Comparing the Security Approaches of CIP and OPC UA

Alex Gebhard
Marquette University
Rockwell Automation
Milwaukee, WI, USA
alexander.gebhard@marquette.edu

Debbie Perouli
Marquette University
Milwaukee, WI, USA
despoina.perouli@marquette.edu

Abstract

As Operational Technology (OT) protocols integrate more closely with conventional Information Technology (IT) networks, they have faced heightened scrutiny regarding their security posture. Most OT protocols were conceived decades ago, often without consideration for security, thereby leaving devices susceptible to exploitation by malicious actors. This work examines two prevalent industrial protocols, CIP and OPC UA, through a concise historical overview, an analysis of their respective security architectures, and a discourse on the challenges and prospective avenues for researchers in fortifying industrial protocols' security. We compare and contrast each protocols' approach to redesign itself with security in mind.

CCS Concepts

• **Security and privacy** → **Security protocols**; • **Applied computing** → *Engineering*.

Keywords

ICS; OT; EtherNet/IP; CIP; security; industrial; protocol; networking

ACM Reference Format:
Alex Gebhard and Debbie Perouli. 2024. Comparing the Security Approaches of CIP and OPC UA. In *Proceedings of the 2024 Workshop on Re-design Industrial Control Systems with Security (RICSS '24), October 14–18, 2024, Salt Lake City, UT, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3689930.3695206

1 Introduction

The amalgamation of IT with OT networks has unveiled previously insecure OT protocols to the broader business network, protocols historically devoid of robust security mechanisms. Security measures are often retroactively incorporated as reactive measures after the widespread adoption of these protocols by thousands or even hundreds of thousands of devices. Consequently, many devices find themselves exposed, deliberately or inadvertently, to the Internet, allowing malicious actors worldwide to exploit these vulnerable industrial devices. Attackers have used these insecure protocols to carry out harmful attacks against defenseless operators.

Dragos and ESET reported in 2017 about a new ICS malware called CRASHOVERRIDE/Industroyer, which targeted electrical grids [5, 10]. The malware had modules to read and write messages using four protocols commonly used in electrical grids: IEC 101, IEC 104, IEC 61850, and OPC. The attackers appear to have been targeting circuit breakers to disrupt power transmission. CRASHOVERRIDE was used to disrupt power in Ukraine on December 17th, 2016. It is unknown if it affected other facilities elsewhere. This past year, Dragos has reported on a new malware called FrostyGoop [6]. Like CRASHOVERRIDE/Industroyer, FrostyGoop targets devices on the electrical grid. FrostyGoop uses the Modbus protocol over TCP to read and modify data on ICS devices. The Security Service of Ukraine and Dragos have identified FrostyGoop being used to target a municipal energy company in Ukraine.

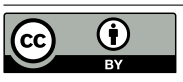
All protocols used in these attacks lack security mechanisms commonly found in traditional IT protocols. This paper analyzes the security aspects of two popular industrial protocols: CIP (occasionally referred to as the Ethernet-specific implementation of CIP, EtherNet/IP) and OPC UA. OT devices such as Rockwell Automation's ControlLogix 5580 series and Schneider Electric's Modicon M262 implement both CIP and OPC UA. We compare and contrast the approaches taken by both protocols to achieve security and discuss their relative strengths.

Our study makes the following contributions: i) we thoroughly explore the CIP and OPC UA protocols; ii) we systematically analyze CIP and OPC UA on their security properties; iii) we compare and contrast the approaches CIP and OPC UA have taken in terms of security; iv) we research the current adoption of CIP Security; v) we present future challenges and new research directions for those investigating OPC UA and CIP Security.

2 Industrial Protocol Security Objectives

In their core objectives, industrial protocols diverge from IT network protocols such as HTTP, DNS, and SSH. While IT network protocols primarily underscore the assurance of data confidentiality and integrity, industrial protocols focuses on operational safety due to their direct control over physical processes. This leads industrial protocols to focus more on integrity, authentication, and availability – ensuring data is not maliciously modified. The critical distinction arises from the potential consequences of unauthorized data manipulation within industrial environments, which can precipitate hazardous scenarios leading to physical harm, infrastructure damage, or even fatalities.

Moreover, industrial equipment emphasize system availability, recognizing that downtime may engender significant financial ramifications. This prioritization necessitates a departure from conventional IT practices, where frequent equipment updates mitigate emerging vulnerabilities. However, such a strategy may be infeasible for mission-critical OT systems tasked with continuous operation. Consequently, mission critical system operators may choose to prioritize system uptime and rely on aging hardware



RICSS '24, October 14–18, 2024, Salt Lake City, UT, USA

ACM ISBN 979-8-4007-1226-5/24/10
https://doi.org/10.1145/3689930.3695206

configurations vulnerable to known exploits, enduring operational lifespans spanning years, if not decades.

Automation devices often exhibit constrained computational resources, including limited RAM, CPU capacity, and storage capabilities. Consequently, even rudimentary cryptographic tasks, such as Advanced Encryption Standard (AES), can incur significant time overheads, rendering them impractical for real-time industrial applications. Manufacturers could revise their automation devices to handle modern cryptographic operations. However, this leads to increased costs and may be impractical due to market constraints. This leads some manufacturers to devise specialized, insecure protocols, exemplified by protocols such as the one referenced in [1], tailored to the unique requirements of industrial environments.

3 Related Work

Researchers have previously studied other industrial protocols to analyze the security properties they provide. Dzung *et al.* [7] provide requirements outlining requirements industrial protocols must have to be secure. The authors list the security properties that a protocol should have, common attack scenarios, and possible mitigations. However, the paper does not examine these security properties in light of industrial protocols in use today.

Volkova *et al.* [39] presents the security properties of six industrial protocols: Modbus, OPC UA, TASE.2, DNP3, IEC 60870-5-101, IEC 60870-5-104, and IEC 61850. The authors give a background to each protocol and synthesize how these protocols can protect (or not) against various attacks or threat actors. Similarly, Li, Wu, and Pan [15] examine industrial protocols such as Profinet, DNP3, Modbus, IEC 60870-5104, IEC 61850, IEC 61400-25, and IEEE C37.118 and the security properties that these protocols provide. The authors also propose countermeasures as a defense-in-depth approach.

Schwarz and Borcsok [37] conduct a review of the OPC and OPC UA protocol. The paper compares how the two protocols differ and discusses the new security features in OPC UA as of 2013. Since then security in OPC UA has changed, as detailed in this paper.

