Internet Draft   Jesse Walker
Expiration: March 2005   Amol Kulkarni, Ed.
File: draft-ietf-rap-cops-tls-09.txt   Intel Corp.

COPS Over TLS

Last Updated: September 24, 2004

Status of this Memo

Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Abstract

This document describes how to use Transport Layer Security (TLS) to secure Common Open Policy Service (COPS) connections over the Internet.

This document also updates RFC 2748 by modifying the contents of

the Client-Accept message.

Table Of Contents

Glossary

COPS -	Common Open Policy Service. See [RFC2748].
COPS/TCP -	A plain-vanilla implementation of COPS.
COPS/TLS -	A secure implementation of COPS using TLS.
PDP -	Policy Decision Point. Also referred to as the Policy Server. See [RFC2753].
PEP -	Policy Enforcement Point. Also referred to as the Policy Client. See [RFC2753].

1 Introduction

COPS [RFC2748] was designed to distribute clear-text policy information from a centralized Policy Decision Point (PDP) to a set of Policy Enforcement Points (PEP) in the Internet. COPS provides its own security mechanisms to protect the per-hop integrity of the deployed policy. However, the use of COPS for sensitive applications such as some types of security policy distribution requires additional security measures, such as data privacy. This is because some organizations find it necessary to hide some or all of their security policies, e.g., because policy distribution to devices such as mobile platforms can cross domain boundaries.

TLS [RFC2246] was designed to provide channel-oriented security. TLS standardizes SSL and may be used with any connection-oriented service. TLS provides mechanisms for both one- and two-way authentication, dynamic session keying, and data stream privacy and integrity.

This document describes how to use COPS over TLS. "COPS over TLS" is abbreviated COPS/TLS.

2 COPS Over TLS

COPS/TLS is very simple: use COPS over TLS similar to how you would use COPS over TCP (COPS/TCP). Apart from a specific procedure used to initialize the connection, there is no difference between COPS/TLS and COPS/TCP.

3 Separate Ports versus Upward Negotiation

There are two ways in which insecure and secure versions of the same protocol can be run simultaneously.

In the first method, the secure version of the protocol is also allocated a well-known port. This strategy of having well-known port numbers for both, the secure and insecure versions, is known as 'Separate Ports'. The clients requiring security can simply connect to the well-known secure port. This method is easy to implement, with no modifications needed to existing insecure implementations. The disadvantage, however, is that it doesn't scale well, with a new port required for each secure implementation. More problems with this approach have been listed in [RFC2595].

The second method is known as 'Upward Negotiation'. In this method, the secure and insecure versions of the protocol run on the same port. The client connects to the server, both discover each others' capabilities, and start security negotiations if desired. This method usually requires some changes to the protocol being secured.

COPS/TLS uses the Upward Negotiation method to secure COPS messages.

4 COPS/TLS Objects and Error codes

This section describes the COPS objects and error codes needed to support COPS/TLS.

4.1 The Security ClientSI Object

The Security ClientSI object is used by the PDP and the PEP to start the TLS negotiation. This object should be included only in the Client-Open or Client-Accept messages. It MUST NOT be included in any other COPS message.

```
      0         1          2          3
+----------+----------+----------+----------+
|    Length (Octets)  | C-Num=9  | C-Type=2 |
+----------+----------+----------+----------+
|       ////////      |        Flags        |
+----------+----------+----------+----------+
Note: //// implies the field is reserved, set to 0 and should be
      ignored on receipt.
```

Flags: 16 bits
     0x01 = StartTLS
     This flag indicates that the sender of the message wishes to
     initiate a TLS handshake.

The Client-Type of any message containing this Named ClientSI object MUST be 0. Client-Type 0 is used to negotiate COPS connection level security and must only be used during the connection establishment phase. Please refer to section 4.1 of [RFC2748] for more details.

4.2 Error Codes

This section adds to the error codes described in section 2.2.8 (Error Object) of [RFC2748].

