Fundamentals of Cryptography: Session 3

NALINI ELKINS

INDUSTRY NETWORK TECHNOLOGY COUNCIL

PRESIDENT@INDUSTRYNETCOUNCIL.ORG

A few words about me

- President: Industry Network Technology Council
- Founder & CEO: Inside Products, Inc.
- Advisory Board: India Internet Engineering Society
- RFCs: RFC8250 (Embedded performance and diagnostics for IPv6) and others
- Product developer (OEMed by IBM and others)
- Working with IPv6 for over 20 years
- Working with network management, diagnostic, cryptography, performance issues at large brick-and-mortar enterprises for over 30 years

Fundamentals of Cryptography

Details:

- DES
- 3DES
- Asymmetric encryption / symmetric encryption
- Elliptic curve cryptography
- Certificate authority
- Diffie-Hellman key exchange
- Diffie-Hellman groups
- Hashed message authentication code (HMAC)
- HMAC MD5
- HMAC_SHA
- Message authentication code (MAC)
- Message digest algorithm 5 (MD5)
- Rivest Shamir Adleman (RSA)
- Secure hash algorithm 1 (SHA1)
- X.500 distinguished name
- X.509 digital certificate

Concepts

- Block cipher
- Encryption
- Hash
- Keys
- Public / private keys
- Tags

Issues

- Key sizes
- Choice of protocol
- End-to-end security

We have gone over this in the last two sessions.

This session's agenda

- Pull things together
- Set the stage for the next webinars
- Discuss implications of quantum computing

Pull things together

- Cryptographic concepts: hashing, signing, keys, etc. are implemented in protocols

- How?

- Protocols are used to secure and transmit traffic. For example:
  - TLS
  - MLS
  - SSH
  - IPSec

TLS: Start

- NIST standardizes ciphers:
  - Symmetric (AES, DES – deprecated)
  - Asymmetric (RSA, Diffie-Hellman, ECC, etc)

- IETF standardizes protocols – TLS
  - Packet exchanges
  - Format of handshake
  - Format of data packet
  - Security analysis

TLS: Middle

- Packet format, protocol exchanges, cipher suites are implemented in libraries. For example:
  - OpenSSL
  - MBedTLS
  - BouncyCastle

Note: Protocols can be embedded in other protocols. For example, DNS over HTTPS (DoH) embeds TLS.

- Libraries are called by programs. Programs are coded in languages which are (generally) compiled. For example:
  - Java
  - C, C++, C#

TLS: End

- Programs can be:
  - Applications
  - Services (web servers, database servers, proxies, etc)
- Implementation can be:
  - Software
  - Hardware
- Platform can be:
  - Windows
  - Linux
  - Mobile
  - IoT
  - Mainframe

TLS: Ongoing Operation

- What happens when there is a problem between client and server? (Mismatch in cipher suites / versions)
- What happens when a vulnerability is found?
- TLS is embedded and can be used by middle boxes
  - IDS / IPS
  - Firewalls
  - Bulk data packet capture tools

This is the only part many enterprises see or know about.

TLS: Ongoing Operation

- Hackers / terrorists, etc. (try to find vulnerabilities)
- Academia:
  - Try to find vulnerabilities before hackers do!
  - Do ongoing security analysis
- Organizations (for-profit and non-profit)
  - NonProfit: CERT (track vulnerabilities)
  - For profit: Router, OS vendors, ISVs (try to make $$)

Many enterprises see only some of this. Question also is “when do you find out?”

One reason to come to IETF is that this is “in the ether”. Hallway conversations, WG presentations.

Some people make things happen,

some watch things happen,

while others wonder what has happened.

— Eleanor Roosevelt

TLS Versions

Version	RFC	Release Date	End of Life
TLS 1.3	RFC 8446	March 21, 2018	
TLS 1.2	RFC 5246	August, 2008	
TLS 1.1	RFC 4346	April, 2006	June 30, 2018
TLS 1.0	RFC 2246	January, 1999	June 30, 2018