4 Protocol Introduction

This section will discuss the history and architecture of the two major industrial protocols, CIP and OPC UA.

4.1 CIP

CIP™, or the Common Industrial Protocol, is maintained by ODVA [20] through a collective of its members. CIP is a transport-independent protocol. It can run through Ethernet (called EtherNet/IP™), Controller Area Network media layer (called DeviceNet™), or via RG-6 coaxial cable (called ControlNet™). CIP is comprised of a wide variety of messages and services to provide functionality to any industrial system, including messages such as control, I/O, security, motion, energy, information, and network management.

The CIP protocol can be considered an "object-oriented" program. CIP defines an *object* as an abstract part or component of a system. For example, some everyday CIP objects are the File Object, the Identity Object, or the Connection Object. Each object has instances, which are the real representation of the object.

These objects have attributes that hold data values and services that take parameters, perform some action, and return a result. See Figure 1 for a conceptual representation of CIP objects, classes, and attributes. For example, files on a CIP device are represented by File Object instances. These File Object instances have attributes such as file name, type, checksum, and permissions. The File Object instances support Save, Create, and Delete services. Every instance of a particular object has its own ID to manage object instances. The CIP specification Volume 1 Chapter 1 [21] and associated public documentation [36] details objects in CIP.

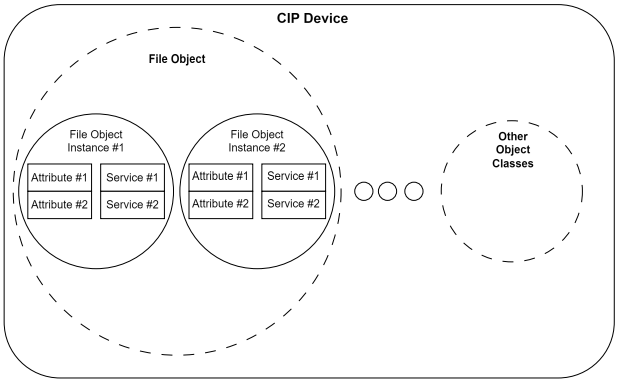


Figure 1: A conceptual example of a CIP Device

4.1.1 Specification. ODVA breaks the CIP specification into nine different volumes. Of particular interest to security researchers are volume one (introduction to CIP), volume two (EtherNet/IP or CIP over Ethernet) and volume eight (CIP Security). These volumes give an overview of the CIP, various message formats for EtherNet/IP, and CIP Security changes. The list of volumes and their topics are below:

(1) Volume 1: Common Industry Protocol (CIP)
(2) Volume 2: EtherNet/IP Adaptation of CIP
(3) Volume 3: DeviceNet Adaptation of CIP
(4) Volume 4: ControlNet Adaptation of CIP
(5) Volume 5: CIP Safety
(6) Volume 6: CompoNet Adaptation of CIP
(7) Volume 7: Integration of Modbus/HART/IO-Link devices into CIP
(8) Volume 8: CIP Security
(9) Volume 9: CIP Motion

Members of ODVA are invited to contribute to the specifications. The specifications themselves are technically open, however they require a fee to access[1].

4.1.2 Connection Types. In practice, EtherNet/IP implementations tend to use TCP port 44818 to achieve insecure, reliable data transmission on the IP layer of unconnected messages. Insecure UDP communication is generally used for messages over port 2222. Port 2221 is used for secure UDP/TCP transmission. CIP supports a UDP-only profile for resource constrained devices. CIP also allows I/O devices to configure the level of reliable data transmission.

[1]The authors have access to the specification for this research.

4.2 OPC UA

Open Platform Communication Unified Architecture (OPC UA) is an open-source, cross-platform protocol that allows devices to exchange data and commands. According to the OPC Foundation, OPC UA was released initially in 2004 [29]. Before OPC UA, vendors were using proprietary communication protocols over Microsoft DCOM. Automation vendors Fisher-Rosemount, Intellution, Opto 22, and Rockwell Software recognized the need for standards and interoperability among their products. The group formed the OPC Foundation on April 22, 1996. The OPC Foundation released the original OPC standard for communicating among Microsoft COM and DCOM, now called OPC Classic.

OPC Classic has several drawbacks that became quickly apparent. First, it used the Distributed Component Object Model (DCOM), a Microsoft proprietary protocol, to transmit data. This made it highly dependent on the Microsoft Windows operating system. Since embedded devices usually use real-time operating systems (RTOS) – or Linux – as their underlying OS, this prevented them from communicating with OPC servers. At that time, Microsoft DCOM lacked necessary security protections such as process authentication [18] that could allow any user to talk to the device on the other end. With these considerations, the OPC Foundation formed working groups to discuss a new protocol in November 2003. The specification for OPC UA was released in 2004.

4.2.1 Specification. The OPC Foundation breaks the OPC UA specification into twenty-four parts, with other documents for specific industries (such as mining, heavy machinery, industrial automation, etc.). Of particular interest to security researchers are the following parts:

- Part 1: Overview and Concepts
- Part 2: Security
- Part 3: Address Space Model
- Part 4: Services
- Part 5: Information Model
- Part 6: Mappings
- Part 7: Profiles
- Part 12: Discovery and Global Services
- Part 18: Role-Based Security
- Part 21: Device Onboarding

These specifications cover the basis of OPC UA and the security features the protocol can provide.

4.2.2 Connection Types. Like CIP, OPC UA is also designed to be independent of the underlying protocol. As specified in Part 1 Section 5.3 [23], OPC UA can run over raw TCP, HTTPS, or WebSockets. It can be encoded in raw binary, XML, or JSON. OPC UA is usually run over TCP encoded in raw binary. Conceptually, OPC UA has a similar structure to ODVA's CIP protocol. OPC UA aims to give a standard view to client applications.

The OPC Foundation defines different "node" classes or types to accomplish this task. Parts 1 & 3 of the OPC UA specification [11, 23] define nodes and their types. The primary node classes are shown in Table 1. The nodes are grouped into namespaces. Every node is given a NodeID and a namespace index. This separates it from other nodes. Namespaces are given a URI and an index (ex `nsu=http://test.org/UA/Data/;i=10853`). This allows a client to query the namespace using a URI with the NodeID it wishes to access. All namespaces belong to the global AddressSpace, encompassing all OPC UA application nodes. See Figure 2 for a conceptual representation.

