Protocols for issuing and managing digital certificates

Max Crone
max.crone@aalto.fi

Tutor: Mohit Sethi

Abstract

Digital certificates have become an essential part of secure communication. Besides the internet, certificates are used in industrial and Internet of Things settings. The increase in usage of certificates calls for effective ways to manage them.

This paper overviews three standardized certificate management protocols and discusses the extent of their deployment and use. In addition, the paper addresses extensions to these protocols and their role in the Internet of Things.

KEYWORDS: Certificate, PKI, Protocol, ACME, EST, CMP

1 Introduction

In recent years, the usage of digital certificates for establishing trust between communication parties has significantly increased. Certificates are used by a variety of different protocols. For example, protocols such as Transport Layer Security (TLS) and Internet Protocol Security (IPsec) use certificates to establish a secure communication between endpoints. Pretty Good Privacy (PGP) uses certificates to provide security for email

and data storage. The most common usage of certificates is on servers, but there are also cases where they are used for clients, such as in network access authentication with the Extensible Authentication Protocol (EAP) using TLS [1].

Secure communication using digital certificates requires a Public Key Infrastructure (PKI). The PKI serves as a system for enabling the use of digital certificates on a global, internet-wide scale. In PKI, Certificate Authorities (CA) are the entities that digitally sign and publish certificates. The trust in a communication party thus relies on the trust in the CA that has verified them. In contrast, enterprises may also operate a private CA that is not part of the PKI. Instead, such a private CA only issues and stores certificates for parties within the enterprise.

The operation of a CA is complex, which can lead to mis-issuance of certificates, with broken security as a consequence. Such incidents have occurred in the past. In 2011, DigiNotar mis-issued a certificate that was used in an attack against users in Iran [2]. TURKTRUST mistakenly issued two intermediate CA certificates instead of regular SSL certificates in 2011, which were misused to impersonate Google [3].

A solution to these security failures is the automation of certificate management processes. Standardized and automated certificate issuance and management processes are less likely to fail than ad-hoc, humanly operated processes. This paper overviews three important protocols for certificate management and compares them against each other.

The paper is organized as follows. Section 2 summarizes the three protocols evaluated in this paper. Section 3 lists the requirements of each protocol and discusses their implementation status in popular software libraries. Section 4 reviews protocol extensions. Section 5 discusses the challenges when applying the protocols on resource-constrained devices that are part of the Internet of Things (IoT). Finally, Section 6 presents the conclusion of this paper.

2 Protocols

There are many stages in the lifecyle of a certificate. These include key generation, certificate request, certificate issuance, key update, and certificate revocation. The research community has developed several certificate management protocols. For example, Wasef et al. propose a scheme for certificate management in vehicular networks [4]. Caballero-Gil and

Hernández-Goya propose a mechanism for certificate management in mobile ad-hoc networks [5]. However, this paper focuses only on standardized protocols. In particular, this section summarizes three important protocols for certificate issuance and management that have been standardized at the Internet Engineering Task Force (IETF).

2.1 CMP

The Certificate Management Protocol (CMP) [6], specifies messages and interactions between PKI components for certificate management and creation. CMP intends to provide a comprehensive set of certificate management operations that can be carried out online. The groups of operations include CA initialization, end entity initialization, certification, certificate discovery, recovery, and revocation. Initialization operations of PKI entities include many non-recurring actions such as export and import of CA root certificates, and generation of an initial Certificate Revocation List (CRL).

RFC 6712 proposes an HTTP transfer protocol for CMP [7], which is needed to enable message passing between all parties.

