsbereiche für Echt-Zeit-Kollaboration ergeben und sich Lösungen dafür gefunden. Zu den anfänglichen Textdokument-basierten Lösungen haben sich Lösung für Tabellenkalkulation gesellt. Aber auch Lösungen für Mindmaps[2], Diagramme[3] und GUI Designer[4] finden sich bereits im Produktivbetrieb.
Alles in allem steigt der Bedarf an Echt-Zeit-Kollaborationslösungen an. Einzig Datensicherheits- und Vertraulichkeitsbedenken scheinen den Durchbruch solcher Lösungen zu behindern.

[1] https://web.archive.org/web/20100102003829/http://etherpad.com/ep/about/company
[2] https://www.mindmeister.com
[3] https://www.gliffy.com
[4] http://www.protoshare.com

3. Bestehende Ansätze

Nachfolgend wird ein Überblick über textbasierte Ansätze für Kollaborationslösungen mit dem Fokus auf Sicherheit und Vertraulichkeit von Daten dargestellt. Es handelt sich dabei hauptsächlich um wissenschaftliche Ansätze. Es gelang nur lediglich eine Lösung zu finden, die produktiv verwendbar ist.
SafeGDocs [2] und SeGoDoc [3] sind Ansätze, die auf existierenden (unsicheren) Kollaborationslösungen aufbauen. Die beiden Ansätze, unter anderen, fangen Änderungspakete ab und leiten diese erst nach einem Verschlüsselungsschritt an den jeweiligen (originalen) Service weiter. Dieser Service kann die übertragenen Daten somit nicht mehr lesen und ein höheres Datensicherheits- und –vertraulichkeitsniveau ist erreicht. SafeGDocs, genauso wie SeGoDoc verwenden Google Docs als Kollaborationsservice und bedienen sich eines Browserplugins, um die Änderungspakete abzufangen und verschlüsselt weiterzusenden. Beide Lösungen eignen sich aber nur bedingt für Echt-Zeit Kollaboration zumal zum Beispiel SafeGDocs Änderungen erst nach einer manuellen Speicher-Prozedur weiterleitet. CipherApps [4], ein weiterer Ansatz der auf Google Docs aufbaut bietet zusätzlich zu den eben diskutierten Ansätzen MACs, es wird aber weder Echt-Zeit-Fähigkeit noch Schlüsselverwaltung dezidiert angesprochen.
SPORC [5], ein eigenständiges Framework, hält sich an das generelle Konzept von Echt-Zeit-Kollaboration, reduziert den zentralen Server aber soweit, dass dieser lediglich für die Verteilung von (von den Teilnehmern und Teilnehmerinnen verschlüsselten) Änderungspaketen zuständig ist. Schlüsselmanagement wird genauso behandelt wie Nachvollziehbarkeit und Authentizität. SPORC ist damit hervorragend geeignet für Echt-Zeit-Kollaboration. Einziges Manko ist, dass SPORC etwaige Konflikte mit allen Nebenwirkungen den Teilnehmern und Teilnehmerinnen selbst überlässt.
Codox[5], ein Add-on für Microsoft Word und auch SubEhtaEdit[6] bedienen sich einer Lösung [6], die auf P2P Netzwerken basiert. Die Autoren geben an, dass, solange zwischen den Knoten des Netzwerks eine gesicherte Verbindung gegeben ist die Daten selbst auch sicher und privat bleiben. Wie auch SPORC leidet auch diese Lösung unter der Eigenschaft, dass jeder Teilnehmer und Teilnehmerinnen Konflikte für sich selbst lösen muss. Das vorgeschlagene Einfügen von „Super-Knoten“, die Konflikte behandeln, senkt die Datensicherheit und –vertraulichkeit deutlich. Die schlussendlich vorgeschlagene Lösung basiert auf einem Sperr-Mechanismus. Ein Textabschnitt darf nur von einem Benutzer geändert werden. Der Bedienkomfort wird dadurch deutlich geschmälert.
Es gibt zusammenfassend meist wissenschaftliche Ansätze, wie eine Echt-Zeit-Kollaboration umgesetzt werden könnte und gleichzeitig die Datensicherheit und –vertraulichkeit gewahrt werden kann. Leider haben alle beleuchteten Ansätze ihre Schwachstellen.

