Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 26, 2018

R. Shoemaker
ISRG

February 22, 2018

ACME TLS ALPN Challenge Extension
draft-shoemaker-acme-tls-alpn-00

Abstract

This document specifies a new challenge for the Automated Certificate Management Environment (ACME) protocol which allows for domain control validation using TLS.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The Automatic Certificate Management Environment (ACME) [I-D.ietf-acme-acme] specification doesn't specify a TLS layer validation method which limits the points at which validation can be performed. This document extends the ACME specification to include a TLS based validation method that uses the Application Level Protocol Negotiation extension.

2. Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in BCP 14, RFC 2119 [RFC2119].

3. TLS with Application Level Protocol Negotiation (TLS ALPN) Challenge

The TLS with Application Level Protocol Negotiation (TLS ALPN) validation method proves control over a domain name by requiring the client to configure a TLS server referenced by the DNS A and/or AAAA Resource Records for the domain name to respond to specific connection attempts utilizing the ALPN extension [RFC7301]. The server validates control of the domain name by connecting to the TLS server and verifying a certificate with specific content is presented.

type (required, string): The string "tls-alpn-01"

token (required, string): A random value that uniquely identifies the challenge. This value MUST have at least 128 bits of entropy. It MUST NOT contain any characters outside the base64url alphabet, including padding characters ("=").

```
GET /acme/authz/1234/1 HTTP/1.1
Host: example.com

HTTP/1.1 200 OK
{
  "type": "tls-alpn-01",
  "url": "https://example.com/acme/authz/1234/1",
  "status": "pending",
  "token": "evaGxfADs6pSRb2LAv9IZf17Dt3juxGJ-PCt92wr-oA"
}
```

The client prepares for validation by constructing a self-signed certificate which MUST contain a acmeValidation-v1 extension and a subjectAlternativeName extension [RFC5280]. The subjectAlternativeName extension MUST contain a single dNSName entry where the value is the domain name being validated. The acmeValidation-v1 extension MUST contain the SHA-256 digest [FIPS180-4] of the key authorization [I-D.ietf-acme-acme] for the challenge. The acmeValidation extension MUST be critical so that the certificate isn't inadvertently used to make trust decisions.

```
id-pe-acmeIdentifier OBJECT IDENTIFIER ::=  { id-pe 30 }

id-pe-acmeIdentifier-v1 OBJECT IDENTIFIER ::=  { id-pe-acmeIdentifier 1 }

acmeValidation-v1 ::= OCTET STRING (SIZE (32))
```

Once this certificate has been created it MUST be provisioned such that it is returned during a TLS handshake that contains a ALPN extension containing the value "acme-tls/1" and a SNI extension containing the domain name being validated.

When ready the client acknowledges this by sending a POST message containing the key authorization, as defined in [I-D.ietf-acme-acme] section 8.1, to the challenge URL.

keyAuthorization (required, string): The key authorization for this challenge. This value MUST match the token from the challenge and the client's account key.

```
POST /acme/authz/1234/1
Host: example.com
Content-Type: application/jose+json

{
  "protected": base64url({
    "alg": "ES256",
    "kid": "https://example.com/acme/acct/1",
    "nonce": "JHb54aT_KTXBWQOzGYkt9A",
    "url": "https://example.com/acme/authz/1234/1"
  }),
  "payload": base64url({
    "keyAuthorization": "evaGxfADs...62jcerQ"
  }),
  "signature": "Q1bURgJoEslbD1c5...3pYdSMLio57mQNN4"
}
```

On receiving this the server MUST verify that the key authorization in the request matches the "token" value in the challenge and the client's account key. If they do not match then the server MUST return a HTTP error in response to the POST request in which the client sent the challenge.

The server then verifies the client's control over the domain by verifying that the TLS server was configured as expected using these steps:

1. Compute the expected SHA-256 [FIPS180-4] digest of the expected key authorization.

2. Initiate a TLS connection with the domain name being validated, this connection MUST be sent to TCP port 443. The ClientHello that initiates the handshake MUST contain a ALPN extension with the value "acme-tls/1" and a Server Name Indication [RFC6066] extension containing the domain name being validated.