TLS Cipher Suites

Deprecation as security issues found

Value	Description	DTLS-OK	Recommended	Reference
0x00,0x00	TLS_NULL_WITH_NULL_NULL	Y	N	[RFC5246]
0x00,0x01	TLS_RSA_WITH_NULL_MD5	Y	N	[RFC5246]
0x00,0x02	TLS_RSA_WITH_NULL_SHA	Y	N	[RFC5246]
0x00,0x03	TLS_RSA_EXPORT_WITH_RC4_40_MD5	N	N	[RFC4346][RFC6347]
0x00,0x04	TLS_RSA_WITH_RC4_128_MD5	N	N	[RFC5246][RFC6347]
0x00,0x05	TLS_RSA_WITH_RC4_128_SHA	N	N	[RFC5246][RFC6347]
0x00,0x06	TLS_RSA_EXPORT_WITH_RC2_CBC_40_MD5	Y	N	[RFC4346]
0x00,0x07	TLS_RSA_WITH_IDEA_CBC_SHA	Y	N	[RFC5469][status-change-tls-des-idea-ciphers-to-historic]
0x00,0x08	TLS_RSA_EXPORT_WITH_DES40_CBC_SHA	Y	N	[RFC4346]
0x00,0x09	TLS_RSA_WITH_DES_CBC_SHA	Y	N	[RFC5469][status-change-tls-des-idea-ciphers-to-historic]
0x00,0x0A	TLS_RSA_WITH_3DES_EDE_CBC_SHA	Y	N	[RFC5246]
0x00,0x0B	TLS_DH_DSS_EXPORT_WITH_DES40_CBC_SHA	Y	N	[RFC4346]
0x00,0x0C	TLS_DH_DSS_WITH_DES_CBC_SHA	Y	N	[RFC5469][status-change-tls-des-idea-ciphers-to-historic]
0x00,0x0D	TLS_DH_DSS_WITH_3DES_EDE_CBC_SHA	Y	N	[RFC5246]
0x00,0x0E	TLS_DH_RSA_EXPORT_WITH_DES40_CBC_SHA	Y	N	[RFC4346]
0x00,0x0F	TLS_DH_RSA_WITH_DES_CBC_SHA	Y	N	[RFC5469][status-change-tls-des-idea-ciphers-to-historic]
0x00,0x10	TLS_DH_RSA_WITH_3DES_EDE_CBC_SHA	Y	N	[RFC5246]
0x00,0x11	TLS_DHE_DSS_EXPORT_WITH_DES40_CBC_SHA	Y	N	[RFC4346]
0x00,0x12	TLS_DHE_DSS_WITH_DES_CBC_SHA	Y	N	[RFC5469][status-change-tls-des-idea-ciphers-to-historic]
0x00,0x13	TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA	Y	N	[RFC5246]
0x00,0x14	TLS_DHE_RSA_EXPORT_WITH_DES40_CBC_SHA	Y	N	[RFC4346]
0x00,0x15	TLS_DHE_RSA_WITH_DES_CBC_SHA	Y	N	[RFC5469][status-change-tls-des-idea-ciphers-to-historic]
0x00,0x16	TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA	Y	N	[RFC5246]
0x00,0x17	TLS_DH_anon_EXPORT_WITH_RC4_40_MD5	N	N	[RFC4346][RFC6347]
0x00,0x18	TLS_DH_anon_WITH_RC4_128_MD5	N	N	[RFC5246][RFC6347]

IETF and Security

https://datatracker.ietf.org/group/sec/about/

The Security Area is the home for working groups focused on security protocols. They provide one or more of the security services:

- integrity,
- authentication,
- non-repudiation,
- confidentiality, and
- access control.

Since many of the security mechanisms needed to provide these security services employ cryptography, key management is also vital.

IETF Security Working Groups

(saag) Security Area Open Meeting
(ace) Authentication and Authorization for Constrained Environments
(acme) Automated Certificate Management Environment
(cose) CBOR Object Signing and Encryption
(dance) DANE Authentication for Network Clients Everywhere
(emu) EAP Method Update
(gnap) Grant Negotiation and Authorization Protocol
(i2nsf) Interface to Network Security Function
(ipsecme) IP Security Maintenance and Extensions
(jose) Javascript Object Signing and Encryption
(kitten) Common Authentication Technology Next Generation
(lake) Lightweight Authenticated Key Exchange
(lamps) Limited Additional Mechanisms for PKIX and SMIME
(mls) Messaging Layer Security
(oauth) Web Authorization Protocol
(ohai) Oblivious HTTP Application Intermediation
(openpgp) Open Specification for Pretty Good Privacy
(ppm) Privacy Preserving Measurement
(pquip) Post-Quantum Use In Protocols
(privacypass) Privacy Pass
(radext) RADIUS EXTensions
(rats) Remote ATtestation ProcedureS
(scitt) Supply Chain Integrity, Transparency, and Trust
(secdispatch) Security Dispatch
(secevent) Security Events
(suit) Software Updates for Internet of Things
(teep) Trusted Execution Environment Provisioning
(tls) Transport Layer Security

