TLS
Internet-Draft
Updates: RFC8446, RFC8705, RFC5280, RFC5246,
RFC7590 (if approved)
Intended status: Standards Track
Expires: 18 December 2025

K. Frank

16 June 2025

Allow using serverAuth certificates for mutual TLS (mTLS) authentication in server-to-server usages. draft-frank-mtls-via-serverauth-extension-00

Abstract

This document aims to standardize the validation of mutual TLS authentication between servers (server-to-server). It outlines recommended validation flows as well as provides practical design recommendations. Basically the EKU id-kp-clientAuth and id-kp-serverAuth get more precisely defined to represent their common understanding by issuing CAs and browsers. id-kp-clientAuth aka. "TLS WWW client authentication" SHOULD mean authentication of a natural or legal entity. id-kp-serverAuth aka. "TLS WWW server authetnication" SHOULD mean authentication of a device. When two id-kp-clientAuth certificates are used this means E2E authentication between two users. Where as two id-kp-serverAuth certificates being used means server-to-server authentication. And one user and one server certificate within one TLS connection means client-to-server (or technically also server-to-client). The term "TLS-Client" SHOULD no longer be used and mean the party sending the initial package while establishing a TLS connection. This helps to avoid design issues moving forward as currently some people thought TLS-Client auth was only ever used in "client-to-server" and never within "server-to-server" context. Which sparked the demand for this document to begin with to keep server-to-server auth with public trusted certificates working.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Copyright Notice

Table of Contents

1. Introduction

Recently public CAs have stopped issuing certificates with both clientAuth and serverAuth from the common web root of trust because of a change in the Chrome Root Program Policy. Some CAs still issue client auth certificates but only towards invididuals and none towards domain names (dNSNames). But even their usage is questionable as they're no longer issued by the an signing CA within the same trust hirarchy. Furthermore the policy now also states that going forward all certificates containing both clientAuth and

serverAuth will be considered invalid. Because of this there are now new challanges for server-to-server usages. This document aims to address this by relaxing the validation requirements for TLS Clientauthentication within the server-to-server usecase. In addition this document also provides some standardized validation flows which formerly weren't formally standardized.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119],[RFC8174] when, and only when, they appear in all capitals, as shown here.

The reader should be familiar with the terms defined in [RFC5280] and [RFC8446]

2. Updates to RFC5280: id-kp-serverAuth for mutual Authentication

Section 4.2.1.12 of [RFC5280] is extended by:

NEW:

> "TLS WWW client authentication" SHOULD mean authentication of a natural or legal entity. "TLS WWW server authetnication" SHOULD mean authentication of a device e.g. client-to-server or server-to-server A certificate with either (or both) of these options MAY be used for either side of a TLS-connection. A server expecting only end user authentication SHOULD decline id-kp-serverAuth certificates and one only expecting server-to-server connections SHOULD decline id-kp-clientAuth certificates. The server SHOULD validate the certificate it received from the TLS connection establishing party (TLS-Client) according to draft-frank-mtls-via-serverAuth-extension.

3. Updates to all RFCs using mutual authentication

The following mutual authentication scenarious should be differentiated

3.1. Basic scenario

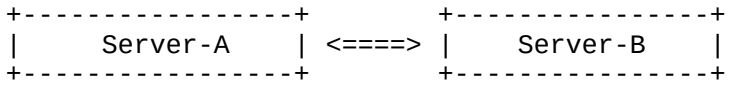

Figure 1: Figure 1: Server-to-server

Figure 1 shows a minimal deployment with two servers.

In the most basic case the authentication is performed between two servers directly. This case can be further split into:

* application to application: In this case there is an application on both servers that performs the authentication and use the tls authentication also for authentication within the application itself. E.g. to grant different access rights.

* server to server: In this scenario the authentication is performed outside of the application, e.g. using a separate application to wrap the connection or by the webserver without forwarding it to the application itself. This can be seen as adding an additional layer of authentication in cases where the application itself may not support authentication or where a 2nd dedicated factor is desired. It is basically similar to a peer-to-peer VPN like IPSec in transport mode.

Where some of these MAY be able to use an internal, private, or enterprise CA this does not apply to all usages. E.g. XMPP servers commonly use mutual TLS but the other parties are not necessarily known beforehand. The same applies to some extend also to SMTP relays between different organizations. The desired level of assurance in these cases is not authentication as in associating a session to a specific user but more generic the same as when a server identifies itself to a end user, to ensure both parties are who they claim to be. e.g. server-a.example.com communicating with server-b.example.com

3.2. Advanced scenario

A slightly more advanced usage is when a 3rd server, like a load-balancer is present within the connection.

Figure 2: Figure 2: Sample Deployment with Load Balancer

Figure 2: An extension of Figure 1 with one more server behind Server-B which serves as load balancer.

