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

KONSOLIDIERUNG VON SKYTRUST WERKZEUGEN

Version 1.0 vom 10.01.2018
Florian Reimair – florian.reimair@iaik.tugraz.at

Der Cryptographic Service Interoperability Layer (CrySIL), ehemals bekannt als Skytrust, hat sich zur Aufgabe gemacht, angewandte Kryptografie so zu ergänzen, dass diese mit Situationen und Anwendungsfällen aus unserem modernen digitalen Leben, die den Stand der Technik teilweise überfordert haben, wieder zurechtkommt. Dieses Projekt konsolidiert viele der bisher geleisteten Beiträge zu CrySIL aus verschiedensten Quellen. Das Ergebnis des Projekts ist eine Informationswebsite, die neben den A-SIT Projekten, wissenschaftlichen Veröffentlichungen oder Studentenprojekten zu CrySIL auch verschiedene Demonstratoren zur Ansicht zur Verfügung stellt.

Inhaltsverzeichnis

1. Einleitung

Der Cryptographic Service Interoperability Layer (CrySIL), ehemals bekannt unter dem Namen Skytrust, fand seinen Anfang in den Jahren 2013/2014. Die Motivation war damals, dass die Smart-Card-Version der österreichischen Bürgerkarte für über die Grund-Funktionalität hinausgehende Anwendungen wie Email-Signaturen verwendet werden kann, speziell für Benutzer und Benutzerinnen, die weniger Technologie-affin sind, und auch Nutzer und Nutzerinnen, die nicht Microsoft Windows als Betriebssystem verwenden. Seither haben sich Motivation als auch zutreffenden Anwendungsfälle erweitert und CrySIL ist zu einem mächtigen Stück Technologie herangewachsen. Erste wissenschaftliche Veröffentlichung folgten dann ab dem Jahr 2015 ([1] und andere) – sowohl A-SIT Projekte, Dissertationsbemühungen, Studentenprojekte und auch eine Industriekooperation haben das Projekt vorangetrieben.
Das Ziel von CrySIL ist, mehr Flexibilität und Benutzbarkeit im Bereich der angewandten Kryptografie zu erreichen. Speziell dort, wo aktuelle Lösungen nicht mehr in der Lage sind, mit den vorherrschenden und damit auch zukünftigen Situationen und Anwendungsfällen Schritt zu halten,

versucht CrySIL neue Möglichkeiten zu schaffen, die einem Anwender oder einer Anwenderin die Nutzung von angewandter Kryptografie weiterhin (wieder) möglich macht, Daten gegen Missbrauch zu schützen.
CrySIL erreicht dies mit zwei dedizierten Eigenschaften. Erstens trennt CrySIL seine API strikt von der tatsächlichen Implementierung. Dabei versteckt sich CrySIL hinter existierenden KryptoAPIs und erlaubt gleichzeitig die Verwendung von geräteübergreifenden Implementierungen. Abbildung 2 illustriert das Konzept. Der Benutzer oder die Benutzerin kann verschiedene APIs verwenden, die

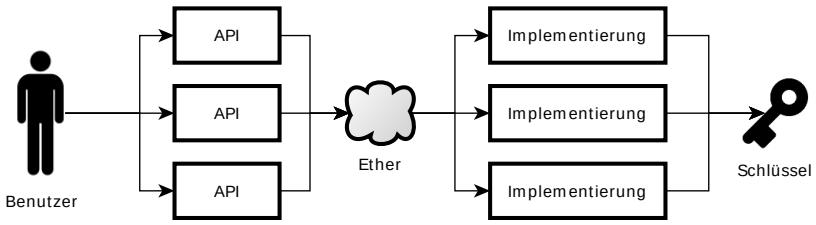

Abbildung 1: Interoperabilitätskonzept von CrySIL

transparent für den Benutzer oder die Benutzerin verschiedene Implementierungen verwenden kann, die unterschiedliche Eigenschaften haben. So können sich solche Implementierungen lokal am selben Gerät befinden, im Unternehmensnetzwerk oder auch in der Cloud, es können software-basierte Implementierungen verwendet werden genauso wie hochsichere Hardware Security Modules (HSMs), bezahlte Services oder eigene Schlüssel.
Zum anderen ergänzt CrySIL Funktionalitäten, die existierende Lösungen nur sehr vereinzelt anbieten. Zum Beispiel übernimmt CrySIL die Schlüsselautorisierung vollständig. CrySIL interagiert selbst mit der Benutzerin oder dem Benutzer, sammelt die Authentifizierungsdaten ein und entscheidet basierend auf diesen Daten, ob ein Schlüssel für eine gegebene Operation genutzt werden darf. Basierend auf diesen Möglichkeiten kann CrySIL die Verantwortung von Sicherheit auf mehrere Parteien aufteilen; die Applikation, die kryptografische Operationen benötigt, die Applikation, die Authentifizierungsdaten sammelt und die Applikation, die die Operation selbst ausführt. Weiters ermöglicht es CrySIL, mehrere verschiedene Schlüsselspeicher hinter einer API zusammenzufassen, sodass lokal und entfernt verfügbare Schlüssel mit gleichem Aufwand verwendet werden können.
Damit ermöglicht CrySIL unter anderem, dass ein Benutzer oder eine Benutzerin Cloud-basierte (ev. kommerzielle) Schlüsselservices, seine Bürgerkarte-Smartcard und einen lokalen Schlüsselspeicher ohne zusätzliche Komplexität und Aufwand verwenden kann, und das von jedem seiner/ihrer Geräte. Mit entfernten Schlüsselspeicher wird es auch möglich, Schlüssel mit anderen zu teilen, sodass zum Beispiel der Sender von Daten entscheiden kann, welcher kryptografische Schutz für die zu senden Daten angemessen ist, indem er einen passenden Schlüssel selbst erstellt und mit dem Empfänger (via Authentifizierung) teilt. CrySIL ermöglicht teilweise auch, Eigenschaften von kryptografischen Methoden nachzubilden, die nicht im Mainstream der angewandten Kryptografie zu finden sind (zb. Identity-Based Encryption (IBE)) bzw. (noch) nicht produktiv einsetzbar sind (zB proxy-reencryption (PRE)).
In den vergangenen Jahren haben viele Beiträge von unterschiedlicher Quelle CrySIL ergänzt und vorangetrieben. Dieses Projekt hat nun einen Teil dieser Beiträge konsolidiert und eine Demonstrationsplattform erstellt, wo einige der Anwendungsfälle direkt ausprobiert werden können. Diese Demonstratoren bauen auf einem konsolidierten Kern von CrySIL auf, der nach Abschluss des Projekts einen Großteil der Beiträge enthält. Informationen zu CrySIL, dessen Source Codes und auch die nachstehen beschriebenen Demonstratoren sind auf der CrySIL Demonstrations-website[1] zu finden.

