PKI and User Access Rights Management for OPC UA based Applications

Gajasri Karthikeyan, Stefan Heiss
inIT Institute Industrial IT
Ostwestfalen-Lippe University of Applied Sciences
32657 Lemgo, Germany
{gajasri.karthikeyan, stefan.heiss}@hs-owl.de

Abstract*—The growing need for end-to-end security in distributed communication for industrial automation as emphasized in Industrie 4.0 requires an investigation of the security features of relevant protocols. One of the security requirements is authentication and authorization of users within and across organizational boundaries. OPC UA (Open Platform Communication Unified Architecture) is a service-oriented architecture for platform independent communication in automation industry. This research work is on OPC UA to understand its security architecture's support for end-to-end communication and an implementation of a demo PKI (Public Key Infrastructure) to illustrate the same. The design and implementation of such a PKI facilitates both, offline and online validation services. This work emphasizes different concepts of PKIs used in enabling security in applications based on OPC UA standards. The information modelling provided by OPC UA has options to enable user access rights. The applicability of access level attributes in differentiating access rights between different users is demonstrated. The results of this work illustrate a single level hierarchical trust model for end-to-end communication using X509IdentityToken authentication for a user to access services provided by an OPC UA server. The demonstration of online validation for X.509 certificate using OCSP (Online Certificate Status Protocol) protocol is illustrated. The offline validation using CRL (Certificate Revocation List) is also illustrated. The X.509 certificates required for OPC UA based applications can be generated using a tool called keytool. A open source project of keytool is used to create the OPC UA specific extensions for the certificates. There are several challenges in implementing such an infrastructure for distributed systems and they are described. The scope for further research is discussed briefly.

I. Introduction

End-to-end security in industrial communication is one of the research areas focused in the context of Industrie 4.0 [1]. The need for cross company communication for automated information exchange has highlighted the importance of enabling trusted communication links. In order to enable trusted communication, detailed investigations of security requirements, secure identities [2] and their applicability are required. Each entity trying to enable a secure communication needs to be associated with a secure identity. A secure identity is a unique identity with additional security properties for trustworthy authentication of the entity [2], and every secure identity requires a secure element as a credential carrier. An important security requirement is to authenticate and authorize users in order to differentiate and restrict the access to data within and across organizational boundaries. There are applications built for different requirements and based on different middlewares enabling different options for secure communication. This paper describes the research work on one such architecture for enabling end-to-end secure communication in industrial automation. OPC UA is a platform independent service-oriented architecture [3]. The security architecture of OPC UA highlights the need for technical and organizational infrastructure like a Public Key Infrastructure (PKI). And, node management provided by OPC UA facilitates differentiation of access level through attributes.

Certificate management for OPC UA based applications as stated in [4] gives an overview of different options available. One of the applicable PKI frameworks using Java keytool utility is discussed in [4], it is studied and implemented in this work. Research on finding a feasible solution for user access rights management for OPC UA based applications is another interesting area of research where working on Role Based Access Control (RBAC) as stated in [5] is one specific example for how AutomationML can be used. Nevertheless, the different methods for implementing user authorization based on RBAC are to be studied for their applicability in general.

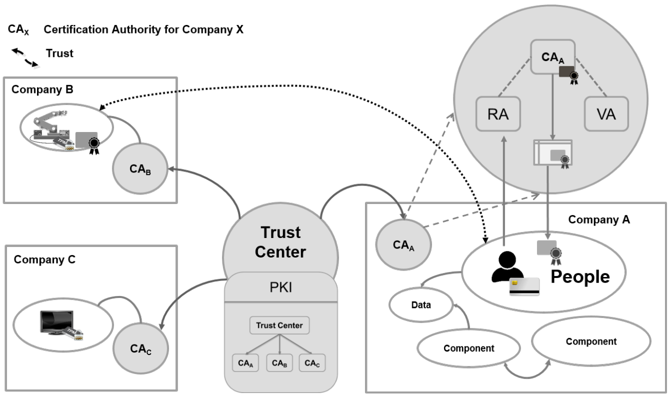


Figure 1. Secure end-to-end communication within and across organizational boundaries [1]

Figure 1 is adapted from publication of the Platform Industrie 4.0 [1]. The example scenario in figure 1 shows 3 different companies establishing trust relationships using a

978-1-5386-7108-5/18/$31.00 ©2018 IEEE

trust center. There are different entities involved in each of these communication links within a company and they can be components, people, etc. Each entity trying to establish a secure end-to-end communication needs to authenticate itself using a secure identity. Depending on the secure identity chosen (e.g. a private key with an associated X.509 v3 certificate), the carrier for the identity is a secure element (e.g. a smart card) [2]. The trust link shown in figure 1 shows a user from company A connecting to a machine in company B using secure identities. In order to design and maintain such a trusted relationship, a PKI to be used is demonstrated in this paper considering the OPC UA architecture.