IETF Security Working Groups

(saag) Security Area Open Meeting
(ace) Authentication and Authorization for Constrained Environments
(acme) Automated Certificate Management Environment
(cose) CBOR Object Signing and Encryption
(dance) DANE Authentication for Network Clients Everywhere
(emu) EAP Method Update
(gnap) Grant Negotiation and Authorization Protocol
(i2nsf) Interface to Network Security Function
(ipsecme) IP Security Maintenance and Extensions
(jose) Javascript Object Signing and Encryption
(kitten) Common Authentication Technology Next Generation
(lake) Lightweight Authenticated Key Exchange
(lamps) Limited Additional Mechanisms for PKIX and SMIME
(mls) Messaging Layer Security
(oauth) Web Authorization Protocol
(ohai) Oblivious HTTP Application Intermediation
(openpgp) Open Specification for Pretty Good Privacy
(ppm) Privacy Preserving Measurement
(pquip) Post-Quantum Use In Protocols
(privacypass) Privacy Pass
(radext) RADIUS EXTensions
(rats) Remote ATtestation ProcedureS
(scitt) Supply Chain Integrity, Transparency, and Trust
(secdispatch) Security Dispatch
(secevent) Security Events
(suit) Software Updates for Internet of Things
(teep) Trusted Execution Environment Provisioning
(tls) Transport Layer Security

OpenSSH 9.3 released March 15, 2023

OpenSSH is the premier connectivity tool for remote login with the SSH protocol. It encrypts all traffic to eliminate eavesdropping, connection hijacking, and other attacks. In addition, OpenSSH provides a large suite of secure tunneling capabilities, several authentication methods, and sophisticated configuration options.

The OpenSSH suite consists of the following tools:

- Remote operations are done using ssh, scp, and sftp.
- Key management with ssh-add, ssh-keysign, ssh-keyscan, and ssh-keygen.
- The service side consists of sshd, sftp-server, and ssh-agent.

What is NOT there

OpenSSH is developed by a few developers of the OpenBSD Project and made available under a BSD-style license.

OpenSSH is incorporated into many commercial products, but very few of those companies assist OpenSSH with funding.

Contributions towards OpenSSH can be sent to the OpenBSD Foundation.

What else is NOT there

Download PuTTY

PuTTY is an SSH and telnet client, developed originally by Simon Tatham for the Windows platform. PuTTY is open source software that is available with source code and is developed and supported by a group of volunteers.

Download PuTTY

Why does this matter?

- SSH is used by many organizations.

- SSH is not standardized by IETF

- What are the cipher suites used? (In particular, quantum safe suites.)

Datatracker Groups Documents Meetings Other User

Transport Layer Security (tls)

About | Documents | Meetings | History | Photos | Email expansions | List archive »

Search

Document		Date	Status
Active Internet-Drafts (11 hits)			
draft-ietf-tls-wkech-02 **A well-known URI for publishing ECHConfigList values.**	8 pages	2023-04-21	I-D Exists WG Document
draft-ietf-tls-esni-16 **TLS Encrypted Client Hello**	48 pages	2023-04-06	I-D Exists WG Document Mar 2021
draft-ietf-tls-rfc8447bis-04 **IANA Registry Updates for TLS and DTLS**	14 pages	2023-03-27	I-D Exists In WG Last Call
draft-ietf-tls-rfc8446bis-07 **The Transport Layer Security (TLS) Protocol Version 1.3**	158 pages	2023-03-26	I-D Exists In WG Last Call
draft-ietf-tls-deprecate-obsolete-kex-02 **Deprecating Obsolete Key Exchange Methods in TLS 1.2**	20 pages	2023-03-25	I-D Exists WG Document
draft-ietf-tls-ctls-08	24 pages	2023-03-	I-D Exists

