International Journal of Service and Knowledge Management
International Institute of Applied Informatics
2022, Vol. 6, No. 2, IJSKM731

Monitoring Encrypted Communication with OPC UA

Toshiaki Honda [*], Yuki Shimazawa [*],
Takashi Hamaguchi [*], Yoshihiro Hashimoto [*]

Abstract

Cyber-attacks on critical infrastructure have been on the rise. Therefore, cyber-security has become very important for Industrial Control Systems. For communication protocol in Industrial Control Systems networks, the Open Platform Communications Unified Architecture communication protocol, which enables secure and platform-independent communications, is expected to be widely used. An important property of Open Platform Communications Unified Architecture is encryption. It is effective in protecting communication data from tampering and eavesdropping but also makes it impossible to monitor communications. In Industrial Control Systems, inappropriate commands to controllers can cause dangerous situations. Even a secure communication protocol cannot guarantee that the data being communicated are safe. There are many types of machines, such as operating support systems and engineering workstations, that can send commands to controllers. They are implemented in common operating systems and may fall victim to a cyber-attack. Therefore, the commands to controllers should be monitored. We monitor the communication by decrypting the encrypted data. In addition, we propose a method of monitoring without communication loads by making the decryption mechanism independent and using the decrypted data to enable flexible integration with other systems such as Security Information and Event Management.

*Keywords: c*yber-security, encrypted data, monitoring system, OPC UA, SIEM

1 Introduction

Cyber-attacks on critical infrastructure have been on the rise. When Industrial Control Systems (ICS) became the targets of cyber-attackers, the incidents this caused were not only security issues but also safety and serious business issues. The cyber-attacks negative impact is extremely high, for example, the occurrence of accidents at manufacturing sites, damage to customers due to inappropriate shipping products, and temporary suspension of manufacturing services [1].

The Open Platform Communications Unified Architecture (OPC UA) communication protocol enables platform-independent and secure communications in the industrial automation field [2]. From OPC UA v1.04, which was released in 2021, Pub/Sub communication has been implemented in addition to Client/Server. It can support a large number of communications.

[*] Nagoya Institute of Technology, Aichi, Japan

Therefore, it can cover all industrial use cases, including real-time communication from the field and simultaneous commands to multiple devices. For this reason, the use of OPC UA has been standardized in the industrial sector, and its performance has been studied in various papers [3][4][5].

Data confidentiality is an important function of OPC UA for ICS. By encrypting the data with OPC UA, the data are not visible to anyone other than authorized clients [6].

Even if the communication protocol is secure, the data in the communication cannot be guaranteed as safe. There are many machines that send commands to controllers. For example, operation support systems, advanced control systems, Manufacturing Execution Systems (MES) servers, engineering workstations of Supervisory Control And Data Acquisition (SCADA) systems, Distributed Control Systems (DCS), and Programmable Logic Controllers (PLCs). Inappropriate commands to controllers can cause dangerous situations. Because they are implemented in common operating systems, it is not difficult to obtain their information in various ways, such as through manuals. An inappropriate command is one that does something that would not be possible in normal control, such as changing the temperature to a set value that exceeds the upper limit. When such commands are hijacked by cyber-attackers, dangerous commands might be sent to controllers even if OPC UA was adopted as the communication protocol for the communication between the controllers and them [7][8].

Therefore, to detect such attacks to ICS, the data in the communications to controllers should be monitored even if they were encrypted with OPC UA [9]. Although OPC UA makes it impossible to intercept the data, we want to monitor the data in OPC UA. It is an issue to solve in this paper.

2 Open Platform Communications Unified Architecture

2.1 Change over from OPC Classic to OPC UA

OPC UA was developed by the OPC Foundation in 2006 as the successor to OPC Classic (DA, AE, HDA). OPC Classic is a communication technology that uses Microsoft's distributed component object model (DCOM) and enables easy interoperability between different products. The OPC Classic specification defines interfaces for ICS data exchange, message notification, and time-series data exchange. Today, it is used in the products of companies around the world and has become the de facto standard in ICS communication technology [10][11][12][13].

The rapid increase in incidents and accidents related to information security, however, has increased the security requirements in the communication environment. As a result, it is difficult to operate OPC Classic because DCOM, which enables OPC Classic’s ease of connection, has serious vulnerabilities [14]. OPC UA is the second-generation OPC specification that takes into account information security measures from the design stage. It also inherits the ease of connection of OPC Classic without the use of DCOM.

2.2 Security architecture of OPC UA

OPC UA is standardized as IEC-62541. OPC UA v1.03 supports a Client/Server protocol, and OPC UA v1.04 supports a Pub/Sub protocol. They are implemented on top of the Transmission Control Protocol/Internet Protocol (TCP/IP) stack [1].

The security architecture of OPC UA adopts Public Key Infrastructure (PKI) and common key cryptography as a hybrid. Common key encryption is used for data encryption, and PKI is used for application authentication and encryption of exchanging information necessary for

generating the common key. The common key is updated each time the OPC UA communication connection is started, so it is not reused. In addition, OPC UA uses multiple common keys used for encryption to increase the confidentiality of data. Therefore, it is extremely difficult for a third party to guess the common key and decrypt the data by OPC UA. In addition, the access level can be set by attribute through node management provided by OPC UA [15].

The security modes of OPC UA are categorized into None, Sign, and SignAndEncrypt. None mode transfers data in plain text. Sign mode transfers data with a digital signature. SignAndEncrypt mode encrypts the data and uses a digital signature to transfer data. OPC UA supports the following security policies for Sign mode and SignAndEncrypt mode:

1. Basic128Rsa15
2. Basic256
3. Basic256Sha256
4. Aes128Sha256RsaOaep
5. Aes256Sha256RsaPss

The server provides these security modes and security policies. The client selects the information provided by the server, and then the client will be able to connect with the selected security mode and security policy. However, it is necessary to note that the communication contents can be intercepted when using None. If you don't want the communication contents to be intercepted, it is recommended to configure the OPC UA server so that the Nonce mode is not used.

