Sichere Benutzerauthentifizierung mit mobilen Endgeräten in industriellen Anwendungen

Andreas Schmelter, Oliver Konradi, Stefan Heiss

Institut für industrielle Informationstechnik (inIT)
Technische Hochschule Ostwestfalen-Lippe
Campusallee 6, 32657 Lemgo
andreas.schmelter@hs-owl.de
oliver.konradi@stud.hs-owl.de
stefan.heiss@hs-owl.de

Zusammenfassung. Ein Anwendungsfeld mobiler Endgeräte im Kontext von Industrie 4.0 ist das Prozessmonitoring. Daten werden in Echtzeit auf mobilen Endgeräten dargestellt und geben Aufschluss über den aktuellen Zustand des Prozesses. Im Falle von Fehlerzuständen kann es zielführend sein, mit dem mobilen Endgerät steuernd in den Prozess einzugreifen; allerdings wird dazu eine sichere Nutzerauthentifikation benötigt, um den Prozess vor unbefugten Zugriffen zu schützen, beziehungsweise ein abgestuftes Rechtesystem zu etablieren. Eine solche Nutzerauthentifikation, die alle in IEC 62443 genannten Anforderungen erfüllt, wurde prototypisch umgesetzt.

1 Einleitung

Mit der Digitalisierung in der Produktion (Industrie 4.0) finden auch mobile Endgeräte in diesem Bereich zunehmend mehr Anwendungsmöglichkeiten. Anwendungen, die zum Beispiel Lehrmaterial in Form von Videos oder Augmented Reality-Funktionen beinhalten und direkt vor Ort angewendet werden können, sind hilfreich für eine Einweisung und Bedienung neuer Anlagen und Maschinen. Einzelne Produktionsschritte und die Anzahl der bearbeiteten und gelagerten Produkte sowie deren Standort im Lager lassen sich auf einem mobilen Endgerät nachvollziehen. Außerdem können im Fehlerfall Informationen abgerufen werden und eine digitale Dokumentation erstellt werden, die bei Bedarf auch mit Fotos oder Videos angereichert werden kann [1].

Ein weiterer Anwendungsfall ist eine Adhoc-Überwachung und -Steuerung von industriellen Prozessen, die prinzipiell mit mobilen Endgeräten wie zum Beispiel Tablets oder Smartphones möglich ist. Für die Übertragung der mit den Endgeräten auszutauschenden Prozessparameter bietet sich hierbei beispielsweise das durch die „Open Platform Communications Unified Architecture“ (OPC UA) definierte Protokoll an. OPC UA bietet eine plattformunabhängige und sichere Vernetzung von industriellen Anlagen [2] und kristallisiert sich immer mehr als wichtiger Bestandteil zukünftiger Industrie-4.0-Standards heraus [3].

Bei einer Nutzung mobiler Endgeräte, deren Bedienung nicht zwingend eine räumliche Nähe zu einem zu überwachenden Prozess voraussetzen, sind besondere Anfor-

J. Jasperneite, V. Lohweg (Hrsg.), *Kommunikation und Bildverarbeitung in der Automation*,
Technologien für die intelligente Automation 12, https://doi.org/10.1007/978-3-662-59895-5_14

derungen an die Authentifizierung berechtigter Nutzer gegeben. Im Rahmen des Protokollschrittes „ActivateSession“ bietet OPC UA verschieden Möglichkeiten der Benutzerauthentifizierung, unter anderem mittels Passwort oder einem sogenannten X509IdentityToken (vgl. [4], 7.3.5, S. 161). Bei einem X509IdentityToken besitzt der Nutzer einen privaten Schlüssel mit dem im Rahmen einer Authentifizierung eine digitale Signatur erstellt wird. Der private Schlüssel muss hierzu auf dem Endgerät verfügbar sein. Wird er auf dem Endgerät gespeichert, so sollte dessen Verwendung zumindest durch ein starkes Passwort geschützt sein, welches von dem berechtigten Nutzer gemerkt und zum Verbindungsaufbau eingegeben werden muss.

Durch die genannten Optionen der Benutzerauthentifizierung kann die in IEC62443 genannte Forderung nach einer eindeutigen Benutzerauthentifizierung [5] erfüllt werden. Eine deutlich sicherere und benutzerfreundlichere Authentifizierung lässt sich allerdings durch eine Nutzung von SmartCards erreichen. Im Zusammenhang mit der o.g. X509IdentityToken-basierten Authentifizierung kann eine solche Zweifaktorauthentifizierung realisiert werden, bei der der private Schlüssel sicher und unabänderlich an eine nutzerspezifische SmartCard gebunden ist. Ein Nutzer muss sich anstelle eines komplexen Passwortes nur noch eine persönliche Identifikationsnummer (PIN) merken. Die Anbindung der SmartCard an verschiedene mobile Geräte kann per Near-Field-Communication (NFC) erfolgen.