Error Code: error-code-TBD-by-IANA = TLS Required

This error code should be used by either PEP or PDP to indicate a security-related connection closure.

5 COPS/TLS Secure Connection Initiation

Security negotiation may be initiated either by the PDP or the PEP. The PEP can initiate a negotiation via a Client-Open message, while a PDP can initiate a negotiation via a Client-Accept message.

Once the TLS connection is established, all COPS data MUST be sent as TLS "application data".

5.1 PEP Initiated Security Negotiation

A PEP MAY initiate security negotiation with a PDP using the Client-Open message. The Client-Open message MUST have a Client-Type of 0 and MUST include the Security ClientSI object.

Upon receiving the Client-Open message, the PDP SHOULD respond with a Client-Accept message containing the Security ClientSI object.

Note that in order to carry the Security ClientSI object, the contents of the Client-Accept message defined in section 3.7 of [RFC2748] need to change to the following:

```
<Client-Accept> ::= <Common Header>
                    <KA Timer>
                    [<ACCT Timer>]
                    [<ClientSI>]
                    [<Integrity>]
```

Upon receiving the appropriate Client-Accept message, the PEP SHOULD initiate the TLS handshake.

The message exchange is as follows:

```
C: Client-Open   (Client-Type = 0, Security)
S: Client-Accept (Client-Type = 0, Security)
<TLS handshake>
C/S: <...further messages...>
```

Instead of sending a Client-Accept message, the PDP may choose to close the connection if it does not wish to open a secure connection with the PEP. It MUST include the error code error-code-TBD-by-IANA in the ensuing Client-Close message.

A PEP expecting the Security ClientSI object in a Client-Accept message MUST close the connection if the ClientSI object is missing. It MUST include the error code error-code-TBD-by-IANA in the ensuing Client-Close message.

5.2 PDP Initiated Security Negotiation

The PEP initially opens a TCP connection with the PDP on the standard COPS port and sends a Client-Open message. This Client-Open message MUST have a Client-Type of 0.

The PDP SHOULD then reply with a Client-Accept message. In order to signal the PEP to start the TLS handshake, the PDP MUST include the Security ClientSI object in the Client-Accept message.

Upon receiving the Client-Accept message with the Security ClientSI object, the PEP SHOULD initiate the TLS handshake. If for any reason the PEP cannot initiate the handshake, it MUST close the connection.

The message exchange is as follows:

```
C: Client-Open   (Client-Type = 0)
S: Client-Accept (Client-Type = 0, Security)
<TLS handshake>
C/S: <...further messages...>
```

Before completion of the TLS handshake, the PEP MUST NOT send any messages other than Client-Close and Keep-Alive. Upon receiving any other message, a PDP expecting a TLS negotiation MUST issue a Client-Close message with an error code of error-code-TBD-by-IANA.

A PDP wishing to negotiate security with a PEP having a non-secure connection MUST send a Client-Close with the error code error-code-TBD-by-IANA and wait for the PEP to reconnect. Upon receiving the Client-Open message, it SHOULD use the Client-Accept message to initiate security negotiation.

6 Connection Closure

TLS provides facilities to securely close its connections. Reception of a valid closure alert assures an implementation that no further data will arrive on that connection. The TLS specification requires TLS implementations to initiate a closure alert exchange before closing a connection. It also permits TLS implementations to close connections without waiting to receive closure alerts from the peer, provided they send their own first. A connection closed in this way is known as an "incomplete close". TLS allows implementations to reuse the session in this case, but COPS/TLS makes no use of this capability.

A connection closed without first sending a closure alert is known as a "premature close". Note that a premature close does not call into question the security of the data already received, but simply indicates that subsequent data might have been truncated. Because TLS is oblivious to COPS message boundaries, it is necessary to examine the COPS data itself (specifically the Message header) to determine whether truncation occurred.

6.1 PEP System Behavior

PEP implementations MUST treat premature closes as errors and any data received as potentially truncated. The COPS protocol allows the PEP system to find out whether truncation took place. A PEP system detecting an incomplete close SHOULD recover gracefully.