Further sections of this paper are described as follows: section II describes PKI concepts, section III gives an overview of the OPC UA security infrastructure, section IV details OPC UA validation services required for application instance certificates and also the different options for authentication and authorization of users, section V illustrates the proposed demo PKI used for OPC UA based applications, section VI consolidates the advantages and challenges in implementing such a security infrastructure. The conclusion gives an overview of the proposed solution and further subjects for research.

II. Public Key Infrastructures and Validation Services

A Public Key Infrastructure (PKI) has different entities like Certification Authorities (CA), Validation Authorities (VA) and Registration Authorities (RA) in order to create and manage X.509 certificates used to establish trust between two communicating partners [6]. Figure 2 shows a simple use case where a user tries to obtain a X.509 certificate signed by a CA. The CSR (Certificate Signing Request) is sent from the user to the CA. The identity of the requesting user is verified by the RA and the verified user is issued a signed certificate. The user uses the signed X.509 certificate to authenticate and establish a secure communication with an application. The application uses validation services provided by the CA. In this illustration a single CA issues certificates and validates the same.

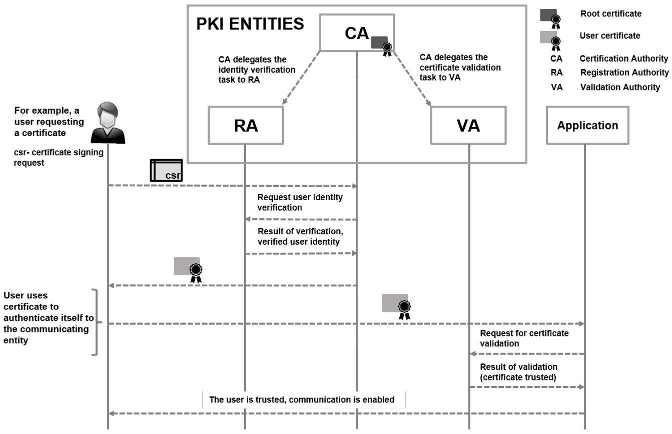

PKI ENTITIES
CA
RA
VA
Application
Root certificate
User certificate
CA Certification Authority
RA Registration Authority
VA Validation Authority
CA delegates the identity verification task to RA
CA delegates the certificate validation task to VA
For example, a user requesting a certificate
csr- certificate signing request
csr
Request user identity verification
Result of verification, verified user identity
User uses certificate to authenticate itself to the communicating entity
Request for certificate validation
Result of validation (certificate trusted)
The user is trusted, communication is enabled

Figure 2. Public Key Infrastructure

For example, figure 3 shows an example PKI that needs to be created and maintained for a company 'A' which is seen in figure 1. It shows different subordinate or intermediate CAs involved in issuing certificates for users and machines. Figure 3 also depicts user access rights management, secure elements as credential carriers and trusted communication links within and across organizational boundaries. This is a simple use case to show the applicability of PKI in an industrial environment.

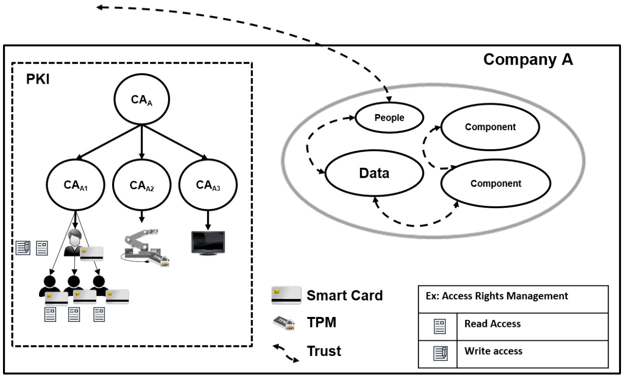


Figure 3. Example PKI - Company A

There are different checks performed in order to determine if the certificate to be validated is trusted. This includes signature verification, time period validity, certificate revocation status and some checks exclusive to certain architectures like OPC UA. OPC UA ApplicationInstanceCertificate validation steps are detailed in further sections. The following sections describe online and offline revocation methods.

A. *CRL - Offline Validation*

Certificate Revocation List (CRL) is an offline approach in order to determine the revocation status of a certificate. A CRL is a time stamped list identifying revoked certificates using serial numbers [6]. It is signed by a CA or CRL issuer [6].

