IETF 101
LONDON
March 17-23, 2018

Compiled & Edited by: Dr. Balaji Rajendran

(with inputs from Mr. Anoop Kumar Pandey, Mr. Sankalp Bagaria, Ms. Ranjana, Ms. Akanksha Gupta)

Table of Contents

Abstract

The IETF 101 meeting in London, was held from March 17-23, 2018. Dr. Balaji Rajendran, C-DAC Bangalore attended the meeting.

IETF 101 meeting was important due to the standardization of work in the TLS working group that is in the process of rolling out TLS 1.3. The following working group's discussions are covered in this report:

- TLS 1.3 (Transport Layer Security)
- UTA (Using TLS in Applications)
- Major and minor issues of UTA (Using TLS in Applications)
- Autonomic Networking Integrate Model and Approach (anima)
- ACME (Automated Certificate Management Environment)
- Trusted execution environment provisioning (TEEP).

1.Major discussions in Different Working Groups

1.TLS 1.3(Transport Layer Security)

TLS 1.2 has been in use for almost a decade now. Over the years, it has faced a number of attacks owing to poor implementations, less-secure crypto algorithms etc... The Transport Layer Security (TLS) Working Group is a key IETF effort developing core security protocols for the Internet. TLS 1.3 has been accepted as a standard by the community.

TLS 1.2 takes 2 round trips to setup a new TLS connection, but TLS 1.3 only takes 1 round trip time (RTT) and that saves around 100-200 milliseconds, which will lead to faster browsing experience. Also the 'resumption' connections in TLS 1.2, though were optimized, TLS 1.3 is able to achieve it with 0 RTT (with data being sent in the first step of the handshake process itself) which is again a performance improvement [1].

A key point to be noted is that in TLS 1.2 resumptions, the same key that was used previously would be used again, and therefore no key shares happen. In TLS 1.3 resumptions, a new and different key is used for each session guaranteeing forward secrecy which means the past conversations cannot be decrypted. This has led to debates and heated discussions which has now taken the shape as the concerns of Data centre operators, who want the keys for a number of reasons including for regulatory compliance.

Two options have emerged from the community, one of them is to have an option for both the client and server to explicitly grant access to TLS sessions which is discussed in the TLS Visibility draft (discussed in detail below). Another is the use of static Diffie-Hellman keys as an optional configuration in TLS 1.3 to enable monitoring. It may be noted that the former allows for online decryption while the latter only allows for offline decryption.

TLS 1.3 Option for Negotiation of Visibility in the Datacentre (draft-rhrd-tls-tls13-visibility-01): Current drafts of TLS 1.3 do not include the use of the RSA Handshake and have instead adopted ephemeral-mode Diffie-Hellman (DHE) and elliptic-curve Diffie-Hellman (ECDHE) as the primary cryptographic key exchange mechanism used in TLS. While (EC) Diffie-Hellman is in nearly all ways an improvement over the TLS RSA handshake, the use of (EC)DH impacts certain enterprise network operational requirements. The TLS Visibility Extension provides an option to enable visibility into a TLS 1.3 session by an authorized third party. Use of the extension requires opt-in by the TLS client when it initiates a TLS 1.3 session. The TLS server then opts-in by including key material that will enable decryption in the TLS Visibility Extension. The presence of the TLS Visibility Extension provides a clear indication that other parties have been granted access to the TLS session plaintext. The keying material in the TLS Visibility Extension is encrypted and can only be decrypted by authorized parties that have been given the private key from a managed Diffie-Hellman key pair.

