Bundesamt
für Sicherheit in der
Informationstechnik

Ein Entwicklungsvorhaben im Auftrag des
Bundesamtes für Sicherheit in der Informationstechnik (BSI)

Projekt 197

Sichere Implementierung einer allgemeinen Kryptobibliothek

Projektzusammenfassung

Version 1.0.0 / 27.03.2017

Zusammenfassung

In diesem Projekt wird die sichere Implementierung einer allgemeinen Kryptobibliothek realisiert, die alle gängigen und für den breiten kryptographischen Einsatz notwendigen kryptographischen Primitive beinhaltet. Dazu zählen unter anderem symmetrische und asymmetrische Verschlüsselungs- und Signaturverfahren, PRFs, Hashfunktionen und RNGs. Zusätzlich müssen verbreitete Sicherheitsstandards wie X.509-Zertifikate oder SSL/TLS und der Einsatz von kryptographischer Spezialhardware unterstützt werden. Die Bibliothek stellt dann außerdem eine umfangreiche Testsuite und eine ausführliche Dokumentation bereit, die eine Evaluierung der Kryptographie im Einsatz erleichtert.

Dieser Bericht fasst die Projektinhalte zusammen.

Autoren

Daniel Neus (DN), Rohde & Schwarz Cybersecurity

Kai Michaelis (KM), Rohde & Schwarz Cybersecurity

René Korthaus (RK), Rohde & Schwarz Cybersecurity

Philipp Weber (PW), Rohde & Schwarz Cybersecurity

Christian Mainka (CM), Hackmanit GmbH

Matthias Gierlings (MG), Hackmanit GmbH

Jörg Schwenk (JSc), Hackmanit GmbH

Juraj Somorovsky (JSo), Hackmanit GmbH

Tobias Niemann (TN), Hackmanit GmbH

Copyright

Änderungshistorie

Version	Autor	Kommentar	Datum
1.0.0	RK	Initiale Dokumentenversion erstellt	20.03.2017

Inhaltsverzeichnis

Überblick

Kryptographische Bibliotheken werden häufig als Kernkomponenten in Sicherheitsanwendungen eingesetzt. Sie sind dort zum Erreichen der Sicherheitsziele von zentraler Bedeutung. In der Umsetzung ist eine Kryptobibliothek jedoch sehr anspruchsvoll und fehlerträchtig, sowohl in der Auswahl geeigneter Verfahren als auch in der Implementierung. Daher haben die Firmen Rohde & Schwarz Cybersecurity und Hackmanit GmbH im Auftrag des BSI eine Kryptobibliothek weiterentwickelt, die in Sicherheitsprodukten eingesetzt werden kann.

Bei der Entwicklung der Kryptobibliothek wurde in mehreren Phasen vorgegangen. Zunächst wurde eine Sichtung und Analyse bestehender Kryptobibliotheken durchgeführt, um einen geeigneten Kandidaten für eine Weiterentwicklung zu identifizieren. Nach Abschluss dieser ersten Phase wurde in enger Abstimmung mit dem BSI die Bibliothek Botan als geeignete Grundlage für die weitere Entwicklung ausgewählt. In der zweiten Phase wurde Botan dokumentiert, in der Analysephase vorgefunden Mängel wurden behoben, weitere Algorithmen und Seitenkanäle wurden analysiert, fehlende Kryptoprimitive und Standards wurden gemäß der technischen Standards und Richtlinien nachimplementiert, die Testsuite wurde verbessert und eine Testspezifikation erstellt. Zur Demonstration der Eignung der Kryptobibliothek für den praktischen Einsatz in Sicherheitsprodukten wurde die Bibliothek in eine bestehende VS-NfD Anwendung eingebunden. In der dritten Phase wird die Kryptobibliothek für mehrere Jahre im Auftrag des BSI gewartet und damit auf dem aktuellen technischen Stand gehalten.