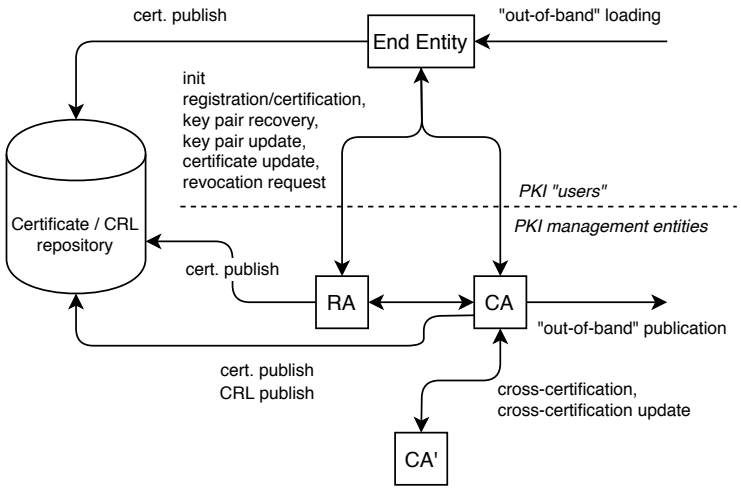
CMP distinguishes three parties involved in PKI management, which are shown in Figure 1. End entities are the subjects to whom certificates are issued. The Certification Authority (CA) is the party that issues the certificates. The protocol makes a further distinction between a root CA, which is a CA that is directly trusted by an end entity, and a subordinate CA. Lastly, CMP mentions a Registration Authority (RA). The RA may carry out functions that would otherwise be provided by the CA, including personal authentication, revocation reporting, token distribution, and key generation. The specific set of functions provided by the RA, if any at all, depends on the context.

An end entity who wants to obtain a certificate utilizing CMP has to take the following steps. First, they must initialize themselves by acquiring PKI information and verifying one root-CA public key via an out-of-band channel. The end entity then sends a certificate request as a *PKIMessage*. The *PKIBody* specifies the requested certificate in the *CertReqMessages* data structure, the syntax of which is specified in Certificate Request Message Format (CRMF) [8]. The CA responds with a message containing the *CertRepMessage* data structure, with relevant information alongside the certificates wrapped in multiple envelopes.

Initial certificate enrollment is only one of the available operations. Fig-

ure 1 shows a comprehensive overview of the messages and interactions defined by CMP.

Figure 1. PKI entities and their interactions in CMP. Adapted from [6].



2.2 EST

Cryptographic Message Syntax (CMS) is used to digitally sign, digest, authenticate, and encrypt message contents [9]. Certificate Management over CMS (CMC) uses CMS to specify interactions between PKI entities as requests and responses pertaining to certification services [10]. Enrollment over Secure Transport (EST) [11] describes a profile of a certificate enrollment client using CMC. EST describes the use of TLS and HTTP in order to establish an authenticated channel for Simple PKI Requests and Responses. EST only addresses certificate provisioning. For additional management operations, the document refers to the CMC standard.

The main use case of the EST protocol is automated certificate provisioning for digital devices in industrial contexts. For example, Cisco supports EST in their Cisco IOS and IOS XE operating systems for network infrastructure devices [12]. The networking devices use the provisioned certificates to secure communication among them.

EST is a successor of the Simple Certificate Enrolment Protocol (SCEP) [13] originally developed by Cisco. While protocol standardization was never completed for SCEP, the EST protocol has been fully standardized at the IETF.

EST distinguishes three different parties in its process. Firstly, the EST client runs on a computer system that wants to obtain a valid X.509 certificate. Secondly, the EST server responds to requests from the EST client and issues certificates. Lastly, EST requires a Certificate Author-

ity. A CA issues the EST server certificate that a client uses to validate the server. Ultimately, the end-entity certificate obtained by the client will also be signed by the CA. Thus, the EST server only acts as a service enabler, not as the Certificate Authority itself.

The EST client and server interact as follows in the scenario of a simple enrollment request. Initially, the EST client opens a TLS session to the EST server. The EST client authenticates the identity of the server by verifying the received EST CA certificate using local Trust Anchor (TA) databases. After authentication, the client makes a `/simpleenroll` request to the server. The request is a Simple PKI Request as defined in CMC [10]. The EST server identifies and authenticates the client using any of three methods: certificate TLS authentication, authentication using a shared key, or authentication with username and password. After successful authentication and authorization of the client, the server acts upon the request by issuing and returning a certificate.

2.3 ACME

Automated Certificate Management Environment (ACME) [14] specifies a protocol for automating interactions between a CA and their users' web servers. ACME allows web servers to prove their ownership of a domain name to a CA, after which they can automatically request and renew TLS certificates. Web servers use these certificates to securely serve websites over HTTPS to their visitors. Since 2016, when Let's Encrypt first introduced ACME and started offering free certificates, the usage of HTTPS has increased significantly [15].