4. Offen-gebliebene Herausforderungen aktueller Ansätze

Die eben diskutierten Ansätze zur sicheren Echt-Zeit-Kollaboration sind auf einem guten Weg zu echter Datensicherheit, haben aber noch Schwachstellen, die einen kryptografisch beweisbare Vertraulichkeit von Daten bislang verhindern. In diesem Kapitel werden die offen-gebliebenen Herausforderungen diskutiert. Als Quelle dienen sowohl die eben diskutierten Ansätze als auch andere Ansätze, die den diskutierten ähnlich sind.
Die wohl größte Herausforderung bleibt das Schlüsselmanagement. Viele Ansätze diskutieren diesen wichtigen Aspekt einer Lösung gar nicht, andere verwenden einfache Lösungen wie Passwörter um entsprechende Schlüssel abzuleiten. Fakt ist, dass kompromittierte (also verlorene oder öffentlich gewordene) kryptografische Schlüssel den Zweck einer Lösung – nämlich ein erhöhtes Sicherheitslevel - zunichtemachen. Lösungen, die sich auf Public Key Kryptografie verlassen, mindern dieses Risiko insofern, als hier nie privates Schlüsselmaterial ausgetauscht werden muss. Die Nachteile einer solchen Lösung zeigen sich aber, sobald sich die Menge der Teilnehmer und Teilnehmerinnen an einer Kollaboration ändert. Öffentliche Schlüssel müssen ausgetauscht, das Vertrauen in diese neuen Schlüssel geprüft, der bestehende Datensatz so angepasst werden, dass auch ein neuer Teilnehmer oder eine neue Teilnehmerin mit seinem oder

[5] http://codoxware.com
[6] http://codingmonkeys.de/subethaedit

ihrem Schlüssel den Datensatz lesen und auch etwas beitragen kann. All dies macht Lösungen, die Public Key Kryptografie verwenden, sicherer, aber auch langsamer, komplexer und damit fehleranfälliger.
Eine weitere Herausforderung ist Leistungsfähigkeit, im Speziellen „Echt-Zeitfähigkeit“. Als Echt-Zeitfähigkeit wird hier verstanden, dass ein Beitrag eines Teilnehmers oder einer Teilnehmerin in einer angemessenen Zeit (meist wenige 10ms) bei allen anderen Teilnehmern und Teilnehmerinnen ankommt. Nur so ist eine reibungsfreie und komfortable Zusammenarbeit möglich. Kryptografie, die es ermöglicht, Daten so zu schützen, dass es einem Angreifer nicht leicht möglich ist, die Daten mitzulesen, hat aber die Eigenschaft, verhältnismäßig viel Zeit zu brauchen. Wo symmetrische Verschlüsselungsalgorithmen, wie z. B. AES, mittlerweile mit Hardwarebeschleunigung und anderen Optimierungen recht schnell sind, bringen asymmetrische Verfahren wie RSA deutliche Performanceeinbußen mit sich. Dazu kommt, dass Verschlüsselung von sehr kurzen Daten, im Fall von Textkollaboration vielleicht nur sehr wenige Buchstaben, einem Angreifer möglicherweise leichtes Spiel bietet um alle Möglichkeiten auszuprobieren. Um dieser Problematik zu entgehen, warten viele Ansätze, bis genügend Änderung passiert ist, vor ein Änderungspaket geschnürt wird. Somit verzögert sich aber auch die Verteilung an andere Teilnehmer, die „Echt-Zeitfähigkeit“ leidet und die Lösung ist nicht mehr komfortabel brauchbar.
Eine dritte Herausforderung ist beispielsweise, dass System, die versuchen, Daten vor Angreifern zu schützen, oft ein recht komplexes Setup benötigen. Es müssen beispielsweise Browser-Plugins installiert, kryptografische Schlüssel ausgetauscht und Teilnehmer und Teilnehmerinnen autorisiert werden. Browser-Plugins müssen entwickelt, gewartet und Browserseitig regelmäßig aktualisiert werden. Austauschen unter Geheimhaltung von kryptografische Schlüssel ist an sich nicht trivial. Sollte das erfolgreich gelungen sein, werden diese Schüssel dem Browser übergeben. Der Browser selbst hat aber nur sehr begrenzte Möglichkeiten, einen solchen Schlüssel zu schützen.
Und als letztes Beispiel muss es im Falle eines Konflikts von Inhalten eine Möglichkeit geben, diesen Konflikt zu lösen. Da der Server die Daten nicht lesen kann, müssen Prozesse zur Verfügung gestellt werden, dass die Teilnehmer und Teilnehmerinnen selbst Konflikte lösen können. Diese Prozesse verlangen wieder nach Kommunikationsaufwand und damit Zeit.
Alles in allem bleiben ein paar wenige aber im Kontext dieser Arbeit wesentliche Herausforderungen offen.