In the future, the OPC Foundation has indicated on its roadmap that it will add Elliptic Curve Cryptography (ECC) to security policy [16].

3 How to Monitor Encrypted Communications

3.1 Basic idea of encrypted communication monitoring

Figure 1 shows a simple configuration of Client/Server communication. This structure is the same for Pub/Sub mode. Even if a communication packet is intercepted, the interceptor will not be able to understand the content because it is encrypted with OPC UA.

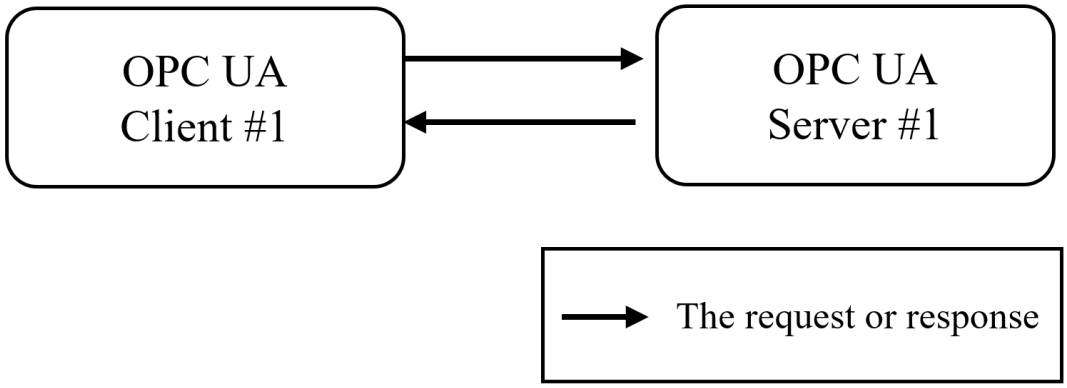

Figure 1: General connection configuration

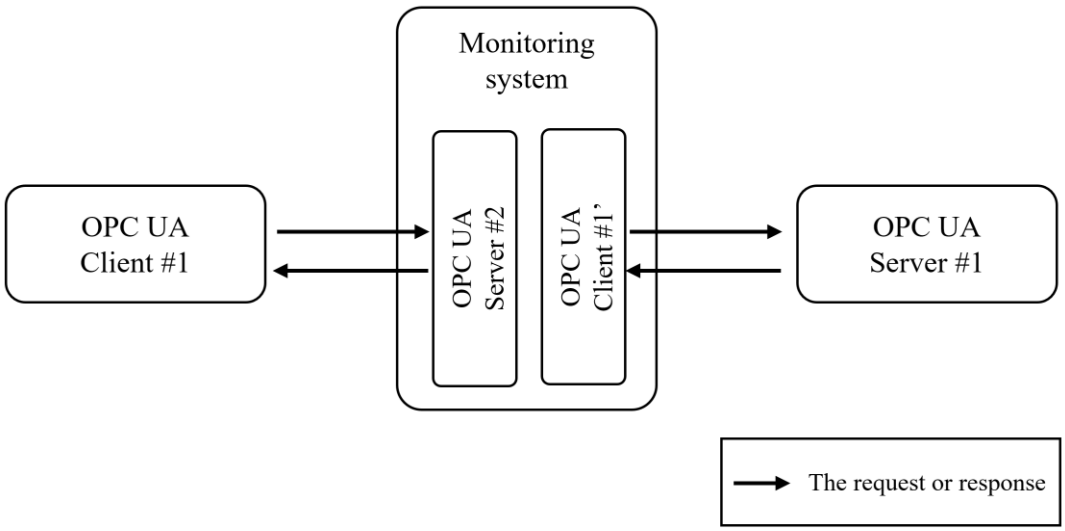

Figure 2: Basic configuration of the monitoring system

Figure 2 shows the basic configuration of the concept of monitoring encrypted communications. This paper, we refer to the concept of monitoring as a monitoring system. An important feature of this monitoring system is that it does not monitor by interception. The client and server do not communicate directly but communicate via the monitoring system. The monitoring system can monitor the data between Client #1 and Server #2, and then send the same data from Client #1' to Server #1. It also can monitor the data between Server #1 and Client #1', and then send the same data from Server #2 to Client #1. So, the monitoring system also can manage the common key for the client and server.

For the communications, TCP data for OPC UA (opc.tcp) are used. The format of opc.tcp has the following structure.

opc.tcp://[Endpoint name]:[port number]/[server name]

The part of server name is called the url-path and is not used to control the route of communication packets. The usage of the url-path in the monitoring system is explained in the next section. The following Uniform Resource Locators (URLs) are examples of opc.tcp expression.

opc.tcp://localhost:51210/UA/SampleServer
opc.tcp://192.168.1.3:4840

More precisely, Server #2 is always listening and waiting for requests. If it receives a connection request from Client #1, the monitoring system creates a proxy Client #1' for Client #1. Then, the monitoring system must send all requests received from the client to the server. The security policy and user token should be able to relate to the configuration specified by Server #1. Also, Client #1 can connect by changing only the endpoint URL because when deploying a monitoring system between a communicating client and server, we want to reduce the risk of operation errors. The risk of operation errors refers to setting a weak security policy that differs from the system specification when changing the security mode or security policy.

The client communicates with the monitoring system, so it is possible to hide the server. If

attacks can be detected at the monitoring system, the communication packets might be rejected before being sent to the server. This can be a defense against cyber-attacks.

3.2 Routing control

Figure 3 shows the communication configuration between multiple clients and multiple servers. In the monitoring system, the same number of clients as servers are to be monitored, and only one server exists. All clients to be monitored communicate with the server in the monitoring system. The following two functions are required to enable route control.