Ein entsprechendes Verfahren einer differenzierten Nutzerauthentifikation, welches durch die Anwender einfach und intuitiv zu handhaben ist, wurde prototypisch umgesetzt und soll in diesem Beitrag detailliert beschrieben werden. Neben der SmartCard, die als sicherer Speicher für den privaten Schlüssel eingesetzt wird, kommt in der prototypischen Umsetzung eine Public-Key-Infrastruktur (PKI) zum Einsatz, durch die die den X509IdentityToken entsprechenden X509-v3 Zertifikate gemanagt und verwaltet werden können. Durch die PKI wird zusätzlich zu der eindeutigen Benutzeridentifizierung, eine zentrale Verwaltung der Identifizierung und Authentifizierung von Nutzern ermöglicht. In IEC 62443 wird eine solche zentrale Verwaltung als optionale Erweiterung der eindeutigen Benutzeridentifizierung vorgeschlagen (vgl. [5], 5.7.2, S. 28).

2 OPC-UA

OPC-UA ist eine plattformunabhängige Norm, die eine sichere Kommunikation zwischen einem Client und einem Server definiert. In industriellen Anwendungen können Prozesse von Benutzern oder technischen Systemen überwacht und gesteuert werden, indem ein OPC-UA-Server einem OPC-UA-Client Dienste bereitstellt [2]. Für den Zugriff und die Nutzung der Dienste wird ein Kommunikationskanal aufgebaut. Dazu können im Vorfeld Informationen über verfügbare Endpunkte des OPC-UA-Servers abgerufen werden. Endpunkte definieren die zur Absicherung des Kommunikationskanals zu verwendenden Mechanismen und Algorithmen sowie die Art der Benutzerauthentifizierung (User Identity Token) [4].

Folgende Typen von User Identity Token sind in OPC UA spezifiziert: AnonymousIdentityToken, UserName-IdentityToken, IssuedIdentityToken und X509IdentityToken (vgl. [4], 7.35, S.161). Das AnonymousIdentityToken bietet dem OPC-UA-

Server keine Informationen über den Benutzer, was dazu führt, dass keine Identifikation möglich ist. Bei einer Nutzung dieses Token müssen die Zugriffsrechte auf vertrauliche Daten und Dienste eingeschränkt werden, um diese zu schützen. Mittels des UserNameIdentityToken erfolgt die Anmeldung am Server durch einen Benutzernamen und ein zugehöriges Passwort. Eine Nutzung dieses Verfahrens erfordert vom Server keine besonderen Funktionen, allerdings hängt die Sicherheit des Verfahrens nur von der Qualität das Passwortes und der Geheimhaltung des selbigen ab. Alternativ dazu kann das IssuedIdentityToken verwendet werden, welches durch einen externen Authentifizierungsdienst erzeugt wird; zum Beispiel mittels des Ticket Granting Services von Kerberos.

Als letzte Option kann ein Benutzer mittels eines X509IdentityToken identifiziert werden. Das X509IdentityToken setzt sich aus zwei Elementen zusammen: Einem X.509-Zertifikat und einer Signatur des Erstellers des Tokens (vgl. Kapitel 3).

Neben des Typs des User Identity Tokens wird durch einen Endpunkt der Message Security Mode festgelegt: Nachrichten können ohne eine Anwendung von Sicherheitsfunktionen übertragen werden (Modus: none), sie können um einen Message Authentication Code ergänzt werden (Modus: sign) oder zusätzlich verschlüsselt werden (Modus: sign and encrypt). Zusätzlich wird ein Application Instance Certificate gesendet, welches sowohl den genutzten Server, als auch den genutzten Dienst identifiziert (vgl. [6], Kap. 2.1).

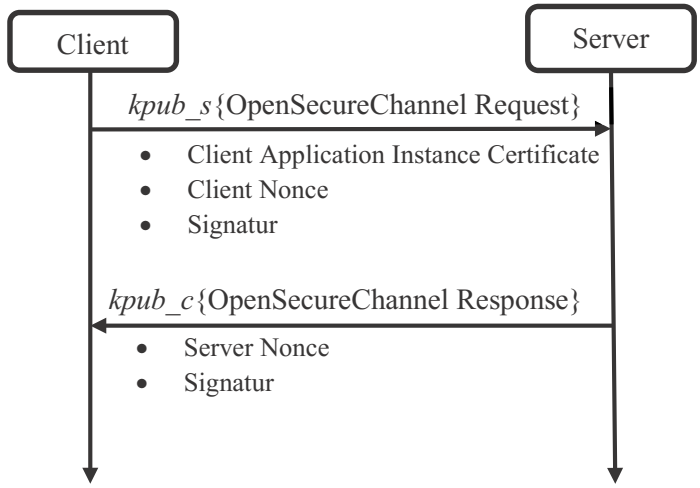

Abb. 1. Aufbau des SecureChannel

Abbildung 1 stellt den Aufbau eines sicheren Kommunikationskanals (SecureChannel) dar, wobei alle Nachrichten durch den öffentlichen Schlüssel des Kommunikationspartners verschlüsselt werden. Der Aufbau wird durch den OpenSecureChannel-Request des Clients initiiert. Innerhalb des Request werden diverse Daten verschickt (Client Application Instance Certificate, Nonce, Signatur der Daten). Die Signatur wird durch den privaten Schlüssel des Client erstellt und umfasst alle vorher aufgezählten