Type	Description
`Object`	Similar to CIP. Objects that have methods, attributes, and references to other objects.
`ObjectType`	Provides a structural definition of an Object
`ReferenceType`	Defines a reference between two nodes
`Variable`	Holds a value
`VariableType`	Provides a structural definition for a Variable
`DataType`	Defines simple and complex data types of Variable values
`Method`	A lightweight function

Table 1: Types of NodeClasses

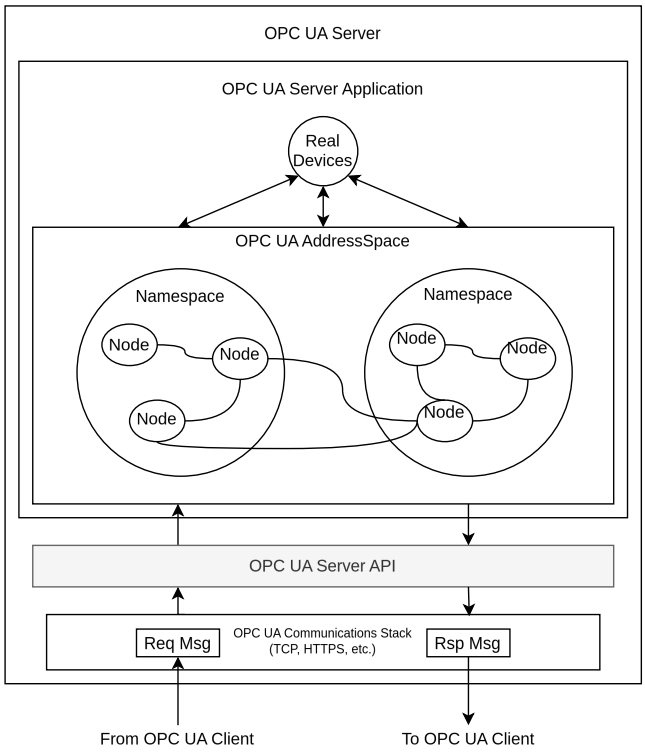


Figure 2: A conceptual representation of the OPC UA Server and AddressSpace

Members of the OPC Foundation are invited to participate in working groups that revise and write the specifications. The specifications themselves are public, unlike CIP, and do not require a fee to access. Specifications can be found here [30].

5 CIP Security Protocol Overview

Initially released in 2015, CIP Security is an *optional* extension to the CIP protocol that provides security properties to the CIP protocol. CIP Security is specified in Volume 8 of the CIP specification [22]. CIP Security uses the following open technologies to provide integrity, confidentiality, authentication, and authorization.

(1) X.509v3 certificates to establish the identity of users and devices
(2) TLS (Transport Layer Security) and Datagram Transport Layer Security (DTLS) cryptography protocols to provide communications integrity and confidentiality
(3) Hashes or HMAC to provide data integrity and message authentication to EtherNet/IP traffic
(4) Proven and standardized encryption algorithms
(5) OAuth2.0 and OpenID Connect for user authentication with JSON Web Tokens (JWT)

CIP Security adds seven new objects, each containing attributes and services to configure and support CIP Security. These are specified in Volume 8 Section 5 [22]. The list of these objects and a description can be found in Table 2.

Object Name	Description
`CIPSecurity`	Provides services to allow exclusive access to modify security configuration
`EtherNetIPSecurity`	Provides attributes and services to modify (D)TLS settings
`CertificateManagement`	Provides an interface to manage a device's X.509v3 certificate
`Authority`	Provides services and attributes to authenticate against an external Authority (See section 6.2)
`PasswordAuthenticator`	Provides services and attributes to manage a local database of users and passwords
`CertificateAuthenticator`	Provides services and attributes to manage a local database of users and their X.509v3 certificates
`IngressEgress`	Provides services and attributes for firewall configuration

Table 2: Objects CIP Security adds to a device

5.1 Confidentiality, Endpoint Authentication, and Integrity

To provide confidentiality, endpoint authentication, and integrity to the CIP Protocol, CIP Security-enabled devices use DTLS/TLS 1.2. TLS 1.2 and DTLS 1.2 are specified in RFC 5246 and RFC 6347 respectively [34, 35]. TLS sends TCP messages, whereas DTLS is used for UDP transport. Together, these cryptographic protocols reside at the transport layer. Thus, it does not change the format of any CIP messages. CIP Security allows device owners to restrict what devices can communicate with the target device via (D)TLS Authentication, which can be done with X.509v3 certificates or Pre-Shared Keys (PSK). A device that does not contain a certificate from a trusted Certificate Authority (CA) or know the PSK can be blocked from communicating with the target device. Devices are required to support both authentication methods as specified in Volume 8 Section 3-4.1 [22]. ODVA requires devices to support at least the following cipher suites for certificate authentication (see [17] for public documentation):

(1) `ECDHE_ECDSA_WITH_NULL_SHA`
(2) `ECDHE_ECDSA_WITH_AES_128_CBC_SHA256`
(3) `ECDHE_ECDSA_WITH_AES_256_CBC_SHA384`
(4) `ECDHE_ECDSA_WITH_AES_128_GCM_SHA256`
(5) `ECDHE_ECDSA_WITH_AES_256_GCM_SHA384`
(6) `ECDHE_ECDSA_CHA20_POLY1305_SHA256`

The following are required for PSK authentication (Volume 8 Section 3-5.5.3):

(1) `ECDHE_PSK_WITH_NULL_SHA256`
(2) `ECDHE_PSK_WITH_AES_128_CBC_SHA256`
(3) `ECDHE_PSK_WITH_AES_128_GCM_SHA256`
(4) `ECDHE_PSK_WITH_CHA20_POLY1305_SHA256`

Vendors can add more cipher suites as they choose; however, the ones listed are required. Notice that CIP Security provides options for users to select cipher suites that provide authentication and integrity but not confidentiality. This is intentional, as confidentiality may be undesirable for several reasons (slower performance, blind to intrusion detection systems (IDS), etc.).

5.2 User Authentication & Authorization

Traditional CIP, without CIP Security, does not have the concept of users. Any client CIP devices are automatically trusted and allowed to perform any CIP command on the target device itself. CIP Security adds a new profile device that can optionally support the CIP User Authentication Profile (specified in Volume 8 Section 4-2 and public documentation [38]). The User Authentication Profile runs at the top of the CIP network stack. The CIP User Authentication profile supports two different authentication types: local authentication and central authentication. To determine which authentication method to use, the originator will query the target's Authority Object to identify the device against which to authenticate.