3.2.1 Route control by static table

OPC UA uses the URL of the web system to point to the location of the server [17]. The OPC UA URL takes the form: <scheme>://<host>:<port>/<url-path> and is called the endpoint URL.

The scheme is an identifier used by a URL to represent the type of protocol. Protocols supported by OPC UA are data for OPC UA, HyperText Transfer Protocol (HTTP), and WebSocket. The scheme name for TCP data for OPC UA is set to "opc.tcp". By default, OPC UA's TCP protocol typically uses port 4840 for TCP connections. The scheme name for HTTP for OPC UA is set to "http". By default, the HTTP protocol typically uses port 80 for HTTP connections, and port 443 for HTTP connections tunneled over Transport Layer Security (TLS). When using TLS, it sets the scheme name to "https". The scheme name for WebSocket for OPC UA is set to "ws". By default, the WebSocket protocol typically uses port 80 for WebSocket connections and port 443 for WebSocket connections tunneled over TLS [18]. When using TLS, it sets the scheme name to "wss". The host is the domain name of a network host or IP address. The port is the port number of the connection destination. Finally, the url-path sets the access path to specific information. However, the url-path is not used in OPC UA. The monitoring system uses this url-path for routing. In OPC UA, the url-path can be acquired from the request so that it can be used for route control of the monitoring system [17][19].

The client connects to the server by specifying the endpoint URL. The client includes the url-path in the endpoint URL. It then determines the endpoint URL of the server to which the request received by the monitoring system should be forwarded. Route control sets fixed route information in the monitoring system and determines the transfer destination. Because this function can uniquely select the transfer destination, the load on the central processing unit (CPU) is small. The route is also fixed, so it is possible to prevent unintentional rewriting.

Figure 3 is a diagram of route control. The numbers separated by periods near the squares mean IP addresses. The number in parentheses means the port number published by the OPC UA server. The monitoring system uses the url-path specified by the client as a key to acquire the endpoint URL, which is the next route, from the static table. The schema uses the same schema that was requested. For example, if Client #1 communicates with Server #1 via the monitoring system, Client #1 sets "Server #1" in the url-path. The key required for route control to determine the next route is "Server #1". Therefore, Client #1 specifies the full endpoint URL, "opc.tcp://192.168.1.4:4850/Server#1". The monitoring system obtains the next destination from the static table using the key. The proxy client (Client #1') for Client #1 created by the monitoring system communicates to Server #1 using the next destinations. Thus, client #1' will connect to the full endpoint URL, "opc.tcp://192.168.2.2:48010".

In addition, if Client #1 communicates with Server #2 via the monitoring system, Client #1 sets "Server#2" in the url-path. The key required for route control to determine the next route is "Server #2". Therefore, Client #2 specifies the full endpoint URL, "opc.tcp://192.168.1.4:4850/Server#2". The monitoring system obtains the next destination from

the static table using the key. The proxy client (Client #1") for Client #1 created by the monitoring system communicates to Server #2 using the next destinations. Thus, client #1" will connect to the full endpoint URL, "opc.tcp://192.168.3.3:48010".
One OPC UA client can connect to multiple OPC UA servers via a monitoring system.

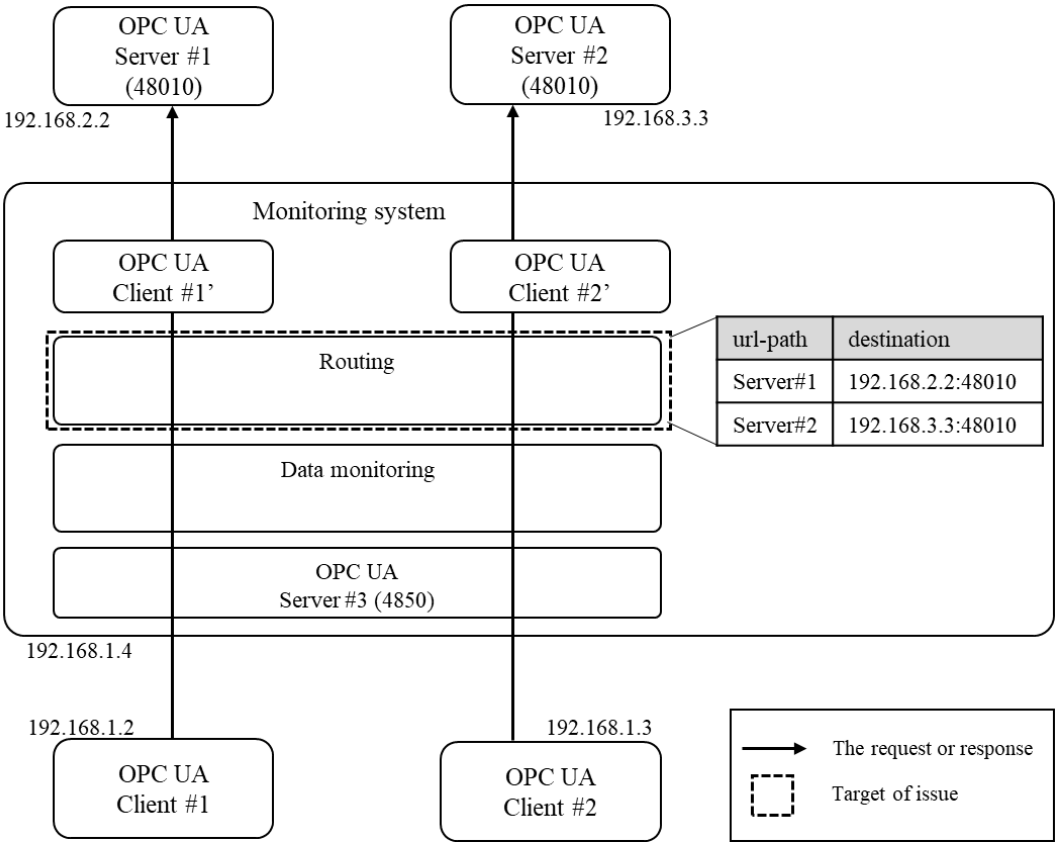

Figure 3: Route control sub-system

Figure 4 shows an actual flow and position of route-control processing. The IP address is shown at the top of each application. The numbers under the monitoring system and Server #1 are port numbers. The black star signifies the position of the route control. This route control executes processing in three places. If Client #1 needs to connect to Server #1, the following process is run:

a) Client #1 sends GetEndpoints request to "opc.tcp://192.168.1.4:4840/Server#1" to connect to the monitoring system

b) The monitoring system obtains Server #1 with Blackstar 1

c) The monitoring system determines the next destination (192.168.2.2:48010) from Server #1

d) The monitoring system sends GetEndpoints request to "192.168.2.2:48010"

e) Client #1 sends OpenSecureChannel request to "opc.tcp://192.168.1.4:4840/Server#1"

f) Blackstar 2 determines the next destination, "192.168.2.2:48010" in the same manner as Blackstar1.

g) Client #1 sends CreateSession request to "opc.tcp://192.168.1.4:4840/Server#1"

h) Blackstar 3 determines the next destination, "192.168.2.2:48010" in the same manner as Blackstar1.

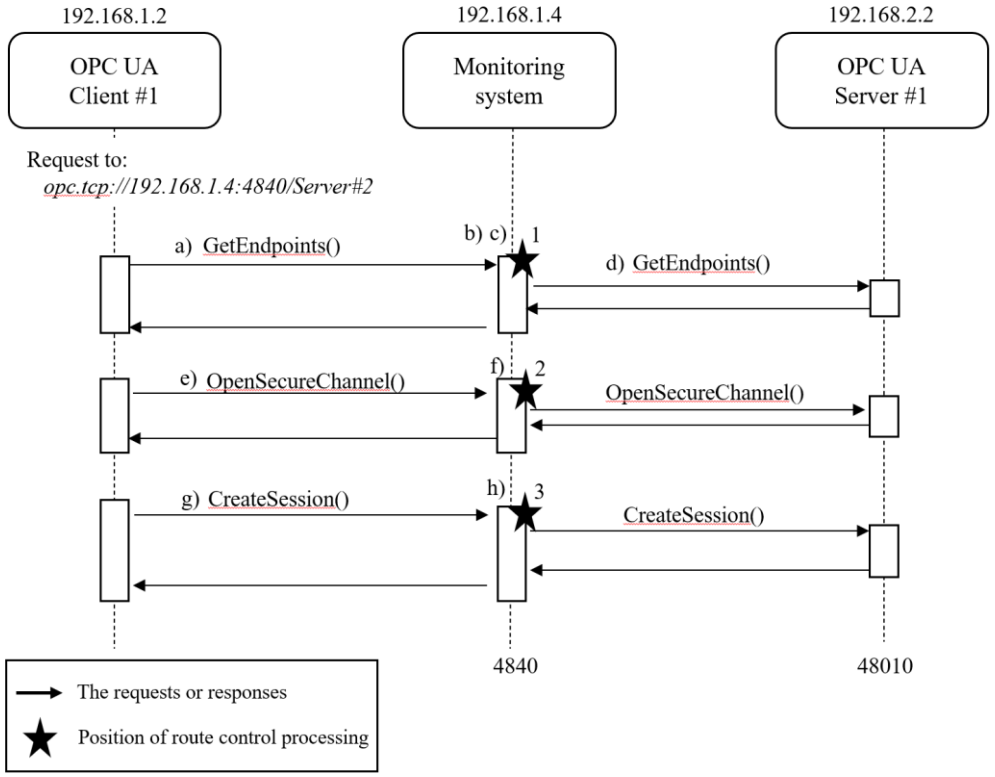

Figure 4: Route-control flow

3.2.2 Routing by GetEndpoints service and Change server certificate

To communicate via the monitoring system, it is necessary to perform important processing for the Get Endpoints communication shown in the flow shown in Figure 4. Specifically, it refers to the communication from a) to d) in Figure 4. The GetEndpoints service returns the endpoints supported by the server and all the configuration information required to establish communication.

The GetEndpoints service returns a response containing the endpoint URL to which the client should connect. The endpoint URL is the session endpoint. The client establishes a connection in accordance with the endpoint URL included in the GetEndpoins service response. If we do not change the endpoint URL, it will connect directly to the server, omitting the monitoring software. Therefore, the endpoint URL included in the GetEndpoins service response needs to be changed [20].

The client expects the monitoring system to return the endpoints supported by a server and all the configuration information required to establish communication. The monitoring

system needs to support all security policies and security modes. The security policies supported by a monitoring system are "Basic128Rsa15", "Basic256", "Basic256Sha256", "Aes128SSha256RsaOaep", and "Aes256SSha256RsaPss".

The GetEndpoints service response contains the following four security configurations:

- Server Application Instance Certificate
- Message Security Mode
- Security Policy
- Supported User Identity Tokens

To establish encrypted communication between the client and the monitoring system, the "Server Application Instance Certificate" in the GetEndpoints service response must be changed to a monitoring system certificate. The "Server Application Instance Certificate" is the information required to use the public key cryptosystem. By changing the information, the client and monitoring system can communicate the encrypted data. The monitoring system and server do not need to change the "Server Application Instance Certificate" [20].

Figure 5 shows the part that should be encrypted. It is necessary to encrypt between the client and the monitoring system.