Daten (vgl. [7], 6.7.2, S.43). Abschließend wird die Nachricht durch den öffentlichen Schlüssel des Servers verschlüsselt und verschickt. Dieser kann die Nachricht nun entschlüsseln und die Signatur des Clients durch dessen öffentlichen Schlüssel prüfen. Sofern das Zertifikat (noch) gültig ist, verläuft der anschließende Validierungsprozess des Servers erfolgreich und der „OpenSecure-Channel-Response“ wird erzeugt. Der „OpenSecureChannel-Response“ beinhaltet eine vom OPC-UA-Server generierte Zufallszahl (nonce) und eine Signatur, welche über die gesamte OpenSecureChannel-Request-Nachricht berechnet wird. Diese Daten werden mit dem öffentlichen Schlüssel des OPC-UA-Clients verschlüsselt und an selbigen gesendet. Anschließend können Client und Server identische symmetrische Schlüssel (*k_symm*) auf Basis der ausgetauschten Zufallszahlen, generieren [7]; der weitere Einsatz der Schlüssel obliegt dem gewählten Security Mode (none, sign, sign and encrypt). Innerhalb des SecureChannel kann eine Session erstellt und aktiviert werden.

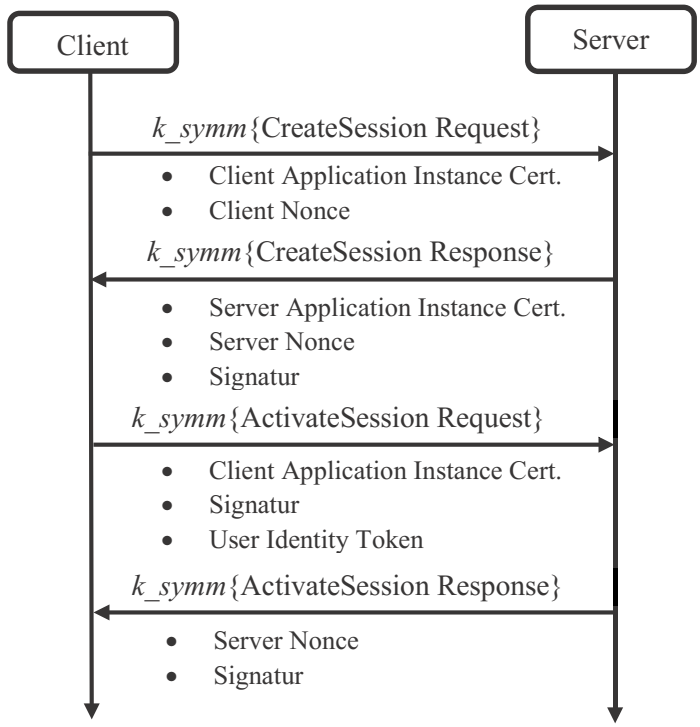

Abb. 2. Erstellen und Aktivieren einer Session

Abbildung 2 stellt den Aufbau einer Session dar. Der Aufbau beginnt mit dem „CreateSession-„Request, welcher unter Anderem das Application Instance Certificate und eine Zufallszahl (nonce) des Clients beinhaltet; der Request und alle folgenden Nachrichten ist mit *k_symm* verschlüsselt. Nachdem der OPC-UA-Server das Application Instance Certificate validiert hat, wird der Response mit einem „CreateSession-„

Response beantwortet. Dieser beinhaltet unter anderem das Application Instance Zertifikat des Servers, eine vom OPC-UA-Server generierte Zufallszahl sowie eine Signatur, erzeugt durch den Server, über das Application Instance Certificate desselben und die Zufallszahl des Clients. Da der Client den öffentlichen Schlüssel des Servers aus dem Zertifikat kennt, kann er die Signatur prüfen und somit sichergehen, dass der Server im Besitz des entsprechenden privaten Schlüssels ist. Innerhalb des nun folgenden „ActivateSession-„Request erbringt der Client ebenfalls einen Identitätsnachweis. Dazu wird eine Signatur, über das enthaltene Application Instance Certificate des Clients, gebildet und zusammen mit dem User Identity Token verschickt. Wird das X509IdentityToken genutzt, so erzeugt der Client eine zweite Signatur und fügt diese mit dem Token hinzu (vgl. Aufbau des SecureChannel) [4].

