Lawrence Livermore National Laboratory

Technical Analysis of SSP-21 Protocol

UNCLASSIFIED

CES-21 TLP WHITE
ORIG LLNL

9 June 2017

Seth Bromberger, David Youd, and David Knapp, LLNL

Disclaimer

Table of Contents

1 Introduction and Background

As part of the California Energy Systems for the Twenty-First Century (CES-21) program, in December 2016 San Diego Gas and Electric (SDG&E) contracted with Lawrence Livermore National Laboratory (LLNL) to perform an independent verification and validation (IV&V) of a white paper describing their Secure SCADA Protocol for the Twenty-First Century (SSP-21) in order to analyze the effectiveness and propriety of cryptographic protocol use within the SSP-21 specification.

SSP-21 is designed to use cryptographic protocols to provide (optional) encryption, authentication, and nonrepudiation, among other capabilities. The cryptographic protocols to be used reflect current industry standards; future versions of SSP-21 will use other advanced technologies to provide a subset of security services.

For this report, the following versions of the SSP-21 protocol specification were evaluated:

- Specification dated 17 November 2016
- Specification dated 18 January 2017

2 Analysis of the SSP-21 Protocol

2.1 About the protocol specification document

The SSP-21 specification document is well-thought out and clear in describing SSP-21 security and performance objectives. The specification goes into detail with respect to the functions, structure, and expected responses; there is a good explanation of the cryptographic decisions that underpin, for example, the choice of parameters for CRC checks.

2.2 SSP-21 Protocol Overview

The SSP-21 protocol distinguishes itself from others in its relative simplicity and ability to be deployed in a secure fashion across networks with varying reliability, latency, and bandwidth. Few assumptions are made with respect to data ordering or the security of the underlying communications path.

According to the specification the SSP-21 protocol is designed to provide the following security guarantees:

- Authentication of messages
- Protection against message replay
- Protection against valid message flooding via timeout protection

The cryptographic algorithms underpinning each of these guarantees are based on industry standards. Where differentiation was important, the evaluation focused on the use of shared secrets as described in the “Alternative: Shared Secrets” section of the 18 January 2017 version of the specification and the “Alternative: Symmetric Keys” section of the 17 November 2016 version.

The protocol itself is defined in two abstraction layers: a *link layer* that is responsible for framing, addressing, and error detection, and a *cryptographic layer* that provides the security guarantees for the protocol. Each of these layers is described in more detail below.

2.3 SSP-21 Link Layer Overview

The SSP-21 link layer is designed for use in environments that do not provide one or more of the following services directly to the SSP-21 cryptographic layer:

- Framing (which is required for stream-based protocols such as TCP) designed to extract an SSP-21 unit message from a stream of data;
- Addressing, required to ensure proper routing of the messages; and
- Error detection, which is used to ensure reliable delivery of cryptographically-signed messages.

In practical terms, this means that the SSP-21 link layer must be implemented if the protocol is used directly above a TCP transport due to the lack of framing support provided by TCP to higher-level protocols (regardless of the underlying link protocol; this is a deliberate design decision and the nature of abstraction in the TCP/IP stack).

Because multiple implementations of UDP do not actually verify the integrity of the UDP checksum, it is recommended that the SSP-21 link layer be considered when the protocol is used directly above a UDP transport as well.

For implementations that integrate SSP-21 between the DNP3 transport and application layers, the SSP-21 link layer is not strictly needed. This is true even if the underlying transport is TCP or UDP, since the DNP link and transport layers provide the three necessary link services.

The error detection capability provided by the SSP-21 link layer is robust and complements the cryptographic layer. A message that passes CRC checks but does not validate cryptographically is statistically very likely the result of a cryptographic attack and not a transmission error. Any implementation of SSP-21 that relies on a weaker error detection mechanism as part of a different link layer protocol will not have this level of assurance.

2.4 SSP-21 Cryptographic Layer Overview

The Cryptographic Layer is responsible for providing the majority of the security guarantees claimed by SSP-21. Sessions are established between an initiator and a responder via a handshaking protocol that establishes the security parameters of the session. Among the decisions to be made for a session are

- The nonce mode;
- The Diffie-Hellman key exchange algorithm;
- The key derivation function, hashing algorithm, and other cryptographic support to guarantee the integrity of the handshake; and
- The algorithms used to provide security guarantees for the overall session.

2.4.1 Session Security

Once a handshake between an initiator and a responder has been completed, both sides have derived two common symmetric keys: one key secures traffic from the initiator to the responder; the other secures traffic in the other direction.

The two symmetric keys generated during the handshake process are shared between initiator and responder. The first key is used by the initiator to (optionally) encrypt and (non-optionally) sign messages and by the responder to decrypt and authenticate messages; the second is used by the initiator to decrypt and authenticate messages (optionally) encrypted and (non-optionally) signed by the responder. In this way, the SSP-21 protocol provides message encryption and authentication for a session.

Each session is time-sensitive: relative timestamps are kept by each side and time-to-live (TTL) values are used in each message to invalidate old messages and prevent rapid replay of deliberately-delayed messages.