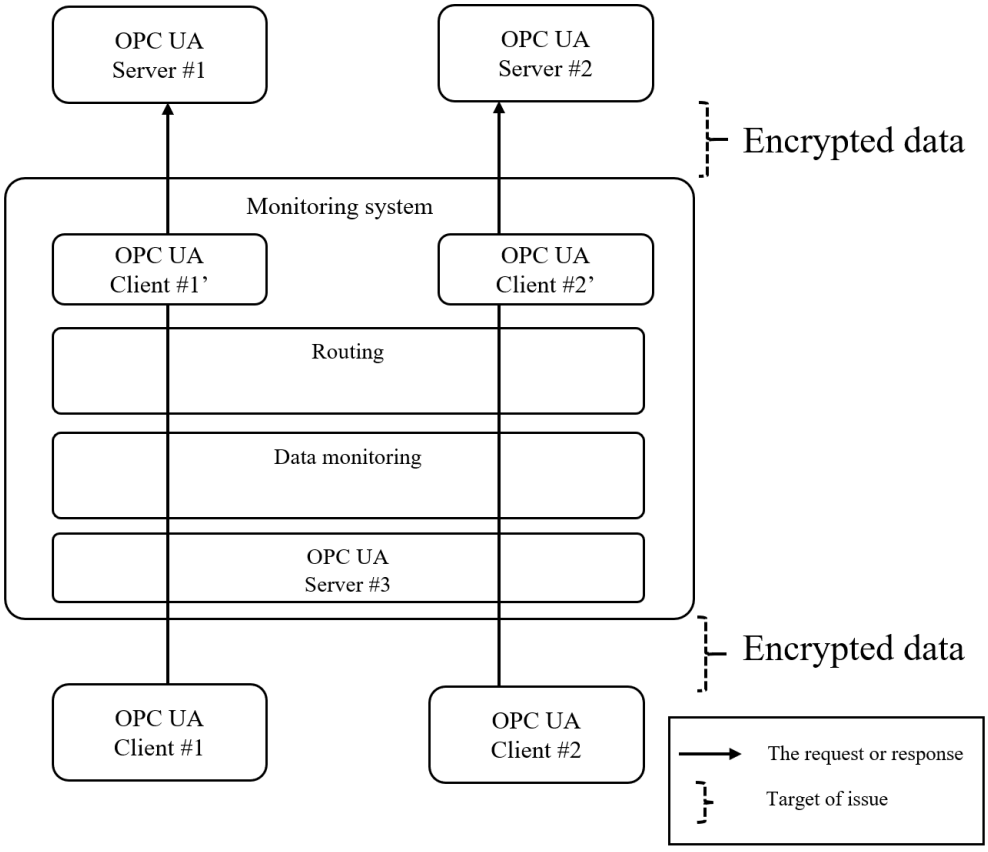

Figure 5: Encrypted data via the monitoring system

3.3 Handling of monitored data

OPC UA can transfer all types of data and parameters. The following types of data are observed in a monitoring system:

- Current data
- Complex structures
- Event data
- Historical data

Complex structures can aggregate a large amount of data. Complex structures are supported by some PLCs and should be supported by monitoring systems as well. Event data are the parameters of events that occur on the server. Complex structures and event data are efficient because a large amount of data can be transferred at once. Historical data is time-series data for a specific period of time.

Figure 6 shows the inclusion of a sub-system for monitoring encrypted data. Encrypted data is not immediately decrypted by Data Monitoring. Encrypted data will be retained in Data Logging and decrypted later. The mechanism for decrypting encrypted data is independent, allowing for flexible integration with other systems. The decrypted data can be output as text or database in the current version. The data format is output in JSON format so that supervisors can easily analyze the data. It can also expect that analysis using Security Information and Event Management (SIEM) will allow early detection of threats that could harm the system, such as inappropriate commands.

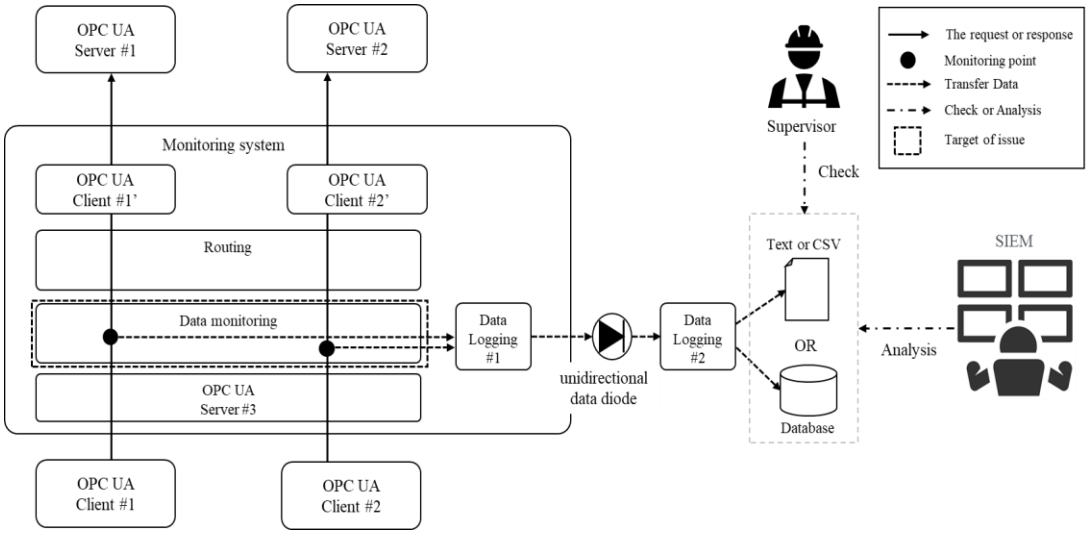

Figure 6: Analyzing Encrypted Data

Encrypted data will be decrypted later by the monitoring system. A session is created between the OPC UA client and the proxy to monitor the OPC UA encrypted data in a typical proxy. Also, a session is created between the proxy and the OPC UA server. Therefore, it is necessary to construct two or more sessions and the communication load increases. It is required to decrypt it once by the proxy, and to manage the session, too. After that, it is