B. *OCSP - Online Validation*

Online Certificate Status Protocol (OCSP) is an online approach to determine revocation status of a certificate in request. The OCSP protocol defines the request and response messages exchanged in order to retrieve the revocation status of a certificate in request. An OCSP response shall be signed by a CA that signed the certificate in request or by a trusted responder (whose public key is trusted by the requestor) or an Authorized responder [7].

III. OPC UA Security Architecture

The OPC UA architecture provides an extensible framework with different protocol options. OPC UA is standardized in IEC 62541 with 13 sub specification of which the first 7 parts are core specifications like Concepts, Security Model [8], Address Space Model, Services [9], Information Model [10], Mappings [11] and Profiles. The other parts are access type specifications like Data Access, Alarm and Conditions, Programs, Historical Access and Aggregates, one part about the Discovery services. Recently, there was a publish/subscribe communication model standardised as Part 14: PubSub.

OPC UA describes its security architecture in a layered model as seen in figure 4 below. The establishment of connection between a OPC UA client and server is explained as follows,

1) The transport layer establishes the socket connection.
2) The communication layer with respect to this architecture enables application authentication. It ensures confidentiality and integrity through options for encryption and digital signatures, and use certificates. Asymmetric keys are used to secure the OpenSecureChannel messages. The secret exchanged using this SecureChannel is used to generate the symmetric keys used to secure further messages exchanged in the application layer.
3) The application layer provides options for user authentication and authorization in order to authenticate and provide access to requesting users with appropriate access rights.

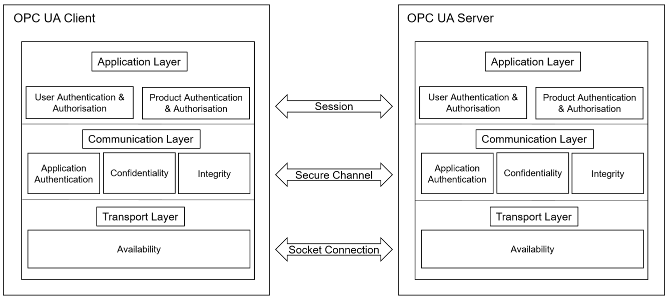


Figure 4. OPC UA Security Architecture [3]

OPC UA defines services [9] where security parameters are exchanged by service messages like OpenSecureChannel Requests/Responses. Security configurations are detailed in the following sections.

A. Security Configuration

In order to request a connection to a server, a client needs to be aware of session endpoints which can be determined using the discovery process by connecting to discovery endpoints of a server [9]. The session endpoints can also be manually configured for a client by an administrator, in this case, the discovery process is disabled. The security configuration of a server session endpoint has the following four parameters,

- Server Application Instance Certificate - X.509 Certificate that authenticates the server
- Message Security Mode - mode which specifies what security mechanism should be applied to the messages exchanged during a session [9]. Clients request to connect to a server in one of message security modes. There are three different modes and they are None, Sign and SignandEncrypt. Security is not applicable for 'None' mode.
- Security Policy - a set of security algorithms [11]
- Supported User Identity Tokens - Client can provide identity of a user to be authenticated using one of the identity token types supported by the server

B. ApplicationInstanceCertificate

The X.509 v3 fields of an ApplicationInstanceCertificate are shown in table I. This certificate is used to identify an instance of an application running on a single host. Table I shows fields of an X.509 v3 certificate and their respective precise descriptions [11].

Table I
APPLICATIONINSTANCECERTIFICATE [11]

Field Name	Description
version	shall be version 3
serialNumber	The serial number assigned by the issuer.
signatureAlgorithm	The algorithm used to sign the certificate.
signature	The signature created by the Issuer.
issuer	The distinguished name of the Certificate used to create the signature.
validity	When the certificate becomes valid and when it expires.
subject	The distinguished name of the Application Instance.
subjectAltName	Alternate names for the Application instance. [applicationUri, hostnames]
publicKey	The public key associated with the certificate.
keyUsage	Specifies how the certificate key may be used.
extendedKeyUsage	Specifies additional key uses for the certificate.
authorityKeyIdentifier	Provides more information about the key used to sign the certificate. It should be specified for self-signed certificates.