2.TLS sessions in IETF 101:

There were two sessions on TLS 1.3. One on Monday 19th march, and another on Wednesday 21st March 2018. On Monday the focus primarily was on the ongoing discussion of data center operator concerning the implementation of TLS 1.3. They have come up with couple of workarounds and solutions which are discussed below:

2.1 A proposal was Record Header extensions for DTLS, wherein the authors were proposing to signal the presence of connection ID (CID) in the packet. This may have implications for packet sniffers in DTLS environment. It faced criticism, and said to have overlapping with SPAKE and therefore postponed to Wednesday. On Wednesday, use of implicit CIDs were discussed and it was mentioned that connection ID alone does not reveal much information and ideas were floated to encrypt the connection ID and also not to use sequential numbers. The idea was to synch with QUIC protocol. However, owing to privacy concerns with respect to implementation the author agreed to put up a next updated version soon.

2.2 The next proposal was to use of SPAKE in TLS 1.3 connections (jointly proposed by members from UC Berkeley and Akamai) which allows a means for two parties that share a password to derive a strong shared key,

and use it for exchange in TLS 1.3 instead of the shared secret key generated newly for every session. It was decided to take up the discussions on the mailing lists.

2.3 An updated proposal on a "**TLS 1.3 Option for Negotiation of Visibility in the Datacentre**" [5]. This allows an intermediary to watch for the extension being present in "Client Hello" or "Server Hello" messages. Clients include an "Empty Structure". A lot of discussions followed and few major concerns expressed were: Increase in Data Exfiltration attacks and increase in attack surface, Analysis of this extension's performance in TLS 1.3, increase in cost as it may require modification in browsers, and the possibility of any number of intermediaries being able to intercept once the client has given the consent. It was called for adoption of the document as a working group item, which means that the draft can be pursued till it matures as a standard. However, it was voted down (opposed) by a large number of members present in the meeting which means now if this draft has to be pursued further, it requires the consent of the Area Directors (TLS falls under Security Area).

2.4 A proposal on Exported Authenticators - **"TLS 1.3 Extension for Certificate-based Authentication with an External Pre-Shared Key":** The TLS 1.3 handshake protocol employs key agreement algorithms that could be broken by the invention of a large-scale quantum computer. These algorithms include Diffie-Hellman (DH) and Elliptic Curve Diffie-Hellman (ECDH. As a result, an adversary that stores a TLS 1.3 handshake protocol exchange today could decrypt the associated encrypted communications in the future when a large-scale quantum computer becomes available. This draft TLS 1.3 extension proposes to protect today's communications from the future invention of a large-scale quantum computer by providing a strong external PSK as an input to the TLS 1.3 key schedule while preserving the authentication provided by the existing certificate and digital signature mechanisms. This solution is improvements.

3.UTA(Using TLS in Applications)

The following issues were raised for the **draft-ietf-uta-mta-sts-14** [2]**.**

Major issue(1)

- In the Section 3.2 it's mentioned that while fetching a policy,sender's should validate that the media type is "text/plain".
- So the concern raised by the **Area Director of UTA Mr.Alexey Melnikov** was that what happens if somebody uses a file in ShiftJS(Shift Japanese Industrial Standards which is a character encoding for the japanese language) charset or one of Unicode-16 variants,it would not be parseable at all.
- He then recommended to update the requirement to say that all parameters other than charset are ignored .
- Additionally require use of charset=utf-8 or charset=us-ascii.

Major issue(2)

- In Section 3.2 it's mentioned that-
- sts-policy-record = *WSP sts-policy-field *WSP *(CRLF *WSP sts-policy-field *WSP)
- Gist of the discussion was that Whitespaces(WSP) will create problem in parsing according to RFC 5322 parser (RFC 5322 tells about the Internet Message Format (IMF), a syntax for text messages that are sent between computer users, within the framework of "electronic mail" messages.)
- Leading Whitespaces are problemetic not the trailing ones.
- So,It's suggetsed that if we really want WSP's then we have to add a line that this format can't be parsed by RFC 5322 parser without stripping off leading whitespaces on each line first.

Minor Isuue(3)

- In Section 3.2 it's mentioned that-
- For example: "mx: mail.example.com mx: .example.net" indicates
- Suggestion has been given by Alexey that between the two mx's shown above ,there should be <CRLF> or \r\n

- as the exmaple is not syntactically valid as per the ABNF(Augmented Backus-Naur Form which is a metalanguage based on Backus -Naur Form consisting of it's own syntax and derivation rules.