necessary to encrypt it again and to connect it to the OPC UA server. Moreover, the common key generated by the OPC UA client and the OPC UA server can be retained in the monitoring system. Thereby it is possible to decrypt the data later. However, we believe that cyber-attacks are more likely to target the monitoring system. If we are logging data externally from our monitoring system as a defense against cyber-attacks, we can combine it with the unidirectional data diode to further enhance security. The OPC UA's Pub/Sub communication supports UDP. UDP is a communication method that does not require a response and allows data to be transferred to the outside world even though a unidirectional data diode. The OPC UA's Pub/Sub communication is encrypted. Unidirectional data diodes do not allow packets from the outside to pass through, so they provide effective protection against cyber-attacks [21].

The center of Figure 6 shows that Data Logging #1 and Data Logging #2 support the OPC UA's Pub/Sub communication, and the data decoded by Data Logging #1 is sent to Data Logging #2 via the unidirectional data diode. Data Logging #2 outputs the received data to CSV or database. The OPC UA's Pub/Sub communication uses Advanced Encryption Standard Counter Mode (AES-CTR) for data encryption. AES-CTR encrypts and decrypts with a pre-exchanged key and Message Nonce contained in the message. Therefore, the encryption of data becomes more complex, which further strengthens the protection against attacks and threats.

4 Evaluation

4.1 Proof of implementation

This section explains the monitoring system. This monitoring system can run on Windows or Linux.

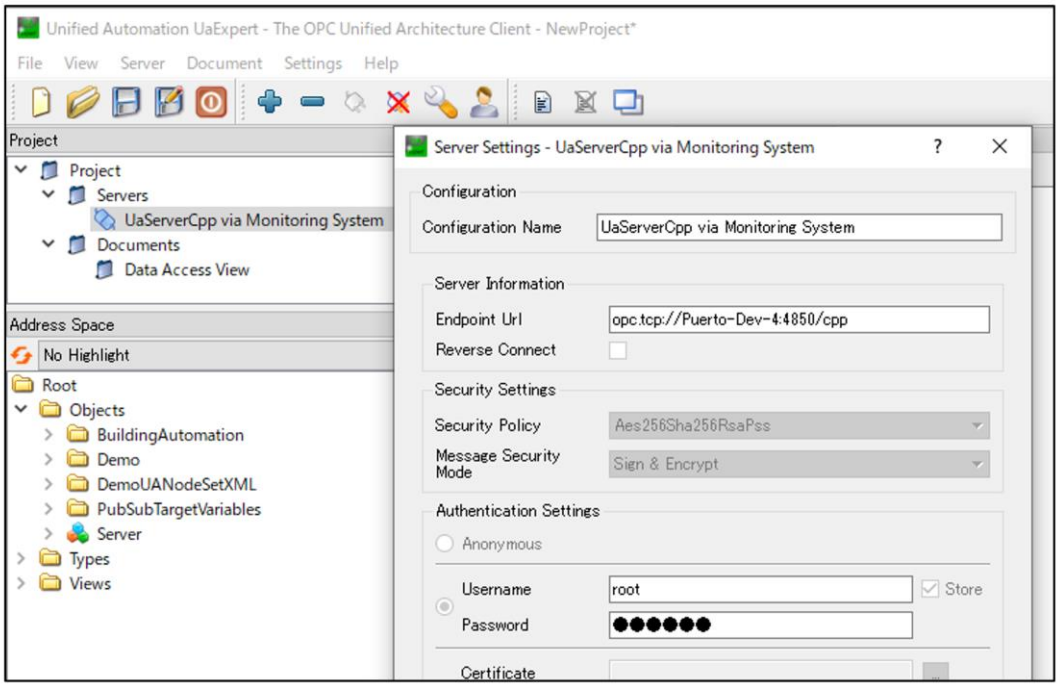

Figure 7: OPC UA client screen connected via the monitoring system

The setting that monitors the OPC UA communication of UaExpert, a general OPC UA client application published for free by Unified Automation GmbH, is shown here. As shown in Figure 7, the system is connected to Unified Automation's OPC UA C++ Demo Server via the monitoring system. The endpoint URL does not specify a direct URL to the OPC UA C++ Demo Server but specifies the endpoint URL of the monitoring system. The endpoint URL of the monitoring system needs to include the forwarding destination. In Figure 7, the endpoint of the monitoring server has a hostname of “Puero-Dev-4” and a port number of “4850”. And set "cpp" to url-path. The full endpoint URL is "opc.tcp://Puerto-Dev-4:4850/cpp". Then, the packet will be forwarded to the endpoint of the OPC UA C++ demo server by specifying the " cpp " of url-path. The security policies and user tokens do not need to be changed with the original settings.

The monitoring system passes the request from the OPC UA client to the OPC UA server. Then the response from the server is returned to the client. For example, by passing through the browsed request and response, the same hierarchical structure as when connecting directly to the OPC UA server can be displayed. In addition, when the OPC UA client does Read or Write, the NodeId included in the request and the value corresponding to it can be monitored. If there is an inappropriate read or write, it will be detected.