2. Demonstratoren

Im Laufe der Zeit haben sich einige Demonstratoren gesammelt, die die Flexibilität und Modularität des Cryptographic Service Interoperability Layers zeigen. Die Demonstratoren sind auf der CrySIL

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

Website veröffentlicht. Hier werden nun die bis jetzt konsolidierten Demonstratoren vorgestellt, deren Funktion erklärt und deren Demonstrationsziel diskutiert.

2.1. Einfacher Schlüsselservice in der Cloud

Dieser Demonstrator ist der einfachste und grundlegendste der Demonstratoren. Es wird ein Schlüsselservice angeboten, das von internet-fähigen Geräten verwendet werden kann. Der Demonstrator bietet nur die grundlegendsten Operationen der CrySIL-Welt an. Zu finden ist der Demonstrator auf der CrySIL Website im Abschnitt *Demonstratoren* mit dem Titel *Simple Static Key RSA CrySIL Node without authentication:* ***https://crysil.iaik.tugraz.at/tomcat/simple-webservice/json.***
Der hinter der URL zur Verfügung gestellte CrySIL Knoten beinhaltet nur wenige Komponenten. Abbildung 2 illustriert den CrySIL Knoten und wie diese zu verwenden ist. Den zum Knoten bildet ein http-Interface, das CrySIL Befehle im JSON Format akzeptiert. Ein zweites Modul bietet kryptografische Operationen an und hängt direkt hinter dem http-Interface. Um den Demonstrator

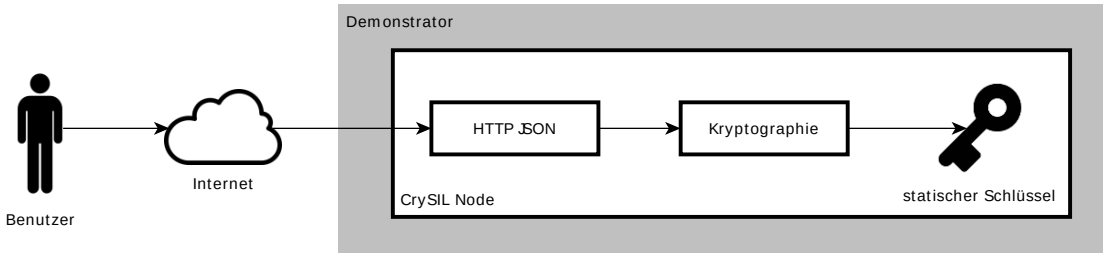

Abbildung 2: Komponenten des einfachen Schlüsselservice

einfach zu halten, verwendet das zweite Modul für alle angebotenen kryptografischen Operationen nur einen einzigen kryptografischen Schlüssel. Dieser Schlüssel ist im Source Code des Demonstrators festgesetzt und kann nur durch erneutes veröffentlichen des Demonstrators geändert werden. Wenn ein CrySIL-Befehl einen anderen Schlüssel verlangt, wird die Schlüsselauswahl überschrieben und nur der eine verfügbare Schlüssel verwendet. Dieser Schlüssel ist vom Typ RSA mit einer Länge von 512 Bit, das Zertifikat ist 10 Jahre gültig. Damit eignet sich der Schlüssel/das Zertifikat nicht für den Produktivbetrieb. Die kryptografischen Operationen werden von der BouncyCastle Kryptolibrary zur Verfügung gestellt.
Ziel des Demonstrators ist es, einem Entwickler schnell eine CrySIL Node zur Verfügung zu stellen, die Angaben zu Schlüsseln und Algorithmenparameter ignoriert. Damit kann ein Entwickler schnell erste Ergebnisse erzielen und Probleme in der Kommunikation mit der Node ausmerzen. Durch den fixen Schlüssel kann ein Entwickler diesen Demonstrator auch schnell und ohne Konfigurationsaufwand auf seinem eigenen Rechner betreiben.
Alles in allem leitet dieser Demonstrator einen Entwickler oder Anwender bei deren ersten vorsichtigen Schritten in der CrySIL Welt ohne diese bereits mit all den Möglichkeiten zu überfahren.