Nonces are used to guarantee sequential processing of messages. Two nonce modes exist: a strict mode (*INCREMENT_LAST_RX*) that requires that a received nonce is exactly equal to the last valid nonce plus one; and a more forgiving mode (*GREATER_THAN_LAST_RX*) that simply verifies that the received nonce is greater than the last valid nonce. Each mode has its advantages and disadvantages: for lossy or unstable link layers where messages might be lost, the strict nonce mode would result in frequent session terminations, and the *GREATER_THAN_LAST_RX* mode would be more appropriate.

Session data is validated by the responder via the following actions:

- Authentication / decryption of the message using the appropriate session key is successful;
- Nonce value is equal to expected value (in strict mode), or greater than expected value; and
- Validity period of the message is not expired.

A message that fails validation will be dropped by the responder. In strict nonce mode, this will cause all subsequent messages to fail nonce validation at the responder, which will eventually result in the session timing out.

2.4.2 Session Termination / Invalidation

SSP-21 sessions can only be invalidated under a specific set of circumstances:

- Either nonce reaches its maximum value;
- The session timeout period expires; or
- The session is reinitialized via a subsequent valid handshake.

Closing the underlying communication session will cause the session expiry via session timeout. There is no explicit signaling between the communications session and the cryptographic layer.

It is important to note that the reception of spurious, malformed, or cryptographically-unsound messages WILL NOT invalidate an existing session. The primary reason for limiting session

invalidation to a defined set of criteria is to limit the effects of a malicious actor sending malformed data to a valid recipient in an attempt to cause a denial of service via unscheduled session termination.

2.4.3 SSP-21 Timing Considerations

SSP-21 relies on accurate time intervals in order to take advantage of session timeouts and message validity periods. While a synchronized absolute time between the initiator and responder is not necessary, both sides should be able to keep track of relative elapsed time to a precision of one millisecond.

The initiator and responder both agree to a common "time anchor" (which may be a different absolute timestamp) during protocol handshake. Message validity periods are simply offsets from this common time reference.

Messages that reach the responder past the validity period are discarded. Because there is no explicit notification back to the initiator that a message has been discarded, in strict nonce mode, subsequent messages sent by the initiator will have nonces greater than what the responder is expecting, and therefore all further messages will be discarded, leading to an eventual session timeout.

3 Comparison with Other Protocols

3.1 TLS

TLS (Transport Layer Security) is a well-known internet protocol that provides network communications security. It was designed to secure TCP transport, but has been modified to secure UDP (via Datagram Transport Layer Security) as well.

The protocol's handshake includes a cipher suite negotiation between the participants, determining the authentication, encryption, message authentication codes, and key exchange algorithms that will be used. Servers (and optionally clients) are authenticated via a chain of digital certificates. Depending on configuration, the protocol may provide forward secrecy.

TLS evolved out of the SSL (Secure Sockets Layer) protocol developed by Taher Elgamal to support ecommerce in the 1995 Netscape web browser. TLS is optimized for the general-purpose task of securing internet application communications, and depending on configuration, may not be suitable for networks with strict latency requirements (*e.g.*, SCADA environments).

At all scales of data messaging, there is a tension between security monitoring and data encryption. SCADA systems tend to favor strong data integrity without confidentiality. While TLS offers integrity and authentication without encryption[1], use of the NULL family of ciphers is not widely supported.

Because TLS was not designed with SCADA requirements in mind, it offers primary security

[1] see, *e.g.*, https://tools.ietf.org/html/rfc4785

features that are less important in control systems environments than in other (traditional IT) networks. The prioritization of confidentiality, as the primary example, leads to computational and network overhead that is unwarranted for common SCADA installations and provides an unnecessary security feature at comparatively significant cost. In addition, because time-to-live limits are not available in TLS, the protocol is unable to detect messages that should be ignored due to expiration.

3.2 DNP3-SA

DNP3 (Distributed Network Protocol 3) is an open and interoperable control system communication protocol, initially designed in the early 1990s for use by the electrical utility industry. It is a 3-layer protocol (link, transport, and application), which can sit atop various, existing transports (serial, TCP, etc.).

The protocol has received wide acceptance, but unfortunately was designed without security considerations in mind. In 2012, DNP3 SAv5 extended the protocol by adding Public Key Infrastructure (PKI) and pre-shared symmetric key mechanisms for authentication. While an improvement, the reliance on shared symmetric keys can result in scalability issues in larger installations.

DNP3 supports considerable flexibility in the construction of its Application Service Data Units (ASDUs), the messages that are sent in the protocol's application layer. While this flexibility meets the messaging requirements for DNP3, it can be at odds with DNP3 SAv5's strong authentication capability (implementation of which is optional for some function codes – the decision as to which messages are authenticated are left up to the vendors and end users). To support the additional complexity, software/hardware vendors must develop more complex parsing functionality that must carefully guard against a larger attack surface. For this reason, LLNL assesses that a lighter-weight protocol that implements a defined set of security guarantees relevant to control systems environments in a simple, straightforward manner could provide a better security posture than DNP3-SA.

4 Next Steps

LLNL recommends continued evaluation of the SSP-21 specification and the SSP-21 reference code as further development occurs. Specifically, the introduction of self-signed certificates for identification (used for authentication), and the use of a formal PKI infrastructure for establishing the trust relationships among the SSP-21 devices, may introduce both advantages as well as potential weaknesses in the protocol (both from a specification and an implementation perspective).