The two main parties involved in ACME are the client and the server. The ACME client runs on a server system that wishes to obtain a valid public key certificate in X.509 format [16]. The ACME server runs within a Certification Authority and handles requests from the ACME clients. Clients must authenticate themselves to the server using an account key pair.

The ACME client must be authorized to respond to a domain validation challenge provided by the ACME server hosted by a CA. Thus the client may also run on a separate server that does not consume the certificate, but is solely meant for managing it.

ACME is built with RESTful interactions in mind. Therefore, the ACME client only needs a single URL that points to the directory resource of the ACME server.

Obtaining a certificate as the client using ACME consists of five phases. Firstly, the client requests an account with an ACME server. Then, they submit an order for a certificate, describing the desired identifiers. An identifier in the context of ACME most often refers to a domain name. The ACME server desires proof of control over the claimed identifiers, which the client must provide. Two examples of such challenges are the HTTP challenge and the DNS challenge. In the HTTP challenge, the client proves ownership by provisioning HTTP resources on a server accessible under the claimed domain name. In the DNS challenge, the client provisions a TXT resource record containing a specific value for a specific claimed domain name. After successfully passing the challenges, the ACME client submits a public key cryptography Certificate Signing Request (CSR) [17] to finalize the order. Ultimately, they can then download and install the server issued certificate.

The ACME client and server exchange JavaScript Object Notation (JSON) messages over HTTPS for all certificate management actions. All messages are protected with JSON Web Signatures (JWS) that provide authentication of the client's payloads, replay protection, and integrity for the HTTPS request URL.

3 Comparison

The following sections compare the discussed certificate management protocols in their defined functionality and investigate existing software implementations and deployment.

The comparison of functionality may convey a skewed perception, because CMP intentionally defines a more comprehensive set of management operations than ACME and EST. However, note the intended difference between the protocols as portrayed in Table 1.

Certificate provisioning	**Certificate management**
ACME	CMP
EST	

Table 1. Protocol classes

3.1 Functionality

This section compares the distinct functionalities of all certificate management protocols introduced in this paper. Table 2 presents an overview.

Function	**ACME**	**EST**	**CMP**
Certificate request	✓	✓	✓
Domain validation	✓		
Certificate update	✓	✓	✓
Certificate revocation	✓		✓
Key update	✓	✓	✓
Key recovery			✓
Server key generation		✓	
Cross-certification			✓
CRL announcement			✓
CA key update			✓

Table 2. Functionality comparison between certificate management protocols.

3.2 Deployment & use

This section discusses and compares implementations of the certificate management protocols discussed in this paper. Additionally, this section discusses the extent to which the protocols are used in industry.

CMP

CMP is widely supported by cryptographic libraries. *Bouncy Castle*[1] provides generators and processors for CMP and CRMF. *cryptlib*[2] allows to issue and revoke certificates using CMP and provides a complete CMP server implementation. Lastly, development of CMP support in the *OpenSSL* library[3] is estimated to finish in April 2020.

The CMP specification is relatively verbose, which makes full compliance by software implementations resource-intensive. In order to improve the practicality of implementations, there exist profiles of CMP. A CMP profile captures a subset of the full CMP specification with the intention of using it in specific application domains. As an example, the IETF is currently specifying a lightweight CMP profile meant for managing cer-

[1] `https://bouncycastle.org/`
[2] `https://www.cs.auckland.ac.nz/~pgut001/cryptlib/`
[3] `https://github.com/mpeylo/cmpossl`

tificates in industrial and IoT scenarios [18]. In order to facilitate interoperability, it only specifies the most crucial operations as mandatory, while all other operations defined in CMP are specified as optional. As another example, the 3GPP organization defines a CMP profile for certificate enrollment of base stations in mobile cellular systems [19].

The open source Certificate Authority software *EJBCA* supports both the full CMP standard and the 3GPP profile [4].