5. Architekturvorschlag

Auf Basis der bisherigen Diskussionen und den identifizierten offen-bebliebenen Herausforderungen wird nun ein weiterer Ansatz zur sicheren Echt-Zeit Textkollaboration vorgestellt. Dieser Ansatz fokussiert sich auf Datensicherheit und –vertraulichkeit und räumt Performance sowie Setup-Komplexität eine zweitrangige Rolle ein. Der Architekturvorschlag scheint vielversprechend und lässt nur wenige Herausforderungen ungelöst.
In diesem Kapitel wird nun die Architektur vorgestellt, der Setup-Prozess beschrieben und der Austausch von Änderungspaketen diskutiert. Den Abschluss bildet eine Diskussion über die erreichten Ziele und offen gebliebenen Punkte.

5.1. Architektur

Der Ansatz versucht, neue Ergebnisse in den Bereichen der Kryptografie zusammenzuführen. Um ein hohes Niveau von Datensicherheit und –vertraulichkeit zu erreichen, wird eine strikte Trennung der Belange eingesetzt. Damit ergeben sich vier große Blöcke, der Benutzer/die Benutzerin, der Kollaborationsservice, der Authentifizierungsservice und der Schlüsselservice. Die Blöcke und deren Kommunikationspfade sind in Abbildung 1 illustriert.
Beim Benutzer/bei der Benutzerin bzw. dem User Agent (Browser) laufen alle Linien zusammen. Der User Agent interagiert mit dem Benutzer/der Benutzerin, fordert die Kollaborationssoftware an und bindet den Authentifizierungsservice ein.

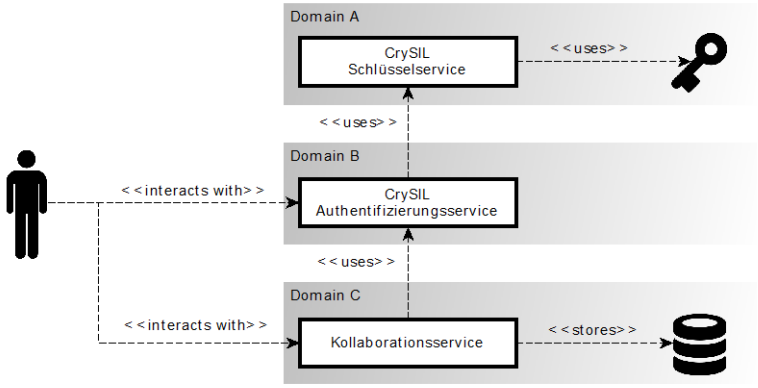

Abbildung 1 Architektur

Das Kollaborationsservice liefert die Webapplikation und übernimmt das Speichern des Dokuments. Die Webapplikation bietet dem Benutzer/der Benutzerin eine grafische Bedienoberfläche die Texterstellung und –bearbeitung ermöglicht. Der Kollaborationsservice empfängt verschlüsselte Änderungspakete, verteilt diese an alle Teilnehmer und Teilnehmerinnen und speichert die Änderungspakete als Liste. Die Webapplikation sendet Änderungspakete aber erst zum Kollaborationsservice, nachdem diese verschlüsselt wurden.
Die Verschlüsselung übernimmt der Cryptographic Service Interoperability Layer CrySIL [1], ehemals Arbeitstitel Skytrust. CrySIL bietet einen Authentifizierungsservice an, der zwischen der Applikation und dem Schlüsselservice angesiedelt ist. Der in Form eines IFrames eingebettete Service wird von der Applikation zu einer Verschlüsselung beauftragt und gibt das Ergebnis wieder an die Applikation zurück. Alle Schritte dazwischen, das heißt, Schlüsselauswahl und Authentifizierung, wird vom IFrame selbst erledigt. Nutzdaten werden von der Applikation aber direkt für den Schlüsselservice verschlüsselt, sodass das IFrame keinen Zugriff auf die Nutzdaten hat.
Schlussendlich kümmert sich das Schlüsselservice um die Verschlüsselung selbst, um das Schlüsselmanagement und um die Autorisierung der Schlüsselnutzung. Das Schlüsselservice kann den Zugriff auf ein und denselben kryptografischen Schlüssel durch verschiedene Authentifizierungsprozesse autorisieren und damit von jedem Teilnehmer und Teilnehmerinnen der Kollaboration eigene Zugangsdaten verlangen. Damit müssen keine Daten mehr zwischen den Teilnehmern und Teilnehmerinnen ausgetauscht werden.

