International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

X.1409

(07/2022)

SERIES X: DATA NETWORKS, OPEN SYSTEM COMMUNICATIONS AND SECURITY

Secure applications and services (2) – Distributed ledger technology (DLT) security

Security services based on distributed ledger technology

Recommendation ITU-T X.1409

ITU-T X-SERIES RECOMMENDATIONS

DATA NETWORKS, OPEN SYSTEM COMMUNICATIONS AND SECURITY

PUBLIC DATA NETWORKS	X.1–X.199
OPEN SYSTEMS INTERCONNECTION	X.200–X.299
INTERWORKING BETWEEN NETWORKS	X.300–X.399
MESSAGE HANDLING SYSTEMS	X.400–X.499
DIRECTORY	X.500–X.599
OSI NETWORKING AND SYSTEM ASPECTS	X.600–X.699
OSI MANAGEMENT	X.700–X.799
SECURITY	X.800–X.849
OSI APPLICATIONS	X.850–X.899
OPEN DISTRIBUTED PROCESSING	X.900–X.999
INFORMATION AND NETWORK SECURITY	
General security aspects	X.1000–X.1029
Network security	X.1030–X.1049
Security management	X.1050–X.1069
Telebiometrics	X.1080–X.1099
SECURE APPLICATIONS AND SERVICES (1)	
Multicast security	X.1100–X.1109
Home network security	X.1110–X.1119
Mobile security	X.1120–X.1139
Web security (1)	X.1140–X.1149
Application Security (1)	X.1150–X.1159
Peer-to-peer security	X.1160–X.1169
Networked ID security	X.1170–X.1179
IPTV security	X.1180–X.1199
CYBERSPACE SECURITY	
Cybersecurity	X.1200–X.1229
Countering spam	X.1230–X.1249
Identity management	X.1250–X.1279
SECURE APPLICATIONS AND SERVICES (2)	
Emergency communications	X.1300–X.1309
Ubiquitous sensor network security	X.1310–X.1319
Smart grid security	X.1330–X.1339
Certified mail	X.1340–X.1349
Internet of things (IoT) security	X.1350–X.1369
Intelligent transportation system (ITS) security	X.1370–X.1399
Distributed ledger technology (DLT) security	**X.1400–X.1429**
Application Security (2)	X.1450–X.1459
Web security (2)	X.1470–X.1489
CYBERSECURITY INFORMATION EXCHANGE	
Overview of cybersecurity	X.1500–X.1519
Vulnerability/state exchange	X.1520–X.1539
Event/incident/heuristics exchange	X.1540–X.1549
Exchange of policies	X.1550–X.1559
Heuristics and information request	X.1560–X.1569
Identification and discovery	X.1570–X.1579
Assured exchange	X.1580–X.1589
Cyber Defence	X.1590–X.1599
CLOUD COMPUTING SECURITY	
Overview of cloud computing security	X.1600–X.1601
Cloud computing security design	X.1602–X.1639
Cloud computing security best practices and guidelines	X.1640–X.1659
Cloud computing security implementation	X.1660–X.1679
Other cloud computing security	X.1680–X.1699
QUANTUM COMMUNICATION	
Terminologies	X.1700–X.1701
Quantum random number generator	X.1702–X.1709
Framework of QKDN security	X.1710–X.1711
Security design for QKDN	X.1712–X.1719
Security techniques for QKDN	X.1720–X.1729
DATA SECURITY	
Big Data Security	X.1750–X.1759
Data protection	X.1770–X.1789
IMT-2020 SECURITY	X.1800–X.1819

For further details, please refer to the list of ITU-T Recommendations.

Recommendation ITU-T X.1409

Security services based on distributed ledger technology

Summary

Distributed ledger technology (DLT) features include immutability, data sharing, decentralization, and tamper-resistance. Certain security services can benefit from the decentralized nodes of DLT to solve problems such as single point of failure, bottleneck performance and tampering.

Recommendation ITU-T X.1409 identifies aspects to be evaluated before delivering a security service based on DLT and provides examples to implement four security services which could be delivered based on DLT, namely:

- DLT-based public-key certificate management;
- DLT-based software defined perimeter;
- DLT-based threat intelligence sharing; and
- DLT-based security audit.

History

Edition	Recommendation	Approval	Study Group	Unique ID*
1.0	ITU-T X.1409	2022-07-29	17	11.1002/1000/15035

Keywords

Blockchain, distributed ledger technology (DLT), security service.

* To access the Recommendation, type the URL http://handle.itu.int/ in the address field of your web browser, followed by the Recommendation's unique ID. For example, http://handle.itu.int/11.1002/1000/11830-en.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure, e.g., interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had not received notice of intellectual property, protected by patents/software copyrights, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the appropriate ITU-T databases available via the ITU-T website at http://www.itu.int/ITU-T/ipr/.

Table of Contents

Introduction

Some traditional (in this Recommendation 'traditional' means 'not using DLT') security services face challenges such as single point of failure problem, bottleneck performance and tampering. Distributed ledger technology (DLT) features include immutability, data sharing, decentralization and tamper-resistance. As a tamper-resistant and auditable technology that is resilient to systemic failures, DLT supplies decentralized solutions to these challenges. Decentralized nodes of DLT could be used to solve single point of failure and bottleneck performance problems. DLT could also be used to improve cooperation among participants.

This Recommendation identifies aspects to be evaluated before delivering a security service based on DLT, and provides examples to implement four security services which could be delivered based on DLT, namely:

- DLT-based public-key certificate management;
- DLT-based software defined perimeter;
- DLT-based threat intelligence sharing; and
- DLT-based security audit.

Recommendation ITU-T X.1409

Security services based on distributed ledger technology

1 Scope

This Recommendation identifies aspects to be evaluated before delivering a security service based on distributed ledger technology (DLT), and provides examples to implement four security services which could be delivered based on DLT, namely:

- DLT-based public-key certificate management;
- DLT-based software defined perimeter;
- DLT-based threat intelligence sharing; and
- DLT-based security audit.

2 References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published. The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

[ITU-T X.509] Recommendation ITU-T X.509 (2019) | ISO/IEC 9594-8:2020, *Information technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks*.

3 Definitions

3.1 Terms defined elsewhere

This Recommendation uses the following terms defined elsewhere:

3.1.1 accepting host [b-SDP Spec 1.0]: A host accepts the communication from the initiating host after the controller authenticates and authorizes the connection.

3.1.2 blockchain [b-ITU-T X.1400]: A type of distributed ledger which is composed of digitally recorded data arranged as a successively growing chain of blocks with each block cryptographically linked and hardened against tampering and revision.

3.1.3 certificate revocation list (CRL) [ITU-T X.509]: A signed list indicating a set of public-key certificates that are no longer considered valid by the issuing certificate authority. In addition to the generic term certificate revocation list (CRL), some specific CRL types are defined for CRLs that cover particular scopes.

3.1.4 distributed ledger [b-ITU-T X.1400]: A type of ledger that is shared, replicated, and synchronized in a distributed and decentralized manner.

3.1.5 distributed ledger technology (DLT) network [b-ISO 22739]: Network of DLT nodes which make up a DLT system.

3.1.6 DLT oracle [b-ITU-T X.1400]: A service that supplies information to a distributed ledger using data from outside of the distributed ledger system.

3.1.7 DLT system [b-ITU-T X.1400]: A system that implements a distributed ledger.

3.1.8 genesis block [b-ITU-T X.1400]: The first block in a blockchain that serves to initialize the blockchain.

3.1.9 initiating host [b-SDP Spec 1.0]: A host that initiates communication to the controller and to the accepting hosts.

3.1.10 ledger [b-ITU-T X.1400]: Information store that keeps final and definitive (immutable) records of transactions.

3.1.11 node [b-ITU-T X.1400]: Device or process that participates in a distributed ledger network.

3.1.12 transaction [b-ITU-T X.1400]: Whole of the exchange of information between nodes. A transaction is uniquely identified by a transaction identifier.

3.2 Terms defined in this Recommendation

This Recommendation defines the following terms:

3.2.1 DLT client: A client to access the distributed ledger technology (DLT) system.

3.2.2 resume (a certificate): An action to make a suspended certificate active.

3.2.3 software defined perimeter (SDP): A security framework that gives application owners the ability to deploy perimeter functionality where needed in order to isolate services from unsecured networks. It replaces physical appliances with logical components that operate under the control of the application owner and provides access to application infrastructure only after device attestation and identity verification.

NOTE – Definition is adapted from [b-SDP Spec 1.0].

3.2.4 software defined perimeter (SDP) controller: A controller that determines which SDP hosts can communicate with each other. It may relay information to external authentication services such as attestation, geo-location and identity servers.