2.2. Komplexerer Schlüsselservice in der Cloud

Dieser Demonstrator erweitert die Funktionalität des vorher beschriebenen Demonstrators. Es wird ein Schlüsselservice angeboten, das von internet-fähigen Geräten verwendet werden kann. Der Demonstrator bietet erweiterte Funktionalität der CrySIL-Welt an. Zu finden ist der Demonstrator auf der CrySIL Website im Abschnitt *Demonstratoren* mit dem Titel *CrySIL Node with authentication and key database:* ***https://crysil.iaik.tugraz.at/tomcat/webservice-with-auth-and-keydatabase/json.***
Die hinter der URL zur Verfügung gestellte CrySIL Node beinhaltet neben einem Interface und einem Modul für Kryptografie zusätzliche Komponenten. Abbildung 3 illustriert die CrySIL Node und wie diese zu verwenden ist. Den Einstieg zur Node bildet wieder ein http-Interface, das CrySIL Befehle im JSON Format akzeptiert. Dahinter findet ein Gatekeeper-Modul seinen Platz. Dieses Modul entscheidet aufgrund der Metadaten in der Schlüsseldatenbank, ob ein Befehl einem Autorisierungs-prozess unterzogen werden muss. Wenn ja, merkt sich das Modul den ursprünglichen Befehl, fordert aber gleichzeitig Authentifizierungsdaten an. Erst wenn diese eingetroffen und korrekt sind, wird der ursprüngliche Befehl an das nächste Modul weitergeleitet. Dieses nächste Modul bietet kryptografische Operationen an. Die verfügbaren kryptografischen Schlüssel werden aus der

Schlüsseldatenbank gelesen. Wenn ein CrySIL-Befehl einen Schlüssel verlangt, der in der Datenbank nicht existiert, führt dies, im Gegensatz zum vorherigen Demonstrator, zum Abbruch der Operation. Die kryptografischen Operationen werden von der BouncyCastle Kryptolibrary zur Verfügung gestellt.
Ziel des Demonstrators ist es, nach erfolgreicher Integration des vorher beschriebenen Demonstrators, mehr aus der CrySIL Welt für Experimente zur Verfügung zu stellen. Der Service kann mehrere kryptografische Schlüssel zur Verfügung stellen. Dabei sind RSA und auch AES möglich mit Schlüssellängen bis 2048 Bit, bzw. 256 Bit. Diesen Schlüsseln können verschiedene

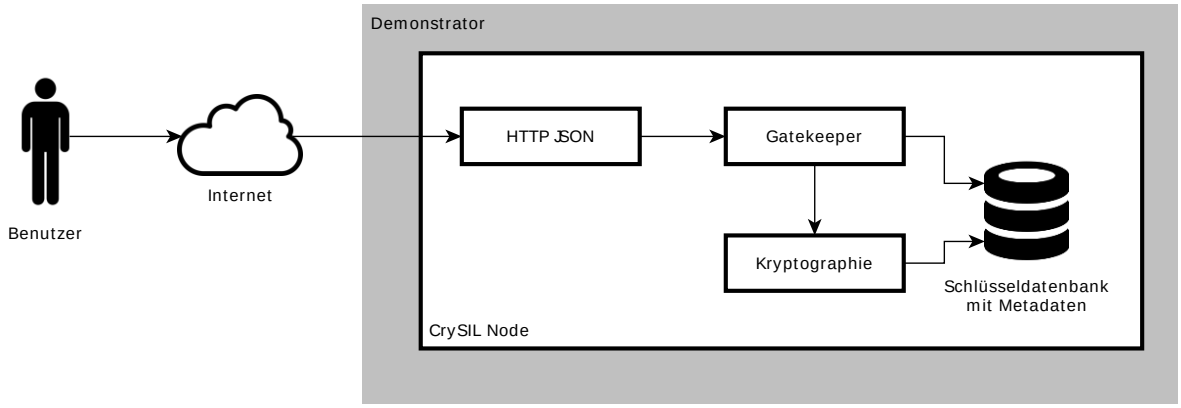

Abbildung 3: Komponenten des komplexeren Schlüsselservice

Verwendungszwecke zugeordnet werden. Damit ist es möglich, dedizierte Signatur- oder Verschüsselungsschlüssel zu haben. Weiters kann jeder der Schlüssel jeweils mit seinen eigenen Autorisierungsanforderungen versehen werden. Und zu guter Letzt hat jeder dieser Schlüssel seine eigene Kennung, die zu dessen Identifikation und Auswahl verwendet werden muss. Für den Entwickler bietet dieser Demonstrator damit umfangreichere Funktionalität, die er für Tests und Entwicklung verwenden kann. Der Demonstrator selbst zeigt die Möglichkeiten auf, die die Welt von CrySIL für den Anwendungsfall für Schlüsseldienste in der Cloud bieten kann.
Alles in allem weist dieser Demonstrator auf die Anwendungsmöglichkeiten hin, die die CrySIL Welt bietet. Damit wird dieser Demonstrator auch in sehr vielen anderen Demonstratoren als Schlüsselservice verwendet.