5.2. Setup

Der Setup-Prozess für den hier beschriebenen Ansatz ist recht einfach. Es muss lediglich ein Schlüssel erstellt und die entsprechenden Zugriffsberechtigung installiert werden. Der Rest wird von der Applikation selbst gemacht.
Um eine Kollaboration verschlüsseln zu können, muss ein Schlüssel ausgewählt werden. Wenn dieser schon existiert, kann dieser später im Betrieb ausgewählt werden, wenn dieser nicht existiert, muss ein symmetrischer Schlüssel am CrySIL Schlüsselservice erstellt werden. Nun müssen dem ausgewählten Schlüssel noch Zugangsdaten für die Teilnehmer und Teilnehmerinnen bekannt gegeben werden, damit jeder Teilnehmer und Teilnehmerinnen mit seinen eigenen Zugangsdaten später den Schlüssel nutzen kann.

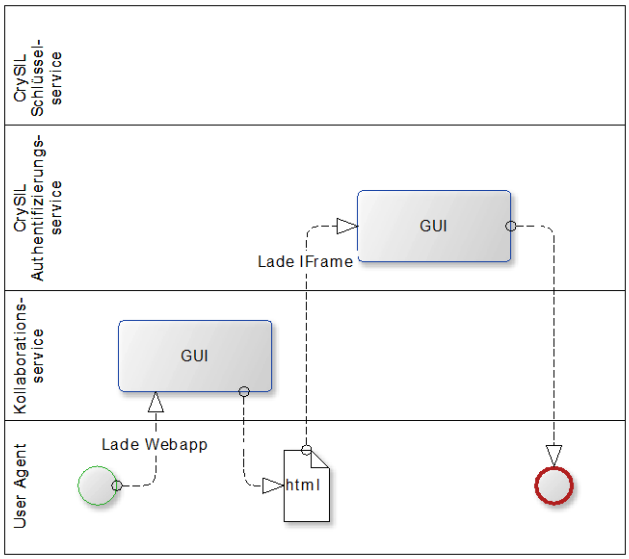

Abbildung 2 Setupprozess

Die weitere Setupprozedur macht die Applikation selbst. Der Benutzer/die Benutzer muss lediglich die Webapplikation vom Kollaborationsservice abrufen und diese lädt selbstständig das Authentifizierungsservice. Damit ist die Lösung einsatzbereit. Ein Flussdiagramm dazu findet sich in Abbildung 2.

5.3. Inhalte erstellen

Wenn der Benutzer/die Benutzerin nun die Applikation bedient, wird, sobald genügend Daten vorhanden sind, ein Änderungspaket an alle anderen Teilnehmer und Teilnehmerinnen versendet. Der Prozess dazu ist in Abbildung 3 illustriert.

