5 Security Threats and Countermeasures

There are two sets of security requirements that need to be considered, those of the applications that use WS-RM and those of the WS-RM protocol itself.

This specification makes no assumptions about the security requirements of the applications that use WS-RM. However, once those requirements have been satisfied within a given operational context, the addition of WS-RM to this operational context should not undermine the fulfillment of those requirements; the use of WS-RM should not create additional attack vectors within an otherwise “secure” system. The sole exception to this is the creation of additional denial of service attack points. For example, a particular WS-RM implementation may implement the Reliable Messaging Destination (RMD) as an independent SOAP-processing node. Once an application begins using this WS-RM implementation it becomes vulnerable to attacks against the machine(s) and services that support the RMD.

5.1 Threats and Countermeasures

The primary security requirement of the WS-RM protocol is to protect the WS-RM semantics and protocol invariants against various threats. The following sections discuss various threats to the integrity and operation of the WS-RM protocol and provide general outlines of the recommended countermeasures to those threats. Readers should keep in mind that every threat is not necessarily applicable in every operational context.

5.1.1 Integrity Threats

In general, any mechanism which allows an attacker to alter the information in a Sequence Traffic Message or Sequence Lifecycle Message, or which allows an attacker to alter the correlation of a RM Protocol Header Block to its intended message represents a threat to the WS-RM protocol.

For example, the WS-RM specification states:

> The RM Source MUST assign each message within a Sequence a message number beginning at 1 and increasing by exactly 1 for each subsequent message. These numbers MUST be assigned in the same order in which messages are sent by the Application Source.

If an attacker is able to swap `<wsrm:Sequence>` headers on messages in transit between the RMS and RMD then they have undermined the implementation's ability to guarantee this invariant. The result is that there is no way of guaranteeing that messages will be delivered to the Application Destination in the same order that they were sent by the Application Source.

5.1.1.1 Countermeasures

Integrity threats are generally countered via the use of digital signatures or encryption at some level of the communication protocol stack. Note that, in order to counter header switching attacks, the signed and/or encrypted block must include both the SOAP body and the relevant SOAP headers (e.g. `<wsrm:Sequence>` header).

5.1.2 Resource Consumption Threats

The creation of a Sequence with an RMD consumes various resources on the systems used to implement that RMD. These resources include network connections, database handles, database tables, etc. This behavior can be exploited to conduct denial of service attacks against an RMD. For example, a simple attack is to repeatedly send `<wsrm:CreateSequence>` messages to an RMD. Another attack is to create a Sequence for a service that is known to require in-order message delivery and use this

Sequence to send a stream of very large messages to that service, making sure to omit message number "1" from that stream.

5.1.2.1 Countermeasures

There are a number of countermeasures against the resource consumption threats. The technique recommended by this specification is for the RM Destination to restrict the ability to create a Sequence to a specific set of entities/principals. This reduces the number of potential attackers and, in some cases, allows the identity of any attackers to be determined.

The ability to restrict Sequence creation depends, in turn, upon the RM Destination's ability identify and authenticate the RM Source that issued the CreateSequence message.

5.1.3 Sequence Spoofing Threats

Sequence spoofing is a class of threats in which the attacker uses its knowledge of the `<wsrm:Identifier>` for a particular Sequence to forge Sequence Lifecycle or Traffic Messages. For example the attacker creates a fake `<wsrm:TerminateSequence>` message that references the target Sequence and sends this message to the appropriate RMD. Some sequence spoofing attacks also require up-to-date knowledge of the current `<wsrm:MessageNumber>` for their target Sequence.

In general any Sequence Lifecycle Message, RM Protocol Header Block, or sequence-correlated SOAP fault (e.g. `<wsrm:InvalidAcknowledgement>`) can be used by someone with knowledge of the Sequence identifier to attack the Sequence. These attacks are "two-way" in that an attacker may choose to target the RMS by, for example, inserting a fake `<wsrm:SequenceAcknowledgement>` header into a message that it sends to the "AcksTo" EPR of an RMS.

5.1.3.1 Sequence Hijacking

Sequence hijacking is a specific case of a sequence spoofing attack. The attacker attempts to "inject" Sequence Traffic Messages into an existing Sequence by inserting fake `<wsrm:Sequence>` headers into those messages.

The following scenario provides an example:

1. An RMS and RMD create a Sequence with ID "urn:uuid:72dfcac0-3d09-11da-8cd6-0800200c9a66".
2. The RMS transmits messages 1-10 under this sequence.
3. An attacker gains knowledge of the above sequence ID and message numbers and transmits messages 11-19 under "urn:uuid:72dfcac0..." to the RMD using fake sequence headers.