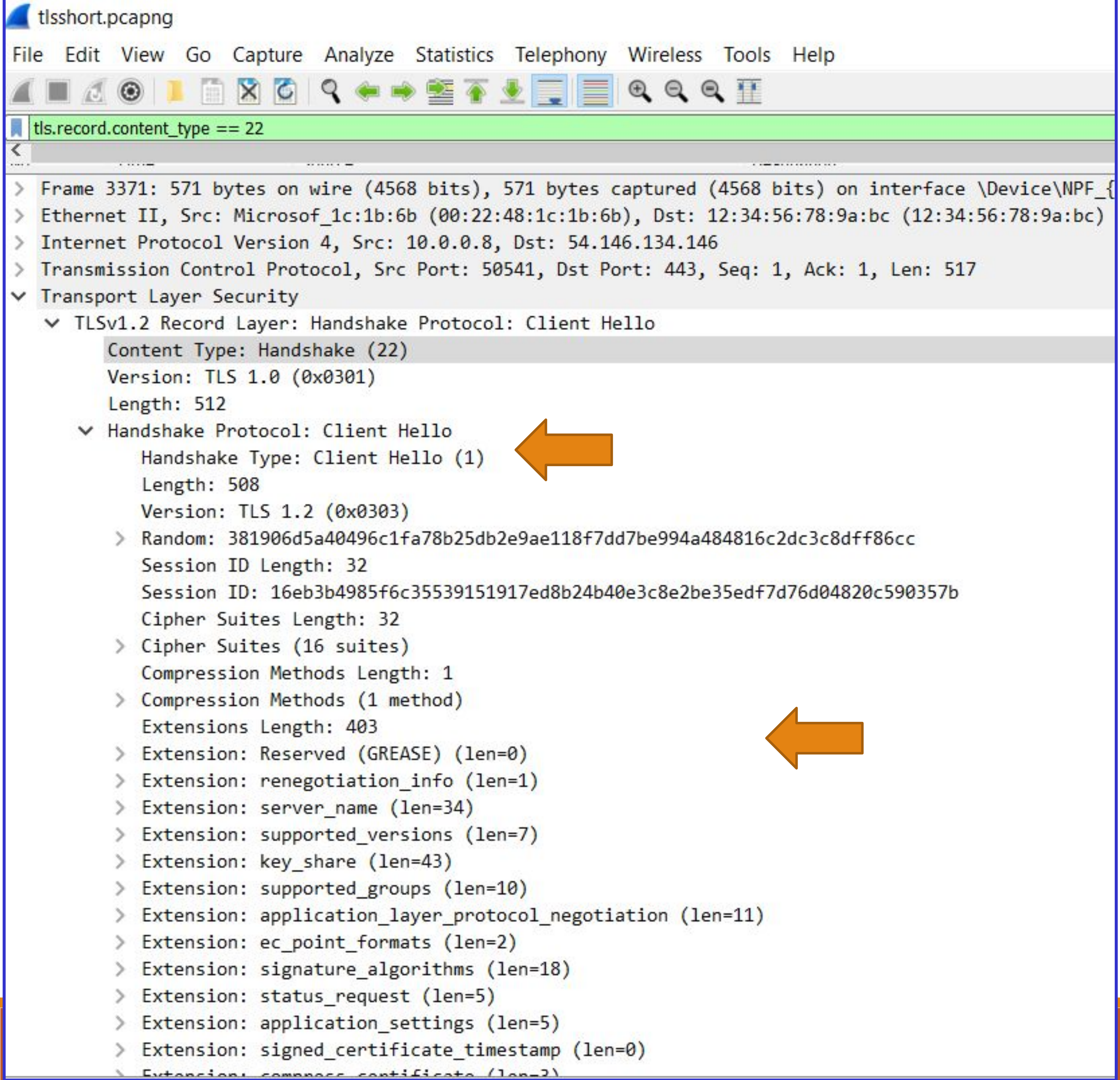
Interesting TLS Drafts

- TLS Encrypted Client Hello (draft-ietf-tls-esni-16)
- Deprecating Obsolete Key Exchange Methods in TLS 1.2 (draft-ietf-tls-deprecate-obsolete-kex-02)
- Hybrid key exchange in TLS 1.3 (draft-ietf-tls-hybrid-design-06)
- Merkle Tree Certificates for TLS (draft-davidben-tls-merkle-tree-certs-00)

TLS Encrypted Client Hello

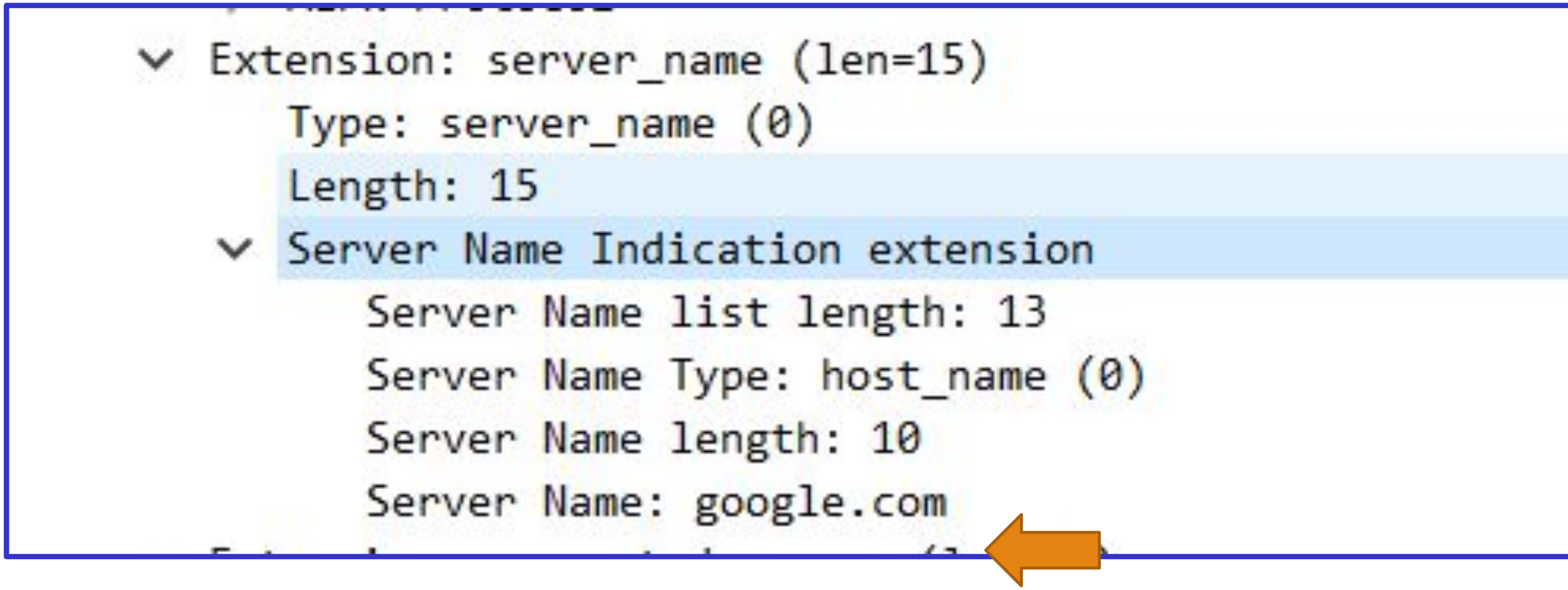
Although TLS 1.3 [RFC8446] encrypts most of the handshake, including the server certificate, there are several ways in which an on-path attacker can learn private information about the connection. The plaintext Server Name Indication (SNI) extension in ClientHello messages, which leaks the target domain for a given connection, is perhaps the most sensitive, unencrypted information in TLS 1.3.