EST

EST is implemented by several software libraries. Cisco developed a reference implementation of EST [5]. Thales eSecurity provides an EST client SDK in Go [6]. Besides these dedicated software implementations, EST is also supported by the cryptographic library *Bouncy Castle*.

Multiple organizations adopt EST in their products or services. Cisco provides EST support for its network infrastructure devices such as enterprise routers and switches [12]. Thales eSecurity leverages EST internally in the registration flow of their secure cloud services [20]. The open source CA software *EJBCA* provides support for the EST protocol [7]. Another CA software vendor, Nexus, also supports EST in their *Certificate Manager* [8].

ACME

The organization Let's Encrypt pioneered the automated and free issuance of TLS certificates [21]. The initial version of ACME was an internal protocol developed for the organization's purpose. This initial protocol was the basis for the final ACME IETF standard. Since they started operation, Let's Encrypt has issued over a billion certificates using ACME [9].

Before the existence of ACME and Let's Encrypt, many certificate authorities were already operational in the issuance of TLS certificates. Upon standardization of ACME, many CAs implemented the protocol in their products and services. Some companies use the ACME protocol as part of a certificate management product with additional features. For exam-

[4] https://doc.primekey.com/ejbca/ejbca-operations/ejbca-ca-concept-guide/protocols/cmp

[5] https://github.com/cisco/libest

[6] https://github.com/thales-e-security/estclient

[7] https://doc.primekey.com/ejbca/ejbca-operations/ejbca-ca-concept-guide/protocols/est

[8] https://doc.nexusgroup.com/display/PUB/Supported+certificate+enrollment+protocols+in+CM

[9] https://letsencrypt.org/2020/02/27/one-billion-certs.html

ple, Sectigo (formerly Comodo) supports ACME in their PKI management platform *Certificate Manager*[10], in addition to other certificate management protocols and extensive lifecyle control features. Table 3 lists certificate authorities that support ACME for automatic issuance of TLS certificates.

Authority	**Notes**
DigiCert	ACME is part of *CertCentral* [11]
GlobalSign	ACME is part of *Auto Enrollment Gateway* [12]
Let's Encrypt	Exclusively issues certificates via ACME [13]
Sectigo	ACME is part of *Certificate Manager* [14]
WISeKey	Support for ACME using EJBCA [15]

Table 3. Overview of Certificate Authorities that support ACME.

Besides support in managed services of already established certificate authorities, ACME is also implemented by many CA server software products intended for self-hosted deployment and enterprise operation. Examples include *Boulder*[16] (which is the server software that runs Let's Encrypt), *Dogtag Certificate System*[17], *EJBCA*[18], *Nexus Certificate Manager*[19], and *Smallstep*[20].

Lastly, ACME is supported by many client software options. Let's Encrypt recommends the *Certbot* client, which is developed by the Electronic Frontier Foundation (EFF) [21]. Besides *Certbot*, Let's Encrypt lists other client options written in many different languages [22]. Thus, most platforms and environments should be suitable to operate an ACME client for automated certificate issuance and renewal.

4 Extensions

This section explores extensions to the certificate management protocols discussed in Section 2. All the following extensions are published by the

[10] `https://sectigo.com/enterprise/sectigo-certificate-manager`
[16] `https://github.com/letsencrypt/boulder`
[17] `https://www.dogtagpki.org/wiki/PKI_Main_Page`
[18] `https://doc.primekey.com/ejbca/ejbca-operations/ejbca-ca-concept-guide/protocols/acme`
[19] `https://doc.nexusgroup.com/display/PUB/ACME+support+in+CM`
[20] `https://github.com/smallstep/certificates`
[21] `https://github.com/certbot/certbot`
[22] `https://letsencrypt.org/docs/client-options/`

IETF, in similar fashion to the original protocols. Many are found by reading through the list of documents that reference the respective protocol.

The number of extensions for CMP is relatively low, compared to the other two protocols. In 2012, the IETF published the aforementioned HTTP transfer document that specifies how to layer CMP over HTTP [7]. This extension came seven years after initial publication of the CMP standard. Currently, a working group from the IETF is developing a set of updates for CMP [22]. The updates allow implementers to more easily use alternative cryptographic algorithms in the future, in case current options are proven insecure. In addition, the updates include the introduction of extended key usages to identify CMP endpoints on CA and RA. While CMP has few extensions, it has many profiles of its specification. The CMP standard is meant to be comprehensive. Thus instead of extending the specification when additional functionality is required, an implementing party may profile CMP for their more specific use case.