Minor Issue(4)

- In Section 3.2 it's mentioned that-
- sts-policy-max-age-value = 1*10(DIGIT)
- Alexey told that the Leading 0's are allowed for above value according to the ABNF.
- If the leading 0's are not OK then we need to add a comment "leading 0's are not allowed like this
- sts-policy-max-age-value = 1*10(DIGIT) ; leading 0s are disallowed

Minor Issue(5)

- In Section 3.2 it's mentioned that-
- sts-policy-ext-value = 1*(%x21-3A / %x3C / %x3E-7E) ; chars, excluding "=", ";", SP, and ; control chars
- Alexey wanted to doublecheck about the restrictiveness in the policy format.
- He says if extensions want to add a field with human readable text, at least allowing for space might be useful.
- Also is it necessary to prohibit "=" and ";" ?.
- D. Morgolis said that we can allow spaces(SP) but not "=" and ";".

Minor Issue(6)

- In Section 3.3 it's mentioned that-
- HTTPS Policy Fetching When fetching a new policy or updating a policy, the HTTPS endpoint MUST present a X.509 certificate which is valid for the "mta-sts" host (e.g. "mta-sts.example.com") as described below, chain to a root CA that is trusted by the sending MTA, and be non-expired.
- And also questioned about the various certificate key usage field.
- People discussed and realised that if HTTPS is mentioned then it already specifies and directs towards all the requirements related to it.

Minor Issue(7)

- In Section 3.4 it's mentioned that-
- Policy Selection for Smart Hosts and Subdomains When sending mail via a "smart host"--an intermediate SMTP relay rather than the message recipient's server--compliant senders MUST treat the smart host domain as the policy domain for the purposes of policy discovery and application.
- Alexey said ,“he doesn't think that the definition of the smart host is quite right as email already uses intermediate SMTP relays which are specified by MX records".These don't have to correspond to "message recipient's server".
- He suggests to use another definition for the smart host.
- When sending mail via a "*smart host*"--an administratively configured intermediate SMTP relay, which is different from the message recipient's server as determined from DNS --compliant senders MUST treat the smart host domain as the policy domain for the purposes of policy discovery and application.

Minor Issue(8)

- Alexey has shown some concerns regarding the use of SNI extension.He has suggested authors to review the whole section(Section 7.1 of SNI Support) for consistency.
- He Also suggested that there should be normative reference to RFC 3207(SMTP Service Extension for Secure SMTP over Transport Layer Security),as it's required to implement and understand this document(mta-sta).

Minor Issue(9)

- Chris has suggested to add a new considerartion in the Security Consideration Section:
- This mechanism causes an MTA (an automated system) to adopt the role of an HTTPS client in a scenario where the HTTPS server may be hostile to operation of the MTA. A full HTTP stack is a large amount of code that may contain coding errors that expose the MTA to new implementation vulnerabilities due to the increased attack surface. This threat can be partially mitigated by using a hardened HTTPS client library that has been tested against a fuzzing HTTPS test server.

This threat can also be partially mitigated by isolating the HTTPS code into a separate process that does not have access to the normal MTA machinery and making sure the MTA machinery gracefully handles a wedged HTTPS co-process.

4.Autonomic Networking Integrated Model and Approach (anima)

Autonomic networking refers to the self-managing characteristics (configuration, protection, healing, and optimization) of distributed network elements, adapting to unpredictable changes while hiding intrinsic complexity from operators and users [3]. Autonomic Networking, which often involves closed-loop control, is applicable to the complete network (functions) lifecycle (e.g. installation, commissioning, operating, etc). An autonomic function that works in a distributed way across various network elements is a candidate for protocol design. Such functions should allow central guidance and reporting, and co-existence with non-autonomic methods of management. The general objective of this working group is to enable the progressive introduction of autonomic functions into operational networks, as well as reusable autonomic network infrastructure, in order to reduce the operational expense.