NOTE – Definition is adapted from [b-SDP Spec 1.0].

3.2.5 suspend (a certificate): An action to cause a certificate to temporarily cease to be active.

4 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

ASDPH	Accepting SDP Host
CRL	Certificate Revocation List
DDoS	Distributed Denial of Service
DLT	Distributed Ledger Technology
H(e)NB	Home NodeB or Home eNodeB
ISDPH	Initiating SDP Host
OCSP	Online Certificate Status Protocol
P2P	Point to Point
PKI	Public Key Infrastructure
SAML	Security Assertion Markup Language
SDP	Software Defined Perimeter
SeGW	Security Gateway
TLS	Transport Layer Security

URL Uniform Resource Locator

5 Conventions

In this Recommendation:

The keywords "**is required to**" indicate a requirement which must be strictly followed and from which no deviation is permitted, if conformance to this Recommendation is to be claimed.

The keywords "**is recommended**" indicate a requirement which is recommended but which is not absolutely required. Thus, this requirement need not be present to claim conformance.

The keywords "**can optionally**" indicate an optional requirement which is permissible, without implying any sense of being recommended. This term is not intended to imply that the vendor's implementation must provide the option, and the feature can be optionally enabled by the network operator/service provider. Rather, it means the vendor may optionally provide the feature and still claim conformance with the specification.

6 Overview

This Recommendation analyses four kinds of typical security services that could benefit from the deployment of distributed ledger technology (DLT): public-key certificate management, software defined perimeter, threat intelligence sharing and security audit. The challenges to these traditional (in this Recommendation 'traditional' means 'not using DLT') security services and the benefits of delivered security services based on DLT are analysed in Appendix I.

Each of the example DLT-based security services is described with the basic concept and a brief description. The basic concept provides the main idea of the DLT-based security service, such as the participants in the DLT system, and information recorded in the DLT ledger. The brief description shows interactions with the DLT system, and considerations related to the storage cost of the DLT nodes. The interactions indicate how the DLT-based security service works. On one side, the nodes send transactions to the DLT system and on the other side the nodes queries information recorded in the DLT ledger. Blockchain, a typical distributed ledger, is recommended to deploy these DLT-based security services.

7 Evaluation on whether to deliver a security service based on DLT

DLT features such as decentralization and tamper-resistance make it an efficient solution to certain security problems, however, it is recommended to take comprehensive considerations into account to deliver a security service based on DLT.

First, the security service needs to have one or more of the following characteristics matching the capabilities of DLT:

- Multiple participants;
- Decentralized deployment and operation;
- Distribution to improve robustness or adaptability;
- Data may be tampered with or may be forged or inconsistent;
- Rules need to be made by multiple participants;
- Requirement of traceability.

Once this pre-requisite is met, it is recommended to evaluate the benefit of delivering the security service based on DLT, including the gains of DLT over the traditional security service against the efficiency and cost of the DLT-based solution. The aim of the evaluation is to ensure that the challenges of a traditional security service can be solved by DLT and that DLT is beneficial to the security service.

Next, it is recommended to consider the performance requirements of the security service, which may help to choose adaptive DLT components. Main concerns may include transaction delay, handing capacity, business scale, bandwidth utilization caused by point to point (P2P) communication, read-only and append-only data, resource consumption restrictions, etc.

It is recommended to use DLT if the evaluation results about the security services are positive.

8 DLT-based public-key certificate management

8.1 Basic concept

In a DLT-based public-key certificate management system, public-key certificates issued by certification authorities are recorded into the ledger after verification and consensus among its participants in this DLT system. The public-key certificates will be trusted by the participants in the DLT system once they are recorded in the ledger.

Figure 1 illustrates the participants of a DLT-based public-key certificate management system. The validator nodes are peer to peer nodes in the distributed ledger network which are responsible for the verification of the submitted public-key certificates, generation of new blocks and maintenance of the ledger. They could be operators, vendors, certification authorities [ITU-T X.509], etc. The user nodes and relying parties [ITU-T X.509] connect to the system as DLT clients via the validator nodes.

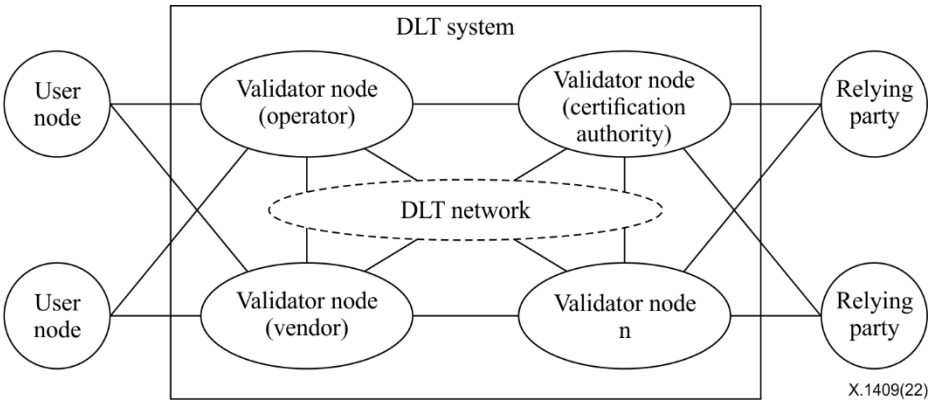

Figure 1 – Participants of the DLT-based public-key certificate management system

NOTE – This Recommendation focuses on the traditional public-key certificates issued by traditional certification authorities. At the time of publication, work on a Recommendation complimentary to [ITU-T X.509] was under development to define a decentralized public key infrastructure (PKI) in such a way that PKI domains will be able to be federated possibly worldwide. This Recommendation recommends using the public-key certificates, syntax and migration methods that will be defined in the future Recommendation once it is published.

8.2 Brief description

8.2.1 Publication of a public-key certificate

The public-key certificate user is required to obtain a public-key certificate issued by the certification authority. The public-key certificate can be published into the ledger in the following process:

1) The public-key certificate is required to be generated by the certification authorities. The private key related to the public-key certificate is required to be kept securely by the public-key certificate user.

2) The user node submits the public-key certificate and the certificate's-status-publish request to one or multiple validator nodes in the DLT network. The request contains the public-key

certificate, its status, intermediate public-key certificates, and root public-key certificate. The validator node receiving the request publishes the certificate's-status-publish request to other validator nodes in the DLT network. The transport layer security (TLS) protocol [b-IETF RFC 8446] is recommended to protect the communications between different nodes, including user nodes, validator nodes and relying parties.

3) The validator nodes in the DLT network verify the request including the public-key certificate and the certificate's status when they receive the request. Each validator node is recommended to verify the request. A list of trusted certification authorities needs to be defined by the participants. The verification includes the check of basic public-key certificates, processing intermediate certificates, explicit policy indicator processing, as defined in [ITU-T X.509]. The policy could be defined by the participants. Optionally, a list of distinguished names [ITU-T X.509] for designated public-key certificates can be defined to ensure only the public-key certificates in the list can be recorded into the ledger.

   NOTE – The consensus mechanism relates to the core layer of the distributed ledgers [b-ITU-T FG DLT D3.1], which has not been included in this Recommendation.

4) When the request is verified as valid, the validator node generates a new block containing the public-key certificate and the certificate's status, or the public-key certificate's identifier (i.e., hash value) and the certificate's status in the request. The public-key certificate's status is recommended to be "normal". Then it sends the new block to other nodes in the DLT network.

The system is recommended to have the capability to revoke, suspend, resume and renew an existing public-key certificate. The methods are defined in clause A.1.

The requests to publish, revoke, suspend and resume a public-key certificate can be seen as different kinds of transactions. Multiple public-key certificates and their statuses from different kinds of requests are recommended to be recorded by a validator node who receives them in a designated time period into one block. The sequence of the public-key certificates and their status is recommended to reflect the sequence of the request. The validator nodes receive and retain all kinds of the above requests in a designated time period. They verify the received requests, generate a new block based on the consensus method and then publish the new block into the DLT network. The new block contains the verified public-key certificates and their statuses from multiple requests in the time period. The public-key certificates are recommended to be recorded into the block according to the time sequence of the requests. Then the blocks are linked into blockchain according to the time sequence of the requests.

The relying party is recommended to have the capability to verify the public-key certificate with the help of a DLT ledger. The verification methods are defined in clause A.2.