https://datatracker.ietf.org/doc/draft-ietf-tls-esni/



What is in the Client Hello?

Server Name Indicator Extension



- Many TLS servers host multiple domains on the same IP address.
- Private origins may also be deployed behind a common provider, such as a reverse proxy. In such environments, the SNI remains the primary explicit signal used to determine the server's identity.

https://datatracker.ietf.org/doc/draft-ietf-tls-esni/

Deprecating Obsolete Key Exchange Methods in TLS1.2

- This document deprecates the use of RSA key exchange and Diffie Hellman over a finite field in TLS 1.2, and discourages the use of static elliptic curve Diffie Hellman cipher suites.
- Note that these prescriptions apply only to TLS 1.2 since TLS 1.0 and   1.1 are deprecated by [RFC8996] and TLS 1.3 either does not use the affected algorithm or does not share the relevant configuration options.

- **https://datatracker.ietf.org/doc/draft-ietf-tls-deprecate-obsolete-kex/**

Hybrid Key Exchange in TLS1.3

- Hybrid key exchange refers to using multiple key exchange algorithms simultaneously and combining the result with the goal of providing security even if all but one of the component algorithms is broken.
- It is motivated by transition to post-quantum cryptography. This document provides a construction for hybrid key exchange in the Transport Layer Security (TLS) protocol version 1.3.

https://datatracker.ietf.org/doc/draft-ietf-tls-hybrid-design/

Merkle Tree Certificates for TLS

- This document describes Merkle Tree certificates, a new certificate type for use with TLS. A relying party that regularly fetches information from a transparency service can use this certificate type as a size optimization over more conventional mechanisms with post- quantum signatures.
- Merkle Tree certificates integrate the roles of X.509 and Certificate Transparency, achieving comparable security properties with a smaller message size, at the cost of more limited applicability.

https://datatracker.ietf.org/doc/draft-davidben-tls-merkle-tree-certs/

Early Proposal: Many Signatures

- Authors' Note: This is an early draft of a proposal with many parts. While we have tried to make it as concrete as possible, we anticipate that most details will change as the proposal evolves.
- A typical TLS [RFC8446] handshake uses many signatures to authenticate the server public key. In a certificate chain with an end-entity certificate, an intermediate certificate, and an implicit trust anchor, there are two X.509 signatures [RFC5280].

Intermediate Certs, CT, OCSP

- Intermediate certificates additionally send an extra public key. If the handshake uses Certificate Transparency (CT) [RFC6962], each Signed Certificate Timestamp (SCT) also carries a signature.
- CT policies often require two or more SCTs per certificate [APPLE-CT] [CHROME-CT].
- If the handshake staples an OCSP response [RFC6066] for revocation, that adds an additional signature.

(Note: OCSP = Online Certificate Status Protocol)

Current and Post Quantum Signatures

- Current signature schemes can use as few as 32 bytes per key and 64 bytes per signature [RFC8032], but post-quantum replacements are much larger.
- For example, Dilithium3 [Dilithium] uses 1,952 bytes per public key and 3,293 bytes per signature.
- A TLS Certificate message with, say, four Dilithum3 signatures (two X.509 signatures and two SCTs) and one intermediate CA's Dilithium3 public key would total 15,124 bytes of authentication overhead.
- Falcon-512 and Falcon-1024 [Falcon] would, respectively, total 3,561 and 6,913 bytes.