The first update to EST defines new CSR attributes [23]. These attributes provide alternatives to the *challengePassword* attribute. Implementers often interpreted the *challengePassword*'s semantics differently, which caused ambiguity between implementations. The update affects the original certificate revocation password, common authentication passwords, and EST-defined linking of transport security identity. In 2018, the IETF published a new set of extensions for EST [24]. The document defines additional PKI services as path components for EST. Among these is the Package Availability List (PAL). The PAL is a resource provided by the server, indexing the actions made available to a client. With the PAL, the server is able to dynamically communicate the list of available actions to a client. For example, the PAL might contain a package which indicates that a new CRL is available for the client by pointing to an applicable URI. Currently, an IETF working group is developing a new transport for EST messages [25]. The EST protocol originally specified transport over HTTP, in which the messages can become relatively large. To support EST on resource constrained devices, this document defines EST transport based on the Constrained Application Protocol (CoAP) rather than on HTTP. In addition, the document profiles the use of EST to solely support certificate-based client authentication.

Multiple standardized extensions for ACME already exist and several others are in progress. A recently published extension provides a new challenge for ACME that enables domain control validation using TLS

[26]. Another recent extension specifies identifiers and challenges for ACME to enable issuance of certificates for IP addresses [27]. Most recently, the IETF released support for Short-Term, Automatically Renewed (STAR) certificates in ACME [28]. ACME support for STAR certificates is posed as an alternative to certificate revocation. In case of a compromise of the private key, the owner terminates the issuance sequence of short-term certificates.

Beside these already standardized extensions to ACME, the ACME working group and several individuals are working on drafts for new extensions. Among these extensions are additional challenges for ACME using an Authority Token [29], identifiers and challenges to enable issuance of end user S/MIME certificates [30], issuance of certificates for subdomains without requiring an explicit ownership challenge [31], and ACME support for end user client, device client, and code signing certificates [32].

5 Challenges with IoT

This section discusses challenges for certificate management in the IoT and what is being done to prepare the current protocols for this new context of execution.

Digital certificates are valuable in securing the Internet of Things. Certificates allow for authentication of devices across networks and without pair-wise configuration between any two nodes [33]. However, certificates and certificate management on IoT devices bring additional challenges with them. Certificates are relatively large, identifiers are not suitable for large numbers of IoT devices and validation of long certificate chains may be outside the processing capabilities of the often resource-constrained devices that make up the IoT [34]. Moreover, as a result of the scale of deployment of IoT devices, manual certificate management must be avoided in favor of automated procedures.

Certificate management protocols discussed in this paper may be the solution for the desired automation of IoT certificate management. However, these protocols are often not directly applicable, because of the resource-constrained nature of devices in the IoT. Therefore, researchers in industry and academia work on suitable solutions for IoT, while building upon already standardized protocols. The rest of this section explores how CMP, EST and ACME will be used in the IoT.

The complete CMP standard is unnecessarily complex for use in IoT, In-

stead, the lightweight CMP profile mentioned in Section 3.2 focuses on managing certificates of devices in IoT and industrial contexts.

The EST protocol operates on HTTP. HTTP messages are relatively large from the perspective of IoT devices, and alternative standards have already been developed for use on constrained devices [35][36]. Currently, a suite of low-overhead networking protocols is used to define public key certificate enrollment procedures based on EST [37]. The new protocol will be suitable for IoT deployments.

ACME might not be as relevant in the IoT as CMP or EST. The actors in ACME are certificate authorities and web servers, both of which are not meant to run on resource-constrained devices. Devices in the IoT require a different type of certificate than the TLS certificates provided by ACME. However, there still exist ACME clients targeting IoT hardware. For example, the *ESP32 ACME client*[23] implements the ACME protocol for the ESP32 series of microcontrollers.