Drafts

Draft 1: An Autonomic Control Plane (ACP)

Contributors: T. Eckert, M. Behringer, S. Bjarnason

Latest Release: draft-ietf-anima-autonomic-control-plane-13 17/12/2017 [4]

Abstract: Autonomic functions need a control plane to communicate, which depends on some addressing and routing. This Autonomic Management and Control Plane should ideally be self-managing, and as independent as possible of configuration. This document defines such a plane and calls it the "Autonomic Control Plane", with the primary use as a control plane for autonomic functions. It also serves as a "virtual out of band channel" for OAM (Operations Administration and Management) communications over a network that is secure and reliable even when the network is not configured, or not misconfigured.

Draft 2: Bootstrapping Remote Secure Key Infrastructures (BRSKI)

Contributors: M. Pritikin, M. Richardson, M. Behringer, S. Bjarnason, K. Watsen

Latest Release: draft-ietf-anima-bootstrapping-keyinfra-13 27/03/2018 [5]

Abstract: This document specifies automated bootstrapping of a remote secure key infrastructure (BRSKI) using manufacturer installed X.509 certificate, in combination with a manufacturer's authorizing service, both online and offline. Bootstrapping a new device can occur using a routable address and a cloud service, or using only link-local connectivity, or on limited/disconnected networks. Support for lower security models, including devices with minimal identity, is described for legacy reasons but not encouraged. Bootstrapping is complete when the cryptographic identity of the new key infrastructure is successfully deployed to the device but the established secure connection can be used to deploy a locally issued certificate to the device as well.

Draft 3: A Generic Autonomic Signaling Protocol (GRASP)

Contributors: C. Bormann, B. Carpenter, B. Liu

Latest Release: draft-ietf-anima-grasp-15 07/07/2017

Abstract: This document specifies the GeneRic Autonomic Signaling Protocol (GRASP), which enables autonomic nodes and autonomic service agents to dynamically discover peers, to synchronize state with each other, and to negotiate parameter settings with each other. GRASP depends on an external security environment that is described elsewhere. The technical objectives and parameters for specific application scenarios are to be described in separate documents. Appendices briefly discuss requirements for the protocol and existing protocols with comparable features.

Draft 4: Generic Autonomic Signalling Protocol Application Program Interface (GRASP API)

Contributors: B. Carpenter, B. Liu, W. Wang, X. Gong

Latest Release: draft-ietf-anima-grasp-api-01 03/03/2018

Abstract: This document is a conceptual outline of an application programming interface (API) for the Generic Autonomic Signalling

Protocol (GRASP). Such an API is needed for Autonomic Service Agents (ASA) calling the GRASP protocol module to exchange autonomic network messages with other ASAs.

Draft 5: Autonomic IPv6 Edge Prefix Management in Large-scale Networks

Contributors: S. Jiang, B. Carpenter, Q. Sun

Latest Release: draft-ietf-anima-prefix-management-07 15/12/2017

Abstract: This document defines two autonomic technical objectives for IPv6 prefix management at the edge of large-scale ISP networks, with an extension to support IPv4 prefixes. An important purpose of the document is to use it for validation of the design of various components of the autonomic networking infrastructure.

Draft 6: A Reference Model for Autonomic Networking

Contributors: M. Behringer, B. Carpenter, T. Eckert, L. Ciavaglia, J. Nobre

Latest Release: draft-ietf-anima-reference-model-06 23/02/2018

Abstract: This document describes a reference model for Autonomic Networking. It defines the behaviour of an autonomic node, how the various elements in an autonomic context work together, and how autonomic services can use the infrastructure.

Draft 7: Using Autonomic Control Plane for Stable Connectivity of Network OAM

Contributors: T. Eckert, M. Behringer

Latest Release: draft-ietf-anima-stable-connectivity-10 05/02 2018

Abstract: OAM (Operations, Administration and Maintenance - as per BCP161, (RFC6291) processes for data networks are often subject to the problem of circular dependencies when relying on connectivity provided by the network to be managed for the OAM purposes.