Schlüsselpaare und Zertifikate, die wie oben beschrieben zur Etablierung sicherer Kommunikationskanäle und zur Authentifizierung von Nutzern eingesetzt werden, müssen im Vorfeld generiert und verteilt werden. Wie diese Aufgaben im Rahmen einer minimalistischen Public-Key-Infrastruktur (PKI) gelöst werden können, wird im nächsten Abschnitt beschrieben.

3 Nutzung einer PKI im Zusammenhang mit einem Konfigurations- und Nutzermanagementsystems

Wie im letzten Abschnitt dargestellt, werden im Rahmen von OPC UA für den Aufbau sicherer Verbindungen und zur Authentifizierung von Nutzern mit einem X509IdentityToken diverse Zertifikate benötigt. Die zur Erzeugung und Verwaltung dieser Zertifikate benötigte PKI wird im einfachsten Fall durch die Etablierung einer einzigen Zertifizierungsstelle (Certification Authority, CA) als vertrauenswürdige Instanz realisiert. Diese CA besitzt ein Schlüsselpaar, dessen öffentlicher Schlüssel (Public Key, k_{pub_CA}) an alle weiteren Kommunikationsteilnehmern zu verteilen ist. Mit ihrem privaten Schlüssel (Private Key, k_{priv_CA}) kann die CA Signaturen erstellen, welche dann von allen weiteren Teilnehmern mithilfe des öffentlichen Schlüssels k_{pub_CA} verifiziert werden können.

Innerhalb einer PKI besitzen alle Kommunikationsteilnehmer ebenfalls ein spezifisches Schlüsselpaar (User *Ui* das Schlüsselpaar k_{priv_Ui}, k_{pub_Ui}), sodass sie sich untereinander mithilfe von Signaturen authentifizieren können. Damit der jeweils für die Verifikation benötigte öffentliche Schlüssel von der Gegenseite nicht vorgehalten werden muss, wird er in einem von der CA signierten Zertifikat $Cert_{CA}(Ui, k_{pub\_Ui})$ (üblicher Weise ein X509-Zertifikat in der Version 3, vgl. [8]) zusammen mit der Signatur an die Gegenseite geschickt. Neben der eigentlichen Verifikation der Signatur mit dem im Zertifikat enthaltenen öffentlichen Schlüssel muss also noch die Signatur des Zertifikats mithilfe des öffentlichen CA-Schlüssels verifiziert werden.

Neue Teilnehmer müssen im Rahmen einer PKI also einerseits mit dem öffentlichen Schlüssel der CA ausgestattet werden, um Signaturen anderer Teilnehmer verifizieren zu können, und andererseits ein Schlüsselpaar generieren und sich ein Zertifikat von der CA ausstellen lassen, um selbst verifizierbare Signaturen erzeugen zu können. Da-

mit die CA ein Zertifikat erzeugen kann, benötigt sie die den Teilnehmer identifizierenden Daten zusammen mit dessen öffentlichen Schlüssel. Häufig werden diese Daten in einem sogenannten Certification Request [9] an eine CA übertragen (s. Abbildung 3).

Soll eine PKI im Rahmen einer wohldefinierten, zentral gemanagten Umgebung genutzt werden, so besteht prinzipiell die Möglichkeit die CA-Funktionalitäten in ein Konfigurations- und Nutzermanagementsystem zu integrieren, sodass die Erzeugung von Schlüsselpaaren und Zertifikaten durch dieses System ausgeführt wird (s. Abbildung 4). Beispielsweise könnten von einem solchen System Smartcards nutzerspezifisch personalisiert und mit den benötigten Schlüsseln und Zertifikaten ausgestattet werden. Ein entsprechendes Szenario liegt der in dieser Arbeit vorgestellten Lösung zugrunde.

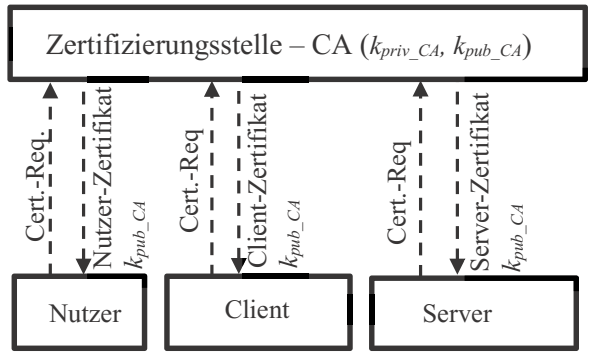

Abb. 3. Ausstellen und Verteilen der Zertifikate

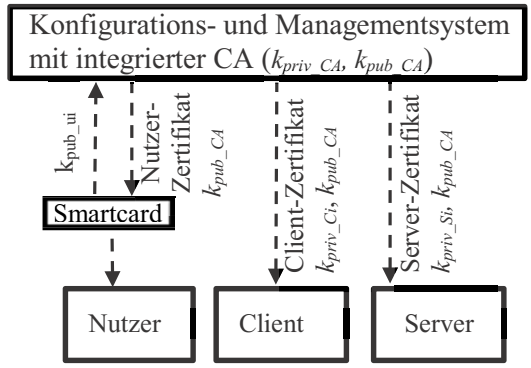

Abb. 4. Ausstellen und Verteilen von Zertifikaten durch ein Konfigurations- und Managementsystem