2.3. Schlüsselmanagement Webapplikation

Dieser Demonstrator arbeitet eng mit dem vorher beschriebenen komplexeren Schlüsselservice in der Cloud zusammen. Es wird ein Web Userinterface bereitgestellt, mit dem die Inhalte der vorher erwähnten Schlüsseldatenbank verwaltet werden können. Zu finden ist der Demonstrator auf der CrySIL Website im Abschnitt *Demonstratoren* mit dem Titel *Simple Webapp for key management for the node above: CloudKS.*
Das Webservice besteht aus 4 großen Bausteinen. Abbildung 4 illustriert diese Bausteine und setzt diese in den Kontext eines Anwendungsfalls. Die grafische Bedienoberfläche bietet dem

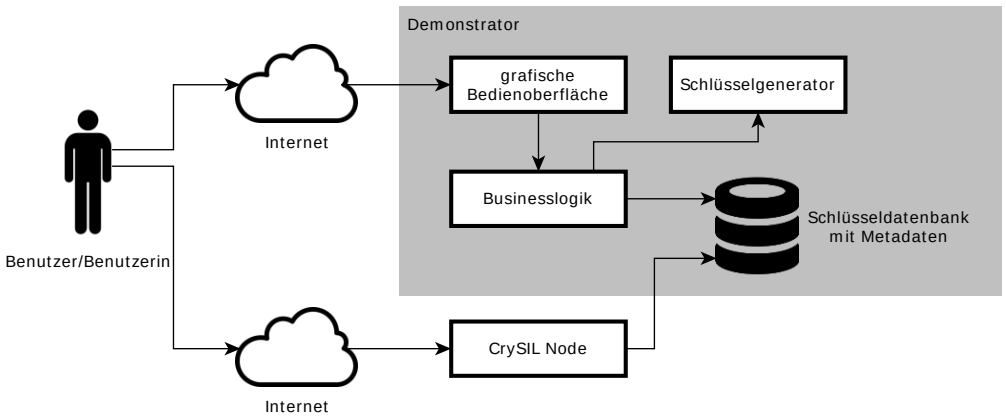

Abbildung 4: Komponenten des Schlüsselmanagement Webservice

Benutzer/der Benutzerin verschiedene Möglichkeiten des Schlüsselmanagements an. Nachdem sich der Benutzer oder die Benutzerin registriert hat, kann dieser neue Schüssel erstellen, Schlüssel mit Autorisierungsanforderungen versehen, einen Verwendungszweck zuweisen. Auch den Namen des Schlüssels kann der Benutzer/der Benutzerin festlegen, der es dem Benutzer/der Benutzerin ermöglicht, den Schlüssel wieder zu erkennen. Zu guter Letzt ermöglicht die Bedienoberfläche auch, ausgewählte Schlüssel zu löschen bzw. zu revozieren. Hinter der Bedienoberfläche führt die Geschäftslogik die von der Bedienoberfläche gesendeten Befehle aus. Ein Schlüsselgenerator hilft dabei, kryptografische Schlüssel zu generieren. Gespeichert werden die generierten Schlüssel und die zugehörigen Metadaten in der Schlüsseldatenbank, die dann von einer CrySIL Node verwendet werden kann.
Dieser Demonstrator ermöglicht es Benutzern/Benutzerinnen bzw. Entwicklern und Entwicklerinnen, das komplexere Schlüsselservice zu konfigurieren. Gleichzeitig zeigt der Demonstrator auf, wie ein kommerzieller Schlüsselservice ausschauen könnte. Für den Produktivbetrieb ist der Demonstrator nicht geeignet. Dies ist einerseits auf die prototypische Implementierung des Service als auch durch die Art der Schlüsselspeicherung bedingt. Schlüssel und deren Metadaten werden ausschließlich in der Datenbank gespeichert ohne zusätzlichen Schutz vor Angriffen.
Alles in allem zeigen dieses Schlüsselmanagement Webservice und der komplexere CrySIL Schüsselservice eine Möglichkeit auf, wie ein Cloudbasiertes Schlüsselservice funktionieren könnte. Dank gilt Herrn Attila Földes, der den Demonstrator ursprünglich implementiert hat.

2.4. Schlüsselservice auf dem eigenen Android Smart Phone

Zu finden ist der Demonstrator auf der CrySIL Website im Abschnitt *Demonstratoren* mit dem Titel *Personal keyservice on your Android phone:* [*Download*]. Der dazu benötigte Relay Service ist wiederum im Abschnitt Demonstratoren unter dem Titel *Crysil Android Relay Service (formerly known as WebVPN) endpoint: https://crysil.iaik.tugraz.at/.../YOURIDHERE* zu finden.
Dieser Demonstrator besteht aus mehreren Bausteinen, die in Abbildung 5 illustriert sind. Der Benutzer oder die Benutzerin bedient eine Applikation. Diese Applikation verwendet für kryptografische Operationen die Welt von CrySIL. Die Applikation ist so konfiguriert, dass Anfragen an einen Schlüsselservice über das Internet mittles http und JSON an den Relay Service abgesetzt