8.2.2 Storage of public-key certificates

The time to record the public-key certificates into the DLT ledger could reflect the public-key certificate validity. Use of the X.509 public-key certificates [ITU-T X.509] is required. The validity field of the public-key certificate could be omitted, if it is assumed that the public-key certificate is valid only if it is recorded in the DLT ledger and the status is "normal". The public-key certificate with no validity field and the expected validity information could be contained in the public-key certificate publish request (short for public-key certificate and the certificate's-status-publish request). The validity information could be used to determine the validity period of the public-key certificate. The validator nodes in the DLT-based public-key certificate management system find the start time of the validity period from the validity information in the received public-key certificate publish request and verify the submitted public-key certificate as in step 3 of publishing a public-key certificate in clause 8.2.1. After the start time of the validity period, according to the verification result, the verified public-key certificate can be recorded into the DLT using the method in step 4 of the process of publishing a public-key certificate described in clause 8.2.1.

The public-key certificate has an expiration date. When the public-key certificate expires, or is revoked, the public-key certificate is invalid and will no longer be trusted. The public-key certificate storage optimization is recommended to be supported as defined in clause A.3.

Use cases can be found in Appendix II.

9 DLT-based software defined perimeter

9.1 Basic concept

Software defined perimeter (SDP) is a security framework that gives application owners the ability to deploy perimeter functionality where needed in order to isolate services from unsecured networks. It replaces physical appliances with logical components that operate under the control of the application owner and provides access to application infrastructure only after device attestation and identity verification. SDP controller is responsible for collecting authorization policies of accepting SDP hosts (ASDPHs, e.g., devices or services deployed in a cloud network), and then authenticating and authorizing the initiating SDP hosts (ISDPHs, e.g., user's terminals) according to the policies [b-SDP Spec 1.0].

By using DLT, SDP controllers, ISDPHs and ASDPHs can connect to each other to form a DLT network, as shown in Figure 2. ISDPHs and ASDPHs work as DLT clients. ASDPHs publish their authorization policies into the DLT ledger. The SDP controllers are peer to peer nodes in the distributed ledger network to verify and endorse the information recorded into the DLT ledger. When ISDPHs request access to ASDPHs, the nodes in the DLT system could perform access control [b-ITU-T X.1252] by using DLT-based SDP according to the policies recorded in the DLT ledger.

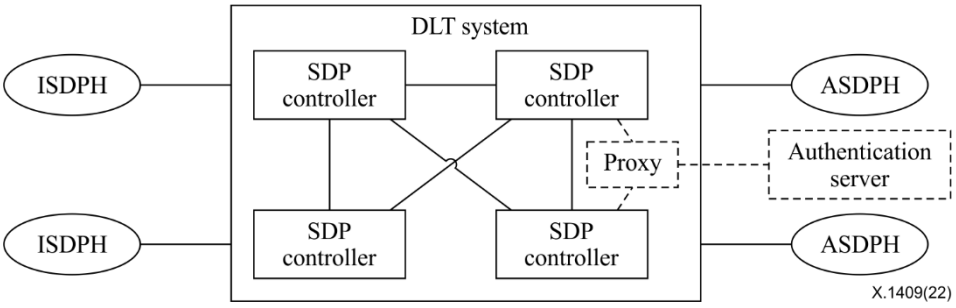

Figure 2 – Participants of the DLT-based SDP system

The information recorded in the ledger can be accessed by all the participants in the system. Some ASDPHs may prefer not to publish their detailed authentication policies, since the policies may reflect the characteristics or privacy of their services. Based on this consideration, the function of authorization could be implemented by the nodes in the DLT system and by the ASDPHs themselves.

9.2 Brief description

9.2.1 Authorization by nodes in the DLT system

The SDP controllers are peer to peer nodes in the DLT system. They authenticate ASDPHs, collect the authorization policies, and then authenticate and authorize the ISDPHs according to the policies. The detailed operations are defined in clause B.1.

1) It is recommended that the ASDPHs authenticate and submit the supported authorization policies to the nodes in the DLT system.
2) It is recommended that each ISDPH authenticate to the nodes in the DLT system.

3) It is recommended that the ISDPH initiate access connection to and obtain the access service from the ASDPH listed in the authorized list.

9.2.2 Authorization by ASDPHs

The nodes in the DLT system authenticate the ISDPHs and verify and endorse the attributes of the ISDPHs. The verified information and attributes are recorded into the DLT ledger. The ASDPHs authenticate and authorize the ISDPHs according to the endorsement recorded in the DLT ledger and their policies. The detailed operations are defined in clause B.2.

1) It is recommended that the ISDPH authenticate to the nodes in the DLT system.

2) It is recommended that the ISDPH initiate access connection to the ASDPH.

3) It is recommended that the ASDPH authorize the ISDPH according to the endorsement recorded in the DLT ledger.

9.2.3 Support for multiple authentication services

Multiple authentication services (e.g., PKI [ITU-T X.509], security assertion markup language (SAML) [b-IETF RFC 7522], OpenID [b-OpenID], OAuth [b-IETF RFC 6749]) are supported in SDP. The nodes in the DLT system are required to authenticate the ISDPH with the help of other entities, such as authentication servers. These entities are responsible for the verification of the ISDPH's authentication credential and attributes. The nodes in the DLT system may also rely on information provided by external servers such as attestation, geolocation and/or identity servers.

In such cases, it is recommended that the nodes in the DLT system send the authentication request received from the ISDPH to a proxy node in the DLT system. The proxy node is a special DLT node which could connect entities outside the DLT system, and supply information to a distributed ledger using data outside the distributed ledger system (i.e., DLT oracle).

The proxy node is recommended to forward the authentication credential and the attributes in the authentication request to corresponding entities, such as authentication servers or attribute attestation servers, according to the proxy's policy. These entities authenticate the credential and attest the attributes respectively and make their own decisions. Each entity will sign the decision using its private key, and then feedback the decision and the signature to the proxy. Then the proxy forwards them to the nodes in the DLT system. The nodes in the DLT system will decide on the authentication credential and the attributes, according to the authentication and authorization policy, and the feedback from the proxy.

A use case can be found in Appendix III.

10 DLT-based threat intelligence sharing

10.1 Basic concept

DLT-based threat intelligence sharing aims to establish trust among threat intelligence providers for a better use of threat intelligence. It could provide a method to share and evaluate threat intelligence, and even provide information for a fine-grained charging method.

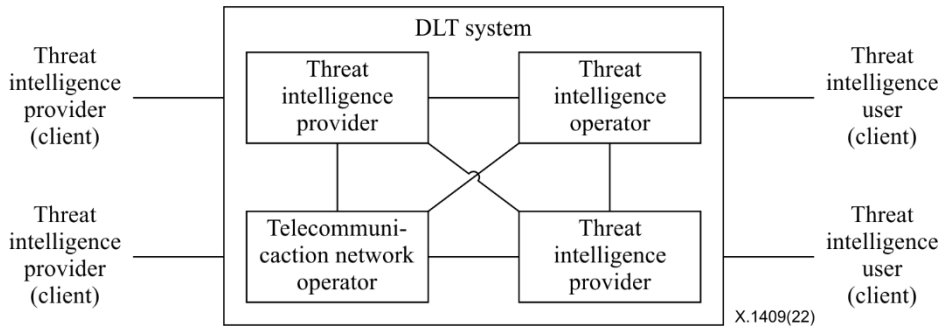

Figure 3 – Participants in DLT-based threat intelligence sharing system

The DLT system consists of threat intelligence providers, threat intelligence operators and telecommunication network operators, as shown in Figure 3. The threat intelligence providers could also act as DLT clients to submit threat intelligence to the DLT system, and then the threat intelligence and its source can be recorded into the ledger. The threat intelligence user acts as a DLT client and it could query and use the threat intelligence in the DLT system. The authorized participants could feedback and evaluate the threat intelligence. A referenced architecture of the DLT-based threat intelligence sharing system can be found in clause IV.1.

10.2 Brief description

10.2.1 Threat intelligence submission

1) A threat intelligence provider generates threat intelligence. It is recommended that the threat intelligence includes information of the malicious IP address, domain name, uniform resource locator (URL), security incident and the vulnerability. A method to collect and process threat intelligence can be found in [b-ITU-T X.1217].
2) The threat intelligence is submitted to the DLT system by the threat intelligence providers.
3) It is recommended that the threat intelligence be recorded into the ledger after consensus. The format of the threat intelligence is specified in [b-ITU-T X.1217].

10.2.2 Threat intelligence feedback

1) Threat intelligence user queries certain threat intelligence from the DLT system.
2) The threat intelligence user generates a feedback after the usage of the intelligence, and sends the feedback to the DLT system.
3) The DLT system generates a description about the threat intelligence and its provider, such as the credibility and quality of threat intelligence provider, etc., by using an existing description recorded in the ledger and the feedback from the user.
4) It is recommended that the new description be recorded into the ledger after consensus.

A use case can be found in clause IV.2.