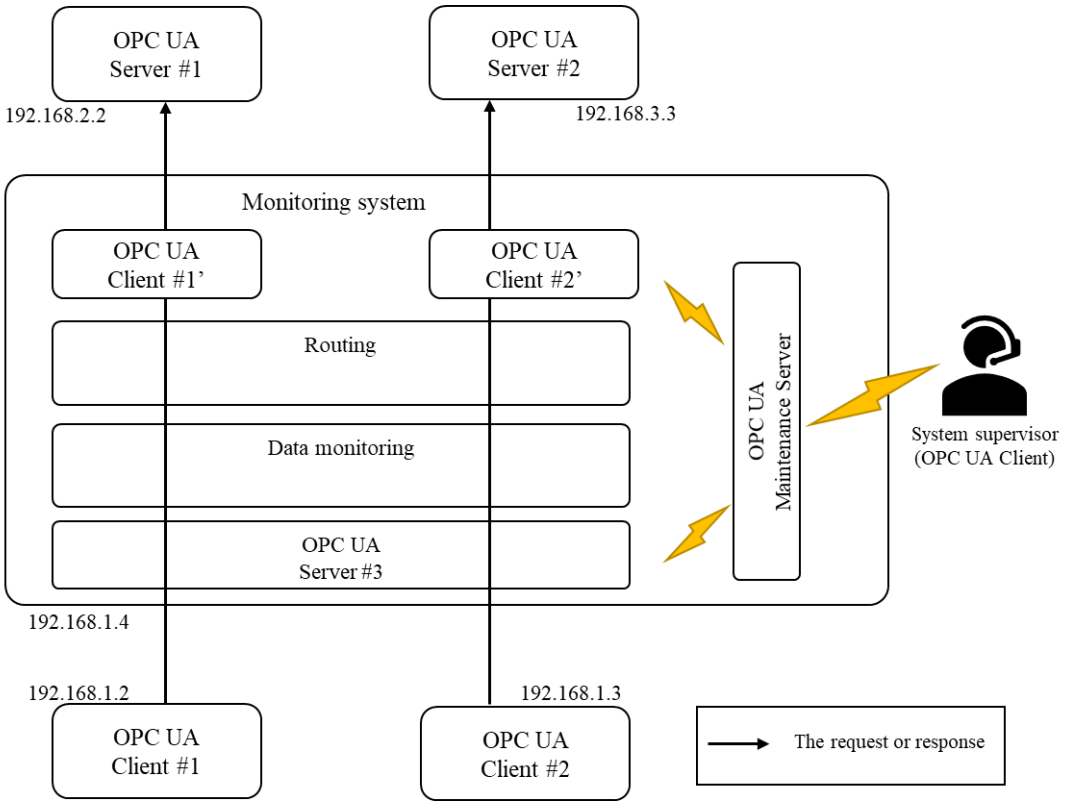

Figure 8: The monitoring system maintenance function

The client communicates with the monitoring system, so it is possible to hide the server. But the monitoring system is a possibility of receiving a cyber-attack instead of the server. Therefore, the monitoring system has the maintenance function to enable monitoring of the status.

Figure 8 shows the monitoring system maintenance function. The numbers separated by periods near the squares mean IP addresses. The maintenance function is an OPC UA server that functions separately from Server #3. This OPC UA server provides an audit function via events. The lightning icon in Figure 8 indicates an event. If the OPC UA server notifies the event, the event is notified to the System supervisor from the OPC UA maintenance server. Event means the information model of conditions, dialogue conditions, and alarms (including acknowledgment functions) as defined in Part 9 of the OPC UA specification released by the OPC Foundation. The difference between an OPC UA's Event and the value of Data Access is that an event can notify the related values together. This event function has the following types:

- Condition Event
- Audit Event (Tracking Event)
- Simple Event

Condition Event is an event with status, such as a process alarm. Audit Event or Tracking Event is an event about the operation which occurred in the OPC UA server. A simple Event is a general event; all events except Condition Event and Audit Event fall under this category.

The maintenance function of the monitoring system uses Audit events to perform audits on operations related to OPC UA. For example, if Server # 3 detects an untrusted client, it notifies the System Supervisor (OPC UA client) of an event. Since the maintenance function is an OPC UA server, it is necessary to dis-close the port number. We will improve the maintenance function.

4.2 Performance

Table 1 shows the system configuration for measuring the performance of the monitoring system.

Table 1: System Configuration

CPU	Memory (RAM)	OS
Intel(R) Core(TM) i7-6800K CPU @ 3.40 GHz 3.40 GHz	32.0 GB	Windows 10 x64 1909

Figure 9 shows the average processing time. Ten OPC UA clients connect to Unified Automation's OPC UA C ++ Demo Server then send 10,000 requests per client to the OPC UA server. Thus, it is time it takes to send 100,000 requests to the OPC UA server and get a response. The OPC UA client and OPC UA server run on the same PC.

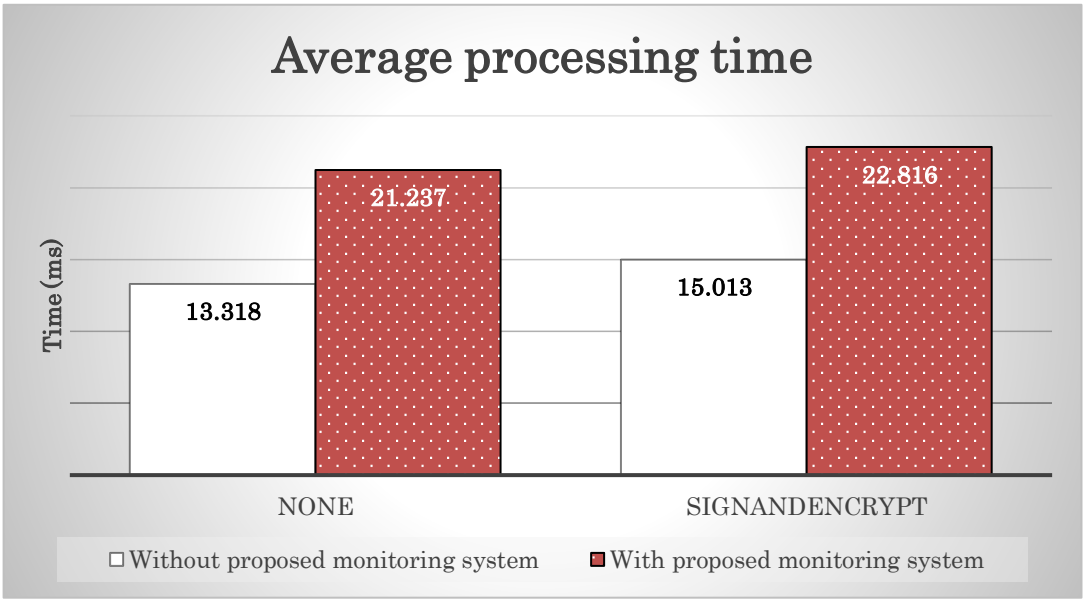

Figure 9: Average processing time with direct connection (without the monitoring system) and with the monitoring system