Merkle Tree Certificates

This document introduces Merkle Tree Certificates, an optimization that authenticates a subscriber key using under 1,000 bytes.

https://datatracker.ietf.org/doc/draft-davidben-tls-merkle-tree-certs/

What is a Binary Search?

Binary search is an algorithm used to find a specific value in a sorted list of elements.   It is an efficient algorithm that quickly locates the desired element by repeatedly dividing the search space in half.

1. **Initial setup**: Assume we have a sorted list of numbers. Let's say we want to find the number 7 in the list

[2, 3, 4, 5, 6, 7, 8, 9, 10].

Divide the List

2. **Divide the list**: We start by looking at the middle element of the list, which is 6. Since 7 is greater than 6, we know it must be in the second half of the list.

[2, 3, 4, 5, 6, 7, 8, 9, 10]

3. **Divide again**: We now divide the second half of the list in half again and look at the middle element, which is 8. Since 7 is less than 8, we know it must be in the first half of the remaining elements.

[2, 3, 4, 5, 6, 7, 8, 9, 10]

Find the answer

4. **Final step**: We repeat the process of dividing the search space until we find the desired element or determine that it doesn't exist. In this case, we find the number 7.

[2, 3, 4, 5, 6, 7, 8, 9, 10]

Binary sort eliminates half of the remaining elements in each iteration, making it highly efficient.

Please note that the list in the example was already sorted. If the list is not sorted, you would need to sort it first before applying binary sort.

Why Binary Search?

- When you have to search a large number of items, it is better to use a binary search rather than sequential.

- Let’s now discuss binary trees and Merkle / rachet trees which are used in current cryptography (MLS / TLS)

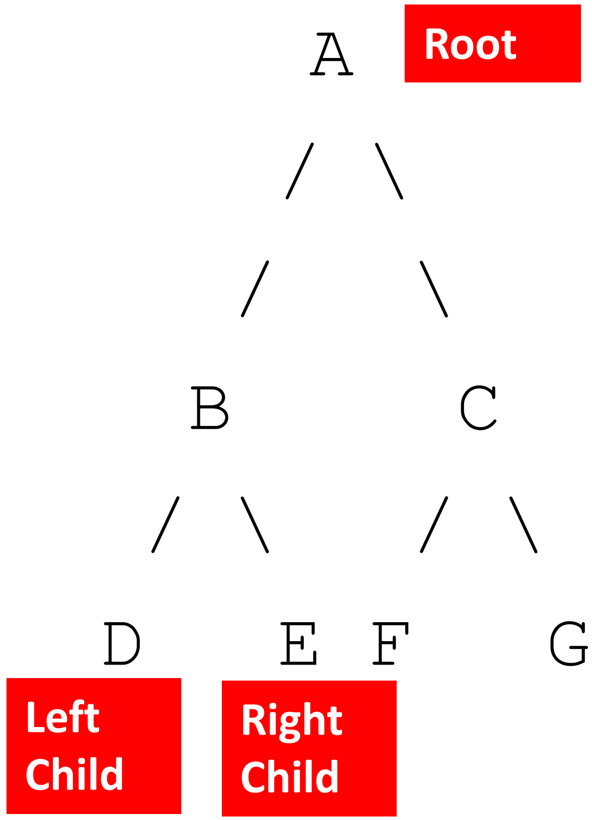
What is a Binary Tree?

- A binary tree is a type of data structure that consists of nodes connected in a hierarchical manner.
- Each node in a binary tree can have at most two children, referred to as the left child and the right child.
- The topmost node of the tree is called the root node.