5.2.1 Local Authentication. With local authentication, the target hosts the authority (the service that provides authentication and authorization services) and does not rely on an external server. Only passwords or X.509v3 certificates are supported. If the device supports local authentication, the Password Authenticator Object and the Certificate Authenticator Object are available to validate users locally. As the environment grows, syncing the local authentication database across many devices becomes increasingly challenging. For this reason, the CIP User Authentication Profile supports using a centralized authentication mechanism.

5.2.2 Central Authentication. Central authentication relies on an external provider to validate user credentials and authorization. When the CIP device is manufactured, it is initialized with the public certificate of the authority that provides authentication. Central authentication uses OAuth2.0 to communicate with the authentication server. An open standard, like OAuth2.0, allows users to configure *any* service that supports OAuth2.0. This gives users a tremendous amount of flexibility with their security. For example, a user can use biometric two-factor authentication against Active Directory. The CIP device does not need to implement various authentication types as it uses OAuth2.0.

Once the CIP originator receives a response from the target's Authority object, it knows the server to authenticate against it. Then, the CIP originator authenticates against the central authority using the preferred authentication mechanism of the authority. Once authentication is successful, the authority will return a JSON Web Token (JWT) signed by the authority's private key. A JWT is a standardized (RFC 7519 [14]) set of properties (or claims) encoded in a JSON object. A JWT is then signed by an authority to prove its authenticity. JWTs returned by the authority must implement specified claims (see Appendix Table 4).

5.3 Provisioning

In order to allow users to configure their devices, CIP security-enabled devices ship with a self-signed or manufacturer signed certificate. The device will not verify certificates or check the Subject Alternative Name (SAN) sent from the user. Once the user sets up the device's security properties, then the device will switch to the user-provided certificates/pre-shared keys. ODVA requires all provisioning certificates to be unique to prevent vendor certificate reuse. Alternatively, CIP Security supports Enrollment over Secure Transport (EST) as specified in RFC 7030 [33]. This allows CIP Security devices to automatically provision themselves via a DNS-SD server to download certificates and configuration details securely.

6 OPC UA Protocol Security Overview

The following section describes the security features of the OPC UA protocol. This section assumes that the client and server communicate using OPC UA binary format over raw TCP. Other transport protocols, like HTTPS or WebSockets, have security properties outside the OPC UA specification.

6.1 Confidentiality, Endpoint Authentication, and Integrity

OPC UA supports three different security modes for transport security: `None`, `Sign`, and `SignAndEncrypt` [27]. `None` does not provide confidentiality or integrity to messages sent within the communications session. `Sign` provides integrity and authentication by adding an HMAC to the end of each message. `SignAndEncrypt` provides confidentiality and integrity by using HMACs and encryption to prevent messages from being read or modified in transit. Giving the user an option for confidentiality is similar to CIP Security, as not all environments need encryption while messages are in transit.

Unlike CIP Security, OPC UA does not rely on TLS/DTLS to provide communication security (OPC UA does use TLS for the PubSub architecture, but that is outside this paper's scope). OPC UA gives vendors some leeway to implement various encryption schemes that OPC UA defines. OPC UA Part 7 [25] defines different "profiles" or features that device vendors can implement. A minimum set of profiles must be implemented depending on the device (i.e., client/server/publisher/subscriber) and the device's capabilities.

According to the Core 2022 Server Facet Profile [26], servers are only required to implement **one** security mode (including the `None` Security Mode). This leaves the choice open to developers. This often leads developers only to support `None` (see Section 7.1 Adoption). Additionally, developers that support Security Modes `Sign` or `SignAndEncrypt` can choose which SecurityPolicies to use for signing and encrypting messages. OPC UA Part 7 specifies the following SecurityPolicy profiles servers and clients could support:

- Aes128_Sha256_RsaOaep
- Aes256_Sha256_RsaPss
- Basic128Rsa15 (deprecated)
- Basic256 (deprecated)
- Basic256Sha256

Additionally, the OPC Foundation released v1.04 of the OPC UA specification in 2022, which added support for the following elliptic curve cryptography (ECC) algorithms:

- ECC_curve25519_ChaCha20Poly1305
- ECC_curve448_ChaCha20Poly1305
- ECC_nistP256
- ECC_nistP384
- ECC_brainpoolP256r1
- ECC_brainpoolP384r1
- ECC_curve448 (deprecated)
- ECC_curve25519 (deprecated)

ECC algorithms do not require large key sizes like RSA, which usually requires 2048-4096 bits to be secure – for industrial devices, having smaller key sizes results in a smaller memory footprint for resource-constrained devices. However, ECC is more computationally complex than RSA which requires more CPU resources.

There is active research into using various cryptographic algorithms to decrease the cost of performing encryption/decryption using RSA. Lou and Zhang [16] propose a combination of RSA and BlowFish (a stream cipher created in 1993) to improve RSA's encryption and decryption time. The authors found that using BlowFish along with RSA improved the encryption and decryption time by a factor of three compared to only using RSA. Paul and Guerin [32] examined four post-quantum secure protocols (NewHope512, NTRU-HRSS, Kyber512-90s, and LightSaber) in OPC UA's key exchange. They found that integrating quantum-resistant key exchanges is generally feasible even in resource-constrained settings. Finally, Wu *et al.* experimented with an implementation using the WHIRLPOOL hash function to sign messages [40]. The authors found that WHIRLPOOL reduces the average message delay compared to SecurityPolicies using SHA-1.

6.1.1 Secure Session Establishment. A high-level overview of establishing a secure session in OPC UA is shown in Figure 3. The secure channel establishment is specified in Part 6 [28].