11 DLT-based security audit

11.1 Basic concept

A DLT-based security audit aims to enhance security of logs and make a decentralized audit available to all participants. The logs are stored in the log servers, while the hash values of the logs could be recorded into the DLT ledger. This could ensure the integrity of the logs since some specific logs could be recorded into the DLT ledger, which make the decentralized audit available. The log servers and devices could act as the DLT clients to submit the logs into the DLT system.

The auditors could be the nodes in the DLT system. They could also act as DLT clients, as shown in Figure 4.

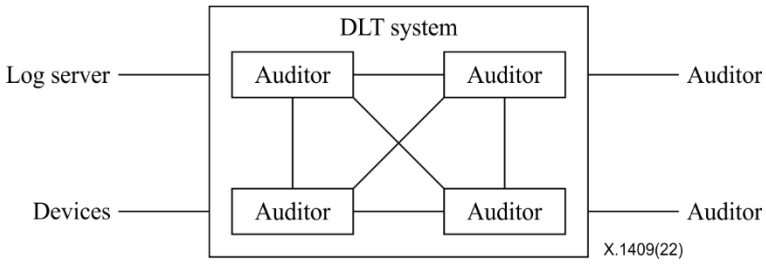

Figure 4 – Participants of the DLT-based security audit system

11.2 Brief description

11.2.1 Log submission

It is recommended that the log server and device be able to submit logs or their hashes into the DLT system.

1) It is required that the logs to be stored in the log server when they are generated. It is recommended that the hash values of the logs be submitted to the DLT system. For the specific logs (e.g., the volume of the logs is small), it is recommended that the logs be submitted to the DLT system.
2) It is recommended that the nodes in the DLT-based security audit system verify the submitted hash values or the logs, according to the verification policy of the audit scenarios.
3) It is recommended that the hash values or logs be recorded into the ledger after verification and consensus.

The logs recorded in the ledger could be used for a decentralized audit. For example, the operation on each public-key certificate could be logged and audited. The operations to generate, revoke, suspend, resume and renew a public-key certificate are recorded in the logs. It is recommended that the public-key certificate and the operations be recorded into the DLT ledger to enable audit. A detailed method can be found in clause V.

11.2.2 Log retention storage

There is usually a desired retention period for logs, because of storage expense. The logs can be deleted after the retention period. Based on this consideration, the storage of the ledger in the DLT nodes can be optimized when necessary.

Once the records in some continuous blocks of the DLT nodes exceed the retention period and can be deleted, it is recommended that an optimized block be generated based on the block information (such as the header and the records) of the original blocks to be deleted.

The optimized blocks are stored in the DLT nodes to formalize an optimized blockchain. The header of each optimized block is required to contain the header's hash value of the previous block. The genesis block in the optimized blockchain could be the same as that of the original blockchain. For the first optimized block in the optimized blockchain, the header is required to contain the header's hash value of the genesis block in the optimized blockchain. For the other optimized blocks, the header of the newly generated optimized block may optionally contain the hash value of the previous block to reflect the integrity of transactions. The body of each optimized block contains the hash value or the header's hash value of the last block in the original continuous blocks to be deleted. When the optimized block is generated, the original blocks to be deleted in the DLT nodes could be deleted. Then, the optimized blockchain contains the original genesis block, optimized blocks and none of the deleted original blocks. It is recommended, when there is a need, to back up the logs and the original blocks.

Annex A

DLT-based public-key certificate management methods

(This annex forms an integral part of this Recommendation.)

A.1 Revoke, suspend, resume or renew an existing public-key certificate

User needs to be able to revoke, suspend, resume or renew a public-key certificate to the certification authority [ITU-T X.509], and get the feedback from the certification authority, before the actions in the DLT-based public-key certificate management system.

1) A user node submits a public-key certificate management request (i.e., revoking, suspending or resuming request) to a validator node in the DLT-based public-key certificate management system. The request contains the feedback from the certification authority and the new status. The validator node will forward the request to multiple validator nodes in the DLT system.
2) The validator nodes in the DLT network receive and verify the user's request, and then record the new status into the DLT ledger after consensus. The new status reflects the result for the user's public-key certificate management request.

Multiple DLT-based public-key certificate management systems could form a composed DLT-based public-key certificate system while keeping their own ledgers. In such a case, a proxy node could be deployed by each of them to connect and communicate with the multiple systems in the composed system. The proxy node could be a validator node. The DLT-based public-key certificate system in which the user resides can be called as the user's host system. The proxy node of the user's host system forwards the user's public-key certificate management request and the result of the request (i.e., new status in the ledger) in the user's host system to the other proxy nodes. Each of the proxy nodes forwards the received request and result to the DLT-based public-key certificate system it resides in, and records the new status into the DLT ledger based on the received request and result.

The procedure to renew a public-key certificate comprises the procedures to publish a new public-key certificate and to revoke an existing public-key certificate. However, the distinguished names [ITU-T X.509] of the two public-key certificates are required to be the same. It is recommended that the request be signed by the private key corresponding to the public-key certificate to be renewed.

A.2 Verification of a public-key certificate at the relying party

When a relying party in the DLT network receives a public-key certificate or its identifier (i.e., a hash value), the relying party has to verify the public-key certificate. If the relying party maintains the DLT ledger, the verification can be done locally. Otherwise, the verification has to be done by the help of validator nodes in the DLT network. The procedures are as follows:

1) The relying party sends a certificate-verification-request to one or multiple validator nodes in the DLT network. The request contains the public-key certificate to be verified or its identifier.
2) The validator node verifies the public-key certificate in the received request, gets the verification result and feedbacks the verification result to the relying party. Then the relying party checks the public-key certificate by using the methods defined in [ITU-T X.509].

   When the public-key certificate verification request is received, the validator node searches the stored blocks of the ledger and checks whether there is a block recording the public-key certificate in the request or not.

If the block exists, the validator node locates the public-key certificate or the certificate's identifier and its status in the block. When the latest status in the block is "normal", the validator node obtains the result that the public-key certificate is valid. When the latest status in the block is "revoked" or "suspended", the validator node obtains the result that the public-key certificate is invalid.

If the block does not exist, the validator node sends a public-key certificate inquiry request to the DLT network. The request contains the public-key certificate's identifier. Another node in the DLT network which maintains the complete public-key certificate verifies the public-key certificate according to the identifier, and then feedbacks the public-key certificate and the verification result to the relying party via the corresponding validator node.

The relying party is recommended to use the verification service from a trusted party. It is recommended to protect the communication between the relying party and the verification node.

A.3 Storage optimization

Public-key certificates recorded in the DLT ledger will consume considerable amounts of storage expense with the increased number of public-key certificates. Public-key certificate has an expiration date. When a public-key certificate expires, or is revoked, it is invalid and will no longer be trusted. When all the public-key certificates in a block are invalid, the body of the block can be deleted. The header of the block remains to ensure the integrity of the ledger, as shown in Figure A.1. The hash pointer to the right is a hash pointer to the latest added block. The Merkle root hash in the header keeps the integrity of the body. The previous header hash (i.e., H(previous header)) keeps the integrity of the blocks. Due to this characteristic, the storage space of validator nodes could be optimized if needed. An example is shown in Figure A.2.

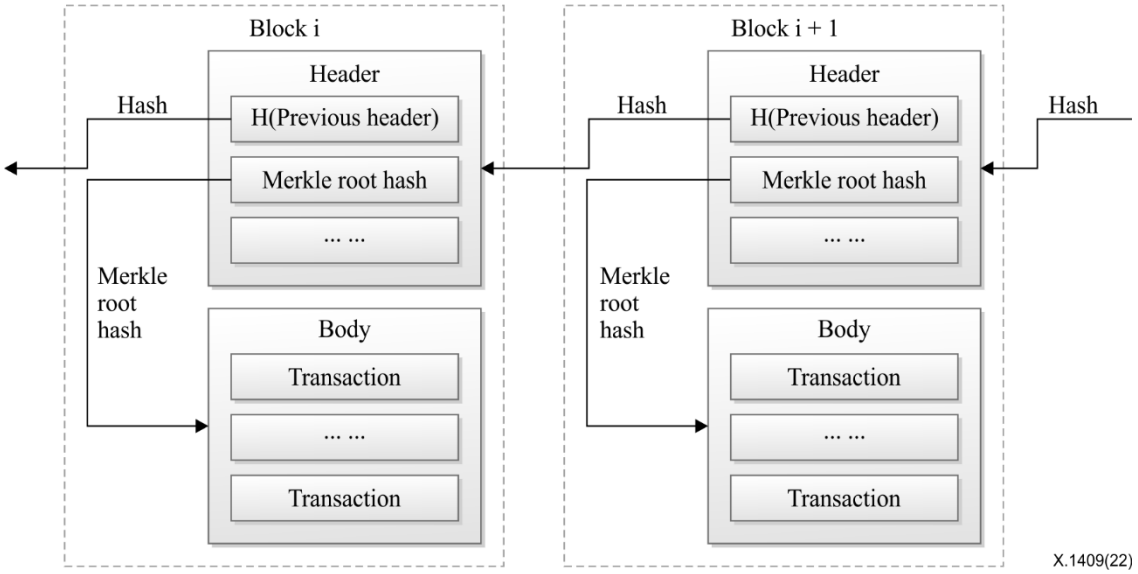

Figure A.1 – The blocks (H( ) means a hash algorithm)