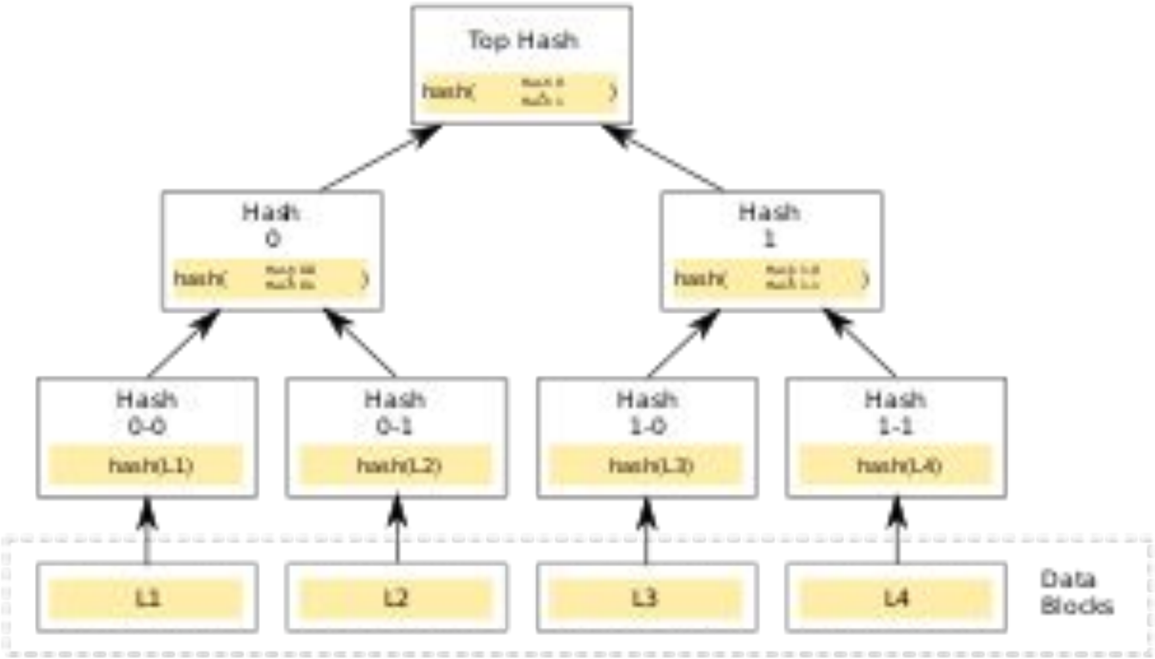
```
         A
        / \
       /   \
      B     C
     / \   / \
    D   E F   G
```

What is a Binary Tree?

- A binary tree is a type of data structure that consists of nodes connected in a hierarchical manner.
- Each node in a binary tree can have at most two children, referred to as the left child and the right child.
- The topmost node of the tree is called the root node.