1. GetEndpoints To establish a session with an OPC UA server, the client will send a packet calling the `GetEndpoints` service. The `GetEndpoints` service will return the security information

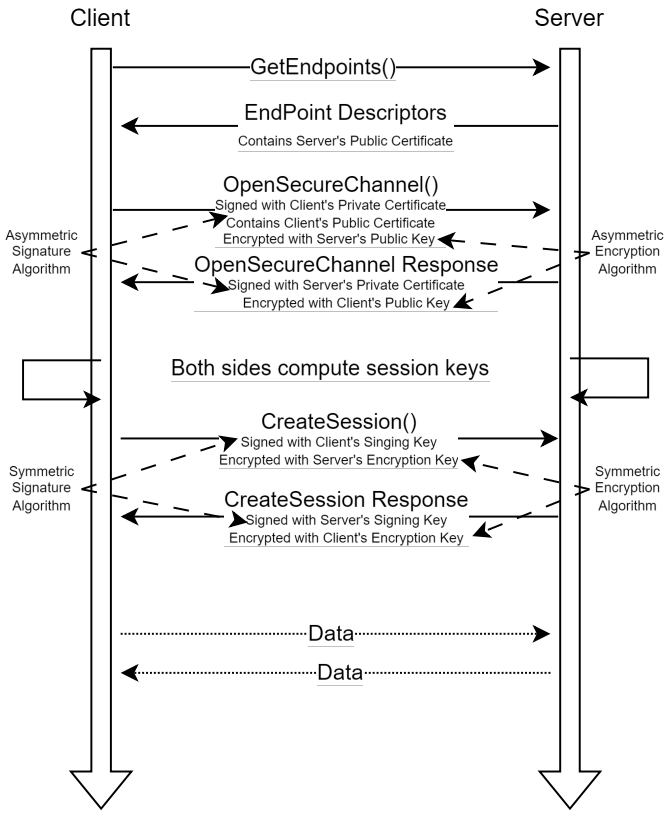


Figure 3: Initial handshake for establishing a secure OPC UA connection

needed to establish a secure connection server: the SecurityMode, the server's preferred SecurityPolicy, and the server's public certificate. The server can return multiple endpoint descriptors if it supports multiple SecurityPolicies and SecurityMethods. The client will use the information from the server to determine if it can securely connect.

2a. OpenSecureChannel Once the client has found a SecurityMode and SecurityPolicy it supports, the client will send a `OpenSecureChannel` request to the server. The request will contain the client's selected SecurityMode, SecurityPolicy, the client's public certificate, and a random nonce. The random nonce will be used to generate symmetric session keys. Once the request is received, the server will determine if the client's certificate is trusted. The server will respond with a nonce if the certificate is trusted. The response will be signed by the server's private certificate and encrypted by the client's public certificate.

2b. Key Generation Once both sides have exchanged nonces, the client and server will independently generate symmetric keys. Each side will follow the algorithm below:

$$SYM\_KEY(secret, seed) = HMAC\_HASH(secret, A(1)+seed)+HMAC\_HASH(secret, A(2)+seed) + HMAC\_HASH(secret, A(3) + seed)$$

where

$$A(0) = seed$$

and

$$A(n-1) = HMAC\_HASH(secret, A(n-1))$$

The client and server's nonce are used for **both** the seed and the secret. The `HMAC_HASH` is defined by the SecurityPolicy agreed upon by the client and the server. Both sides will use the `SYM_KEY` algorithm to generate the signing and encryption keys used for each side.

3. CreateSession Once both sides have computed the symmetric keys, asymmetric keys are no longer needed. The client will use its symmetric keys to sign and encrypt its `CreateSession` request. The server will process the `CreateSession` request and return the response signed and encrypted with the server's symmetric keys.

6.1.2 OPC UA Secure Conversation (UASC). As OPC UA does not use TLS for client-server communications, it must define its packet format for exchanging security data. OPC UA Part 6 Section 6 [28] defines a packet format called OPC UA Secure Conversation (UASC) for data transmission. UASC is designed to secure an OPC UA session over various underlying transport protocols.

The packet structure of UASC depends on whether the underlying encryption algorithm provides authenticated encryption. Authenticated encryption algorithms encrypt the given data and generate and encrypt a message authentication code (MAC). The MAC lets the receiver detect if the message has been altered. Examples of authenticated encryption algorithms are AES-GCM and ChaCha20Poly1305. These algorithms have shown promise in critical infrastructure: researchers have shown how they can encrypt, transmit, and decrypt under the 3ms standard in an electrical substation network [13].

Suppose the underlying encryption algorithm does not use authenticated encryption. In that case, the UACS packet includes the signature in the data to encrypt (thus achieving the same results as an authenticated encryption algorithm). The UACS packet format is shown in Figure 4.

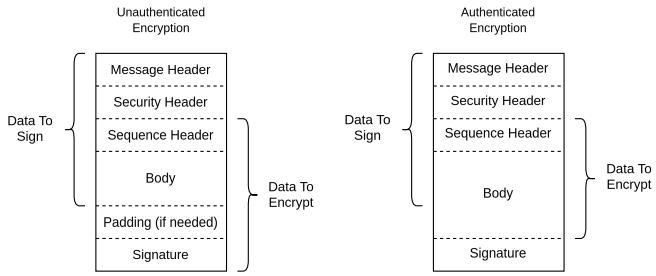


Figure 4: Packet structure of a UASC packet

The UACS packet comprises a message header, security header, sequence header, body, and signature. The message header contains data about the type of packet being transmitted. It contains a three-byte message type field, a length field, and an ID field for the secure channel. The server assigns the secure channel ID at the start of the connection. It allows both sides to look up the connection settings if multiple connections are open.

Security Header: The security header contains information exchanged during the initial handshake. In most cases, the security header only contains a token ID assigned by the server in its

response to the `OpenSecureChannel` service. The client uses the token ID in every subsequent request to the server to validate its identity. The only time the contents of the security header differ occurs when a client initially calls the `OpenSecureChannel` service. During the client's call to `OpenSecureChannel`, the client will put its requested SecurityPolicy and the client's certificate (with a thumbprint). This allows the server to validate the client's certificate and establish the connection.

Sequence Header: The security header is always followed by a sequence header. The sequence header allows the receiver to reassemble the packet if it was fragmented before transmission. OPC UA calls each transmission a MessageChunk. The sequence header contains a thirty-two-bit sequence number and a thirty-two-bit request ID. The sequence number is a randomly generated value incremented for every MessageChunk sent. The sender generates the request ID, and it remains the same across all MessageChunks for a given Message. Both allow the receiver to reassemble the complete Message and to detect replay attacks.

Padding: For unauthenticated algorithms, padding may be added to ensure the length of the message is an integer multiple of the encryption block size. If padding is needed, the first byte of the padding section will indicate the amount of padding added. Following the padding size, extra padding bytes will be added.

Signature: The final block in every signed MessageChunk contains the signature. The algorithm and length of the signature are specified in the chosen SecurityPolicy.

All messages will use the UASC structure even if the security mode is set to `None`. No encryption or signatures will be applied if the SecurityMode is set to `None`. The SecureChannelId and the TokenId are still assigned, but no security is applied to Messages exchanged via the connection. Additionally, any security fields, such as authentication tokens or nonces, shall be set to NULL and ignored by the receiver.