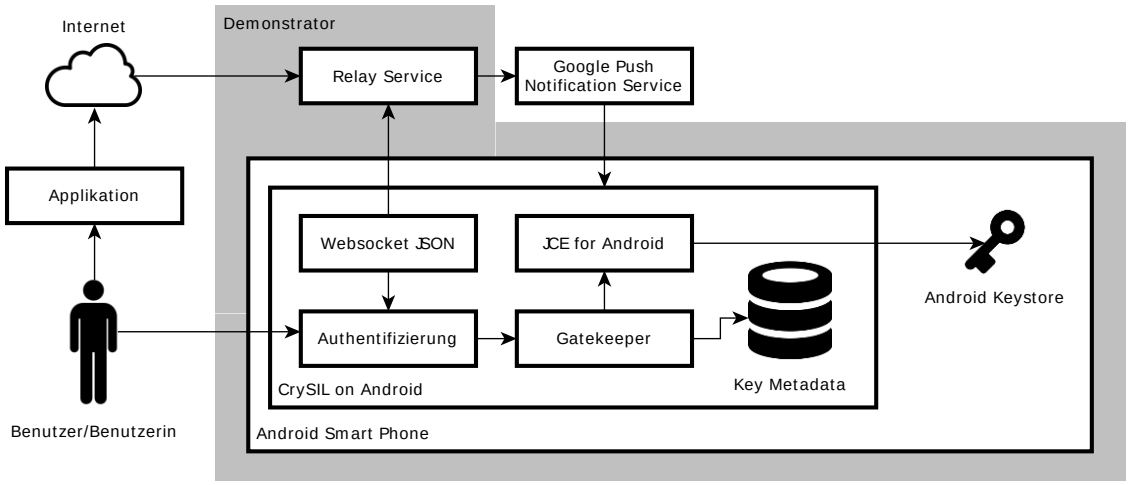

Abbildung 5: Komponenten des Android Schlüsselservice

werden. Der Relay Service nutzt den Google Push Notification Service, um das entsprechende Smart Phone bzw. die CrySIL on Android-App auf diesem Smart Phone über einen eingehenden Befehl zu informieren. Das Websocket JSON Modul baut daraufhin eine Websocket-Verbindung zum Relay Service auf und der Relay Service sendet den originalen Befehl über die Websocket-Verbindung an das Smart Phone. Dort verlangt der Gatekeeper für jeden CrySIL Befehl, der nicht nach den verfügbaren Schlüsseln fragt, eine One-Click Authentifizierung, bevor der Befehl ausgeführt wird. Diese Authentifizierungsanfrage wird zurückgeschickt und gleich vom Authentifizierungsmodul direkt am Smartphone behandelt, indem der Benutzer oder die Benutzerin am Smartphone direkt gefragt wird, ob der Befehl ausgeführt werden soll oder nicht. Wenn der Befehl ausgeführt werden soll, verwendet das Kryptografiemodul die Android JCE und damit den Android Keystore um

dort den geforderten Schlüssel für die Operation zu verwenden. Neben dieser One-Click Autorisierungsanforderung werden natürlich auch alle anderen Autorisierungsmethoden unterstützt. Diese werden dann aber nicht vom Authentifizierungsmodul am Smartphone behandelt, sondern gehen an den Benutzer bzw. an die Benutzerin zurück.
Ziel dieses Demonstrators ist, zu zeigen, dass ein Schlüsselservice in der Cloud nicht unbedingt eine dritte Partei und ein entsprechendes Vertrauensverhältnis zu dieser Partei fordert. Indem die kryptografischen Schlüssel am eigenen Smartphone verwaltet werden, ist die Verwendung dieser Schlüssel unter der alleinigen Kontrolle des Benutzers bzw. der Benutzerin. Weiters weist die One-Click Autorisierung darauf hin, dass eine solche Methodik zu einer vollen Mehrfaktor-Authentifizierung genutzt werden kann, indem zum Beispiel eine Zufallszahl am Smartphone angezeigt wird, die in der Applikation eingegeben werden muss.
Alles in allem handelt es sich bei diesem Demonstrator um ein Proof-of-Concept. Dennoch zeigt dieser Proof-of-Concept Wege auf, die es bisher noch nicht gegeben hat. Dank gilt hierfür den Herren Christoph Thaller und Christian Kollmann, die beide Maßgeblich bei der Realisierung dieses Demonstrators beteiligt waren.

2.5. CrySIL Javascript IFrame

Um die Verwendung von Kryptografie in Browserapplikation sowohl für den Anwender/die Anwenderin als auch für Entwickler und Entwicklerinnen noch einfacher zu gestalten, wurde ein weiterer Demonstrator geschaffen. Dieser Demonstrator ermöglich es, hinter der W3C Web Cryptography API die Welt von CrySIL nutzbar zu machen, wobei die Applikation selbst sich weder um Schlüsselmanagement, noch um Bereitstellung von Authentifizierungsdaten kümmern muss. Zu finden ist der Demonstrator auf der CrySIL Website im Abschnitt *Demonstratoren* mit dem Titel *CrySIL Javascript IFrame: https://crysil.iaik.tugraz.at/iframe/credential-node/credential-node.html*.
Die CrySIL Node im IFrame besteht aus mehreren Komponenten. Abbildung 6 illustriert die Komponenten und bettet diesen in einen Anwendungsfall ein. Die Javascript CrySIL Bibliothek wird in der Browser-Applikation initiiert. Damit wird der W3C Cryptography API eine JCE-ähnliche