C. UserIdentityToken Types

There are four different user identity token types described in [9]. They are as follows,

1) AnonymousIdentityToken: No user identity information
2) UserNameIdentityToken: Username and password
3) X509IdentityToken: X.509 version 3 certificate and userTokenSignature, ex: a smart card
4) IssuedIdentityToken: WS- Security Token as user identity information, ex: Kerberos token

IV. OPC UA CERTIFICATE VALIDATION SERVICES

Figure 5 shows the different service messages exchanged between OPC UA based client and server applications. It shows the security related parameters exchanged for each message. The first two messages are the GetEndpoints request/response messages which are part of the discovery services. This is disabled if the client is configured manually as mentioned earlier. The client validates the server certificate before initiating an OpenSecureChannel request. The server validates the client certificate before sending the OpenSecureChannel response.

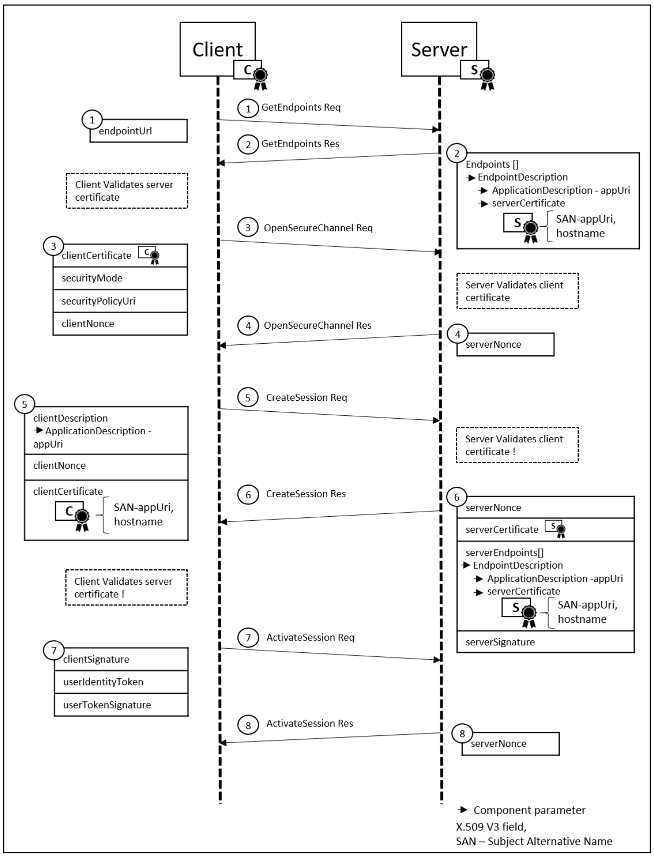


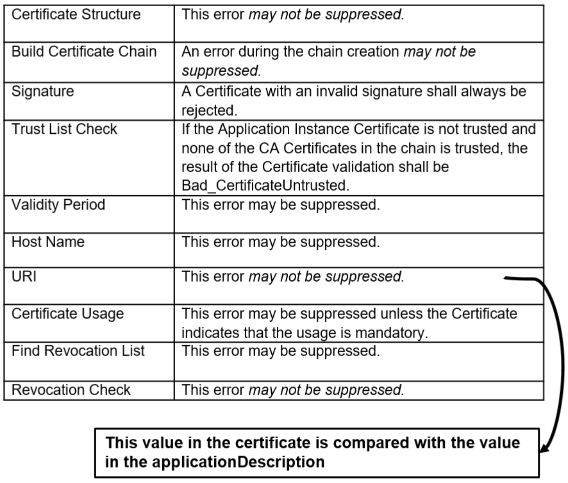
Figure 5. OPC UA - Connection Establishment

A. ApplicationInstanceCertificate Validation Steps

Figure 6 shows OPC UA application instance certificate validation steps according to [9]. The certificate structure is verified and followed by building the certificate chain depending on the trusted certificate found in the trust list. A Certificate Trust List (CTL) [9] is a list of certificates that are trusted by client/server applications. The certificate chain depends on the design of PKI implemented specific to the application environment. The certificate signature and validity period are verified. Hostname can also be verified against the value in subjectAltName field but an error may be suppressed. The application URI in the subjectAlternativeName field of certificate is validated against applicationUri in applicationDescription of the application. The certificate keyusage is checked. The revocation status of the certificate is determined. Figure 6 shows steps and describes if they are mandatory to be checked or may be suppressed [11]. Any failed validation returns respective StatusCode. A StatusCode is a built-in datatype which is a numeric identifier for a error or condition that is associated with a value or an operation with respect to OPC UA architecture [9].

B. User Authentication and Authorization

As mentioned is section III-C, there are 4 different options for user authentication in OPC UA including anonymous user.