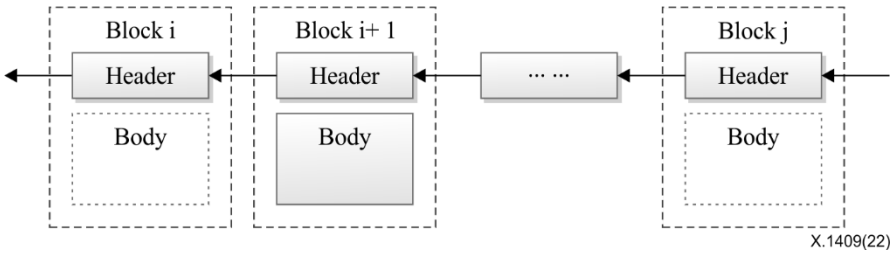

Figure A.2 – An example of the optimized storage of blocks

When the ledger records complete public-key certificates, the optimization could be done by the validator nodes themselves.

When the ledger records public-key certificate identifiers such as hash values, the optimization needs the help from another node, which backs up all the public-key certificates recorded in every block of each validator node. The node could be called a full node or backup node.

The backup node checks its stored blocks and public-key certificates periodically and verifies the status and expiration date of the public-key certificates recorded in each block, to determine whether all the public-key certificates recorded in a block are invalid or not. If it is determined that all the public-key certificates recorded in the block are invalid, the backup node sends an optimization notice containing deletion information to all the validator nodes. The deletion information contains the identifier of the block, and optionally contains all the public-key certificates recorded in the block. Every validator node verifies the deletion information to determine whether all the public-key certificates recorded in the block its stored are invalid. When it is determined that they are all invalid, the validator node deletes the body of the block corresponding to the identifier.

Annex B

Authentication and authorization by DLT-based software defined perimeter

(This annex forms an integral part of this Recommendation.)

B.1 Authorization by nodes in the DLT system

SDP controller nodes in the DLT system authenticate ASDPHs, collect the authorization policies, and then authenticate and authorize the ISDPHs according to the policies, as shown in Figure B.1.

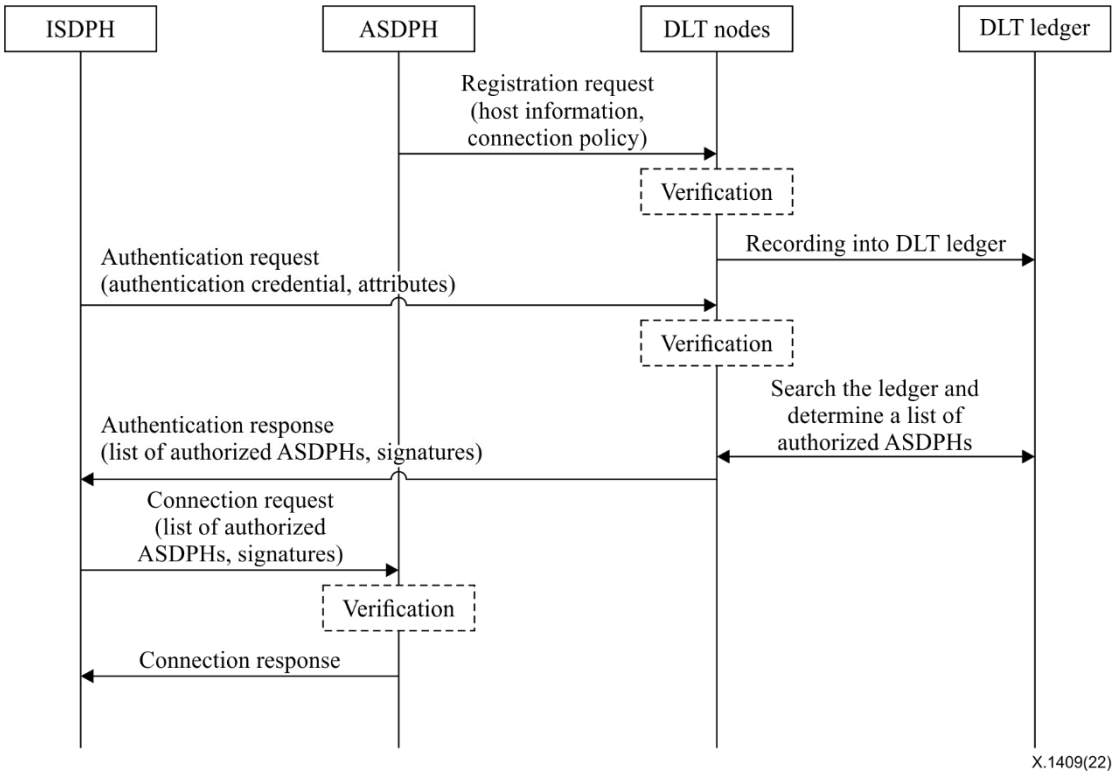

Figure B.1 – Authorization by nodes in the DLT system

1) ASDPH registers itself to and submits the supported authorization policies to the node in the DLT system.

   ASDPH sends a registration request to an SDP controller node in the DLT system. The registration request contains the ASDPH's host information (IP address, port, protocol, etc.) and supported policies for ISDPH's connection. It is recommended that the policies include the login identity, IP address, geo-position of the ISDPH, and the verification/endorsement policies of the DLT system. The host information and supported authorization policies are required to be signed by the ASDPH. It is recommended to include the signature in the registration request.

   The registration request will be verified by the nodes in the DLT system and then be recorded into the DLT ledger after verification and consensus.

2) ISDPH authenticates itself to an SDP controller node in the DLT system.

   ISDPH sends an authentication request containing its authentication credential and attributes (identity, IP address, geo-position, etc.) to an SDP controller node in the DLT system. The SDP controller node forwards the authentication request to multiple SDP controller nodes in the DLT system.

The SDP controller nodes in the DLT system verify the authentication request submitted by the ISDPH. After verification, the nodes search the DLT ledger and determine a list of ASDPHs which the ISDPH is authorized to access. Then the nodes send the authorized list of ASDPHs to the ISDPH. Each ASDPH in the list will be signed by one or multiple nodes in the DLT system, according to the ASDPH's authorization policy or the verification/endorsement policy of the DLT node.

Otherwise, the nodes in the DLT system will verify the authentication request submitted by the ISDPH and feedback the authentication result to the ISDPH. This happens when the authentication is implemented by the designated nodes. If the authentication is successful, the ISDPH queries the authorized list of ASDPHs which the ISDPH is authorized to access by sending the authentication result to the nodes in the DLT system. The nodes verify the authentication result in the query. When the authentication result is verified to be successful, the nodes query the authorized list of ASDPHs (i.e., by a smart contract) and then record the list into the DLT ledger.

3) An ISDPH initiates an access to and obtains the access service from a specific ASDPH listed in the authorized list recorded in the DLT ledger.

The ISDPH sends the initial access connection request to the ASDPH. The request contains the signatures signed by one or multiple nodes in the DLT system in the previous step for the ASDPH which is in the authorized ASDPHs list. If the ASDPH is in the list, it verifies the signature according to the supported connection policy and responds to the ISDPH after successful verification.

When there are no signatures in the initial access connection request message, the ASDPH queries and determines the authorized list of ASDPHs for the ISDPH in the DLT ledger. The ASDPH will respond and provide access service to the ISDPH only if the ASDPH is in the list.

B.2 Authorization by ASDPHs

The nodes in the DLT system authenticate the ISDPHs, verify and endorse the attributes of the ISDPHs. The verified information and attributes are recorded into the DLT ledger. The ASDPHs authenticate and authorize the ISDPHs according to the endorsement recorded in the DLT ledger and their policies, as shown in Figure B.2.