4 Smartcards als Schlüssel zum Zugriff auf Komponenten eines Netzwerks und ihre Personalisierung

Im Zusammenhang mit der Authentifizierung von Personen ist eine Smartcard als Träger des der Person zugeordneten privaten Schlüssels sehr gut geeignet, denn der private Schlüssel ist (nachweisbar) nur

- mit Besitz der Smartcard und
- nach Eingabe einer PIN des Besitzers

nutzbar. Smartcards stellen eine physikalisch wohldefinierte und abgegrenzte Umgebung dar, innerhalb der ein privater Schlüssel genutzt werden kann. Sie repräsentieren somit eine für Nutzer offensichtliche Entsprechung physikalischer Schlüssel, mit denen der Zugang zu Gebäudeteilen geregelt wird, nur dass sie eben die Zugangsregelung zu technischen Systemen nutzerspezifisch ermöglichen. Gegenüber einem üblichen Schlüssel besitzen Smartcards, die erst nach Eingabe einer nutzerspezifischen PIN eine Verwendung des gespeicherten privaten Schlüssels erlauben, einen wesentlichen Schutz vor ihrem Missbrauch nach einem Verlust oder Diebstahl (Zwei-Faktor-Authentifizierung). Zusätzlich erlaubt der beschriebene Einsatz von Smartcards die sichere Authentifizierung von Nutzern, die diese auch im Nachhinein nicht abstreiten können.

Lösungen, die auf eine hardwaregestützte Sicherung der privaten Schlüssel verzichten, müssen diese in geeigneter Form direkt auf den Geräten der Nutzer speichern. Neben der Bindung eines Schlüssels an ein Gerät sind auch viele der im letzten Absatz genannten vorteilhaften Eigenschaften nicht mehr realisierbar. Um einen gewissen Schutz vor einer missbräuchlichen Verwendung der privaten Schlüssel zu erreichen, werden diese häufig mithilfe nutzerspezifischer Passwörter verschlüsselt gespeichert. Um hier nun einen ausreichenden Schutz gegen Brute-Force-Angriffe zu gewährleisten, sind diese Passwörter hinreichend komplex zu wählen. Demgegenüber kann der Zugriffsschutz auf die privaten Schlüssel bei einer Verwendung von Smartcards durch relativ kurze PINs erreicht werden, da Smartcards zur Abwehr von Brute-Force-Angriffen über Fehlbedienungszähler verfügen.

Von einem Konfigurations- und Managementsystem, wie es in Kapitel 3 beschrieben ist, kann die Personalisierung einer Smartcard nach Erfassung der einen Nutzer identifizierenden Daten durchgeführt werden. Im Rahmen der hier beschriebenen Implementierung werden die in Abbildung 5 dargestellten Kommandos mit einer Smartcard ausgetauscht. Am Ende des dargestellten Prozesses erhält ein Nutzer eine Smartcard mit seinem Schlüsselpaar, seinem Zertifikat und dem öffentlichen Schlüssel der CA.

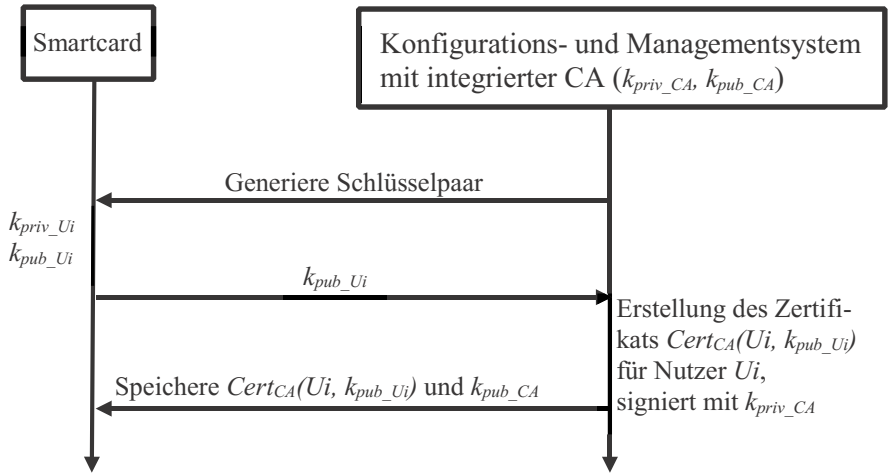

Abb. 5. Personalisierung einer Smartcard

5 Benutzerauthentifizierung mit einem mobilen Endgerät und einer Smartcard

Das im Folgenden beschriebene Konzept der Benutzerauthentifizierung wurde mit einem einfachen Demonstrator für einen industriellen Prozess umgesetzt. Abbildung 6 stellt diesen Demonstrator dar.