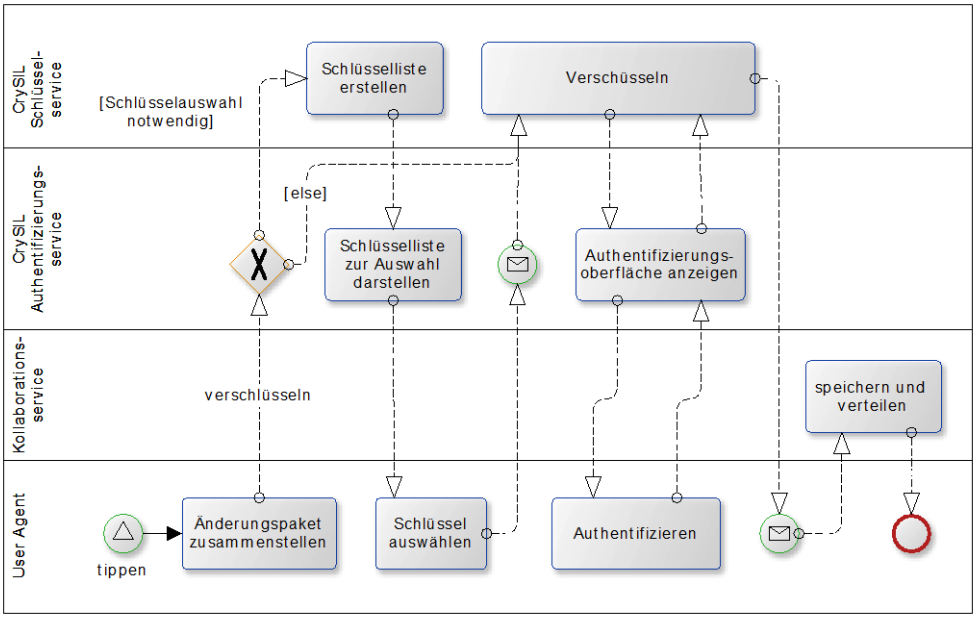

Abbildung 3 Änderungen erstellen

Im ersten Schritt wird von der Applikation ein Änderungspaket zusammengestellt und eine Verschlüsselung angefordert. Das Authentifizierungsservice entscheidet, ob eine Schüsselauswahl notwendig ist oder ob vorher bereits eine Schlüsselauswahl getroffen wurde. Wenn eine Schlüsselauswahl notwendig ist, fragt das CrySIL Authentifizierungsservice direkt im IFrame den CrySIL Schlüsselservice nach den verfügbaren Schlüsseln um diese dem Benutzer/der Benutzerin zur Auswahl anzuzeigen. Nach erfolgter Auswahl veranlasst das CrySIL Authentifizierungsservice

den Verschlüsselungsprozess und behandelt mit Hilfe des Benutzers/der Benutzerin etwaige Authentifizierungswünsche des CrySIL Schlüsselservice. Nach erfolgter Verschlüsselung leitet die Webapplikation des Kollaborationsservice das (jetzt verschlüsselte) Änderungspaket an die Kollaborationsservice weiter. Das Kollaborationsservice speichert das Paket und leitet es allen Teilnehmern und Teilnehmerinnen weiter. Bis zu diesem Zeitpunkt wird die Änderung in der Webapplikation nur als Vorschau angezeigt. Erst wenn ein Änderungspaket vom Kollaborationsservice empfangen wird, wird dieses final in das Dokument eingebaut.

5.4. Inhalte empfangen

Wenn immer der Kollaborationsservice ein Änderungspaket entgegennimmt verteilt er dieses sofort an alle Teilnehmer. Der Prozess, der in der Webapplikation des Kollaborationsservice dazu angestoßen wird, ist in Abbildung 4 dargestellt und nachfolgend beschrieben.