Certificate Structure	This error *may not be suppressed.*
Build Certificate Chain	An error during the chain creation *may not be suppressed.*
Signature	A Certificate with an invalid signature shall always be rejected.
Trust List Check	If the Application Instance Certificate is not trusted and none of the CA Certificates in the chain is trusted, the result of the Certificate validation shall be Bad_CertificateUntrusted.
Validity Period	This error may be suppressed.
Host Name	This error may be suppressed.
URI	This error *may not be suppressed.*
Certificate Usage	This error may be suppressed unless the Certificate indicates that the usage is mandatory.
Find Revocation List	This error may be suppressed.
Revocation Check	This error *may not be suppressed.*

This value in the certificate is compared with the value in the applicationDescription

Figure 6. Certificate Validation Steps

Authorization describes different access level enabled for each authenticated entity.

1) Illustration OPC UA Node Management

The information model is provisioning of organized data to requesting client using the collection of Nodes within an OPC UA server AddressSpace. NodeClasses of type Variable and Method have mandatory attributes to describe access levels and user access levels. Figure 7 shows an example scenario where there are three different Nodes within a OPC UA address space. Each node has different access rights for different users. For example, Node 3 has read and write access for Admin A and only write access for the group X. When a client application tries to read data from Node 3, the access approved for Admin and denied for group X. The different access level attributes required for NodeClasses are briefed in the following section.

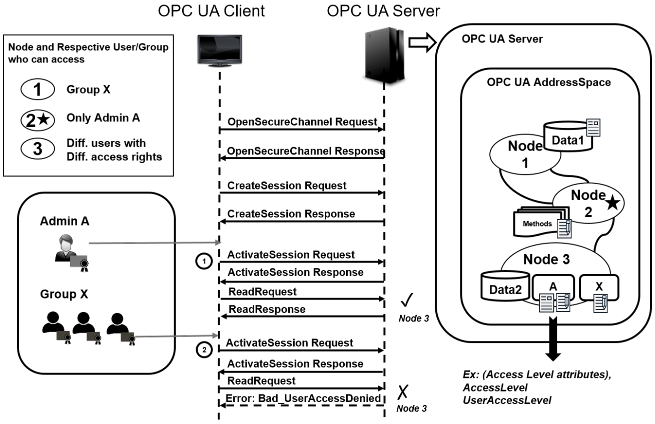


Figure 7. OPC UA Address Space - access level attributes

2) Access Level Attributes of NodeClasses

These mandatory attributes are to be set according to the required access levels determined by design. The different access level options supported by OPC UA NodeClass, access level attributes are defined in [12].

For Variable NodeClass, the following attributes are defined (attribute data type - Byte) [12],

- AccessLevel: This attribute is used to indicate how the value of a variable can be accessed
- UserAccessLevel: This attribute is used to indicate how the value of a variable can be accessed by considering a user's access level

For Method NodeClass, the following attributes are defined (attribute data type - Boolean) [12],

- Executable: This attribute determines if a method is executable
- UserExecutable: This attribute determines if a method is executable for the user trying to execute

V. Demo PKI

In order to illustrate a PKI for OPC UA based applications, an open source implementation of OPC UA, namely the Eclipse Milo project is used in this work [13]. The establishment of secure connection between client and server applications is enabled. Figure 8 shows the demonstrator described for this work. It shows the CA involved in issuing the certificates to the entities, online/offline validation involved in both sides and the usage of trust stores in the context of CTL. Figure 1 and Figure 3 shows an example scenario where CA_A is the trust anchor corresponding to company A and issuing the required intermediate CAs. Figure 8 is an example for one such intermediate CAs. The demo PKI can be extended to the trust model seen in figure 1 and 3.

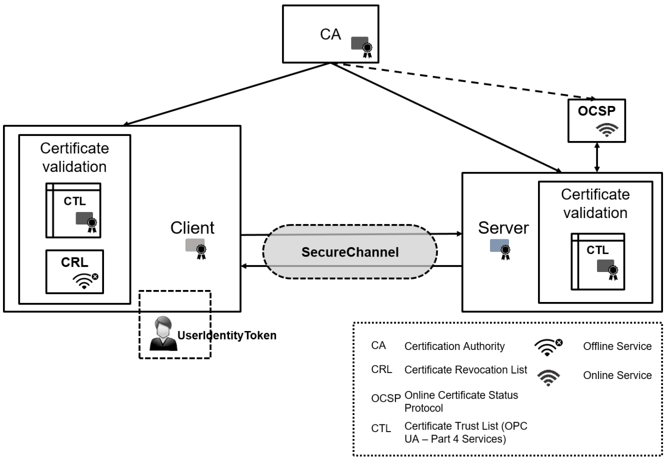

CA
OCSP
Certificate validation
CTL
CRL
Client
SecureChannel
Server
Certificate validation
CTL
UserIdentityToken
CA Certification Authority
CRL Certificate Revocation List
OCSP Online Certificate Status Protocol
CTL Certificate Trust List (OPC UA – Part 4 Services)
Offline Service
Online Service

Figure 8. Demonstration

A. Single CA Architecture

The trust relationship is enabled between the demo CA and end entities in this architecture [14]. The demo CA is a self signed certificate. This CA is also used for demonstration of the OCSP service which means the OCSP responses are signed using the demo CA's private key.