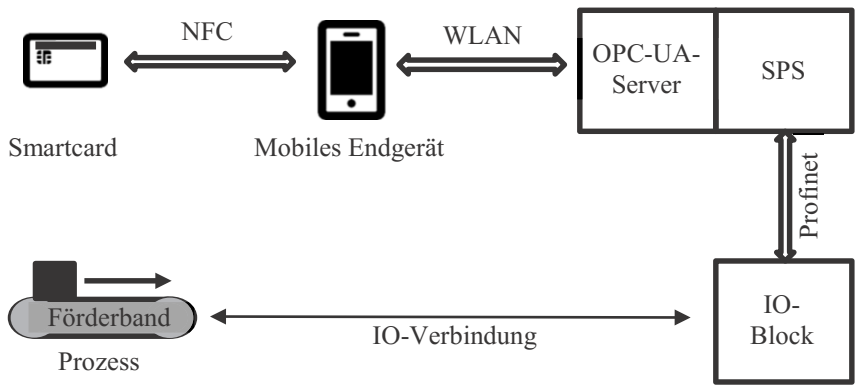

Abb. 6. Aufbau des Demoprozesses

Ein Prozess ist an einen IO-Block angebunden und wird von einer speicherprogrammierbaren Steuerung (SPS) angesteuert, wobei die generierten Prozessdaten als Nodes eines OPC-UA-Servers bereitgestellt werden; neben der Bereitstellung der Prozessdaten dienen die Nodes auch zur Steuerung des Prozesses. Der OPC-UA-Client kann je nach Status der Benutzerauthentifizierung und der zugeteilten Benutzerrechte lesend oder schreibend auf die Nodes zugreifen. Anhand der nutzerspezifischen Daten in den

X.509-Zertifikaten können die Nutzer nicht nur als anonym oder authentifiziert kategorisiert werden, sondern es kann auch zwischen authentifizierten Nutzern individuell unterschieden werden, sodass sich prinzipiell ein feingranulares Zugriffsrechtesystem realisieren lässt.

Im Zusammenhang mit dem Demonstrator kommen handelsübliche mobile Endgeräte (Tablets, Smartphones) als OPC-UA-Clients zum Einsatz. Diese bauen zunächst einen SecureChannel mit dem Message Security Mode sign and encrypt zu dem OPC-UA-Server auf. Hierzu benötigt der Client ein spezifisches Client Application Instance Certificate, den dazugehörigen privaten Schlüssel und das Root-Zertifikat des OPC-UA-Servers, welches den öffentlichen Schlüssel k_{pub_CA} enthält. Diese kryptographischen Schlüssel und Zertifikate werden zusammen mit der OPC-UA-Clientapplikation auf dem Endgerät fest (und möglichst sicher) installiert.

Auf Basis dieses Secure Channels kann bereits ohne weitere Nutzerauthentifikation (Token-Typ: Anonymous-IdentityToken) auf die Prozessdaten zugegriffen werden. Für einen schreibenden Zugriff, der einen aktiven Eingriff in den Prozess erlaubt, muss sich ein Nutzer mit einem X509IdentityToken authentifizieren. Dazu greift das mobile Endgerät mittels NFC auf die Smartcard des Nutzers zu und bildet das benötigte X509IdentityToken, welches anschließend, im Protokollschritt ActivateSession, als User Identity Token genutzt wird. Den Kommunikationsablauf zwischen Smartcard, mobilem Endgerät und Server zeigt Abbildung 7.