Sichtung & Analyse

Ziel der Sichtungs- und Analysephase war es, unter vorhandenen offenen Kryptobibliotheken einen geeigneten Kandidaten für eine Weiterentwicklung zu finden. In einem ersten Schritt wurden dafür 18 Kryptobibliotheken ausgewählt und einer Analyse unterzogen. Aus den 18 Kryptobibliotheken wurden anschließend drei Bibliotheken für eine detailliertere Analyse ausgewählt. Abschließend wurde aus diesen drei Bibliotheken eine Bibliothek zur Weiterentwicklung ausgewählt.

Für eine Vorauswahl wurden zunächst alle in Tabelle 1 aufgeführten 18 auf dem Markt befindlichen Kryptobibliotheken nach vorher festgelegten Kriterien untersucht. Die Bibliotheken wurden in Abstimmung mit dem BSI ausgewählt. Zu den untersuchten Kriterien zählten vor allem die Implementierung der durch das BSI in den relevanten Technischen Richtlinien empfohlenen kryptographischen Verfahren und Protokolle, wie z.B. die OAEP- und PSS-Paddingverfahren für asymmetrische Verschlüsselung und Signaturen, die Hashfunktionen der SHA-1 und SHA-2 Familie, und das TLS-Protokoll in Version 1.2, was beispielsweise in Webbrowsern zur Absicherung von verschlüsselten Verbindungen zum Einsatz kommt. Weitere Bewertungskriterien betrafen beispielsweise die Qualität und den Grad der mitgelieferten Testsuite und Dokumentation, die Portierbarkeit, die Aktivität des Projektes und die Häufigkeit von Releases und eine zur kommerziellen Verwertung geeignete Quelltextlizenz.

Beecrypt	LibreSSL	Nettle
Botan	Libsodium	NSS
Cryptlib	Libtomcrypt	OpenSSL
Crypto++	MatrixSSL	Poco
GnuTLS	mbed TLS	s2n
Libgcrypt	Nacl	WolfSSL

Tabelle 1: In der Vorauswahl untersuchte Kryptobibliotheken

Nach der Auswertung erhielten Botan, OpenSSL und NSS die höchsten Punktwertungen. In Absprache mit dem BSI wurde statt OpenSSL jedoch die LibreSSL-Bibliothek, ein Fork von OpenSSL, ausgewählt, da OpenSSL bereits in einer vorherigen Studie im Auftrag des BSI ausführlich analysiert wurde. LibreSSL wurde entwickelt mit dem Ziel, eine zuverlässigere und sicherere aber weniger komplexe und gleichzeitig API-kompatible Alternative zu OpenSSL zu bieten. Die Bibliotheken Crypto++, Libtomcrypt, Libgcrypt, Nettle, Beecrypt, Libsodium und NaCl bieten keine Unterstützung für das TLS-Protokoll und wurden daher nicht weiter betrachtet, da der Implementierungsaufwand eines neuen TLS-Stacks im Rahmen des

Projektes zu aufwändig erschien. Die s2n-Bibliothek war zum Zeitpunkt der Analyse noch als experimentell eingestuft. Die Bibliotheken Cryptlib (Berkeley DB) und WolfSSL (GPL v2) standen unter keiner geeigneten Lizenz.

In der detaillierten Analyse lag der Fokus u.a. auf der Implementierung von Gegenmaßnahmen gegen gängige Seitenkanalangriffe, Design und Komplexität der Bibliotheken, enthaltene Zufallsgeneratoren, die Unterstützung von kryptographischer Hardware und mögliche Fehler in der Zertifikats- oder TLS-Protokollvalidierung.

Bei der Untersuchung der Gegenmaßnahmen gegen gängige Seitenkanalangriffe wurden folgende Angriffe betrachtet: Timing-Angriffe, Bleichenbacher-Angriffe, Invalid Curve-Angriffe, Small Subgroup-Angriffe und Padding Oracle-Angriffe auf CBC. In Botan wurden Schwächen in den Gegenmaßnahmen gegen Timing-basierte Bleichenbacher-Angriffe und RSA-Angriffe gefunden, die zu potentiellen Problemen führen konnten. Weiterhin war der Botan TLS-Server für direkte Padding Oracle-Angriffe verwundbar. Andere Angriffe waren hingegen nicht möglich. In LibreSSL und NSS stellten sich Schwächen bei der Resistenz gegen Timing-basierte Bleichenbacher-Angriffe heraus.