Figure 9 shows the demo CA and the different entities of the PKI illustrated. For the purpose of demonstration, the demo CA issues certificates to users, clients and servers. It also generates CRLs and OCSP responses in order to enable certificate validation.

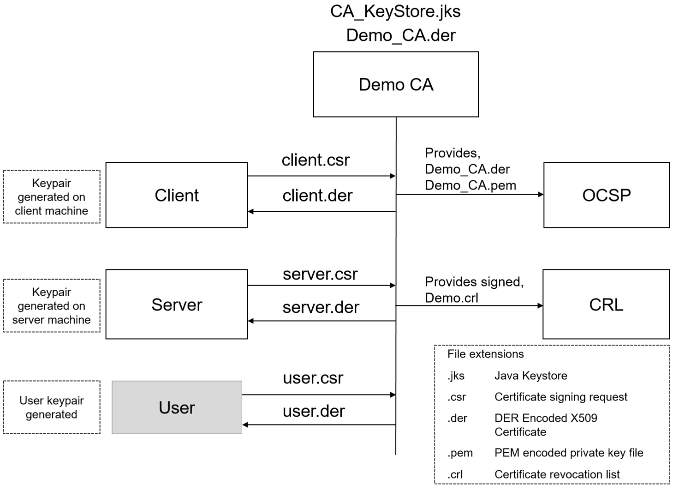

CA_KeyStore.jks
Demo_CA.der
Demo CA
Keypair generated on client machine
Client
client.csr
client.der
Provides, Demo_CA.der Demo_CA.pem
OCSP
Keypair generated on server machine
Server
server.csr
server.der
Provides signed, Demo.crl
CRL
User keypair generated
User
user.csr
user.der
File extensions
.jks Java Keystore
.csr Certificate signing request
.der DER Encoded X509 Certificate
.pem PEM encoded private key file
.crl Certificate revocation list

Figure 9. Demo CA and End Entities

The open source project OpenJDK [15], [16] provides a keytool utility used to generate keystores, keypairs, certificates and CRLs. The keytool utility of the OpenJDK project needs to be updated with the extensions, which enables generation of required certificates used by OPC UA based applications as mentioned in table I.

B. User Authentication X509IdentityToken

The usage of X509IdentityToken for user authentication is illustrated using the demo PKI. When X509IdentityToken is to be used for authentication then the proof is a signature (userTokenSignature [9]) generated with the private key associated with the certificate. The data to sign is created by appending the last serverNonce to serverCertificate returned in a CreateSession response [9]. Thus the identity token type is not applicable for 'None' mode. Authorization is enabled using different access level attributes. Here, access level attributes are demonstrated for few users in order to illustrate the differentiation of access rights. For example, a list of serial numbers in combination with issuer names to identify users assigned to certain roles like 'admin' can be configured for OPC UA servers, where each role is configured with certain access rights and so their access levels are differentiated. A database can also be used to implement such an approach. Further research is required to implement users/ user groups on a suitable scale. Storage of information (like certificates) using LDAP (Lightweight Directory Access Protocol) [17] servers or similar methods need to be investigated for a feasible approach.

C. Certificate Validation: Online/Offline

The mandatory certificate validation steps are implemented according to the steps described in IV-A. The certificate to be validated is checked if it was revoked. The revocation status is determined offline by client and online by server in this demonstration in order to illustrate the use of both methods. The advantages and challenges of these methods are discussed in the next section.

1) Offline Validation

The offline validation of certificates is implemented using CRLs. The demo CA issues the CRL and OPC UA client applications use it in order to validate server certificates. Here, it is checked if the serial number of a certificate in question is not present in the CRL. The CRL is stored locally for the demo PKI. A CRL distribution point extension can also be added to the certificate in order to determine URL location of CRLs which can be used to download CRLs in regular intervals.

2) Online Validation

The OCSP implementation is adopted from Slonvpalto's implementation [18]. The certificate revocation status has to be configured for the certificate in question. The standalone OCSP server runs on a PC using a simple TCP connection. The connection URL for the OCSP server is specified in the Authority Information Access extension of the client certificate. The OCSP request message is generated by a OPC UA server for validation of the client certificate. The OCSP server generates the corresponding response and signs it with private key of the demo CA. On receiving the response, the server determines revocation status of the certificate in question.