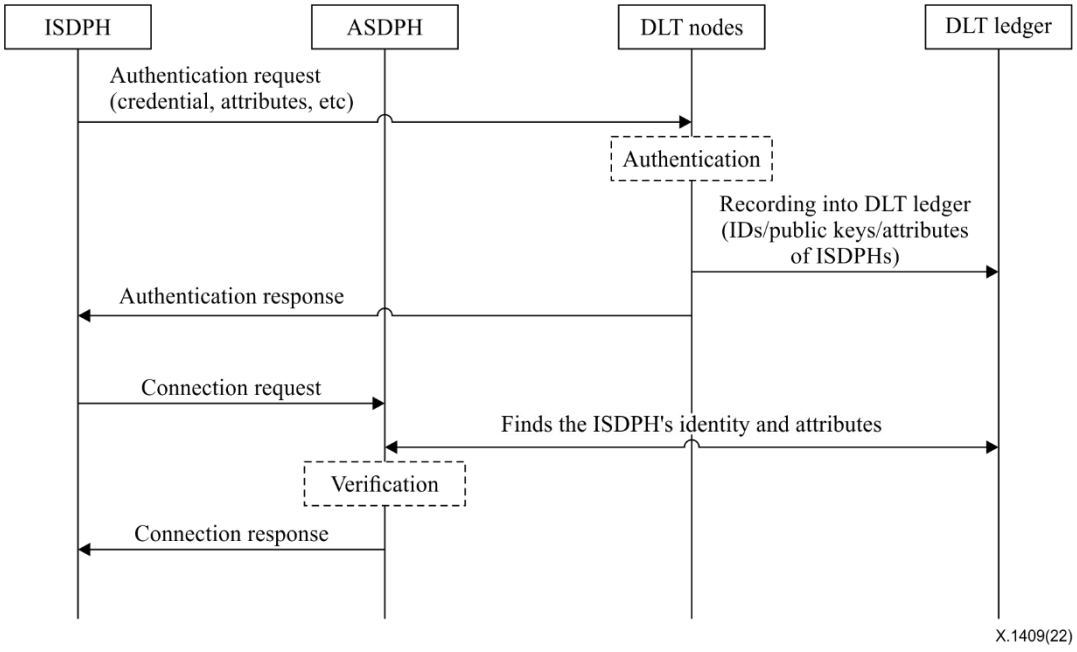

Figure B.2 – Authorization by ASDPHs

1) ISDPH authenticates itself to the nodes in the DLT system.

ISDPH sends an authentication request containing its authentication credential and attributes to the nodes in the DLT system. The authentication credential contains the ISDPH's identity (i.e., derived from the public key of the ISDPH). The request will be authenticated and attested by the nodes (such as SDP controllers) in the DLT system. After authentication and attestation, the ISDPH's identity, attributes in the request and their validity period will be recorded into the DLT ledger.

2) An ISDPH initiates access connection to an ASDPH.

The ISDPH sends an access connection request containing its authentication information to the ASDPH. During implementation, the authentication information could be the signature of the timestamp or a nonce, which is signed by the ISDPH's private key. When it is a signature of a nonce, the ISDPH needs to send another access connection request before the above request. Then the ASDPH provides a nonce and feedbacks the nonce to the ISDPH, according to the indicators in the request. The authentication information also contains the index to identify the record location of the ISDPH's identity and attributes recorded in the DLT ledger.

3) The ASDPH authorizes the ISDPH with the help of the information recorded in the DLT ledger.

The ASDPH finds the ISDPH's identity recorded in the DLT ledger by using the index in the authentication information, and then verifies the signature by using the corresponding public key. The ISDPH is verified only if the signature is valid, and the public key is consistent to the ISDPH's identity.

After verification, the ASDPH finds the ISDPH's attributes in the DLT ledger according to the ISDPH's identity. The ASDPH sends the decision for the access connection to the ISDPH as the feedback for the access control [b-ITU-T X.1252] request, while the decision is made according to the ISDPH's attributes and ASDPH's policy.

Appendix I

Using DLT for security services: Challenges and benefits

(This appendix does not form an integral part of this Recommendation.)

I.1 Using DLT for public-key certificate management

I.1.1 Challenges

The main challenges of traditional public-key certificate management are as follows.

- Single point of failure: Centralized certification authority is the root of trust. Once a certification authority is compromised, the public-key certificates issued by this certification authority will be insecure and cannot be used any longer.
- CRL/OCSP service unavailable due to intranet implementation: This occurs if devices are deployed in the operator's core network with no connection to the Internet, which means both CRL and online certificate status protocol (OCSP) services are unavailable. The system will not be as secure as expected.
- No previsioned trust anchor: Devices in mobile networks prefer to use the public-key certificate issued by the operator's certification authority due to the cost. Devices when manufactured may not be provisioned with the trust anchor of the operator's certification authority, since it is not known where they will be deployed, nor in which operator's network they will be used. The reason behind this issue is the lack of trust among manufacturers, vendors and operators.

I.1.2 Benefits

The DLT-based public-key certificate management can benefit from the feature of decentralization, tamper-resistance and non-repudiation.

There is no centralized node in the DLT. The ledger is recorded in a decentralized manner. Even if some of the nodes are compromised, the ledger will not be tampered with. Thus, DLT could improve the robustness of PKI, and avoid a single point of failure.

An edge entity, which is deployed on the edge of the intranet and Internet, could be launched to provide a public-key certificate's status inquiry service for the intranet. The edge entity could provide a public-key certificate inquiry service for devices in the operator's core network.

NOTE – This Recommendation focuses on the traditional public-key certificates which contain CRL or OCSP information. A Recommendation under development, which is complimentary to [ITU-T X.509], will not use CRL or OCSP for public-key certificate revocation checking.

I.2 Using DLT for software defined perimeter

I.2.1 Challenges

SDP is a set of a network security isolation framework capable of on-demand dynamic configuration. Connectivity in an SDP is based on a need-to-know model, in which device posture and identity are verified before access to application infrastructure (e.g., enterprise application, private cloud and hybrid cloud) is granted. This can isolate the network and services to be protected from an insecure environment in order to mitigate the most common network-based attacks.

In its simplest form, the architecture of the SDP consists of two components: SDP hosts and SDP controllers. SDP hosts can either initiate connections (initiating SDP hosts, ISDPH) or accept connections (accepting SDP hosts, ASDPH). These actions are managed by interactions with the SDP controllers via a control channel, as shown in Figure I.1.

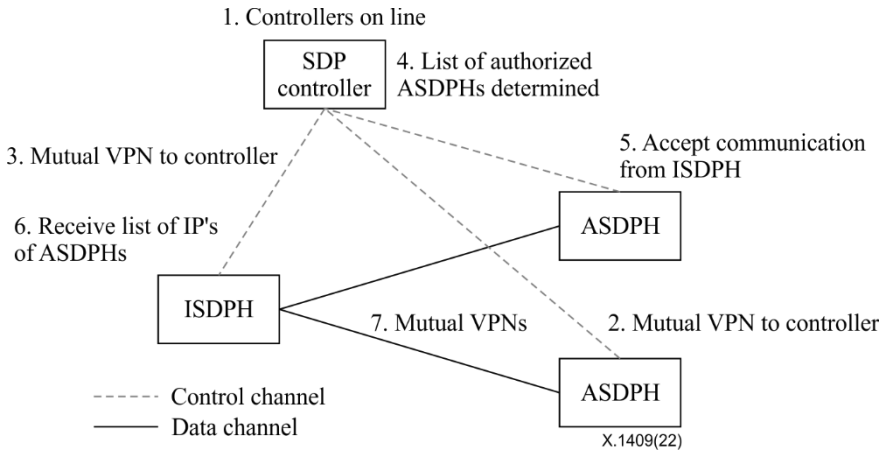

Figure I.1 – Architecture and workflow of SDP [b-SDP Spec 1.0]

The SDP controller will authenticate the initiating SDP host (ISDPH) and determine which SDP hosts can communicate with each other. The controller is centralized and exposed in the external network. It may be vulnerable to distributed denial of service (DDoS) attacks and become the bottleneck of the system. A single point of failure of the SDP controller may paralyse the whole SDP service and render it unavailable.

I.2.2 Benefits

The function of SDP controllers could be implemented by using DLT. Then the decentralized nodes in DLT may prevent DDoS attacks. The access records could be recorded into the ledger. This may be helpful to audit the system.

I.3 Using DLT for threat intelligence sharing

I.3.1 Challenges

There are many scattered threat intelligence providers, most of which are independent. Integration of threat intelligence is needed to perform deep analysis in order to find valuable or elusive attack events. Much work is needed to integrate the threat intelligence provided by different providers.

The threat intelligence providers may have different capabilities, and the quality of threat intelligence from different providers varies. Some of the threat intelligence may be fake, so all intelligence needs further verification.

There is no evaluation of threat intelligence. Providers may have less motivation to publish threat intelligence for free. On the other hand, the motivation to pay for threat intelligence may be lower unless it is evaluated to be genuine and useful.