Bei der automatisierten Analyse der Zertifikatskettenprüfung mittels des x509test[1] Tools wurden in Botan 7 und in LibreSSL 19 Fehler gefunden. NSS führte als einzige Bibliothek alle Tests erfolgreich aus.

Botan ist sehr gut modularisiert. Bei der Kompilation kann man einzelne der über 50 Module deaktivieren, diese sind dann nicht Teil des Kompilats. Diese Modularisierung erleichtert sowohl die Entfernung nicht zugelassener Algorithmen aus der Bibliothek als auch die Evaluierbarkeit der Implementierung einzelner Algorithmen und Protokolle. LibreSSL erwies sich als weniger gut modularisiert und bot nur wenige Compilerflags an. Es ist unklar, ob die Entfernung von bestimmten Funktionen überhaupt einfach möglich ist. NSS ist am wenigsten gut modularisiert und bietet keine Konfiguration der enthaltenen Funktionen über Compilerflags an. Die einzelnen Module haben untereinander viele Abhängigkeiten. Daher ist unklar, ob die Entfernung von bestimmten Funktionen überhaupt einfach möglich ist.

Botan und NSS implementierten einen NIST SP 800-90A kompatiblen Zufallsgenerator, NSS implementiert keinen zugelassenen Zufallsgenerator. NSS bot dagegen Unterstützung für kryptographische Hardware mittels PKCS#11, Botan und LibreSSL nicht.

Botans Testsuite deckte zum Zeitpunkt der Analyse bereits 75% des Codes mit Tests ab, LibreSSL nur etwa 30% und NSS etwa 50%. Eine hohe Testabdeckung stellt die Qualität der Bibliothek sicher und beugt Implementierungsfehlern vor. Dies ist gerade bei einer Kryptobibliothek von hoher Relevanz, da ein Fehler in der Implementierung gravierende Auswirkung auf die Sicherheit des Gesamtsystems haben kann. In der Literatur finden sich unterschiedliche Angaben für eine ausreichende Testabdeckung, meist wird eine Abdeckung von 70% bis 80% angegeben. Für eine Kryptobibliothek erachten wir eine Abdeckung von 80% als ausreichend.

1 https://github.com/yymax/x509test

Zur Bewertung der einzelnen Kriterien wurde der Aufwand in Personentagen abgeschätzt der benötigt würde um die Missstände in der jeweiligen Bibliothek zu beheben und fehlende Implementierungen zu ergänzen, sodass die Bibliothek den Anforderungen des BSI genügt. Die Ergebnisse wurden anschließend mit denen aus der Voranalyse zusammengeführt und ausgewertet.

Die Botan Kryptobibliothek schnitt zwar bei der Resistenz gegen Seitenkanalangriffe schlechter ab als LibreSSL und NSS, bot jedoch deutliche Vorteile bei Design und Komplexität. Auch die Testsuite deckte mit 75% deutlich mehr Quelltext ab als bei LibreSSL (30%) und NSS (50%). Sie implementierte außerdem von den drei Kandidaten die meisten der vom BSI empfohlenen Algorithmen und Protokolle. Insgesamt ist auch die Codequalität bei Botan am besten zu bewerten. Daher wurde schließlich Botan zur Weiterentwicklung in eine allgemeine Kryptobibliothek, die die Empfehlungen aus den relevanten Technischen Richtlinien des BSI berücksichtigt, ausgewählt.