PEP systems MUST send a closure alert before closing the connection. PEPs unprepared to receive any more data MAY choose not to wait for the PDP system's closure alert and simply close the connection, thus generating an incomplete close on the PDP side.

6.2 PDP System Behavior

COPS permits a PEP to close the connection at any time, and requires PDPs to recover gracefully. In particular, PDPs SHOULD be prepared to receive an incomplete close from the PEP, since a PEP often shuts down for operational reasons unrelated to the transfer of policy information between the PEP and PDP.

Implementation note: The PDP ordinarily expects to be able to signal end of data by closing the connection. However, the PEP may have already sent the closure alert and dropped the connection.

PDP systems MUST attempt to initiate an exchange of closure alerts with the PEP system before closing the connection. PDP systems MAY close the connection after sending the closure alert, thus generating an incomplete close on the PEP side.

7 Endpoint Identification and Access Control

All PEP implementations of COPS/TLS MUST support an access control mechanism to identify authorized PDPs. This requirement provides a level of assurance that the policy arriving at the PEP is actually valid. PEP deployments SHOULD require the use of this access control mechanism for operation of COPS over TLS. When access control is enabled, the PEP implementation MUST NOT initiate COPS/TLS connections to systems not authorized as PDPs by the access control mechanism.

Similarly, PDP COPS/TLS implementations MUST support an access control mechanism permitting them to restrict their services to authorized PEP systems only. However, deployments MAY choose not to use an access control mechanism at the PDP, as organizations might not consider the types of policy being deployed as sensitive, and therefore do not need to incur the expense of managing credentials for the PEP systems. If access controls are used, however, the PDP implementation MUST terminate COPS/TLS connections from unauthorized PEP systems and log an error if an auditable logging mechanism is present.

Implementations of COPS/TLS MUST use X.509 v3 certificates conforming to [RFC3280] to identify PDP and PEP systems. COPS/TLS systems MUST perform certificate verification processing conforming to [RFC3280].

If a subjectAltName extension of type dNSName or iPAddress is present in the PDP's certificate, it MUST be used as the PDP identity. If both types are present, dNSName SHOULD be used as the

PDP identity. If neither of the types is present, the most specific Common Name field in the Subject field of the certificate SHOULD be used.

Matching is performed using the matching rules specified by [RFC3280]. If more than one identity of a given type is present in the certificate (e.g. more than one dNSName name in the subjectAltName certificate extension), a match in any one of the provided identities is acceptable. Generally, the COPS system uses the first name for matching, except as noted below in the IP address checking requirements.

7.1 PDP Identity

Generally, COPS/TLS requests are generated by the PEP consulting bootstrap policy information that identifies PDPs that the PEP is authorized to connect to. This policy provides the PEP with the hostname or IP address of the PDP. How this bootstrap policy information arrives at the PEP is outside the scope of this document. However, all PEP implementations MUST provide a mechanism to securely deliver or configure the bootstrap policy.

All PEP implementations MUST be able to securely acquire the signing certificates of authorized Certificate Authorities that issue PDP certificates. Also, the PEPs MUST support a mechanism to securely acquire an access control list or filter identifying the CA's set of authorized PDPs.

PEP deployments that participate in multiple domains, such as those on mobile platforms, MAY use different CAs and access control lists in each domain.

If the PDP hostname or IP address is available via the bootstrap policy, the PEP MUST check it against the PDP's identity as presented in the PDP's TLS Certificate message.

In some cases the bootstrap policy will identify the authorized PDP only by an IP address of the PDP system. In this case, the subjectAltName MUST be present in the certificate, and it MUST include an iPAdress format matching the expected name of the policy server.

If the hostname of the PDP does not match the identity in the certificate, a PEP on a user oriented system MUST either notify the user (PEP systems MAY afford the user the opportunity to continue with the connection in any case) or terminate the connection with a bad certificate error. PEPs on unattended systems MUST log the error to an appropriate audit log (if available) and MUST terminate the connection with a bad certificate error. Unattended PEP systems MAY provide a configuration setting that disables this check, but then MUST provide a setting which enables it.