At this point one or both of the following could occur:

- The RMS transmits the "legitmate" message number 11 under "urn:uuid:72dfcac0...". This message is silently ignored by the RMD since it considers message number 11 to have already been received. The result is that the "real" message number 11 is not received by the RMD and the RMS has no way of determining this fact.
- The RMS receives acknowledgments for messages 12-19 thus causing it to send a `<wsrm:InvalidAcknowledgement>` to the RMD (see Section 4.4 of [WS-RM]) and close the sequence.

Note that "sequence hijacking" should not be equated with "security session hijacking". Although a Sequence may be bound to a security session in order to counter the threats described in this section, applications MUST NOT rely on WS-RM-related information to make determinations about the identity of

the entity that created a message; applications SHOULD rely only upon information that is established by the security infrastructure to make such determinations. Failure to observe this rule creates, among other problems, a situation in which the absence of WS-RM may deprive an application of the ability to authenticate its peers even though the necessary security processing has taken place.

5.1.3.2 Countermeasures

There are a number of countermeasures against sequence spoofing threats. The technique recommended by this specification is to consider the Sequence to be a shared resource that is jointly owned by the RM Source that initiated its creation (i.e. that sent the CreateSequence message) and the RM Destination that serves as its terminus (i.e. that sent the CreateSequenceResponse message). To counter sequence spoofing attempts the RM Destination must ensure that every Sequence Lifecycle Message and Sequence Traffic Message originates from the RM Source that jointly owns the affected Sequence. For its part the RM Source must ensure that every Sequence Lifecycle Message, Sequence Acknowledgment Message, and Sequence-related fault originates from the RM Destination that jointly owns the affected Sequence.

For the RM Destination to be able to identify its sequence peer it must be able to identify and authenticate the entity that sent the CreateSequence message. Similarly for the RM Source to identify its sequence peer it must be able to identify and authenticate the entity that sent the CreateSequenceResponse message. For either the RM Destination or the RM Source to determine if a message was sent by its sequence peer it must be able to identify and authenticate the initiator of that message and, if necessary, correlate this identity with the sequence peer identity established at sequence creation time.

5.2 Security Solutions and Technologies

The security threats to the WS-RM protocol are neither new nor unique. The solutions that have been developed to secure other SOAP-based protocols can be used to secure WS-RM as well. In this section we map the facilities provided by common security solutions against countermeasures described in the previous sections.

Before continuing this discussion, however, some examination of the underlying requirements of the previously described countermeasures is necessary. Specifically it should be noted that the technique described in Section 5.1.2.1 has two components. The first part is the ability of the RM Destination to identify and authenticate the issuer of a CreateSequence request. This requirement is shared by the technique described in Section 5.1.3.2 and will be discussed in the following sections. The second part is the ability of the RM Destination to perform an authorization check against this authenticated identity to determine if the requesting entity is permitted to create Sequences with the RM Destination. Since the facilities for performing this authorization check (runtime infrastructure, policy frameworks, etc.) lie completely within the domain of individual implementations, any discussion of such facilities is considered to be beyond the scope of this specification.

5.2.1 Transport Layer Security

This section describes how the the facilities provided by SSL/TLS [RFC 2246] can be used to implement the countermeasures described in the previous sections. The description provided is general in nature and is not intended to serve as a complete definition on the use of SSL/TLS to protect WS-RM. In order to interoperate implementations must agree on the choice of features as well as the manner in which they will be used.

5.2.1.1 Model

The basic model for using SSL/TLS is as follows:

1. The Initiating party (RM Source or RM Destination) establishes SSL/TLS session with accepting party (RM Destination or RM Source). Authentication information may be exchanged during this step.

2. The SSL/TLS session is used to transmit one or more, WS-RM-related SOAP messages from the initiating party to the accepting party. Authentication information (i.e. a HTTP BasicAuth header) may accompany this transmission.

5.2.1.2 Countermeasure Implementation

Used in its simplest fashion (without relying upon any authentication mechanisms), the per-packet encryption performed by SSL/TLS provides the necessary integrity qualities to counter the threats described in Section 5.1.1. Note, however, that the nature of SSL/TLS limits the scope of this integrity protection to a single transport level session. If SSL/TLS is the only mechanism used to provide integrity, any intermediaries between the RM Source and the RM Destination must be trusted to preserve the integrity of the messages that flow through them.

As noted, the techniques described in Sections 5.1.2.1 and 5.1.3.2 both involve the use of authentication. This specification recommends either of two mechanisms for authenticating entities using SSL/TLS. In both of these methods the SSL server (the party accepting the SSL connection) authenticates itself to the SSL client using an X.509 certificate that is exchanged during the SSL handshake.

- **HTTP Basic Authentication:** This method of authentication presupposes that a SOAP/HTTP binding is being used as part of the protocol stack beneath WS-RM. Subsequent to the establishment of the the SSL session, the sending party authenticates itself to the receiving party using HTTP Basic Authentication [RFC 2617]. For example, a RM Source might authenticate itself to a RM Destination (e.g. when transmitting a Sequence Traffic Message) using BasicAuth. Similarly the RM Destination might authenticate itself to the RM Source (e.g. when sending an acknowledgement) using BasicAuth.