In this setup it can be differentiated between these scenarious:

* with application layer authentication: In this case the load balancer uses a mechanism to forward the authentication towards the application. E.g. by passing the connection through as is or by using another standard that is outside of the scope of this document. The goal in this case is almost always to authenticate a user.

* without application layer authentication: This usage is basically identical to the basic scenario above and the goal is most of the time to add an additional layer of authetnication. The load-balancer will terminate the TLS connection and establish a separate connection with the backend server. The backend server does not get informed about the presence or state of the mutually authenticated TLS connection at all. This is similar to using IPSec or an HTTPS-proxy.

* with mTLS only between load-balancer and backend server: mTLS is used between the load-balancer and the backend. This is commonly used within e.g. a service mesh or where the backend may be spread accross multiple different cloud providers. It also allows a backend server to reject all connections that fail mTLS authentication (e.g. are from either another system within the service mesh, nor from the load-balancer). It also serves as a simpler and more scalable alternative to IP allow lists.

* with mTLS only between requestor and load-balancer: In this case the backend server is either only unidirectionally authenticated or may not be authenticated at all. Also the laod-balancer MAY forward information of the client certificate to the application or not. Depending on this it is either the same as the first or the second case above. Such a configuration is commonly used to secure an otherwise insecure application. In such deployments the backend server is commonly isolated and only exposed through a dedicated transfer network to the loadbalancer without any other systems within that network. Even if the application requires an outbound internet connection this almost always allows for selectively binding the insecure listening socket of the application towards only the transfer network and not towards the uplink network used for outbound connections. Depending on the application the load-balancer (or reverse proxy) MAY be installed on the same system as the insecure application. In these cases the transfer network MAY be either a dedicated network namespace, a unix domain socket, or a dummy network interface. Usage of the loopback interface with "127.0.0.1" and "::1" SHOULD be avoided as many legacy applications have special handling for them and therefore MAY expose more control than expected.

3.3. Validation of a certificates with only EKU serverAuth received by a server from the connection establishing party.

The validation of the TLS certificate received by the connection initiating party is already sufficiently specified within the original RFCs. However the validation of the certificate received from the client by the server is not. Because this document focuses on the server-to-server usage it is possible to specify the validation criteria in more detail:

1. If the received certificate specifies the Extended Key Usage of "clientAuth" any generic application like e.g. a web server SHOULD assume this meaning end user authentication. Without additional configuration it SHOULD reject any otherwise valid certificate that is not issued towards either an individual or organization. Aka. certificates with an FQDN or IP-Address within the CN or SAN section. A more specialized application can decide for itself if it wants to accept either a certificate for a system or an enduser.

2. If the received certificate is otherwise valid, has Extended Key Usage "serverAuth" and is issued towards an entity of type dNSName the receiving server SHOULD perform a forward-confirmed reverse DNS lookup. This discovered FQDN is then used for validating the certificate as if it was a certificate received for an outbound conenction towards that FQDN.

3. If the received certificate is otherwise valid, has Extended Key Usage "serverAuth" and is issued towards an entity of type ipAddress the receiving server MUST use the source IP-Address of the connection to validate the certificate against.

4. A generic application MUST NOT pass authentications using certificates of kind "serverAuth" towards applications. It SHOULD pass authentication for "clientAuth". It MAY allow for a configurable overwrite to be configured to also pass certificates of "serverAuth" towards to the application.

5. Applications SHOULD provide a configuration option to reject all certificates with both clientAuth and serverAuth.

6. A certificate that is issued towards either dNSName or iPAddress as well as any other type is invalid

7. A server performing a forward-confirmed reverse DNS lookup SHOULD also query the relevant CAA, TLSA, SVCB, SRV, CERT, and DNSSEC records. Supporting applications SHOULD check their application specific records as well (e.g. HTTPS, SSHFP, ...).

8. In case the certificate is issued towards an entity of type ipAddress the server SHOULD query the reverse lookup zone for these DNS records. It MAY also try to perform a forward-confirmed reverse DNS lookup but it failing because of missing PTR/A/AAAA-RRs SHOULD not fail the entire validation.

9. In case DNSSEC validation fails the forward-confirmed reverse DNS lookup fails. Therefore the certificate validation fails as well.

10. If for a specific DNS zone DNSSEC is not configured at all the validation SHOULD NOT fail. An application however MAY specify a more strict requirement depending on its security requirements. Howver it then MUST allow for a configurable overwrite.

11. If a TLSA record is received it’s certificate usage value MUST be honored. Except the zone is not DNSSEC signed then only certificate usage 0 and 1 are honored. When the zone is signature certificate usage 2 and 3 also MUST be honored. See [RFC7671], [RFC7672], and [RFC7673].