6 Conclusion

This paper presented three standardized certificate management protocols; CMP, EST, and ACME. Section 3 compared their functionality and discussed the extent of their deployment and use in practice. It has become clear that all three protocols serve distinct purposes, even though they all relate to the management of digital certificates. The paper continued by addressing existing extensions and future extensions, which indicated the directions these standards would move in. CMP and EST are increasingly being used and profiled for IoT settings, while ACME is mainly extended with additional identifiers, challenges and certificate types. Section 5 additionally shows the importance of certificate management in IoT, and discusses how mainly CMP and EST contribute to solving the associated challenges.

Future work should again evaluate the updated state-of-the-art of certificate management protocols and what use cases may have been introduced.

[23] `https://sourceforge.net/projects/esp32-acme-client/`

References

[1] Daniel Simon, Ryan Hurst, and Dr. Bernard D. Aboba Ph.D. *The EAP-TLS Authentication Protocol*. RFC 5216. Mar. 2008. DOI: 10.17487/RFC5216. URL: https://rfc-editor.org/rfc/rfc5216.txt.

[2] Ben Laurie. "Certificate transparency". In: *Communications of the ACM* 57.10 (Sept. 23, 2014), pp. 40–46. ISSN: 00010782. DOI: 10.1145/2659897. URL: http://dl.acm.org/citation.cfm?doid=2661061.2659897 (visited on 03/17/2020).

[3] Adam Langley. *Enhancing digital certificate security*. Google Online Security Blog. Jan. 3, 2013. URL: https://security.googleblog.com/2013/01/enhancing-digital-certificate-security.html (visited on 03/17/2020).

[4] Albert Wasef, Yixin Jiang, and Xuemin Shen. "ECMV: Efficient Certificate Management Scheme for Vehicular Networks". In: *IEEE GLOBECOM 2008 - 2008 IEEE Global Telecommunications Conference*. IEEE GLOBECOM 2008 - 2008 IEEE Global Telecommunications Conference. ISSN: 1930-529X. Nov. 2008, pp. 1–5. DOI: 10.1109/GLOCOM.2008.ECP.129.

[5] P. Caballero-Gil and C. Hernández-Goya. "Efficient Public Key Certificate Management for Mobile Ad Hoc Networks". In: *EURASIP Journal on Wireless Communications and Networking* 2011.1 (Dec. 2011). ISSN: 1687-1499. DOI: 10.1155/2011/935457. URL: https://jwcn-eurasipjournals.springeropen.com/articles/10.1155/2011/935457 (visited on 03/19/2020).

[6] Tero Mononen, Tomi Kause, Stephen Farrell, and Dr. Carlisle Adams. *Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP)*. RFC 4210. Sept. 2005. DOI: 10.17487/RFC4210. URL: https://rfc-editor.org/rfc/rfc4210.txt.

[7] Tomi Kause and Martin Peylo. *Internet X.509 Public Key Infrastructure – HTTP Transfer for the Certificate Management Protocol (CMP)*. RFC 6712. Sept. 2012. DOI: 10.17487/RFC6712. URL: https://rfc-editor.org/rfc/rfc6712.txt.

[8] Jim Schaad. *Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF)*. RFC 4211. Sept. 2005. DOI: 10.17487/RFC4211. URL: https://rfc-editor.org/rfc/rfc4211.txt.

[9] Russ Housley. *Cryptographic Message Syntax (CMS)*. RFC 5652. Sept. 2009. DOI: 10.17487/RFC5652. URL: https://rfc-editor.org/rfc/rfc5652.txt.

[10] Michael Myers and Jim Schaad. *Certificate Management over CMS (CMC)*. RFC 5272. June 2008. DOI: 10.17487/RFC5272. URL: https://rfc-editor.org/rfc/rfc5272.txt.

[11] Max Pritikin, Peter E. Yee, and Dan Harkins. *Enrollment over Secure Transport*. RFC 7030. Oct. 2013. DOI: 10.17487/RFC7030. URL: https://rfc-editor.org/rfc/rfc7030.txt.

[12] Panos Kampanakis. "Cisco IOS EST Certificate Provisioning with the libEST CA Server". en. In: (2016), p. 12.