- **SSL Client Authentication:** In this method of authentication, the party initiating the connection authenticates itself to the party accepting the connection using an X.509 certificate that is exchanged during the SSL handshake (the SSL server is configured to request the client's certificate during the handshake).

To implement the countermeasures described in section 5.1.2.1 the RM Source must authenticate itself using one the above mechanisms. The authenticated identity can then be used to determine if the RM Source is authorized to create a Sequence with the RM Destination.

When considering the countermeasures described in section 5.1.3.2 we need to consider the relationship of the Sequence lifetime to that of the SSL session lifetime. If we define the Sequence lifetime to be bounded by the lifetime of the SSL session, we can simplify the authorization decisions to a check for SSL session ID equivalence. That is, a WS-RM node's Sequence peer is equivalent to their SSL session peer. Any message arriving over the same SSL session must have originated from the co-owner of the Sequence.

If we define the Sequence lifetime to span multiple SSL sessions then we must have the ability to compare the various forms of authentication information supplied by the SSL Server Authentication and either HTTP BasicAuth or SSL Client Authentication. This is necessary because, outside of the scope of a particular security session, the only way to determine who sent a message is via the sender's authenticated identity.

For example, suppose RMS A and RMD B establish a mutually-authenticated SSL session with ID 'β'. RMS A authenticates as “/o=bitparts/cn=b2bgtw3" and RMD B authenticates as “/o=fabrikam/cn=wsrmG”. They then use this session to establish a Sequence with ID “urn:uuid:3b757de4-52be-41b0-b83a-5fa7b35bb21d”. As long as session β is valid, RMD B can be certain that any messages arriving over β

for Sequence “urn:uuid:3b757de4..” must have originated from RMS A. Now suppose that RMS A and RMD B negotiate a new SSL session with ID 'ε' using the same credentials. RMD B has no way of knowing, simply by looking at the SSL session ID, whether messages arriving over ε can take part in Sequence “urn:uuid:3b757de4..” since ε wasn't used to create the Sequence. It's only when RMD B takes into account the fact that its authenticated peer for both β and ε is “/o=bitparts/cn=b2bgtw3" that it can determine that the messages arriving over ε originated from RMS A and can therefore participate in Sequence“urn:uuid:3b757de4..”.

5.2.2 SOAP Message Security

The mechanisms described in WS-Security [WS-Security] may be used in various ways to implement the countermeasures described in the previous sections. Specifically, this specification recommends the protocol described by WS-SecureConversation [WS-SecureConverstaion] (in conjunction with WS-Trust [WS-Trust]) as a mechanism for protecting WS-RM Sequences. The description provided is general in nature and is not intended to serve as a complete definition on the use of WS-SecureConversation/WS-Trust to protect WS-RM. In order to interoperate implementations must agree on the choice of features as well as the manner in which they will be used.

5.2.2.1 Model

The basic model for using WS-SecureConversation is as follows:

1. The RM Source and the RM Destination jointly create a Security Context. This may involve the participation of third parties such as a security token service. The tokens exchanged may contain authentication claims (e.g. X.509 certificates or Kerberos service tickets).
2. The RM Source and the RM Destination use the session key(s) associated with the Security Context to either sign or encrypt (as defined by WS-Security) at least the body and any relevant WS-RM-defined headers of any messages they send to each other. A single Security Context should be used to transmit and receive multiple WS-RM-related messages.
3. The RM Source and RM Destination may renew or amend the Security Context as necessary (subject to policy constraints).

5.2.2.2 Countermeasure Implementation

Without relying upon any authentication information, the per-message signatures (or encryption blocks) provide the necessary integrity qualities to counter the threats described in Section 5.1.1.

To implement the countermeasures described in section 5.1.2.1 some mutually-agreeable form of authentication claims must be provided by the RM Source to the RM Destination during the establishment of the Security Context. These claims can then be used to determine if the RM Source is authorized to create a Sequence with the RM Destination.

As with transport-level security, the implementation of the countermeasures 5.1.3.2 depends on the relationship between the Sequence lifetime and the lifetime of the WS-SecureConversation Security Context. If we define the lifetime of the Sequence to be bounded by the lifetime of the Security Context, we can simplify the authorization decisions to a check for Security Context equivalence. That is, a WS-RM node's Sequence peer is equivalent to their Security Context peer. Any messages signed/encrypted using the same Security Context must have originated from the co-owner of the Sequence.

If we define the Sequence lifetime to span multiple Security Contexts then, just as with the transport-level security sessions, we must have the ability to compare the authentication claims exchanged during the establishment of the Security Contexts in order to determine who owns the Sequence and who sent the messages that relate to that Sequence.