D. Implementation

The PKI was designed as stated in the above section. The client/server applications and OCSP server are running in three different PCs. The certificates, keystores and CRL files were manually provisioned for client and server applications. The demo CA certificate is configured to be the trusted certificate and placed in so called trusted folder of the example applications. The session endpoint for client to connect to server is also configured and so discovery process is disabled. The demo CA's certificate and private key was provided to the OCSP server which has certificate revocation status. A keypair and a X.509 certificate is generated for the user and used for user authentication. The user certificate is also signed and generated by the demo CA. While both OPC UA server and OCSP server were running, OPC UA client application was tested to check for the secure connection establishment. For this demonstration, connection establishment fails if the OCSP server is offline. It can also be configured to use offline method in case of unavailable online service. Thus, a PKI was implemented and tested to establish secure end-to-end communication between OPC UA based client/server applications. For example different smart cards can be used to realize authorization facilitated by access level attributes.

The following points need to be considered for this demo PKI,

1) The IP address in the application URI of the application description and the subjectAltName extension of the application instance certificates are to be same. This enables the validation of URI as seen in figure 6
2) Certificates and keystores need to be provisioned appropriately before testing

VI. Advantages and Challenges

There are different advantages and challenges in establishing a security infrastructure with a PKI. Though a single CA architecture is detailed, one of the advantages of a PKI is when a CA in an intermediary level is compromised then it does not affect the trust ensured by other CAs in trust model. One of the challenges is to manage the distribution of certificates or CRLs to entities. Some of the considerations before designing a PKI are as follows:

1) Implementation challenges in creating and maintaining a certificate infrastructure, like challenges in online and offline validation
2) Feasibility of different options for secure identities like TPM (Trusted Platform Module), smart cards etc.
3) Requirements of the communication links like determining the security level required for each link between the entities

Table II shows different advantages and limitations of online and offline certificate validation services.

Table II
Advantages and Limitations: CRL, OCSP

CRL	OCSP
Advantages	
The information is available offline for validation	Since the information is available online, local storage is not required
Delta CRLs [6] are available for updated information between this update and the next update	The current revocation status information is available
Limitations	
Space required in local caching of the CRLs when the list is large.	The OCSP service is provided online and hence when there are many requests to be processed, there is a huge over load on the OCSP responding server.
The time when a certificate can be revoked is limited, for example, if a revocation is reported now, that revocation will not be reliably noticed to the certificate using systems until all currently issued CRLs are scheduled to be updated which depends on the frequency that CRLs are issued.	Continuous availability of OCSP responder is a consideration.

In addition to ensuring confidentiality and integrity, authorization of entities to access data or to restrict access needs to be enabled. The access level attributes provided in OPC UA architecture can be used to differentiate the access rights between different users or user groups. However, a feasible solution is required to manage a considerable scale of users. The further scope for research is discussed briefly in the next section.

VII. Conclusion

This work emphasizes the necessity of a security infrastructure to enable end-to-end communication for distributed systems. PKI and its entities are described. One of the widely used communication frameworks, OPC UA is detailed. Its security architecture is studied in order to understand the usage of a PKI to enable secure connection establishment between client and server applications. A working example of the PKI is described. Hence, the minimalistic implementation of the demo PKI serves as proof for further extension of the architecture as seen in figure 1. The advantages and limitations vary for every PKI design depending on how they are implemented. Thus the proposed work ensures secure end-to-end communication for OPC UA based applications.

There is scope for different areas of research. The distribution of X.509 certificates and CRLs to the devices where life cycle management of certificates is crucial for such an infrastructure. Redundancy of data required, availability of services (online/offline), feasible maintenance in case of distributed systems are to be studied in detail. The Discovery services of OPC UA described in [19] are facilitating certificate management using GlobalDiscoveryServer (GDS). A study on GDS and its feasibility for security infrastructure needs to be investigated.

RBAC [20], [21] is to be studied in order to understand its applicability with Attribute Certificates (AC) [22] for authorization services. The usage of ACs with X.509 certificates in order to provide authentication and authorization services can be one of the good options while creating a security infrastructure. Roles specification described in [23] is one of the latest updates from OPC Foundation to highlight its support for role based authorization. The OPC UA address space model describes a Base NodeClass from which all other NodeClasses are derived. While comparing the possible attributes (which means both the mandatory and optional attributes) of a Base NodeClass defined in table 2 in [12] with table 7 in [23], additional optional attributes are seen to support roles and access restrictions. These attributes are now optional for any derived Nodeclass. Its advantages and challenges lies in the clarity of implementation. The different approaches in enabling RBAC model and its support for X.509 certificate infrastructure needs to be investigated.