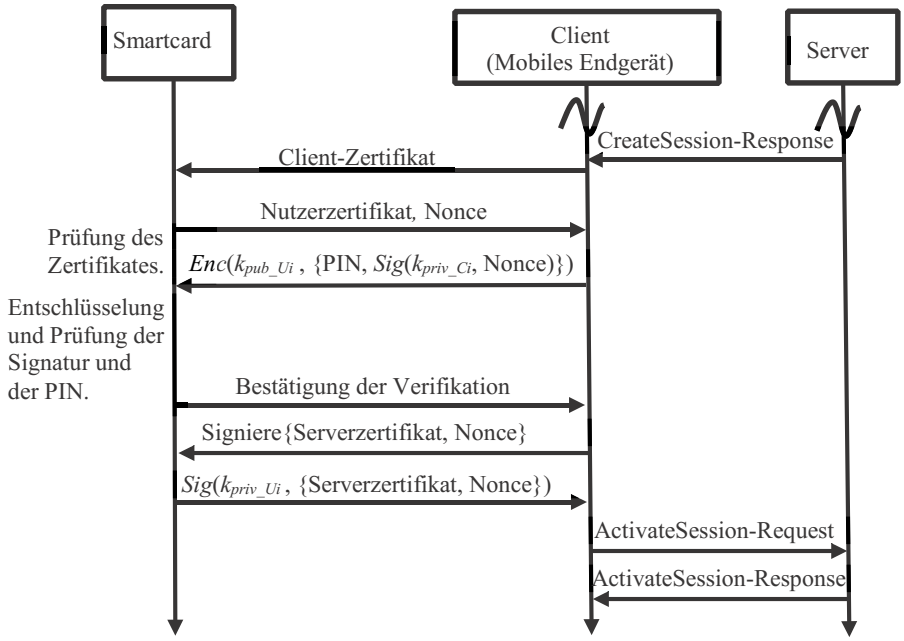

Abb. 7. Erzeugen des X509IdentityToken

Die in Abbildung 7 dargestellte Kommunikation mit der Smartcard beginnt nach dem Empfang des „CreateSession“-Response. Zuerst sendet das mobile Endgerät sein

Client Application Instance Certificate an die Smartcard, damit diese über den öffentlichen Client-Schlüssel k_{pub_Ci} verfügt. Nachdem die Smartcard die Vertrauenswürdigkeit des Zertifikates mit dem öffentlichen Schlüssel k_{pub_CA} der CA geprüft hat, schickt sie das Nutzerzertifikat und einen Zufallswert (Nonce) an das mobile Endgerät. Im Anschluss erzeugt das mobile Endgerät eine Nachricht bestehend aus der PIN und einer Signatur über den zuvor empfangenen Nonce und überträgt diese Nachricht verschlüsselt, mit dem öffentlichen Schlüssel k_{pub_Ui}, an die Smartcard. Die Smartcard entschlüsselt die Nachricht mit k_{priv_Ui} und prüft die Signatur. Ist die Signatur gültig, erfolgt die Prüfung der PIN. Nach der Bestätigung der Nutzeridentität, können die nachfolgenden Befehle ausgeführt werden. Zur Erzeugung der Signatur werden das Server Application Instance Zertifikat und die Zufallszahl des OPC-UA-Servers (Nonce) aus der „CreateSession Response"-Nachricht an die Smartcard gesendet (vgl. Abbildung 2). Für diese Daten erzeugt die Smartcard mit dem privaten Schlüssel k_{priv_Ui} eine Signatur. Aus dieser Signatur und dem Nutzerzertifikat wird das X.509IdentityToken gebildet und innerhalb des ActivateSession-Request an den Server übertragen. Anschließend verifiziert der Server die im X.509IdentityToken enthaltene Signatur, um so den Nutzer zu authentifizieren.

Eine Begründung für das oben skizzierte Protokoll zur Übertragung der PIN an die Smartcard erfolgt im Rahmen der sicherheitskritischen Betrachtung der Near-Field-Kommunikation im nächsten Abschnitt.

6 Sicherheitskritische Betrachtung von NFC im Kontext der Applikation

In der Literatur [11, 12, 13] werden verschieden Angriffsmöglichkeiten gegen NFC-Tags selbst (Cloning) als auch gegen die Übertragungsstrecke (Datenmodifikation, Datainjection, Abhören (Eavesdropping), Relay-Attack) betrachtet. Ein Cloning kann allerdings bei einer Smartcard ausgeschlossen werden [12]. Bei Angriffen auf der Luftschnittstelle geht es entweder darum, übertragene Daten mitzulesen oder zu manipulieren.

Ein Abhören (Eavesdropping) der einer Funkübertragung zugrundeliegenden elektromagnetischen Wellen ist prinzipiell immer möglich und daher bei einer Sicherheitsbetrachtung zu berücksichtigen. Die Funkübertragung von NFC kann, je nach Gerätetyp (aktiv/passiv), in einer Entfernung von bis zu 10 Metern abgehört werden [10].

Vor der eigentlichen Nutzung der Smartcard zur Signaturerzeugung muss sich der Besitzer der Smartcard durch der Übertragung der PIN gegenüber der Smartcard authentifizieren. Die PIN ist bei dieser Übertragung vor einem möglichen Mitlesen zu schützen und wird deshalb verschlüsselt übertragen. Die Möglichkeit eines Replay-Angriffs durch die Wiederverwendung einer mitgeschnittenen PIN-Übertragung ist durch die Integration (einer Signatur von) einer durch die Smartcard zuvor generierten Nonce ausgeschlossen.

Würden PIN und Nonce lediglich verschlüsselt übertragen werden, böte sich jedem potentiellen Angreifer, der in die Nähe der Smartcard gelangen kann, die Möglichkeit der Durchführung eines Denial-of.Service-Angriffs auf die Smartcard, indem er das PIN-Übertragungsprotokoll so oft mit beliebigen (falschen) PIN-Werten durchführt, bis

die Smartcard für eine weitere Nutzung gesperrt ist. Durch die Verwendung der Signatur wird dieses Angriffsszenario auf Angreifer beschränkt, die im Besitz eines privaten Schlüssels sind für dessen zugehörigen öffentlichen Schlüssel ein durch die eingesetzte CA erstelltes Zertifikat existiert.