6.2 User Authentication & Authorization

Authentication in OPC UA is similar to authentication with CIP Security. OPC UA supports multiple authentication mechanisms. End users can choose between anonymous authentication (no authentication), username and password, X.509 certificates, or OAuth 2.0. Like CIP Security, OAuth 2.0 uses JWTs to pass identity information from the client to the server. The properties of the JWT are documented in Appendix Table 3 as defined by Part 6 Section 6-4 [28]. When a JWT is issued to the client, it is signed by the authentication service. The server will verify the authentication service's signature to ensure it is a valid JWT.

Here, we see a difference from CIP Security. OPC UA supports allowing users to restrict access to individual nodes. CIP Security does not yet support this granular level of permissions. Thus, OPC UA has data in the JWT describing user permissions via the `groups`, `roles`, and `scp` claims. Once CIP Security implements this granular level of permissions, it will need to pass the group information similarly to OPC UA.

Once the client has the authentication token (whether it is a JWT, username & password, or X.509 certificate), it will pass it to the server via the `ActivateSession` service. The `ActivateSession` service immediately follows the `CreateSession` response in Figure 3. The server will respond to the `ActivateSession` service indicating whether the authentication was successful. If the SecurityMode is `None`, then this transaction occurs in plaintext, allowing an attacker to capture credentials from the network.

6.3 Provisioning

Similarly to CIP Security, OPC UA defines a method for servers to automatically pull configuration data, certificates, and keys from a central server. OPC UA calls these servers Global Discovery Servers (GDS) defined in Part 12 [24]. These GDS servers allow an application to find other servers and pull its configuration and certificate data. To prevent rogue GDS servers on a network, OPC UA recommends that users initialize applications with a trusted GDS server. Alternatively, OPC UA applications can be manually configured if a GDS server is not on a network.

7 Discussion

The following section describes the different security approaches of these protocols and weighs the pros and cons of both approaches.

7.1 Adoption

Numerous research studies have been conducted on the security of OPC UA deployments. Dahlmanns *et al.* [3] conducted an internet-wide scan of devices exposed on TCP 4840 (the default port OPC UA uses to communicate). The authors conducted weekly measurements on the IPv4 address space for seven months, from February 2020 to August 2020. The authors found that 92% of 1114 reachable OPC UA devices have some security misconfiguration that compromises the device's security principles.

Namely, 26% of devices did not use OPC UA security. 25% of devices that use OPC UA security primitives rely on insecure cryptographic primitives (such as SHA-1) that provide weak communication security. 35% reuse certificates (such as the default certificates provided by a vendor), which allow any attacker with knowledge of this certificate to eavesdrop, modify, or silently insert malicious packets between two devices. Furthermore, Dahlmanns *et al.* found that 44% of the 1114 devices do not use an allow list to restrict devices that are allowed to communicate with the server. This allows an attacker to edit attributes or call services on the OPC UA server. This is especially troubling as these devices are *publicly connected* to the Internet, allowing anyone to modify these devices.

Ebra *et al.* [9] conducted a follow-up study to examine library and device support for OPC UA features. The authors looked at 16 open source libraries ranging from OPC UA Foundation's official C# library [31] (UA.NET) to community-created libraries such as Python-OPCUA [12]. The authors created the following criteria for library support:

(1) Support of PubSub
(2) Compatibility with Global Discovery Server (GDS)
(3) Server security features
(4) Client security features

The authors examined 16 of the most popular OPC UA libraries and found that four do not support security features. At the same time, the other twelve do offer some of the security features.

On the other hand, CIP Security has minimal hardware and software support. According to ODVA, Rockwell Automation is the only company certified to support CIP Security in 14 of its product lines [19]. We analyzed the user manuals of commonly used CIP protocol stacks to determine their support for CIP Security. The results are listed in Appendix Table 5. Out of the four CIP protocol stacks, only NetStax™ has support for CIP Security. Popular open-source CIP libraries such as OpENer [8] do not support CIP Security despite having an open GitHub issue since 2018.

Likewise, we analyzed the user manuals of popular CIP PLC vendors to determine their support for CIP Security. We examined the manuals for the Rockwell Automation CompactLogix 5360, Rockwell Automation 1756-L8, Automation Direct Productivity3000, and Schneider Electric Modicon M262. The results are listed in Appendix Table 6. Out of the four devices we've analyzed, Rockwell Automation is the only vendor to support the EtherNet/IP Confidentiality profile. These devices also support the EtherNet/IP Pull Model, which allows for automatic configuration of security settings on the PLC. None of the devices manuals we examined expressed support for the EtherNet/IP User Authentication profile.

A notable contrast emerges between OPC UA and CIP regarding message security. OPC UA offers extensive flexibility, allowing developers to select supported SecurityPolicies and SecurityModes. However, this allows developers to choose insecure or now obsolete algorithms. Conversely, CIP Security imposes the adoption of multiple algorithms for message transmission at the cost of increased complexity for developers, which in turn means significantly fewer products for end users if they wish to implement CIP Security.

7.2 Philosophy

CIP and OPC UA take two different approaches to security. CIP Security attempts to leverage proven open-source specifications such as DTLS, TLS, X.509v3 certificates, EST, and standardized cipher suites. OPC UA, on the other hand, leverages custom protocols for secure conversation and server enrollment. Both approaches have their benefits and drawbacks. By leveraging custom technologies, OPC UA has complete control over the technology used in the protocol. This makes the protocol nimble to adapt to new technologies.

On the other hand, CIP Security's use of standardized technologies allows developers to drop in proven third-party libraries that support such technologies. This alleviates developers from having to write custom cryptographic protocols. The downside of this implementation is CIP Security's reliance on standardized technologies is that the technologies do not always have resource-constrained devices in mind. A perfect example of this is highlighted in the difference between message security in the two protocols.

7.3 Message Security

Perhaps the most noticeable difference between CIP Security and OPC UA is the use of message security for encryption. CIP Security uses standard protocols like (D)TLS 1.2 to achieve authentication, authenticity, integrity, and access control. On the other hand, OPC UA specifies its handshake process as OPC UA UASC (as detailed in Section 6.1.2). Both approaches have their benefits. For CIP Security, using (D)TLS 1.2 provides a well-established, trusted protocol to secure CIP communications. Vendors can use libraries such as OpenSSL, wolfSSL, or mBedTLS to secure CIP communications. The downside, however, is that devices must be able to handle the cryptographic operations that (D)TLS 1.2 requires. This may severely limit the support of CIP Security on older ICS devices. For example, TLS 1.3 removed many uncommonly used cipher suites. This resulted in the removal of integrity-only cipher suites. CIP Security recommends integrity-only cipher suites for environments that do not need confidentiality or handle the performance impacts of encryption. ODVA and Cisco have authored RFC 9150 [2] as an **optional** extension to TLS 1.3 to support integrity-only cipher suites. It is up to each TLS library to support this optional extension.