Provisioning while bringing up devices and networks tends to be more difficult to automate than service provisioning later on, changes in core network functions impacting reachability cannot be automated because of ongoing connectivity requirements for the OAM equipment itself, and widely used OAM protocols are not secure enough to be carried across the network without security concerns.

This document describes how to integrate OAM processes with an autonomic control plane in order to provide stable and secure connectivity for those OAM processes. This connectivity is not subject to aforementioned circular dependencies.

Draft 8: Voucher Profile for Bootstrapping Protocols

Contributors: K. Watsen, M. Richardson, M. Pritikin, T. Eckert

Latest Release: draft-ietf-anima-voucher-07 24/02/2018

Abstract: This document defines a strategy to securely assign a pledge to an owner, using an artefact signed, directly or indirectly, by the pledge's manufacturer. This artefact is known as a "voucher".

This document defines an artefact format as a YANG-defined JSON document that has been signed using a CMS structure. Other YANG-derived formats are possible. The voucher artefact is normally generated by the pledge's manufacturer (i.e. the Manufacturer Authorized Signing Authority).

This document only defines the voucher artefact, leaving it to other documents to describe specialized protocols for accessing it.

5. ACME WG: Authority Tokens for ACME

ACME is a mechanism for automating certificate management on the Internet [6]. It enables administrative entities to prove effective control over resources like domain names, and automates the process of generating and issuing certificates. The STIR (Secure Telephone Identity) problem statement identifies the need for Internet credentials that can attest authority for telephone numbers in order to detect impersonation, which is currently an enabler for common attacks associated with illegal rob calling (A rob call is a phone call that uses a computerized autodialer to deliver a pre-recorded message, as if from a robot.), voicemail hacking, and swatting(it is a criminal harassment tactic of deceiving an emergency service (via such means as hoaxing an emergency services dispatcher) into sending a police and emergency service response team to another person's address).

The aim of the initial challenges specified is not to prove the assignment and delegation of resources in the telephone network: it is instead establishing whether Internet-enabled entities have effective control over the devices associated with those resources. The likely challenges for proving effective control over a telephone number therefore rely largely on routing some kind of secret to the telephone number in question and requesting that the receiving device play that secret back to the ACME server as the assignment of numbering resources can change over time, demonstrations of effective control must be regularly refreshed.

Communications Service Providers (CSPs) can delegate authority over numbers to their customers, and those CSPs who support ACME can then help customers to acquire certificates for those numbering resources with ACME. The token must contain the delegated telephone number or number range, the SPC of the CSP, a nonce, the signature of the CSP with its SPC credential, and a link to a resource where relying parties can acquire the SPC credential.

With web-based telephone number rout ability validation, the client in an ACME transaction proves its control over a telephone number by proving that it can receive traffic sent to that number over the PSTN. type (required, string): The string "sms-link-00" token (required, string): A random value that uniquely identifies the challenge. This value MUST have at least 128 bits of entropy, in order to prevent an attacker from guessing it. It MUST NOT contain any characters outside

the URL-safe Base64 alphabet and MUST NOT contain any padding characters ("=").

```
{
  "type": "sms-link-00",
}
```

A client's response to this challenge simply acknowledges that it is ready to receive the validation SMS from the server. On receiving a response, the server sends an SMS message to the TN (Telephone Number) being validated containing a URL that the client must have a user access in order to complete the challenge. This URL is intended to be opened in a web browser so that the user can have an interaction with the CA; it is not sufficient for the client to simply send a GET request to the URL.

Because SMS return rout ability tests are becoming more common in two-factor authentication systems, they have also become an attractive target for attackers to try to compromise them. Using short-lived certificates for this function, and requiring the client to perform this validation repeatedly, would help to mitigate associated risks.