12. For Zeroconf ([RFC6762] and [RFC6763]) domains the DNSSEC requirements MUST be ignored.

13. A configuration option listing domains for which DNSSEC is not evaluated MUST be provided. The default list SHOULD include ".local", ".localhost", and ".onion". It MAY be disabled by default though.

14. Applications SHOULD provide a configuration option to specify a separate dns resolver to be used for only these specific domains and MAY provide a configuration option to use for all others instead of using the DNS resolver of the system.

15. A SRV/SVCB/... record containing the source port SHOULD be considered while evaluating. It’s absence is not an error (except when it should be present as determined by DNSSEC). It being present but it not validating doesn’t cause the validation to fail either. However it CAN be used while retrieving the TLSA-RR or for the forward-confirmed reverse DNS lookup. See [RFC9460] and the IANA "Underscored and Globally Scoped DNS Node Names" registry from [RFC8552]. e.g. "_8443._tcp.1.2.0.192.in-addr.arpa" or "_8443._tcp.example.com" when "1.2.0.192.in-addr.arpa" is pointing towards "example.com".An application specific to some protocol like e.g. XMPP SHOULD check the associated SRV-RR but MAY also check other less specific ones if applicable. It should be noted that [RFC6762] for SRV records

specifies "Port numbers SHOULD NOT be used in place of the symbolic service or protocol names". Therefore even though unlikely it MAY still be used in some rare cases. Some MAY use a SRV record with port numbers for reverse lookups instead of a NAPTR record.

16. Except for the above described extensible reverse lookup and validation of the received mapping the certificate validation is exactly the same as for any certificate the system would have received as a TLS-client for an outbound connection by a server. The retrieved domain name is used as a replacement for the desired remote target server name.

17. Applications MAY ONLY validate the source IP and hostname. However applications SHOULD at least offer the additional validation mechanisms outlined above to validate the source port and service as well. This provides additional security against protocol confusion and binds certificates (e.g. when the TLSA record is specified for the SVCB/SRV record) not just to the host but to the source port and service as well.

4. Contributors and Acknowledgements

A special thanks goes to everyone participating in the discussion on the mailing lists as well as Mohamed Boucadair for proofreading, suggested changes, and helping with the submission process itself.

5. References

5.1. Normative References

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <https://www.rfc-editor.org/info/rfc5246>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/info/rfc5280>.

[RFC6762] Cheshire, S. and M. Krochmal, "Multicast DNS", RFC 6762, DOI 10.17487/RFC6762, February 2013, <https://www.rfc-editor.org/info/rfc6762>.

[RFC6763] Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", RFC 6763, DOI 10.17487/RFC6763, February 2013, <https://www.rfc-editor.org/info/rfc6763>.

[RFC7590] Saint-Andre, P. and T. Alkemade, "Use of Transport Layer Security (TLS) in the Extensible Messaging and Presence Protocol (XMPP)", RFC 7590, DOI 10.17487/RFC7590, June 2015, <https://www.rfc-editor.org/info/rfc7590>.

[RFC7671] Dukhovni, V. and W. Hardaker, "The DNS-Based Authentication of Named Entities (DANE) Protocol: Updates and Operational Guidance", RFC 7671, DOI 10.17487/RFC7671, October 2015, <https://www.rfc-editor.org/info/rfc7671>.

[RFC7672] Dukhovni, V. and W. Hardaker, "SMTP Security via Opportunistic DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS)", RFC 7672, DOI 10.17487/RFC7672, October 2015, <https://www.rfc-editor.org/info/rfc7672>.

[RFC7673] Finch, T., Miller, M., and P. Saint-Andre, "Using DNS-Based Authentication of Named Entities (DANE) TLSA Records with SRV Records", RFC 7673, DOI 10.17487/RFC7673, October 2015, <https://www.rfc-editor.org/info/rfc7673>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC8552] Crocker, D., "Scoped Interpretation of DNS Resource Records through "Underscored" Naming of Attribute Leaves", BCP 222, RFC 8552, DOI 10.17487/RFC8552, March 2019, <https://www.rfc-editor.org/info/rfc8552>.

[RFC8705] Campbell, B., Bradley, J., Sakimura, N., and T. Lodderstedt, "OAuth 2.0 Mutual-TLS Client Authentication and Certificate-Bound Access Tokens", RFC 8705, DOI 10.17487/RFC8705, February 2020, <https://www.rfc-editor.org/info/rfc8705>.

[RFC9460] Schwartz, B., Bishop, M., and E. Nygren, "Service Binding and Parameter Specification via the DNS (SVCB and HTTPS Resource Records)", RFC 9460, DOI 10.17487/RFC9460, November 2023, <https://www.rfc-editor.org/info/rfc9460>.

Author’s Address

Klaus Frank

Email: draft-frank-mtls-via-serverauth-extension@frank.fyi