7.2 PEP Identity

When PEP systems are not access controlled, the PDP need have no external knowledge of what the PEP's identity ought to be and so checks are neither possible nor necessary. In this case, there is no requirement for PEP systems to register with a certificate authority, and COPS over TLS uses one-way authentication, of the PDP to the PEP.

When PEP systems are access controlled, PEPs MUST be PKI clients in the sense of [RFC3280]. In this case, COPS over TLS uses two-way authentication, and the PDP MUST perform the same identity checks for the PEPs as described above for the PDP.

When access controls are in effect at the PDP, PDP implementations MUST have a mechanism to securely acquire the signing certificates of the Certificate Authorities issuing certificates to any of the PEPs they support.

8 Backward Compatibility

The PEP and PDP SHOULD be backward compatible with peers that have not been modified to support COPS/TLS. They SHOULD handle errors generated in response to the Security ClientSI object.

9 IANA Considerations

The IANA shall add the following Error-Code to the cops-parameters registry located at http://www.iana.org/assignments/cops-parameters.

Error-Code: error-code-TBD-by-IANA
Description: TLS Required

For the Named ClientSI object for Client-Type 0, the IANA shall add the following Flags value:
0x01 = StartTLS

Further values for the Flags field and the reserved field can only be assigned by IETF Consensus rule as defined in [RFC2434].

10 Security Considerations

A COPS PDP and PEP MUST check the results of the TLS negotiation to see whether an acceptable degree of authentication and privacy have been achieved. If the negotiation has resulted in unacceptable algorithms or key lengths, either side MAY choose to terminate the connection.

A man-in-the-middle attack can be launched by deleting the Security ClientSI object from the Client-Open or Client-Accept messages. To prevent this, the PEP and PDP MUST use the Integrity object as defined in [RFC2748].

11 References

11.1 Normative References

[RFC2026] Bradner, S., "The Internet Standards Process - Revision 3", RFC 2026, October 1996

[RFC2119] Bradner, S., "Key Words for use in RFCs to indicate Requirement Levels", RFC 2119, March 1997.

[RFC2748] Durham, D., Boyle, J., Cohen, R., Herzog, R., Rajan, R., Sastry, A., "The COPS (Common Open Policy Service) Protocol", RFC 2748, January 2000.

[RFC3280] Housley, R., Ford, W., Polk, W., Solo, D., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile ", RFC 3280, April 2002.

[RFC2246] Dierks, T., Allen, C., "The TLS Protocol", RFC 2246, January 1999.

11.2 Informative References

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000.

[RFC2595] Newman, C., "Using TLS with IMAP, POP3 and ACAP", RFC 2595, June 1999.

[RFC3207] Hoffman, P., "SMTP Service Extension for Secure SMTP over Transport Layer Security", RFC 3207, February 2002.

[RFC2434] Alvestrand, H., Narten, T., "Guidelines for writing an IANA Considerations Section in RFCs", BCP 26, RFC 2434, October 1998.

12 Author Addresses

Jesse R. Walker
Intel Corporation
2111 N.E. 25th Avenue
Hillsboro, OR 97214
USA
jesse[dot]walker[at]intel[dot]com

Amol Kulkarni
Intel Corporation
JF3-206
2111 N.E. 25th Avenue
Hillsboro, OR 97214
USA
amol[dot]kulkarni[at]intel[dot]com

13 IPR Disclosure Acknowledgement

14 Disclaimer of Validity

15 Copyright Statement

16 Disclaimer

17 Acknowledgements

This document freely plagiarizes and adapts Eric Rescorla's similar

document [RFC2818] that specifies how HTTP runs over TLS.
Discussions with David Durham, Scott Hahn and Ylian Sainte-Hillaire
also lead to improvements in this document.

The authors wish to thank Uri Blumenthal for doing a thorough security review of the document.

Funding for the RFC Editor function is currently provided by the Internet Society.