5 Conclusion

We explained the monitoring of encrypted data by OPC UA. OPC UA is difficult to eavesdrop on the network unless the secret key and the common key are stolen, but the monitoring system can be installed between the client and the server to monitor the encrypted data. The monitoring system is also easy for system administrators, as the client only needs to change the URL of the endpoint.

Furthermore, decryption of encrypted data can be done later, thus reducing processing delays. By making the decryption mechanism independent and using the decrypted data, we are proposing a way to monitor the data without incurring communication loads. In addition, it can be flexibly linked with other systems such as SIEM, and the server can be hidden because the client communicates with the monitoring system.

The next step also examines how supervisors can recognize cyber-attacks at an early stage. Also, the monitoring system is implemented in C#. Thus it needs to be implemented in a low-level computer language (e.g., C++, Rust) to improve the processing speed. We will then evaluate the effectiveness of the defense against cyber-attacks. Also, it is difficult to ensure complete security with such a system alone. Therefore, it is necessary for organizations to establish guidelines for proper operation.

6 References

[1] Wataru Machii, Isao Kato, Masahito Koike, Masafumi Matta, Tomomi Aoyama, Hidemasa Naruoka et al., “Dynamic zoning based on situational activitie for ICS security” 10th Asian Control Conference (ASCC), May 2015.

[2] OPC Foundation, “OPC Unified Architecture Part 1: Overview and Concepts Release 1.04”, November 2017.

[3] Hermann Haskamp, Michael Meyer, Romina Möllmann, Florian Orth, Armando Walter Colombo, “Benchmarking of existing OPC UA implementations for Industrie 4.0-compliant digitalization solutions” 2017 IEEE 15th International Conference on Industrial Informatics (INDIN), July 2017.

[4] Haoyu Yu, Dong Yu, Yi Hu, Chuting Wang, “Research on CNC Machine Tool Monitoring System Based on OPC UA” 2019 Chinese Control And Decision Conference (CCDC), June 2019.

[5] Hsien-I Lin, Yu-Che Hwang, “Integration of Robot and IIoT over the OPC Unified Architecture” 2019 International Automatic Control Conference (CACS), November 2019.

[6] Anna Volkova, Michael Niedermeier, Robert Basmadjian, Hermann de Meer, “Security Challenges in Control Network Protocols: A Survey” IEEE Communications Surveys & Tutorials, Vol. 21, pp. 619-639.

[7] Shingo Abe, Mariko Fujimoto, Shinichi Horata, Yukako Uchida, Takuho Mitsunaga, “Security threats of Internet-reachable ICS” 2016 55th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE), September 2016

[8] Yaodong Tao, Wei Xu, Hongbin Li, Shenglong Ji, “Experience and Lessons in Building an ICS Security Testbed” 2019 1st International Conference on Industrial Artificial Intelligence (IAI), July 2019.

[9] Akira Yamada, Yutaka Miyake, Masahiro Terabe, Kazuo Hashimoto, “Experience and Lessons in Building an ICS Security Testbed” IPSJ Journal, Vol. 49, No. 3, pp. 1144-1154, Mar 2008.

[10] M. H. Schwarz, J. Börcsök, “A survey on OPC and OPC-UA: About the standard, developments and investigations” 2013 XXIV International Conference on Information, Communication and Automation Technologies (ICAT), November 2013.

[11] OPC Foundation, “OPC Unified Architecture Part 8: Data Access Release 1.04”, November 2017.

[12] OPC Foundation, “OPC Unified Architecture Part 9: Alarms and Conditions Release 1.04”, November 2017.

[13] OPC Foundation, “OPC Unified Architecture Part 11: Historical Access Release 1.04”, January 2018.

[14] D.P.Zegzhda, M.O.Kalinin, M.V.Levykin, “Actual Vulnerabilities of Industrial Automation Protocols of an Open Platform Communications Series” Automatic Control and Computer Sciences Vol. 53, pp. 972–979.

[15] OPC Foundation, “OPC Unified Architecture Part 2: Security Model Release 1.04”, August 2017.

[16] OPC Foundation, "OPC UA Roadmap (2021)", Available: https://opcfoundation.org/about/opc-technologies/opc-ua/opcua-roadmap/, June 2021.

[17] Internet Engineering Task Force, "RFC1738 Uniform Resource Locators (URL) ", Available: https://datatracker.ietf.org/doc/html/rfc1738, December 1994.

[18] Internet Engineering Task Force, "RFC2818 HTTP Over TLS", Available: https://datatracker.ietf.org/doc/html/rfc2818, May 2000.

[19] Internet Engineering Task Force, "RFC2396 Uniform Resource Identifiers (URI): Generic Syntax", Available: https://datatracker.ietf.org/doc/html/rfc2396, August 1998.

[20] OPC Foundation, "OPC Unified Architecture Part 4: Services Release 1.04", November 2017.

[21] Toshiaki Honda, Takashi Hamaguchi, Yoshihiro Hashimoto, "OPC UA information transfer via unidirectional data diode for ICS cyber security", PSE2021+, June 2022.