[13] Peter Gutmann. *Simple Certificate Enrolment Protocol*. Internet-Draft draft-gutmann-scep-15. Work in Progress. Internet Engineering Task Force, Feb. 2020. 48 pp. URL: https://datatracker.ietf.org/doc/html/draft-gutmann-scep-15.

[14] Richard Barnes, Jacob Hoffman-Andrews, Daniel McCarney, and James Kasten. *Automatic Certificate Management Environment (ACME)*. RFC 8555. Mar. 2019. DOI: 10.17487/RFC8555. URL: https://rfc-editor.org/rfc/rfc8555.txt.

[15] Internet Security Research Group. *Let's Encrypt Stats*. https://letsencrypt.org/stats. Retrieved: 2020-02-20.

[16] ISO/IEC. *Recommendation ITU-T X.509 (2019)*. Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks.

[17] Magnus Nystrom and Burt Kaliski. *PKCS #10: Certification Request Syntax Specification Version 1.7*. RFC 2986. Nov. 2000. DOI: 10.17487/RFC2986. URL: https://rfc-editor.org/rfc/rfc2986.txt.

[18] Hendrik Brockhaus, Steffen Fries, and David von Oheimb. *Lightweight CMP Profile*. Internet-Draft draft-ietf-lamps-lightweight-cmp-profile-01. Work in Progress. Internet Engineering Task Force, Mar. 2020. 72 pp. URL: https://datatracker.ietf.org/doc/html/draft-ietf-lamps-lightweight-cmp-profile-01.

[19] 3GPP. *Universal Mobile Telecommunications System (UMTS); LTE; Network Domain Security (NDS); Authentication Framework (AF)*. Technical Specification (TS) 33.310. Version 15.2.0. 3rd Generation Partnership Project (3GPP), Apr. 2019. URL: https://www.etsi.org/deliver/etsi_ts/133300_133399/133310/15.02.00_60/ts_133310v150200p.pdf.

[20] Duncan Jones. *EST: The Forgotten Standard*. Thales eSecurity. Jan. 17, 2019. URL: https://www.thalesesecurity.com/about-us/information-security-research/blogs/est-the-forgotten-standard (visited on 04/03/2020).

[21] Josh Aas et al. "Let's Encrypt: An Automated Certificate Authority to Encrypt the Entire Web". In: *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. CCS '19: 2019 ACM SIGSAC Conference on Computer and Communications Security. London United Kingdom: ACM, Nov. 6, 2019, pp. 2473–2487. ISBN: 978-1-4503-6747-9. DOI: 10.1145/3319535.3363192. URL: http://dl.acm.org/doi/10.1145/3319535.3363192 (visited on 03/30/2020).

[22] Hendrik Brockhaus. *CMP Updates*. Internet-Draft draft-ietf-lamps-cmp-updates-01. Work in Progress. Internet Engineering Task Force, Mar. 2020. 16 pp. URL: https://datatracker.ietf.org/doc/html/draft-ietf-lamps-cmp-updates-01.

[23] Max Pritikin and Carl Wallace. *Alternative Challenge Password Attributes for Enrollment over Secure Transport*. RFC 7894. June 2016. DOI: 10.17487/RFC7894. URL: https://rfc-editor.org/rfc/rfc7894.txt.

[24] Sean Turner. *EST (Enrollment over Secure Transport) Extensions*. RFC 8295. Jan. 2018. DOI: 10.17487/RFC8295. URL: https://rfc-editor.org/rfc/rfc8295.txt.

[25] Peter Van der Stok, Panos Kampanakis, Michael Richardson, and Shahid Raza. *EST over secure CoAP (EST-coaps)*. Internet-Draft draft-ietf-ace-coap-est-18. Work in Progress. Internet Engineering

Task Force, Jan. 2020. 51 pp. URL: https://datatracker.ietf.org/doc/html/draft-ietf-ace-coap-est-18.

[26] Roland Bracewell Shoemaker. *Automated Certificate Management Environment (ACME) TLS Application-Layer Protocol Negotiation (ALPN) Challenge Extension*. RFC 8737. Feb. 2020. DOI: 10.17487/RFC8737. URL: https://rfc-editor.org/rfc/rfc8737.txt.