OPC UA defines its protocol for message security as UASC. By defining its message protocol, OPC UA can tailor cryptographic operations to support low-resource devices. Defining one's own cryptographic protocol can be complex, however. Analysis by the German Federal Office for Information Security has found OPC UA UASC to be secure [4]. The Federal Office for Information Security is concerned about OPC UA's ability to deprecate and support new cryptographic algorithms, as OPC UA releases a new version of the specification every three years. Current versions of OPC UA do not support perfect forward secrecy (CIP Security does support perfect forward secrecy via its selection of TLS 1.2 algorithms). OPC UA product developers can **optionally** support TLS 1.2 for "high security" configurations.

Here again, there is a notable contrast between CIP Security and OPC UA. CIP Security mandates TLS 1.2 algorithms, which can limit its adoption, especially in older, resource-constrained devices. OPC UA allows developers to choose which profiles to support, giving developers more choices. This choice, however, leads to developers choosing insecure options, as stated in the previous section.

8 Conclusion

This paper has provided a comprehensive historical analysis of two widely utilized Operational Technology (OT) protocols: CIP (Common Industrial Protocol) and OPC UA (Open Platform Communications Unified Architecture). Additionally, it has offered a succinct overview of the semantics inherent to each protocol. The study further delineates the divergent methodologies employed by these protocols in addressing security concerns. A comparative analysis of these methodologies highlights their respective advantages and limitations. While neither approach can be deemed unequivocally superior, the insights gained from this comparison may guide the development and enhancement of other OT protocols seeking to adopt more secure practices.

9 Acknowledgements

The first author is supported by a scholarship through NSF award #2235080. This project has also received support from Rockwell Automation. The authors bear the sole responsibility for the content presented in this paper, and any interpretations or conclusions drawn from it do not reflect the official position of Rockwell Automation. The authors would also like to thank the anonymous reviewers for their comments.

References

[1] Adeen Ayub, Hyunguk Yoo, and Irfan Ahmed. 2021. Empirical Study of PLC Authentication Protocols in Industrial Control Systems. In *2021 IEEE Security and*