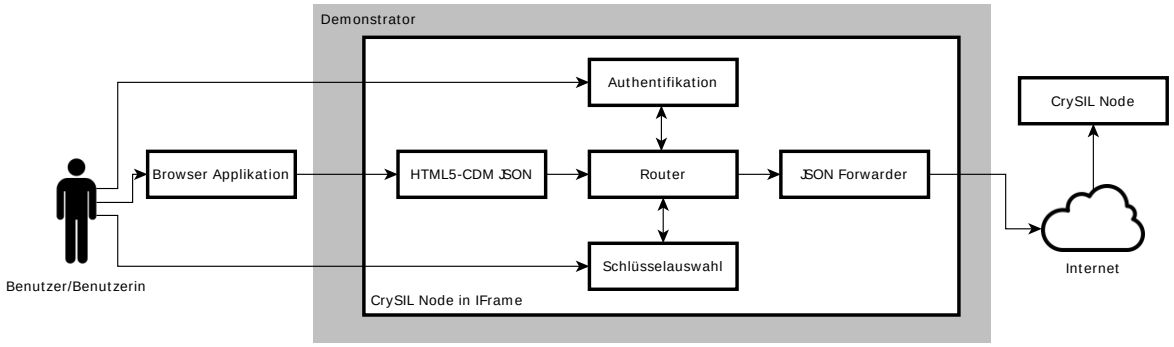

Abbildung 6: Komponenten des IFrame Demonstrators

Providerstruktur übergestülpt und neben der browsereigenen Implementierung der W3C Cryptography API der CrySIL Provider registriert. Weiters wird der konfigurierte Platzhalter durch das IFrame ersetzt. Das IFrame selbst ist nur eine Website, die von einer dritten Partei angeboten werden kann. Ist das IFrame geladen, kann die Browserapplikation die Welt von CrySIL nutzen. Der Einsprung-Punkt der CrySIL Node im IFrame ist ein Modul, das als Endpunkt für die Cross-Domain Messaging Funktionalität (CDM) des HTML5 Standards (auch bekannt als HTML5 PostMessage) fungiert. Die CrySIL Bibliothek in der Browserapplikation sendet so dem IFrame CrySIL Befehle im JSON Format. Das CDM-JSON-Empfangsmodul leitet die Anfrage nun an einen Router weiter. Dieser entscheidet, ob der Befehl bereits einen Schlüssel enthält. Wenn nicht, gibt der Router den Befehl an das Schlüsselauswahl-Modul weiter. Dieses merkt sich den originalen Befehl und fragt beim CrySIL Node des Schlüsselservice (über den JSON Forwarder) nach einer Liste von verfügbaren Schlüsseln. Nach Erhalt der Schlüsselliste wird dem Benutzer/der Benutzerin im IFrame ein Auswahlfeld angezeigt, in dem der zu verwendende Schlüssel gewählt werden muss. Die Schlüsselkennung wird in den originalen Befehl eingefügt und selbiger an das Schlüsselservice weitergeleitet. Sollte der Schlüsselservice Authentifizierungsdaten benötigen, so gibt der Router eine solche Anfrage an das Authentifikationsmodul weiter, welches wiederum direkt im IFrame mit dem

Benutzer bzw. der Benutzerin kommuniziert. Der zu verwendende Schlüsselservice kann direkt nachdem das IFrame geladen wurde mittels Auswahlfeld gewählt werden.
Dieser Demonstrator verfolgt zwei Ziele. Zum einen versucht der Demonstrator aufzuzeigen, dass Kryptografie und Schlüsselmanagement nicht schwierig und kompliziert sein muss. Dies gilt für den Entwickler bzw. die Entwicklerin, da sich die Applikation selbst (bei Verwendung dieses IFrames) nicht um Schlüsselmanagement oder Authentifizierung kümmern muss und so die Applikation einfacher gehalten werden kann. Für den Benutzer bzw. die Benutzerin bietet das IFrame eine widerkehrende Bedienoberfläche die es schnell und einfach ermöglicht, die gerade notwendigen Interaktionen am Weg zur Verschlüsselung oder Signatur zu bewältigen. Zum zweiten zeigt der Demonstrator auf, dass es möglich ist, nicht nur die kryptografischen Schlüssel selbst von der Applikation zu entfernen, sondern auch die Authentifizierungsdaten, die in einem solchen Setup genauso sensibel sind wie die Schlüsseldaten. Die Applikation sieht somit weder Schlüssel noch Authentifizierungsdaten. Angriffe gegen so ein System werden damit erschwert. Zumindest müssen zwei der drei Parteien zusammenarbeiten, um an sensible Daten zu gelangen.
Alles in allem zeigt dieser Demonstrator sowohl, dass Kryptografie (vor allem Schlüsselverwaltung) im Browser vernünftig machbar ist, als auch, dass Kryptografie weder für den Benutzer/die Benutzerin noch für den Entwickler/die Entwicklerin komplex sein muss. Besonderer Dank gilt den Herrn Felix Hörandner und Stefan Gruber, die an der Verwirklichung dieses Demonstrators beteiligt gewesen sind.