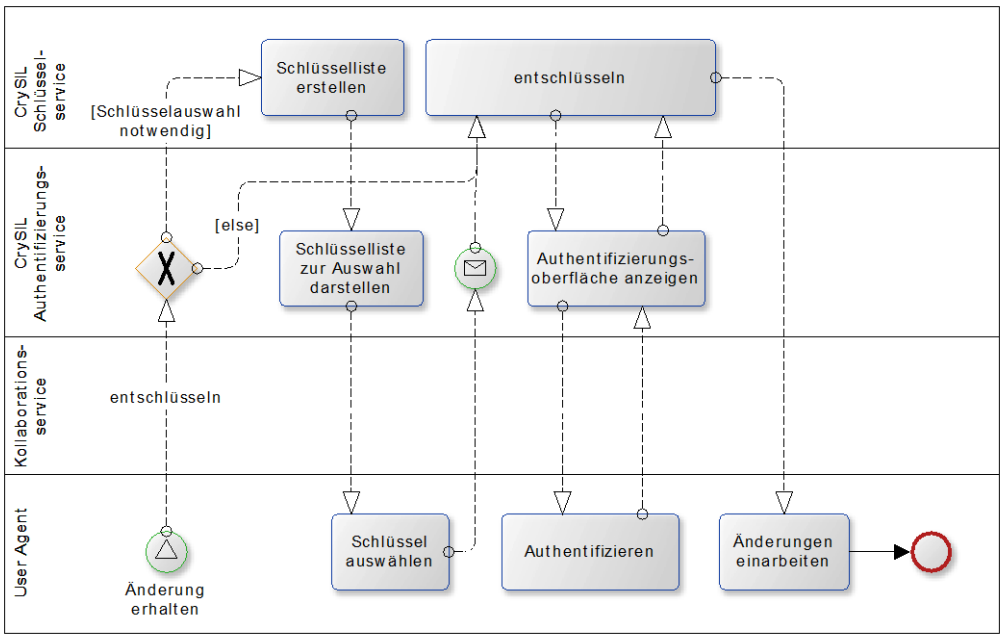

Abbildung 4 Änderungen erhalten

Sobald in der Webapplikation des Kollaborationsservice, d. h. im Browser des Benutzers/der Benutzerin, ein Änderungspaket ankommt, wird eine Entschlüsselung angefordert. Wieder überprüft das CrySIL Authentifizierungsservice, ob bereits ein kryptografischer Schlüssel ausgewählt wurde und initiiert entsprechend entweder den Schlüsselauswahlprozess oder den Entschlüsselungsprozess. Wieder handelt das CrySIL Authentifizierungsservice allfällige Interaktionen zwischen dem CrySIL Schlüsselservice und dem Benutzer/der Benutzerin, wie zum Beispiel Schlüsselauswahl und Authentifizierung, ab. Das nun entschlüsselte Änderungspaket kann nun von der Webapplikation des Kollaborationsservice ins vorliegende Dokument eingearbeitet werden.

5.5. Diskussion

In diesem Kapitel werden nun die Vor- und Nachteile dieser Architektur diskutiert. Diskutiert werden die Eigenschaften des Systems hinsichtlich Vertrauensanforderungen, Benutzbarkeit, Datensicherheit und offener Probleme.
Durch die klare Trennung der Belange ist eine feingranulare Definition von Vertrauensanforderungen an die einzelnen Blöcke möglich. Die Webapplikation des Kollaborationsservice sieht zum Beispiel weder kryptografische Schlüssel, noch Authentifizierungsdaten des Benutzers/der Benutzerin. Somit kann diese Applikation, solange sie sich funktional korrekt verhält keine Informationen über den Benutzer/die Benutzerin extrahieren. Das Kollaborationsservice selbst hat zusätzlich keinen Zugriff auf die Nutzdaten. Lediglich die Metadaten der Änderungspakete (Position, Länge, hinzufügen/löschen, …) sind dem Service einsehbar. Solange sich das Service also funktional

korrekt verhält, ist es dem Service nur sehr beschränkt möglich, Informationen über die Inhalte des Dokuments zu gewinnen.
Das CrySIL Authentifizierungsservice kann ebenfalls in seinen Angriffsvektoren eingeschränkt werden. Jegliche Nutzdaten, die durch das Authentifizierungsservice gehen, sind von der Applikation für das CrySIL Schlüsselservice verschlüsselt. Da nur Nutzdaten verschlüsselt werden und keine Metadaten mitgeliefert werden, kann das Authentifizierungsservice keine Information über die Inhalte des Dokuments lernen. Dem Authentifizierungsservice bleiben lediglich Seitenkanäle wie beispielsweise die Frequenz der Änderungen.
Das CrySIL Schlüsselservice selbst kennt prozessbedingt Authentifizierungsdaten, Schlüssel und die Nutzdaten eines vorliegenden Änderungspakets, kann aber aufgrund fehlender Metadaten nur schwer die Änderungen genau verfolgen. Alles in allem muss dem CrySIL Schlüsselservice das größte Vertrauen entgegengebracht werden. CrySIL bietet aber den Ausweg, das Schlüsselservice auf dem Smart Phone eines Teilnehmers oder einen Teilnehmerin zu betreiben [7] und somit ist der Schlüsselservice unter alleiniger Kontrolle des Benutzers/der Benutzerin.
Vorläufige Performancetests zeigen, dass die Benutzbarkeit des Systems gegeben ist und eine recht gute Echt-Zeitabwicklung der Änderungspakete möglich ist. Der verwendete Verschlüsselungsalgorithmus AES ist durch Hardwarebeschleunigung mittlerweile fast verlustfrei einzusetzen, eine schnelle Internetverbindung ermöglicht auch die recht aufwändige Kommunikation des vorgestellten Ansatzes.
Offen bleibt lediglich die Problematik eines Konflikts. Nachdem der Kollaborationsservice, als zentraler Verwaltung von Änderungspaketen, die Inhalte nicht lesen kann, ist es schwierig, einen klassischen Merge-Algorithmus einzusetzen. Ein alternativer Algorithmus, der nur Metadaten benötigt, scheint aber aus heutiger Sicht erreichbar, wenngleich ein solcher Algorithmus nicht im Rahmen dieses Projekts gefunden werden konnte.
Alles in allem vermag diese Architektur, mit relativ wenig Implementierungsaufwand, eine hinreichend sichere Kollaborationsplattform zu schaffen. Die Lösung ist damit, wenn die offenen Probleme erfolgreich gelöst werden können, durchaus in der Lage, kryptografisch sichere Echt-Zeit Textkollaboration zu ermöglichen.