Umsetzung

In der Implementierungsphase wurde die Architektur Botans ausführlich analysiert und dokumentiert. Das Buildsystem der Bibliothek ist dabei besonders hervorzuheben. Es bietet die Möglichkeit einzelne Algorithmen wie z.B. AES, Protokolle wie z.B. TLS, oder andere Bibliotheksfunktionen vor der Kompilation auszuwählen. Voraussetzung dafür ist die große Modularität der Bibliothek mit über 50 einzelnen Modulen. Abhängigkeiten zwischen Modulen werden automatisch aufgelöst, nicht kompatible Module angezeigt. Darauf aufbauend wurde eine Policy entwickelt, die lediglich vom BSI empfohlene Algorithmen und Protokolle enthält. Die Policy ist vor der Kompilation der Bibliothek auswählbar und bietet damit für Anwender der Bibliothek weiterhin die Flexibilität, andere Algorithmen und Protokolle zur Nutzung auszuwählen. Damit ist kein dauerhafter Fork der Bibliothek notwendig, der den Wartungsaufwand der Bibliothek erhöhen würde.

In der detaillierten Analyse gefundene Mängel in Botan wurden behoben. Dazu gehören Verbesserungen beim RSA-Blinding, bei der Resistenz gegen Bleichenbacher-Angriffe, Small Subgroup-Angriffe und Padding Oracle-Angriffe auf CBC. Die enthaltenen Zufallszahlengeneratoren wurden um zusätzliche Sicherheitsmaßnahmen ergänzt die verhindern sollen, dass nach einem fork()-Aufruf zwei Prozesse die gleichen Zufallszahlen ausgeben. Fehler in der Zertifikatskettenvalidierung wurden genauso behoben wie Fehler in der TLS-Protokollvalidierung. Teile des Quelltextes wurden umstrukturiert, um bestimmte als zu komplex eingestufte Funktionen zu vereinfachen.

In der nächsten Phase des Projektes wurde eine weitere Untersuchung ausgewählter Algorithmen und Angriffe durchgeführt. Dabei wurde ein völlig neuartiges Framework zur Schwachstellenanalyse von TLS-Implementierungen, „TLS-Attacker“, eingesetzt. Die Ergebnisse bestätigen die hohe Qualität und Sicherheit dieser kryptographischen Bibliothek. TLS-Attacker hat während des TLS-Fuzzings einen Buffer Overread gefunden, welcher sich bei einer weiteren Analyse als harmlos gezeigt hat. Desweiteren wurden aktuelle TLS-Angriffe wie Logjam, FREAK, CRIME, BEAST oder DROWN getestet. AES-GCM wurde auf den Umgang mit Nonces überprüft. Keine dieser Analysen hat Fehler aufgefunden. Potentielle Probleme bereiteten jedoch Timing-basierte Seitenkanalangriffe wie Lucky 13, für den eine Gegenmaßnahme implementiert wurde.

Einige in den Technischen Richtlinien BSI-TR-02102-1 und BSI-TR-02102-2 aufgeführten Algorithmen waren bisher nicht in Botan implementiert. Die Implementierungen von KDFs nach NIST SP800-56C, dem hybriden Verschlüsselungsverfahren ECIES, den Signaturverfahren ECGDSA, ECKCDSA und Merkle-Signaturen, dem Message Authentication Code Schema GMAC und der TLS-Erweiterung Encrypt-then-MAC wurden nach aktuellem Stand der Technik umgesetzt und in die Bibliothek integriert. Besonders hervorzuheben ist die Unterstützung für kryptographische Hardware in Form einer PKCS#11-Schnittstelle, die ebenfalls zur Bibliothek hinzugefügt wurde. Damit ist es möglich, statt der durch die

Bibliothek zur Verfügung gestellten Softwareimplementierungen von Kryptoalgorithmen mittels einer geeigneten Middleware diejenigen einer Smartcard, eines Tokens, eines Kryptobeschleunigers oder einer anderen geeigneten kryptographischen Hardware zu nutzen. Auch Schlüssel und Zertifikate können damit in der kryptographischen Hardware verwaltet werden. Die implementierten Schnittstellen bieten außerdem die Möglichkeit zur Initialisierung und Deinitialisierung einer kryptographischen Hardware, beispielsweise zur Personalisierung einer Smartcard.