Privacy Workshops (SPW). 383–397. https://doi.org/10.1109/SPW53761.2021.00058
[2] Nancy Cam-Winget and Jack Visoky. 2022. TLS 1.3 Authentication and Integrity-Only Cipher Suites. RFC 9150. https://doi.org/10.17487/RFC9150
[3] Markus Dahlmanns, Johannes Lohmöller, Ina Berenice Fink, Jan Pennekamp, Klaus Wehrle, and Martin Henze. 2020. Easing the Conscience with OPC UA: An Internet-Wide Study on Insecure Deployments. In *Proceedings of the ACM Internet Measurement Conference* (Virtual Event, USA) *(IMC '20)*. Association for Computing Machinery, New York, NY, USA, 101–110. https://doi.org/10.1145/3419394.3423666
[4] Damm and Gappmeier and Zugfil and Plöb and Fiat and Störtkuhl. 2017. OPC UA Security Analysis. https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/Studies/OPCUA/OPCUA.pdf
[5] Dragos. 2017. CRASHOVERRIDE: Analysis of the Threat to Electric Grid Operations. https://www.dragos.com/wp-content/uploads/CrashOverride-01.pdf?utm_referrer=https%3A%2F%2Fwww.dragos.com%2F
[6] Dragos. 2024. Impact of FrostyGoop ICS Malware on Connected OT Systems. https://hub.dragos.com/hubfs/Reports/Dragos-FrostyGoop-ICS-Malware-Intel-Brief-0724_r2.pdf
[7] D. Dzung, M. Naedele, T.P. Von Hoff, and M. Crevatin. 2005. Security for Industrial Communication Systems. *Proc. IEEE* 93, 6 (2005), 1152–1177. https://doi.org/10.1109/JPROC.2005.849714
[8] EIP Stack Group. 2024. OpENer. Available at https://github.com/EIPStackGroup/OpENer (2005/06/12).
[9] Alessandro Erba, Anne Müller, and Nils Ole Tippenhauer. 2022. Security Analysis of Vendor Implementations of the OPC UA Protocol for Industrial Control Systems. In *Proceedings of the 4th Workshop on CPS & IoT Security and Privacy* (Los Angeles, CA, USA) *(CPSIoTSec '22)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3560826.3563380
[10] ESET. 2017. Industroyer: Biggest malware threat to critical infrastructure since Stuxnet. https://www.eset.com/int/industroyer/
[11] OPC Foundation. 2023. OPC 10000-3 - UA Specification Part 3 - Address Space Model 1.05.03. https://reference.opcfoundation.org/Core/Part3/v105/docs/
[12] Free OPC-UA Library. 2024. Python OPC UA. Available at https://github.com/FreeOpcUa/opcua-asyncio (2005/06/12).
[13] Daniel Hunter, Jack Parry, Kenneth Radke, and Colin Fidge. 2017. Authenticated encryption for time-sensitive critical infrastructure. In *Proceedings of the Australasian Computer Science Week Multiconference* (Geelong, Australia) *(ACSW '17)*. Association for Computing Machinery, New York, NY, USA, Article 19, 10 pages. https://doi.org/10.1145/3014812.3014832
[14] Michael B. Jones, John Bradley, and Nat Sakimura. 2015. JSON Web Token (JWT). RFC 7519. https://doi.org/10.17487/RFC7519
[15] Yang Li, Shihao Wu, and Quan Pan. 2023. Network Security in the Industrial Control System: A Survey. arXiv:2308.03478 [cs.CR]
[16] Zhiyong Luo and Xue Zhang. 2020. Research on OPC UA Security Encryption Method. In *2020 IEEE International Conference on Information Technology,Big Data and Artificial Intelligence (ICIBA)*, Vol. 1. 287–292. https://doi.org/10.1109/ICIBA50161.2020.9276932
[17] Michael Mann, Ron Floyd, Jack Visoky, and Joakim Wiberg. 2017. A Practical Guide for CIP Security Device Developers. https://www.odva.org/wp-content/uploads/2021/09/2017-ODVA-Conference_Floyd_Mann_Visoky_Wiberg_CIP_Security_Guide_FINAL.pdf
[18] Microsoft and contributors. 2020. DCOM Security Enhancements in Windows XP Service Pack 2 and Windows Server 2003 Service Pack 1. https://learn.microsoft.com/en-us/windows/win32/com/dcom-security-enhancements-in-windows-xp-service-pack-2-and-windows-server-2003-service-pack-1
[19] ODVA. [n. d.]. ODVA CIP Security Marketplace. Available at https://marketplace.odva.org/products/#?vendors=all&productTypes=all&deviceTypes=all&docYears=all&categories=all&services=security&page=1&lang=en&view=search&productDisplay=all/ (2024/04/08).
[20] ODVA Inc. [n. d.]. Industrial Automation, Technologies and Standards. Available at https://odva.org (2024/03/12).
[21] ODVA, Inc. 2023. The CIP Networks Library Volume 1: Common Industrial Protocol Edition 3.35. Uponrequesttolicensedparties
[22] ODVA, Inc. 2023. The CIP Networks Library Volume 8: CIP Security Edition 1.17. Uponrequesttolicensedparties
[23] OPC Foundation. 2022. OPC 10000-1 - UA Specification Part 1 - Overview and Concepts 1.05.02. https://reference.opcfoundation.org/Core/Part1/v105/docs/
[24] OPC Foundation. 2022. OPC 10000-12 - UA Specification Part 12 - Discovery and Global Services 1.05.02. https://reference.opcfoundation.org/GDS/v105/docs/
[25] OPC Foundation. 2022. OPC 10000-7 - UA Specification Part 7 - Profiles 1.05.02. https://reference.opcfoundation.org/Core/Part7/v105/docs/
[26] OPC Foundation. 2022. Profile Core 2022 Server Facet. https://profiles.opcfoundation.org/profile/1322
[27] OPC Foundation. 2023. OPC 10000-2 - UA Specification Part 2 - Security Model 1.05.03. https://reference.opcfoundation.org/Core/Part2/v105/docs/
[28] OPC Foundation. 2023. OPC 10000-6 - UA Specification Part 6 - Mappings 1.05.03. https://reference.opcfoundation.org/Core/Part6/v105/docs/
[29] OPC Foundation. 2024. History. Available at https://opcfoundation.org/about/opc-foundation/history/ (2024/04/08).
[30] OPC Foundation. 2024. OPC Online Reference. Available at https://reference.opcfoundation.org/ (2024/04/08).
[31] OPC Foundation. 2024. OPC Unified Architecture .NET Standard. Available at https://github.com/OPCFoundation/UA-.NETStandard (2005/06/12).
[32] Sebastian Paul and Esther Guerin. 2020. Hybrid OPC UA: Enabling Post-Quantum Security for the Industrial Internet of Things. In *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Vol. 1. 238–245. https://doi.org/10.1109/ETFA46521.2020.9212112
[33] Max Pritikin, Peter E. Yee, and Dan Harkins. 2013. Enrollment over Secure Transport. RFC 7030. https://doi.org/10.17487/RFC7030
[34] Eric Rescorla and Tim Dierks. 2008. The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246. https://doi.org/10.17487/RFC5246
[35] Eric Rescorla and Nagendra Modadugu. 2012. Datagram Transport Layer Security Version 1.2. RFC 6347. https://doi.org/10.17487/RFC6347
[36] Viktor Schiffer. 2016. The Common Industrial Protocol (CIP™) and the Family of CIP Networks. https://www.odva.org/wp-content/uploads/2020/06/PUB00123R1_Common-Industrial_Protocol_and_Family_of_CIP_Networks.pdf
[37] M. H. Schwarz and J. Börcsök. 2013. A survey on OPC and OPC-UA: About the standard, developments and investigations. In *2013 XXIV International Conference on Information, Communication and Automation Technologies (ICAT)*. 1–6. https://doi.org/10.1109/ICAT.2013.6684065
[38] Jack Visoky and Joakim Wiberg. 2020. CIP Security and IEC 62443-4-2. https://www.odva.org/wp-content/uploads/2020/05/2020-ODVA-Conference_CIP_Security_and_IEC_62443_Visoky_Wiberg_Final.pdf
[39] Anna Volkova, Michael Niedermeier, Robert Basmadjian, and Hermann de Meer. 2019. Security Challenges in Control Network Protocols: A Survey. *IEEE Communications Surveys & Tutorials* 21, 1 (2019), 619–639. https://doi.org/10.1109/COMST.2018.2872114
[40] Kehe Wu, Yi Li, Long Chen, and Zhuxiao Wang. 2015. Research of Integrity and Authentication in OPC UA Communication Using Whirlpool Hash Function. *Applied Sciences* 5 (2015), 446–458. https://api.semanticscholar.org/CorpusID:15798962

A Appendix

Claim	Name	Explanation
sub	Subject	The subject identifier for which the token was issued. A unique identifier for the user
aud	Audience	The audience for the token. The audience is a server (or a group of servers) which are allowed to accept the token
name	Name	Human-readable name for the user
scp	Scopes	A list of scopes for the user. This is usually a list of permissions or restrictions that limit user access rights
nonce	Nonce	A nonce to prevent replay attack
groups	Groups	A list of groups which the user belongs
roles	Roles	A list of roles which are assigned to the subject

Table 3: JWT claims for OPC UA

Claim	Name	Explanation
sub	Subject	User name
iss	Issue	The name of the Authority that issued the token
nonce	Nonce	A nonce to prevent replay attack
iat	Issued At Time	The UTC at which the token was issued
exp	Expiration Time	The UTC at which the token expires.

Table 4: JWT claims for CIP Security

Library	Language	EtherNet/IP Confidentiality	EtherNet/IP User Authentication	EtherNet/IP Pull Model Support
OpENer	C/C++	○	○	○
NetStaX ™	C	●	○	○
CODESYS	Unknown	○	○	○
Real Time Automation EtherNet/IP Stack	C	○	○	○

Table 5: Library support for CIP Security

Device	Vendor	EtherNet/IP Confidentiality	EtherNet/IP User Authentication	EtherNet/IP Pull Model Support
ControlLogix 5580	Rockwell Automation	●	○	●
CompactLogix 5380	Rockwell Automation	●	○	●
Modicon M262	Schneider Electric	○	○	○
Productivity3000	Automation Direct	○	○	○

Table 6: PLC support for CIP Security