I.3.2 Benefits

A DLT-based threat intelligence sharing platform can help to establish trust among participants in the platform. All the threat intelligence providers connected to the DLT-based platform could publish the threat intelligence to the platform. The threat intelligence recorded in the ledger is tamper-resistant. Participants who have joined the DLT-based threat intelligence sharing platform could access the record in the ledger, which may help to integrate the scatter of threat intelligence from different providers.

The participants could query and use the threat intelligence and submit feedback to the platform. The feedback could be recorded into the ledger, which could be used to evaluate the quality of the threat intelligence and identify fake information.

The querying of the threat intelligence could be recorded into the ledger. This could help to establish a fine-grained charging method and increase the motivation to publish more threat intelligence.

I.4 Using DLT for security audit

I.4.1 Challenges

Traditional log-based audit mechanisms are mostly centralized. They place a high reliance on the security of the individual log server. If the log server is compromised, the logs may be tampered with or destroyed, and the security of the logs cannot be guaranteed. The centralized system may lead to a severe vulnerability to single point of failure attacks.

I.4.2 Benefits

The security of the log audit may benefit from the DLT's characteristic of decentralization and tamper-resistance. The feature of decentralization makes the audit available in the nodes of multiple participants. This makes decentralized local audit possible. In some use cases, it can be used to achieve local verification based on the logs recorded in the ledger. The feature of tamper-resistance ensures the integrity of the logs. Tampering with the logs could be detected.

Appendix II

H(e)NB device authentication and verification by public-key certificate identifier – Two use cases of DLT-based public-key certificate management service

(This appendix does not form an integral part of this Recommendation.)

II.1 H(e)NB device authentication

Home NodeB or Home eNodeB (H(e)NB) [b-3GPP TS 33.320] has to be provisioned with a public-key certificate which will be used to authenticate the H(e)NB to the security gateway (SeGW).

Figure II.1 shows the flow of H(e)NB device authentication by using a DLT-based public-key certificate management system.

1) The H(e)NB sends its public-key certificate or public-key certificate identifier to the SeGW in the authentication request.
2) The SeGW verifies the H(e)NB's public-key certificate based on the received authentication request. The SeGW sends an inquiry message to the DLT-based public-key certificate management system so as to inquire about the public-key certificate or the certificate's status. The inquiry may contain the public-key certificate or the public-key certificate identifier derived from the received authentication request.
3) The DLT-based public-key certificate management system finds the inquired public-key certificate and its status, by using the public-key certificate or the public-key certificate identifier. The response to SeGW could contain the public-key certificate and/or its status, according to the inquiry. If the authentication request contains the public-key certificate, the response could only contain the status. If the authentication request contains the public-key certificate identifier and no certificate, the response is recommended to contain the public-key certificate and its status.
4) SeGW verifies the H(e)NB's public-key certificate only if the certificate's status is normal. Otherwise, the verification will fail. Then, the SeGW sends the authentication response to the H(e)NB.

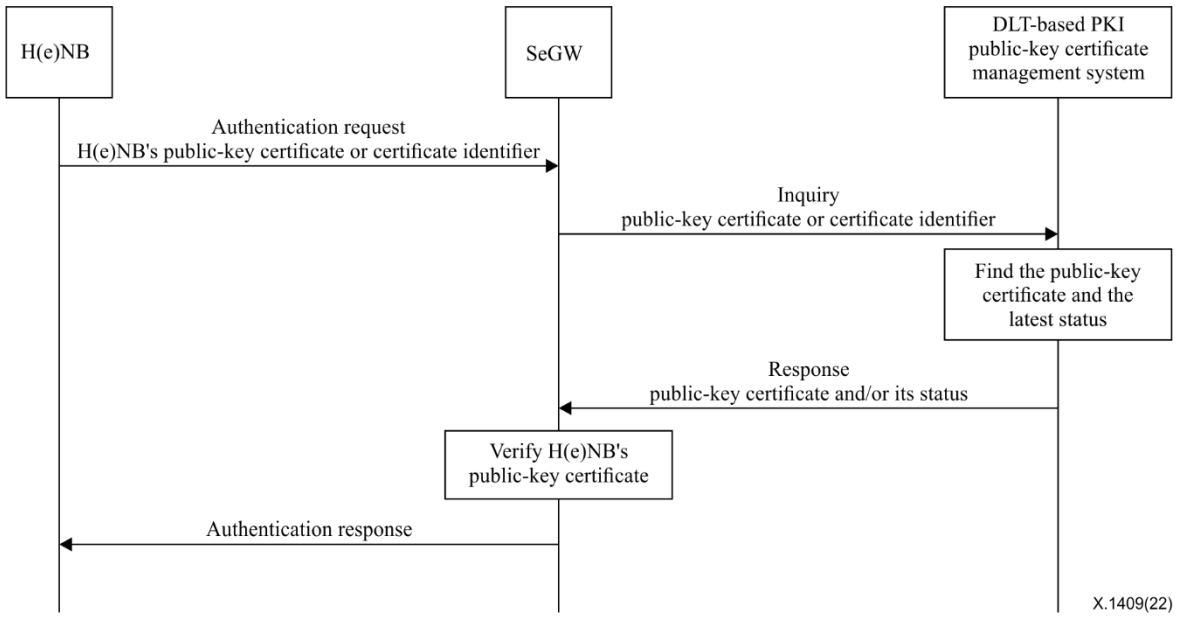

Figure II.1 – Public-key certificate verification in H(e)NB device authentication

II.2 Verification by the public-key certificate identifier

In many protocols supporting PKI technology, such as TLS [b-IETF RFC 8446], the entity public-key certificate and the intermediate public-key certificates need to be transmitted to the relying party. By using DLT-based public-key certificate management system, the public-key certificate identifier instead of the entity public-key certificate can be transmitted between the user and the relying party.

Figure II.2 shows the flow to verify a public-key certificate at the relying party by using the public-key certificate identifier.

1) The public-key certificate user sends the public-key certificate identifier to the relying party in the request (for example, the ClientHello message in TLS protocol).
2) The relying party has to verify the public-key certificate by checking the public-key certificate in the DLT ledger. The relying party sends a public-key certificate verification request to the DLT system (such as a validator node). The validator node locates and verifies the public-key certificate.
   a) Case I: If the validator node keeps all the public-key certificates in the DLT ledger, the verification can be done locally.
   b) Case II: If the validator node does not keep all the public-key certificates in the DLT ledger, it needs to transmit the request to a full node, which keeps all the public-key certificates. The full node checks, verifies the public-key certificate and feedbacks the result to the validator node.
3) The verification node feedbacks the result to the relying party.
4) The relying party checks the public-key certificate by using the methods defined in [ITU-T X.509].

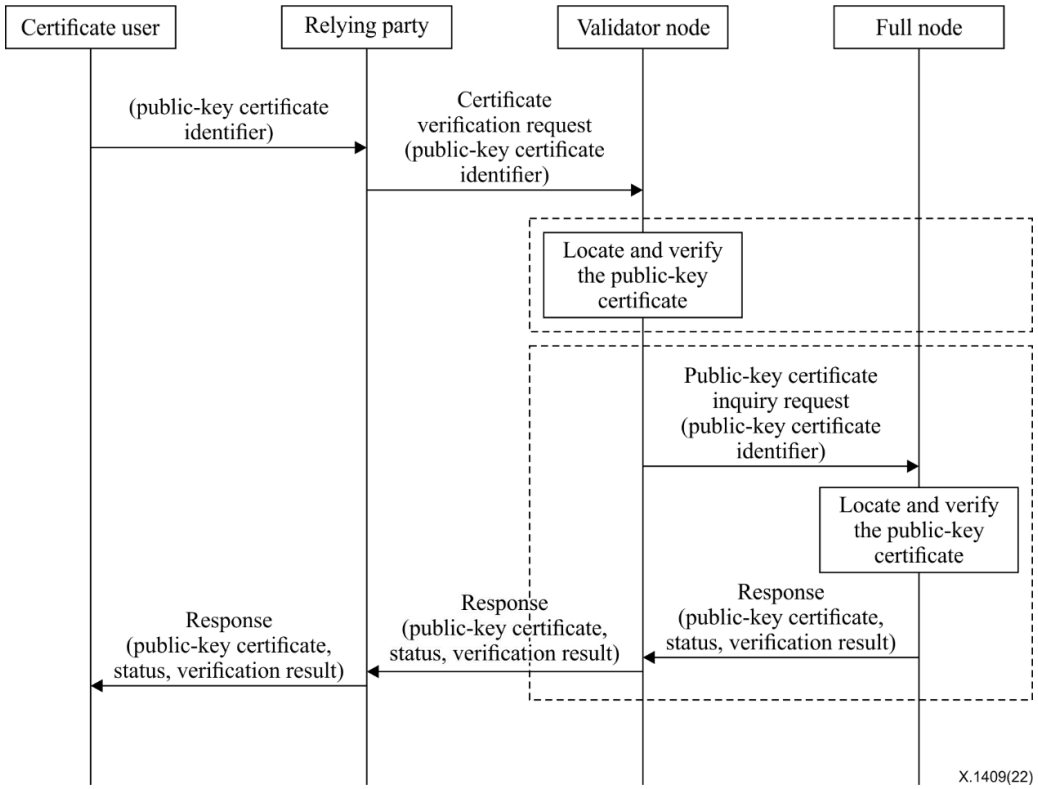

Figure II.2 – Verification of a public-key certificate at the relying party by using the public-key certificate identifier

Appendix III

Developer access to private cloud network as a use case of DLT-based software defined perimeter service

(This appendix does not form an integral part of this Recommendation.)