Acknowledgment

The work presented in this paper is a part of the project IT SIVA (Security infrastructure for distributed automation systems) funded by the Federal Ministry for Economic Affairs and Energy (BMWI, IGF grant 19117 N/2).

References

[1] Federal Ministry for Economic Affairs and Energy BMWi, “Technical Overview: Secure cross-company communication”, Apr. 2016. [Online]. Available: http://www.plattform-i40.de/I40/Navigation/EN/InPractice/Online-Library/online-library.html.

[2] Federal Ministry for Economic Affairs and Energy (BMWi), “Technical Overview: Secure Identities”, Apr. 2016. [Online]. Available: http://www.plattform-i40.de/I40/Navigation/EN/InPractice/Online-Library/online-library.html.

[3] Wolfgang Mahnke, Stefan-Helmut Leitner, Matthias Damm, “OPC Unified Architecture”, 2009.

[4] A. Fernbach, W. Kastner, “Certificate Management in OPC UA Applications: An Evaluation of different Trust Models”, Aug. 2012.

[5] M. Schleipen, E. Selyansky, R. Henssen, T. Bischoff, “Multi-level user and role concept for a secure plug-and-work based on OPC UA and AutomationML”, Aug. 2015.

[6] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, W. Polk, “Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. Request for Comments 5280.”, May 2008. [Online]. Available: https://www.ietf.org/rfc/rfc5280.txt.

[7] S. Santesson, M. Myers, R. Ankney, A. Malpani, S. Galperin, C. Adams, “X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP. Request for Comments 6960”, Jun. 2013. [Online]. Available: https://tools.ietf.org/html/rfc6960.

[8] DIN IEC 62541-2, “OPC Unified Architecture Specification - Part 2: Security Model”, Dec. 2008.

[9] DIN IEC 62541-4, “OPC Unified Architecture - Part 4: Services”, Nov. 2015.

[10] DIN IEC 62541-5, “OPC Unified Architecture - Part 5: Information Model”, Nov. 2015.

[11] DIN IEC 62541-6, “OPC Unified Architecture - Part 6: Mappings”, Nov. 2015.

[12] DIN IEC 62541-3, “OPC Unified Architecture - Part 3: Address Space Model”, Nov. 2015.

[13] Kevin Herron, “Eclipse Milo - an open source implementation of OPC UA (IEC 62541) ”, Last accessed: 28.07.2017. [Online]. Available: https://github.com/eclipse/milo.

[14] M. Shimaoka, N. Hastings, R. Nielsen, “Memorandum for Multi-Domain Public Key Infrastructure Interoperability. Request for Comments 4511”, Jul. 2008. [Online]. Available: https://tools.ietf.org/html/rfc5217.

[15] “OpenJDK source files ”, Last accessed: 16.12.2016. [Online]. Available: http://hg.openjdk.java.net/jdk8/jdk8/jdk/.

[16] “Keytool Documentation for Command and Options”, Last accessed: 16.12.2016. [Online]. Available: https://docs.oracle.com/javase/8/docs/technotes/tools/windows/keytool.html.

[17] J. Sermersheim, “Lightweight Directory Access Protocol (LDAP): The Protocol. Request for Comments 4511.”, Jun. 2006. [Online]. Available: https://tools.ietf.org/html/rfc4511.

[18] “OCSP service - Blog post”, Last accessed: 16.12.2016. [Online]. Available: http://olegpekar.blogspot.de/2014/03/implementing-ocsp-responder.html.

[19] DIN IEC 62541-12, “OPC Unified Architecture - Part 12: Discovery”, Jul. 2015.

[20] David F.Ferraiolo, D. Richard Kuhn, Ramaswamy Chandrmouli, “Role-Based Access Control”, 2007.

[21] D. Chadwick, A. Otenko, E. Ball, “Role-based access control with X.509 attribute certificates”, *IEEE Internet Computing*, vol. 7, pp. 62–69, 2003.

[22] Farrell, S., Housley, R. u. Turner, S., “An Internet Attribute Certificate Profile for Authorization. Request for Comments 5755.”, Jan. 2010. [Online]. Available: http://www.ietf.org/rfc/rfc5755.txt.

[23] OPC Foundation, “OPC Unified Architecture Specification-Part 3: Address Space Model”, Nov. 2017.