2.6. Verschlüsselungsdemonstratoren

Diese Demonstratoren verwenden sowohl das CrySIL Javascript IFrame als auch verschiedene Schlüsselservice Demonstratoren. Es werden einfache Webapplikationen bereitgestellt, die einfache Verschlüsselungsoperationen ausführen können. Zu finden ist der Demonstrator auf der CrySIL Website im Abschnitt *Demonstratoren* mit dem Titel *Encryption demos using the IFrame:* [Demos.]
Jeder Demonstrator ist prinzipiell gleich aufgebaut. Abbildung 7 illustriert deren Bausteine. Der Benutzer/die Benutzerin bedient die Bedienoberfläche in seinem/ihrem Web Browser. Die Bedienoberfläche kommuniziert mit der Programmlogik, die wieder mit dem CrySIL IFrame kommuniziert.
Ziel dieser Demonstratoren ist es, zu zeigen, wie einfach, schnell und unkompliziert einer Browserapplikation der Zugang zu kryptografischen Methoden ermöglicht werden kann. Damit sind diese Demonstratoren hauptsächlich an Entwickler und Entwicklerinnen gerichtet.
Alles in allem zeigen die Demonstratoren, wie eine Webapplikation Zugang zur CrySIL Welt und

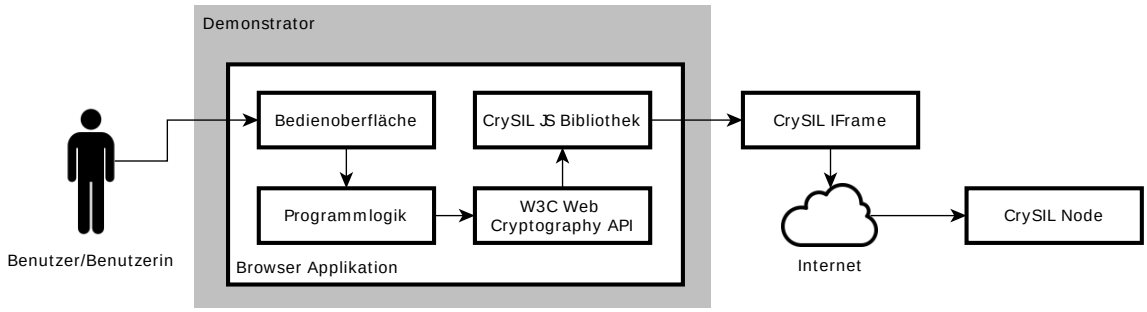

Abbildung 7: Komponenten der browserbasierten Verschlüsselungsapplikationen

damit zu Kryptografie bekommen kann. Dank gilt Herrn Stefan Gruber, der an der Verwirklichung dieser Demonstratoren maßgeblich beteiligt war.

2.7. Verschlüsselungsapplikation im Browser

Dieser Demonstrator bietet eine Webapplikation, mit der kryptografische Operationen auf lokal gespeicherten Dateien ausgeführt werden können. Zu finden ist der Demonstrator auf der CrySIL Website im Abschnitt *Demonstratoren* mit dem Titel *Browser-based Crypto-Application:* [Crypto for CrySIL.]

Dieser Demonstrator folgt im Wesentlichen dem Aufbau der vorhergehend beschriebenen Demonstratoren. Abbildung 7 illustriert auch hier die Bausteine. Wieder erlaubt eine Webapplikation die Verwendung von Kryptografie mit Hilfe des CrySIL IFrames. Besonders bei diesem Demonstrator ist aber, dass dieser auch lokal generierte Schlüssel mit der browsereigenen Implementierung verwenden kann.
Dieser Demonstrator demonstriert wieder, wie einfach, schnell und unkompliziert einer Browserapplikation der Zugang zu kryptografischen Methoden ermöglicht werden kann. Im Gegensatz zu den vorhergehenden Demonstratoren ist dieser Demonstrator aber an den Benutzer und an die Benutzerin gerichtet. Er zeigt damit, dass eine browserbasierte Applikation ohne Installation verwendet werden kann, um private Daten zu schützen.
Alles in allem zeigt dieser Demonstrator auf, wie eine browserbasierte Verschlüsselungsapplikation ausschauen könnte. Dank gilt hier Herrn Daniel Anthofer, der an der Bereitstellung dieses Demonstrators maßgeblich beteiligt war.

2.8. CMS Verschlüsselung am Desktop

Dieser Demonstrator zeigt, wie eine einfache Desktop-Applikation die Welt von CrySIL nutzen kann, um Dateien mit Hilfe des CMS-Formats zu verschlüsseln. Der Demonstrator ist damit funktional ähnlich zur den Verschlüsselungsdemonstratoren im Browser, tritt aber als eigenständiges Programm auf, das von Benutzern und Benutzerinnen eines Desktop-PC-artigen Geräts verwendet werden kann. Zu finden ist der Demonstrator auf der CrySIL Website im Abschnitt *Demonstratoren* mit dem Titel *Java Desktop Encryption Application:* *Download.*
Dieser Demonstrator folgt im Wesentlichen dem Aufbau des vorhergehend beschriebenen Demonstrators, der Verschlüsselungsapplikation für den Browser. Jedoch existiert kein Äquivalent für das IFrame und es wird nur CMS Verschlüsselung/Entschlüsselung mit kryptografischen Schlüsseln eines Schlüsselservice angeboten. Abbildung 8 illustriert die Bausteine. Der Benutzer