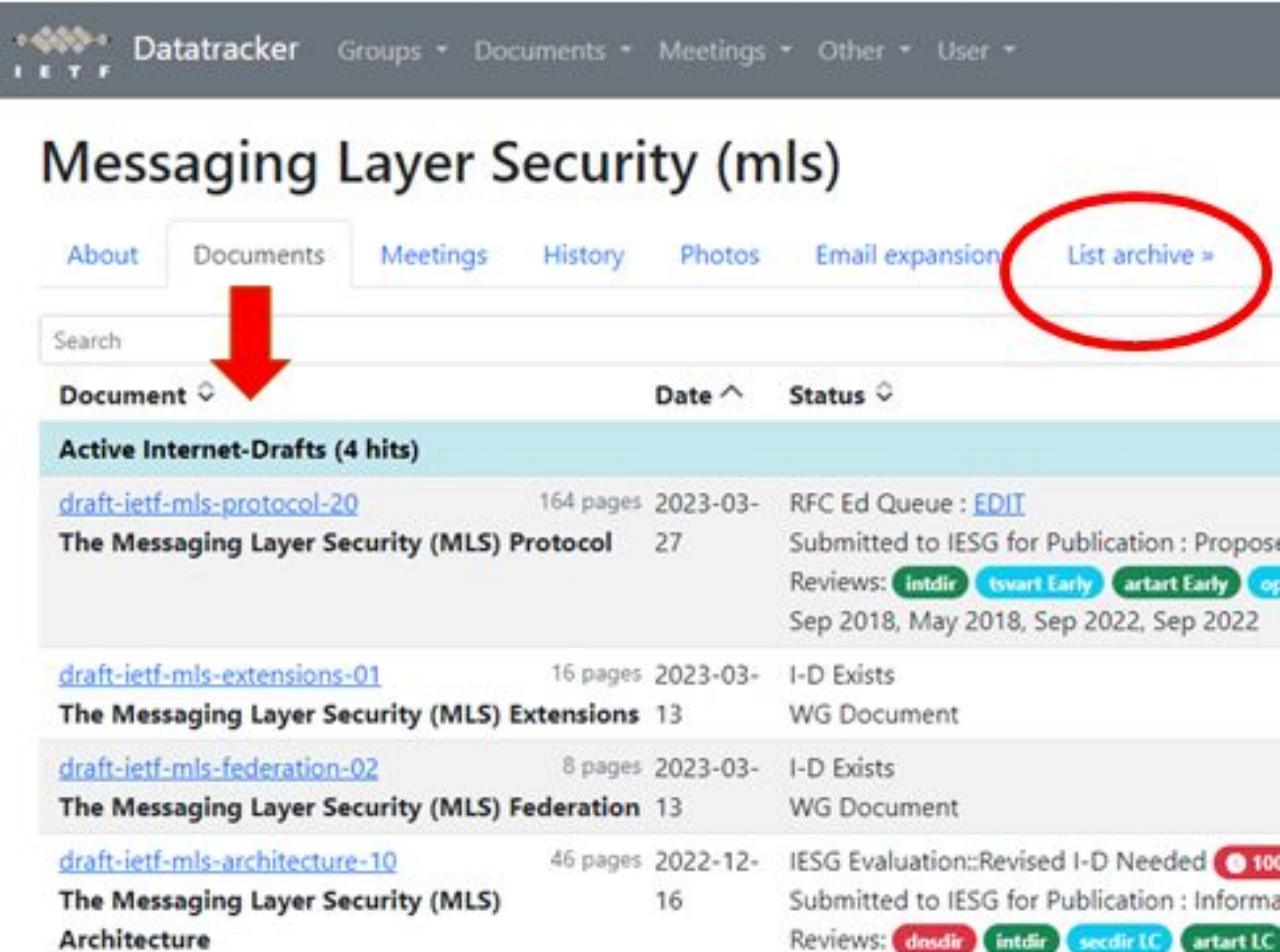


What is a Merkle Tree?



A hash or Merkle tree is a tree of hashes in which the leaves (i.e., leaf nodes, sometimes also called "leafs") are hashes of data blocks in, for instance, a file or set of files.

https://en.wikipedia.org/wiki/Merkle_tree


Datatracker
Groups
Documents
Meetings
Other
User
Messaging Layer Security (mls)
About
Documents
Meetings
History
Photos
Email expansion
List archive »
Search
Document
Date
Status
Active Internet-Drafts (4 hits)
draft-ietf-mls-protocol-20
164 pages
2023-03-27
The Messaging Layer Security (MLS) Protocol
RFC Ed Queue : EDIT
Submitted to IESG for Publication :
Reviews: intdir tsvart Early artart Early
Sep 2018, May 2018, Sep 2022, Sep 2022
draft-ietf-mls-extensions-01
16 pages
2023-03-13
The Messaging Layer Security (MLS) Extensions
I-D Exists
WG Document
draft-ietf-mls-federation-02
8 pages
2023-03-13
The Messaging Layer Security (MLS) Federation
I-D Exists
WG Document
draft-ietf-mls-architecture-10
46 pages
2022-12-16
The Messaging Layer Security (MLS) Architecture
IESG Evaluation::Revised I-D Needed
Submitted to IESG for Publication :
Reviews: dnsdir intdir secdir LC artart LC

Interesting MLS Drafts

- The Messaging Layer Security (MLS) Protocol (draft-ietf-mls-protocol-20)
- The Messaging Layer Security (MLS) Extensions (draft-ietf-mls-extensions-01)
- The Messaging Layer Security (MLS) Federation (draft-ietf-mls-federation-02)
- The Messaging Layer Security (MLS) Architecture (draft-ietf-mls-architecture-10)

Datatracker Groups Documents Meetings Other User

Post-Quantum Use In Protocols (pquip)

About | Documents | Meetings | History | Photos | Email expansions | List ar

Document		Date
Active Internet-Draft (1 hit)		
draft-ietf-pquip-pqt-hybrid-terminology-00 **Terminology for Post-Quantum Traditional Hybrid Schemes**	13 pages	2023-
Related Internet-Draft (1 hit)		
draft-driscoll-pqt-hybrid-terminology-02 **Terminology for Post-Quantum Traditional Hybrid Schemes**	13 pages	2023-

Terminology for Post Quantum

One aspect of the transition to post-quantum algorithms in cryptographic protocols is the development of hybrid schemes that incorporate both post-quantum and traditional asymmetric algorithms. This document defines terminology for such schemes. It is intended to be used as a reference and, hopefully, to ensure consistency and clarity across different protocols, standards, and organisations.

https://datatracker.ietf.org/doc/draft-ietf-pquip-pqt-hybrid-terminology/

What Terminology?

Traditional Algorithm: An asymmetric cryptographic algorithm based on integer factorisation, finite field discrete logarithms or elliptic curve discrete logarithms.

Post-Quantum Algorithm: An asymmetric cryptographic algorithm that is believed to be secure against attacks using quantum computers as well as classical computers.

Component Algorithm: Each cryptographic algorithm that forms part of a cryptographic scheme.

Single-Algorithm Scheme: A cryptographic scheme with one component algorithm. A single-algorithm scheme could use either a traditional algorithm or a post-quantum algorithm.

Multi-Algorithm Scheme: A cryptographic scheme with more than one component algorithm.

https://datatracker.ietf.org/doc/draft-ietf-pquip-pqt-hybrid-terminology/

Changes for Post Quantum

- Need cipher suites
- Need TLS changes (key exchange)
- Need certificate / signing changes
- Need implementation in crypto libraries (OpenSSL, etc)
- Need changes to compilers (Java, C++, etc)
- Need changes to web servers (Apache), data base servers, etc.
- Need to change application programs

Questions?

Contact:

info@iiesoc.in

president@industrynetcouncil.org