6. Demonstrator

Im Rahmen des Projekts ist ein Demonstrator entstanden der sich den aus vergangenen Projekten entstandenem CrySIL-Implementierungen (ehemals Arbeitstitel Skytrust) Lösungen bedient. Der CrySIL Schlüsselservice verwendet eine Datenbank um Schlüssel und Authentifizierungsdaten zu speichern, die Kryptographie wird mit BouncyCastles JCE Provider umgesetzt. Es existiert eine Weboberfläche für Schlüsselmanagement. Das Authentifizierungsservice ist als IFrame mit HTML und JavaScript umgesetzt. Die Webapplikation des Kollaborationsservice baut ebenfalls auf HTML und JavaScript und kommuniziert über JSON-Strukturen mit einer Java Webapplikation. Der Demonstrator kann auf der CrySIL Informationsseite[7] ausprobiert werden.

7. Zusammenfassung

Im Rahmen dieses Projekts wurden einige Lösungen für sichere Echt-Zeit Textkollaboration gesichtet und gemeinsame Unzulänglichkeiten identifiziert. Auf dieser Basis wurde eine Architektur vorgeschlagen, die bisher offen-gebliebenen Herausforderungen adressiert. Der Vorschlag scheint vielversprechend, obschon es noch Probleme gibt, die gelöst werden müssen. Ein Demonstrator macht den Vorschlag greifbar.

Referenzen

[1] F. Reimair, P. Teufl und T. Zefferer, „CrySIL: Bringing Crypto to the Modern User," in *Lecture Notes in Business Information Processing*, Bd. 246, Springer, 2016, pp. 70-90.

[7] https://crysil.iaik.tugraz.at

[2] L. Adkinson-Orellana, D. A. Rodríguez-Silva und F. Gil-Castineira, „Privacy for google docs: Implementing a transparent encryption layer," in *Proceedings of the 2nd International Conference on Cloud Computing*, 2010.

[3] G. D'Angelo, F. Vitali und S. Zacchiroli, „Content cloaking: preserving privacy with Google Docs and other web applications," in *Proceedings of the 2010 ACM symposium on applied computing*, 2010.

[4] M. Clear, K. Reid, D. Ennis, A. Hughes und H. Tewari, „Collaboration-Preserving Authenticated Encryption for Operational Transformation Systems," in *Lecture Notes in Computer Science*, Bd. 7483, Springer Berlin Heidelberg, 2012, pp. 204-223.

[5] A. J. Feldman, W. P. Zeller, M. J. Freedman und E. W. Felten, „SPORC: Group Collaboration using Untrusted Cloud Resources.," in *Proceedings of the 9th USENIX Symposium on Operating Systems Design and Implementation*, Vancouver, BC, Canada, 2010, pp. 337-350.

[6] C. Zhang, J. Jin, E.-C. Chang und S. Mehrotra, „Secure Quasi-Realtime Collaborative Editing over Low-Cost Storage Services," in *Secure Data Management*, Springer, 2012, pp. 111-129.

[7] F. Reimair, P. Teufl und T. Zefferer, „MoCrySIL - Carry Your Cryptographic Keys in Your Pocket," in *12th International Conference on Security and Cryptography.*, 2015.