6.Trusted Execution Environment Provisioning (TEEP)

A trusted execution environment provisioning (TEE) is a secure area of the main processor. It must have such properties so that the device must have a unique security identity (firmware or hardware based unique key). The only authorized code can be executed inside the TEE. Any data inside the TEE cannot be read outside the TEE [7].

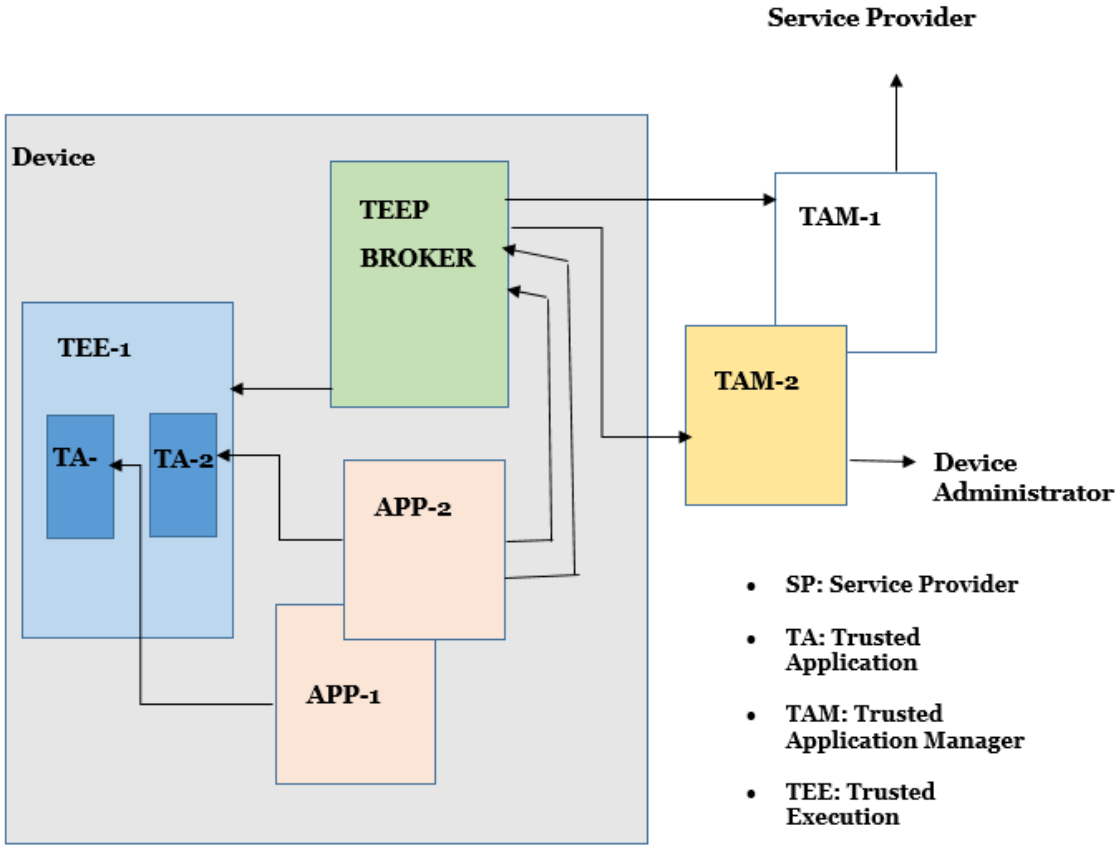

Figure: Notional Architecture of TEEP

System Components:

- Service Providers (SP) and Device Administrators (DA) utilize the services of a TAM to manage TAs on Devices. SPs do not directly interact with devices. DAs may elect to use a TAM for remote administration of TAs instead of managing each device directly.
- TAM: A TAM is responsible for performing lifecycle management activity on TA's and SD's on behalf of Service Providers and Device Administrators. This includes creation and deletion of TA's and SD's, and may include, for example, over-the-air updates to keep an SP's TAs up-to-date and clean up when a version should be removed. TAMs may provide services that make it easier for SPs or DAs to use the TAM's service to manage multiple devices, although that is not required of a TAM.
- TEEP Broker: The TEEP Broker is an application running in a Rich Execution Environment that enables the message protocol exchange between a TAM and a TEE in a device. The TEEP Broker does not process messages on behalf of a TEE, but merely is responsible for relaying messages from the TAM to the TEE, and for returning the TEE's responses to the TAM.