Bei einem Relay-Angriff wird die Übertragungsstrecke zwischen Smartcard und Smartcardterminal überbrückt, um eine Aktion ohne direkten Besitz der Smartcard auszuführen. Da jedoch alle sicherheitsrelevanten Operationen nur nach Eingabe der PIN möglich sind, ist dieser Angriff im Zusammenhang mit der in dieser Arbeit beschriebenen Lösung nicht durchführbar.

Um eine aus Sicht eines Angreifers erfolgreiche Modifikation oder Einfügung von Daten (Datainjection) prinzipiell zu verhindern, werden die mit der Smartcard ausgetauschten Daten mit einem MAC (Message Authenticaton Code) versehen. (Der Einfachheit halber wurde auf deren Darstellung in Abb. 7 verzichtet.)

7 Zusammenfassung

Die vorliegende Arbeit zeigte, wie eine differenzierte Nutzerauthentifikation, welche durch die Anwender einfach und intuitiv zu handhaben ist, in aktuellen industriellen Anwendungen mit einer Nutzung handelsüblicher mobiler Endgeräte umgesetzt werden kann. Diese Nutzerauthentifikation kann, je nach gewünschtem Anwendungsfall, weiter verfeinert werden und lässt sich gut im Zusammenspiel mit einem feingranularen Rechtemanagementsystem nutzen.

Die Integration einer eigenen CA-Funktionalität in Rechtemanagement- und Konfigurationssysteme erspart Aufwände und Abhängigkeiten gegenüber Dritten bei der Erstellung von Authentifizierungsmitteln (Zertifikate). Die Nutzung von Smartcards bietet den Vorteil einer Zwei-Faktor-Authentifizierung und gibt den Nutzern darüber hinaus einen physikalischen „Schlüssel“ an die Hand, mit dem sie sich die ihnen jeweils gewährten Zugang zum System freischalten können. Im Zusammenhang mit der Nutzung der NFC-Schnittstelle wurden mögliche Angriffsszenarien untersucht. Durch eine Verwendung geeigneter kryptographischer Mechanismen können diese jedoch ausgeschlossen werden.

Literatur

[1] „Mobile devices in Industry 4.0,“ 31. Mai 2016. [Online]. Available: http://www.criticalmanufacturing.com/pt/newsroom/blog/posts/blog/mobile-devices-in-industry-4-0#.W59e8fZCTUg. [Zugriff am 17. 09. 2018].

[2] OPC Unified Architecture – Teil 1: Übersicht und Konzepte (IEC 62541-1:2010).

[3] „Verband Deutscher Maschinen- und Anlagenbau (VDMA): Industrie 4.0 Kommunikation mit OPC UA - Leitfaden zur Einführung in den Mittelstand,“ 2017. [Online]. Available: https://industrie40.vdma.org/documents/4214230/16617345/1492669959563_2017_Leitfaden_OPC_UA_LR.pdf/f4ddb36f-72b5-43fc-953a-ca24d2f50840. [Zugriff am 22. 04. 2018].

[4] „OPC Unified Architecture - Teil 4: Dienste, (IEC 62541-4:2015)“.

[5] „IEC 62443 Part 3: Industrial communication network - Network and system security - Part 3-3 System Security requirements and security levels, Edition 1.0," 2013.

[6] OPC Foundation, „The OPC UA Security Model for Administrators v1.0," 7. July 2010.

[7] „OPC Unified Architecture - Teil 6: Protokollabbildungen, (IEC 62541-6:2015)".

[8] „RFC 5280," Mai 2008. [Online].Available: https://www.ietf.org/rfc/rfc5280.txt. [Zugriff am 10. 10. 2018].

[9] „RFC 2986," 11 2000. [Online]. Available: https://www.ietf.org/rfc/rfc2986.txt. [Zugriff am 10 10 2018].

[10] Google, „Android Developers," 14. 8. 2018. [Online]. Available: https://developer.android.com /training/articles/keystore#SecurityFeatures. [Zugriff am 5. 10. 2018].

[11] E. H. a. K. Breitfuß, „Security in Near Field Communication (NFC)," Gratkorn, Austria.

[12] K. M. Keith Mayes, Smart Cards, Tokens, Security and Applications; Seccond Edition, Springer, 2017.

[13] G. K. U. K. S. R. Anusha Rahul, „NEAR FIELD COMMUNICATION (NFC) TECHNOLOGY: A SURVEY," *International Journal on cybernetics & Informatics (IJCI) Wol. 4, No. 2,* April 2015.