Bei der Implementierung der Verfahren und Funktionen wurde auf eine ausreichende Testung in Form von Unit Tests geachtet. Implementiert wurden sowohl Ein-/Ausgabetests mit Testvektoren (Known Answer Tests), als auch einfache Funktionstests der implementierten Klassen und Methoden. Des Weiteren wurden bestehende Module mit wenigen Tests identifiziert und um weitere Ein-/Ausgabetests und Funktionstests ergänzt. Als erstrebenswerte Testabdeckung wird in der Literatur 70-80% genannt. Mit 87,8% liegt die Bibliothek mit der Modulpolicy nach den Anforderungen des BSI sogar deutlich über diesem Wert. Die Testsuite wurde außerdem um das TLS-Attacker Framework ergänzt, welches automatisiert etwaige Anfälligkeiten gegen gängige TLS-Angriffe untersucht.

Die Bibliothek wurde mit der Festplattenverschlüsselung TrustedDisk exemplarisch in eine bestehende Anwendung mit einer Zulassung zur Verarbeitung von Verschlusssachen (VS) bis zum Geheimhaltungsgrad VS-NfD integriert. Dabei wurde die in TrustedDisk eingesetzte PKCS#11-Funktionalität zur Ansteuerung einer CardOS-Smartcard, der AES-XTS Betriebsmodus, die SHA-2 Hashfunktion, X.509-Zertifikate und die RSA-Verschlüsselung von anderen Open Source-Bibliotheken bzw. einer hauseigenen Bibliothek auf die neue Bibliothek umgestellt.

Eine gute Dokumentation ist mindestens genauso wichtig wie eine gute Implementierung. Die mangelhafte Ausführung von API-Dokumentation und die Abwesenheit eines qualitativ guten Handbuches für Entwickler sind ein häufiger Kritikpunkt bei auf dem Markt befindlichen Kryptobibliotheken. Nicht selten entstehen Fehler oder gar Sicherheitslücken in IT-Sicherheitsprodukten durch fehlerhafte, veraltete oder unverständliche Dokumentation der Schnittstellen einer Bibliothek. Daher wurde nicht nur auf eine möglichst vollständige API-Dokumentation mittels Doxygen geachtet, sondern auch ein Handbuch zur Verfügung gestellt, was die Module, Klassen und Schnittstellen außerdem anhand von Codebeispielen beschreibt. Zusammen mit der Architekturdokumentation soll dies dem Anwender der Bibliothek größtmögliche Sicherheit bei der Benutzung der Bibliothek gewährleisten und Fehler bei der Benutzung der Schnittstellen vermeiden. Zusätzlich wurde eine Dokumentation der kryptographischen Algorithmen in der Bibliothek angefertigt, die dabei helfen soll, die Korrektheit der kryptographischen Implementierungen der Bibliothek in Übereinstimmung mit den Technischen Richtlinien des BSI nachvollziehen und prüfen zu können.

In der Umsetzungsphase wurde stets eng mit dem Botan Maintainer zusammengearbeitet. Alle im Rahmen des Projektes an Botan durchgeführten Änderungen und Erweiterungen sind an

das Originalprojekt zurückgeflossen. Dadurch entspricht die im Projekt weiterentwickelte Bibliothek im Wesentlichen der aktuellen Version 2.0.1 von Botan.

Wartung & Support

In der dritten Phase, nach Abschluss der Implementierungsarbeiten und der Erstellung der Dokumentation, wird Botan für einen Zeitraum von mehreren Jahren im Auftrag des BSI gewartet. Dazu gehört die Umsetzung von Änderungen an den implementierten Standards - etwa durch neue wissenschaftliche Erkenntnisse zu einzelnen Kryptoverfahren oder durch neue Einsatzszenarien. Inbegriffen ist auch das Schließen von Sicherheitslücken in Botan und das Beheben von funktionalen Fehlern in einem Zeitraum von maximal 4 Wochen nach Bekanntwerden. Dadurch wird die Bibliothek stets auf dem aktuellen technischen Stand gehalten.