6.1 Benefits of TEE.

A TEE Provides hardware-enforcement that The device has a unique security identity, the code executing inside the TEE is trusted and authorized. Any data inside the TEE cannot be read by code outside the TEE (Safe area of the device to protect assets). Compromising REE (Rich Execution Environment) and normal apps don't affect TEE and code (called Trusted Application) running inside TEE.

Trusted App development and distribution are harder than normal apps via App Store. Trust and management issues due to multiple parties involved in the scenario.

6.2 Use cases for TEE apps

TEE can be used to protect transaction methods because only authorized code can make payments or see payment data, if TEE in evolved in payment modes.

In IoT TEE can also use to reduce safety risk's if in the implementation only authorized code can access physical actuator/sensor, Confidential

cloud computing can be achieved by developing TEE enabled cloud in future where only tenant (not cloud hoster) can access data.

Device/TEE admin requirements.

Device/TEE wants to manage the allowed list of Trusted Application (TA's) which can be executed inside the TEE. and want to Empower Trusted App author to keep the TA code and/or its configuration encrypted and only let it be decrypt able with a kind of TEE that is trusted to keep the info.

Rich Application author-

(REE) Client app author wants to depend on a TA from another vendor and expresses a dependency either at install time or at runtime.

TEE chip vendor-

A TEE chip vendor wants to only allow authorized TA's to run in its chip.

Device OEM:

A device OEM wants to only allow authorized TA's to run in the TEE on its devices.

2. Acknowledgement

We would like to thank Internet Governance division, Ministry of Electronics and Information Technology (MietY), Government of India.

3. Glossary

ACME- Automated Certificate Management Environment

ANIMA-Autonomic Networking Integrated Model and Approach

CSP-Communications Service Providers

DHE-Diffie-Hellman Ephemeral

ECDHE- Elliptic-Curve Diffie-Hellman

GRASP API- Generic Autonomic Signalling Protocol Application Program Interface

IETF-internet engineering task forum

IIREF- Indian Internet Research and Engineering Forum

RTT-Round Trip Time

TA- Trusted Application

TEEP-Trusted Execution Environment Provisioning

TLS- Transport Layer Security

UTA-Using TLS in Applications

References

1. TLS 1.3 -
   https://datatracker.ietf.org/doc/draft-ietf-tls-tls13/

2. Using TLS in Applications (uta)
   https://datatracker.ietf.org/wg/uta/about/

3. Autonomic Networking Integrated Model and Approach (anima)
   https://datatracker.ietf.org/wg/anima/about/

   https://tools.ietf.org/html/draft-ietf-acme-service-provider-02

   https://datatracker.ietf.org/meeting/101/materials/slides-101-acme-authority-tokens-for-acme-00

4. draft-ietf-anima-autonomic-control-plane-13
   https://datatracker.ietf.org/doc/draft-ietf-anima-autonomic-control-plane/13/

5. draft-ietf-anima-bootstrapping-keyinfra-13
   https://datatracker.ietf.org/doc/draft-ietf-anima-bootstrapping-keyinfra/13/

6. Automated Certificate Management Environment (ACME)
   https://datatracker.ietf.org/wg/acme/documents/

7. Trusted Execution Environment Provisioning (TEEP)
   https://datatracker.ietf.org/wg/teep/documents/

8. RFC 8461 SMTP MTA Strict Transport Security (MTA-STS)
   https://datatracker.ietf.org/doc/rfc8461/