3. Verify that the ServerHello contains a ALPN extension containing the value "acme-tls/1" and that the certificate returned contains a subjectAltName extension containing the dNSName being validated and no other entries and a critical acmeValidation extension containing the digest computed in step 1. The comparison of dNSNames MUST be case insensitive [RFC4343]. Note that as ACME doesn't support Unicode identifiers all dNSNames MUST be encoded using the [RFC3492] rules.

If all of the above steps succeed then the validation is successful, otherwise it fails. Once the handshake has been completed the

connection should be immediately closed and no further data should be exchanged.

3.1. acme-tls/1 Protocol Definition

The "acme-tls/1" protocol MUST only be used for validating ACME tls-alpn-01 challenges. The protocol consists of a TLS handshake in which the required validation information is transmitted. Once the handshake is complete the client MUST not exchange any further data with the server and MUST immediately close the connection.

4. Security Considerations

The design of this challenges relies on some assumptions centered around how a server behaves during validation.

The first assumption is that when a server is being used to serve content for multiple DNS names from a single IP address that it properly segregates control of those names to the users on the server that own them. This means that if User A registers Host A and User B registers Host B the server should not allow a TLS request using a SNI value for Host A that only User A should be able to serve that request. If the server allows User B to serve this request it allows them to illegitimately validate control of Host A to the ACME server.

The second assumption is that a server will not blindly agree to use the acme-tls/1 protocol without actually knowing about the protocol itself, which is a violation of [RFC7301].

5. IANA Considerations

5.1. SMI Security for PKIX Certificate Extension OID

Within the SMI-numbers registry, the "SMI Security for PKIX Certificate Extension (1.3.6.1.5.5.7.1)" table is to be updated to include the following entry:

Decimal	Description	References
30	id-pe-acmeIdentifier	RFC XXXX

5.2. ACME Validation Method

The "ACME Validation Methods" registry is to be updated to include the following entry:

Label	Identifier Type	Reference
tls-alpn-01	dns	RFC XXXX

6. Appendix: Design Rationale

The TLS ALPN challenge exists to replace the TLS SNI challenge defined in the original ACME document. This challenge allowed validation of domain control purely within the TLS layer which provided convenience for server operators who were either operating large TLS layer load balancing systems at which they wanted to perform validation or running servers fronting large numbers of DNS names from a single host.

A security issue was discovered in the TLS SNI challenge which allowed users of certain service providers to illegitimately validate control of the DNS names of other users, as long as those users were also using those service providers. When the TLS SNI challenge was designed it was assumed that a user would only be able to claim TLS traffic via SNI for domain names they controlled (i.e. if User A registered Host A with a service provider they wouldn't be able to claim SNI traffic for Host B). This turns out not to be a security property provided by a number of large service providers. Because of this users were able to claim SNI traffic for the non-valid SNI names the TLS SNI challenge used to signal what was being validated to the server. This meant that if User A and User B had registered Host A and Host B respectively User A would be able to claim the SNI name for a validation for Host B and when the validation connection was made to the shared IP address that User A would be able to answer, proving control.

7. Normative References

[FIPS180-4]
Department of Commerce, National., "NIST FIPS 180-4, Secure Hash Standard", March 2012, <http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf>.

[I-D.ietf-acme-acme]
Barnes, R., Hoffman-Andrews, J., McCarney, D., and J. Kasten, "Automatic Certificate Management Environment (ACME)", draft-ietf-acme-acme-09 (work in progress), December 2017.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3492] Costello, A., "Punycode: A Bootstring encoding of Unicode for Internationalized Domain Names in Applications (IDNA)", RFC 3492, DOI 10.17487/RFC3492, March 2003, <https://www.rfc-editor.org/info/rfc3492>.

[RFC4343] Eastlake 3rd, D., "Domain Name System (DNS) Case Insensitivity Clarification", RFC 4343, DOI 10.17487/RFC4343, January 2006, <https://www.rfc-editor.org/info/rfc4343>.

[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>.

[RFC6066] Eastlake 3rd, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, DOI 10.17487/RFC6066, January 2011, <https://www.rfc-editor.org/info/rfc6066>.

[RFC7301] Friedl, S., Popov, A., Langley, A., and E. Stephan, "Transport Layer Security (TLS) Application-Layer Protocol Negotiation Extension", RFC 7301, DOI 10.17487/RFC7301, July 2014, <https://www.rfc-editor.org/info/rfc7301>.

Author's Address

Roland Bracewell Shoemaker
Internet Security Research Group

Email: roland@letsencrypt.org