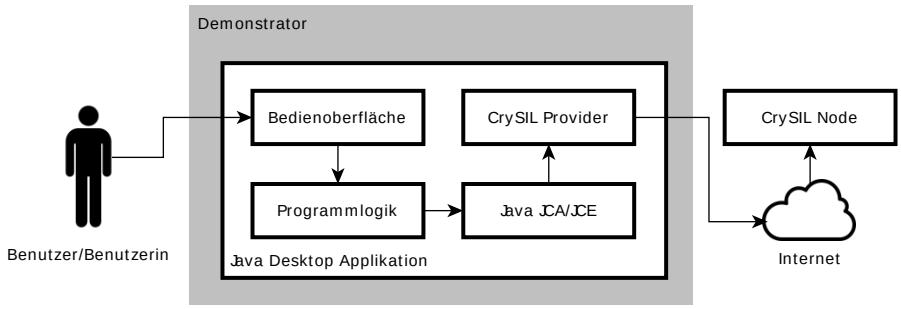

Abbildung 8: Komponenten der CMS Verschlüsselungsapplikation als Java Desktop Applikation

oder die Benutzerin bedient wieder die Bedienoberfläche. Diese ist mit JavaFX realisiert und bietet eine Schritt-für-Schritt Begleitung durch den Verschlüsselungsprozess. Zuerst wird nach der Datei gefragt, die behandelt werden soll, dann nach dem Schlüsselservice, der verwendet werden soll. Im nächsten Schritt wird der Schüsselservice nach verfügbaren Schlüsseln befragt und eine Auswahl geboten. Dieser Schritt wie auch folgende Schritte verbergen sich implementierungstechnisch hinter der von Java angebotenen JCA/JCE. Der Entwickler oder die Entwicklerin verwendet den CrySIL Provider als Keystore, um beispielsweise verfügbare Schlüssel abzufragen oder als Cipher für die Verschlüsselungsoperation selbst. Nach erfolgter Schüsselauswahl wird die Datei an den Schlüsselservice gesendet der diese ver- oder entschlüsselt. Das Sammeln von etwaigen Authentifizierungsdaten übernimmt der CrySIL Provider bzw. die dort enthaltene CrySIL Node.
Die Ziele dieses Demonstrators sind einerseits, eine Applikation zur Verfügung zu stellen, die einem Benutzer oder einer Benutzerin einfache Verschlüsselung von Dateien bietet in einem übersichtlichen Programmablauf und andererseits, Entwicklern und Entwicklerinnen zu zeigen, wie einfach wiederum es ist, von der CrySIL Welt zu profitieren, ohne sich um das Sammeln von Authentifizierungsdaten zu kümmern oder neue APIs erlernen zu müssen. Die Applikation zeigt auch die Möglichkeiten auf, wie eine existierende Applikation, die Java‘s JCA/JCE verwendet, ohne Aufwand zu einer Applikation gemacht werden kann, die die Welt von CrySIL nutzt.

Alles in allem zeigt auch dieser Demonstrator die Flexibilität und Einfachheit von CrySIL auf. Einerseits in der Bedienung für Benutzer und Benutzerinnen und andererseits in der Integration in bestehenden Java Applikationen.

2.9. Weitere Demonstratoren

Neben den hier im Detail beschriebenen Demonstratoren gibt es noch weitere, die aber bis jetzt noch nicht konsolidiert werden konnten. Darunter sind Demonstratoren, die zum Beispiel die Welt von CrySIL in eine PKCS#11 Bibliothek verpacken, damit Programme wie Mozilla Thunderbird oder Schlüsselringe von Betriebssystemen CrySIL's Möglichkeiten nutzen können. Ein weiterer Demonstrator ermöglicht es der Microsoft Windows Schlüsselverwaltung, CrySIL zu verwenden. Damit kann zum Beispiel Microsoft Outlook Emails mit Hilfe von CrySIL und damit mit Schlüsselservices der Cloud signieren. Ein Demonstrator erkundet das Authentifizierungsverfahren Universial Second Factor (U2F) der FIDO Alliance und demonstriert, wie ein U2F USB Token mit CrySIL ergänzt werden kann. Der Demonstrator zeigt, wie ein U2F-Token von einem Schüsselservice, der österreichischen Bürgerkarte-Smart Card, oder einer NFC-fähige Kryptosmartcard via Mobiltelefon emultiert werden kann, und damit eine Authentifizierung bei U2F-fähigen Webapplikationen als auch bei Installationen von Microsofts Windows 10 möglich wird. Dazu kommen noch Applikationen, die die Funktionalität der oben beschriebene Java Desktop CMS Applikation als auch des CrySIL IFrames auf Smart Phones bringt. Weitere Demonstratoren zeigen die Integration der eXtensible Access Control Modeling Language (XACML) in den Authentifizierungsprozess von CrySIL, die Emulation von Proxy-Reencryption und Identity-based Encryption und zu guter Letzt ein Showcase einer CrySIL-enabled Org, wo eine neue Art von Schlüsselmanagement in Unternehmen demonstriert wird.

3. Zusammenfassung

Viele Beiträge haben den Cryptographic Service Interoperability Layer ergänzt und erweitert. Dieses Projekt hat viele davon soweit konsolidiert, dass die jeweiligen Demonstratoren miteinander verwendbar sind. Damit wird über das Projekt die Vielseitigkeit von CrySIL demonstriert.

Literaturverzeichnis

[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.