[27] Roland Bracewell Shoemaker. *Automated Certificate Management Environment (ACME) IP Identifier Validation Extension*. RFC 8738. Feb. 2020. DOI: 10.17487/RFC8738. URL: https://rfc-editor.org/rfc/rfc8738.txt.

[28] Yaron Sheffer, Diego Lopez, Oscar Gonzalez de Dios, Antonio Pastor, and Thomas Fossati. *Support for Short-Term, Automatically Renewed (STAR) Certificates in the Automated Certificate Management Environment (ACME)*. RFC 8739. Mar. 2020. DOI: 10.17487/RFC8739. URL: https://rfc-editor.org/rfc/rfc8739.txt.

[29] Jon Peterson, Mary Barnes, David Hancock, and Chris Wendt. *ACME Challenges Using an Authority Token*. Internet-Draft draft-ietf-acme-authority-token-05. Work in Progress. Internet Engineering Task Force, Mar. 2020. 12 pp. URL: https://datatracker.ietf.org/doc/html/draft-ietf-acme-authority-token-05.

[30] Alexey Melnikov. *Extensions to Automatic Certificate Management Environment for end user S/MIME certificates*. Internet-Draft draft-ietf-acme-email-smime-06. Work in Progress. Internet Engineering Task Force, Nov. 2019. 10 pp. URL: https://datatracker.ietf.org/doc/html/draft-ietf-acme-email-smime-06.

[31] Owen Friel, Richard Barnes, Tim Hollebeek, and Michael Richardson. *ACME for Subdomains*. Internet-Draft draft-friel-acme-subdomains-02. Work in Progress. Internet Engineering Task Force, Mar. 2020. 11 pp. URL: https://datatracker.ietf.org/doc/html/draft-friel-acme-subdomains-02.

[32] Kathleen Moriarty. *ACME End User Client and Code Signing Certificates*. Internet-Draft draft-moriarty-acme-client-04. Work in Progress. Internet Engineering Task Force, Nov. 2019. 14 pp. URL: https://datatracker.ietf.org/doc/html/draft-moriarty-acme-client-04.

[33] René Hummen, Jan H. Ziegeldorf, Hossein Shafagh, Shahid Raza, and Klaus Wehrle. “Towards viable certificate-based authentication for the internet of things”. In: *Proceedings of the 2nd ACM workshop on Hot topics on wireless network security and privacy - HotWiSec ’13*. the 2nd ACM workshop. Budapest, Hungary: ACM Press, 2013, p. 37. ISBN: 978-1-4503-2003-0. DOI: 10.1145/2463183.2463193. URL: http://dl.acm.org/citation.cfm?doid=2463183.2463193 (visited on 04/02/2020).

[34] Michael Schukat and Pablo Cortijo. “Public key infrastructures and digital certificates for the Internet of things”. In: *2015 26th Irish Signals and Systems Conference (ISSC)*. 2015 26th Irish Signals and Systems Conference (ISSC). June 2015, pp. 1–5. DOI: 10.1109/ISSC.2015.7163785.

[35] Zach Shelby, Klaus Hartke, and Carsten Bormann. *The Constrained Application Protocol (CoAP)*. RFC 7252. June 2014. DOI: 10.17487/RFC7252. URL: https://rfc-editor.org/rfc/rfc7252.txt.

[36] Carsten Bormann, Angelo P. Castellani, and Zach Shelby. “CoAP: An Application Protocol for Billions of Tiny Internet Nodes”. In: *IEEE Internet Computing* 16.2 (Mar. 2012), pp. 62–67. ISSN: 1089-7801. DOI: 10.1109/MIC.2012.29. URL: http://ieeexplore.ieee.org/document/6159216/ (visited on 04/02/2020).

[37] Göran Selander, Shahid Raza, Martin Furuhed, Mališa Vučinić, and Timothy Claeys. *Protecting EST Payloads with OSCORE*. Internet-Draft draft-selander-ace-coap-est-oscore-03. Work in Progress. Internet Engineering Task Force, Mar. 2020. 19 pp. URL: https://datatracker.ietf.org/doc/html/draft-selander-ace-coap-est-oscore-03.