Figure III.1 shows a scenario of developers accessing a private cloud network. Developers Alice (IH_A) and Bob (IH_A) have to access to servers in the cloud. Alice is in a coffee shop, while Bob is in the corporate headquarter. According to the access policy, both Alice and Bob could access AH_A and AH_B. However, AH_B could only be accessed in the corporate HQ. All the nodes including ISDPHs and ASDPHs could access the DLT-based SDP.

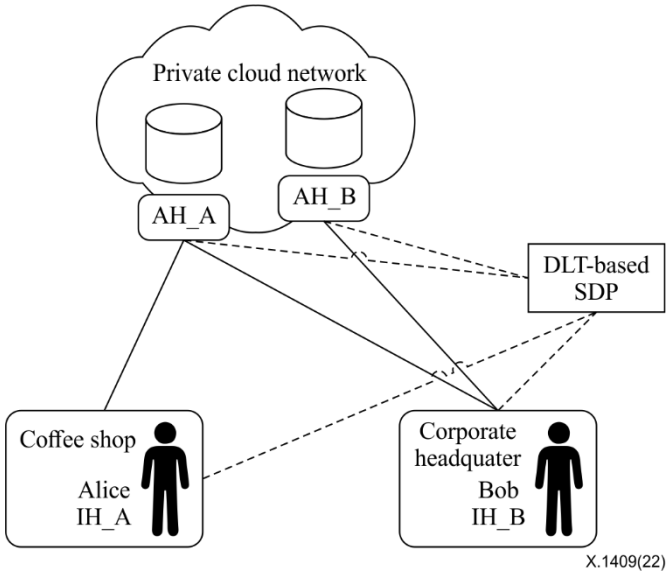

Figure III.1 – Developer access to a private cloud network

1) The ASDPHs authenticate to and submit the supported authorization policies to the nodes in the DLT system.

AH_A and AH_B send their registration requests containing ASDPH's host information (IP address, port) and supported policies to the DLT system. The policy of AH_A includes the login identity of the ISDPH. The policy of AH_B includes the login identity, IP address and geolocation of the ISDPH. AH_B could designate the node to verify the request of ISDPHs. For example, the geolocation of the ISDPH needs to be verified by some capable nodes.

The ASDPH's host information and supported policies will be recorded into the DLT ledger after verification and consensus.

2) Each ISDPH authenticates to the nodes in the DLT system.

IH_A and IH_B send authentication requests containing authentication credentials and attributes (identity, IP address, geo-position, etc) to the nodes in the DLT system.

The nodes in the DLT system verify the authentication request submitted by the ISDPH. After verification, the nodes search the DLT ledger and determine a list of ASDPHs which the ISDPH is authorized to access. The authorized list of IH_A contains AH_A. The authorized list of IH_B contains AH_A and AH_B. Then the nodes send the authorized list of ASDPHs to ISDPH. The lists will be recorded into the ledger.

3) The ISDPH initiates access connection to the ASDPH listed in the authorized list.

IH_A sends an initial access connection request to the AH_A. AH_A queries and determines the authorized list of ASDPHs for IH_A in the DLT ledger. AH_A is in the list, so it responds and provides access service to IH_A.

Appendix IV

Architecture and use case of DLT-based threat intelligence sharing platform

(This appendix does not form an integral part of this Recommendation.)

IV.1 DLT-based threat intelligence sharing platform architecture

Figure IV.1 shows the architecture of a DLT-based threat intelligence sharing platform. The resource layer contains the necessary resources, such as DLT nodes and networks, to run a DLT system. DLT nodes contain the storage and computation resources. The network provides interconnection for the DLT nodes. The protocol layer includes account management and consensus. The resource layer and protocol layer are similar to those in [b-ITU-T FG DLT D3.1].

The function layer contains the function to manage and query the threat intelligence.

- Threat intelligence submission: This module deals with the information submitted by the threat intelligence providers and records it into the ledger. The threat intelligence providers submit the threat intelligence into the platform. The threat intelligence and its source is recommended to be recorded into the ledger.
- Threat intelligence format: This module verifies the format and ensures the threat intelligence is recorded in a standard format.
- Threat intelligence feedback: The feedback module is optional to be used to evaluate the threat intelligence. Some tags could be defined to identify the quality of the threat intelligence and be used as the feedback.
- Permission control: Permission control is recommended when a participant in the platform queries the threat intelligence. The rules and policies could be defined by the threat intelligence providers.
- Record query: When the participant in the platform queries the threat intelligence, the query record is recommended to be recorded into the ledger. The record could be used to charge for the query, and then increase the incentive of the providers to submit more threat intelligence.

The API layer defines the interface to submit, feedback and query the threat intelligence.

- Threat intelligence submission interface: This interface is used by threat intelligence providers to submit threat intelligence.
- Threat intelligence feedback interface: This interface is used by the participants to evaluate threat intelligence. Only the participants who have previously required the threat intelligence can feedback and evaluate it.
- Threat intelligence query interface: This interface is used by participants to query the threat intelligence.

The management function aims to manage the whole DLT system so as to ensure the sustainable and secure running of the DLT system. The functions consist of permission control, operation management, node management, etc.

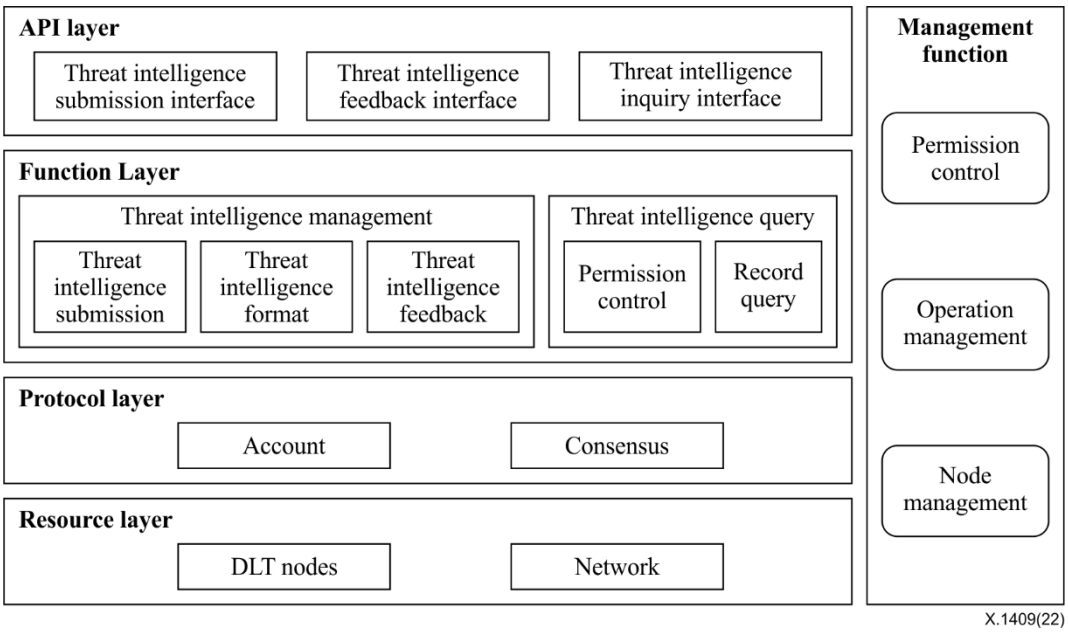

Figure IV.1 – A DLT-based threat intelligence sharing platform architecture

IV.2 Threat intelligence sharing and rating

Threat intelligence providers and threat intelligence users can share and rate threat intelligence by using the DLT-based threat intelligence sharing platform. The nodes of DLT can be the professional threat intelligence manufacturer, antivirus manufacturer, anti-APT manufacturer, detection product manufacturer, free intelligence alliance, intelligence agency, industry alliance, customer, government, operator, finial institution, etc.

Figure IV.2 shows the flow of threat intelligence sharing and rating by using a DLT-based platform. It is composed of some entities and their interactions from Interaction 1 to Interaction 6.

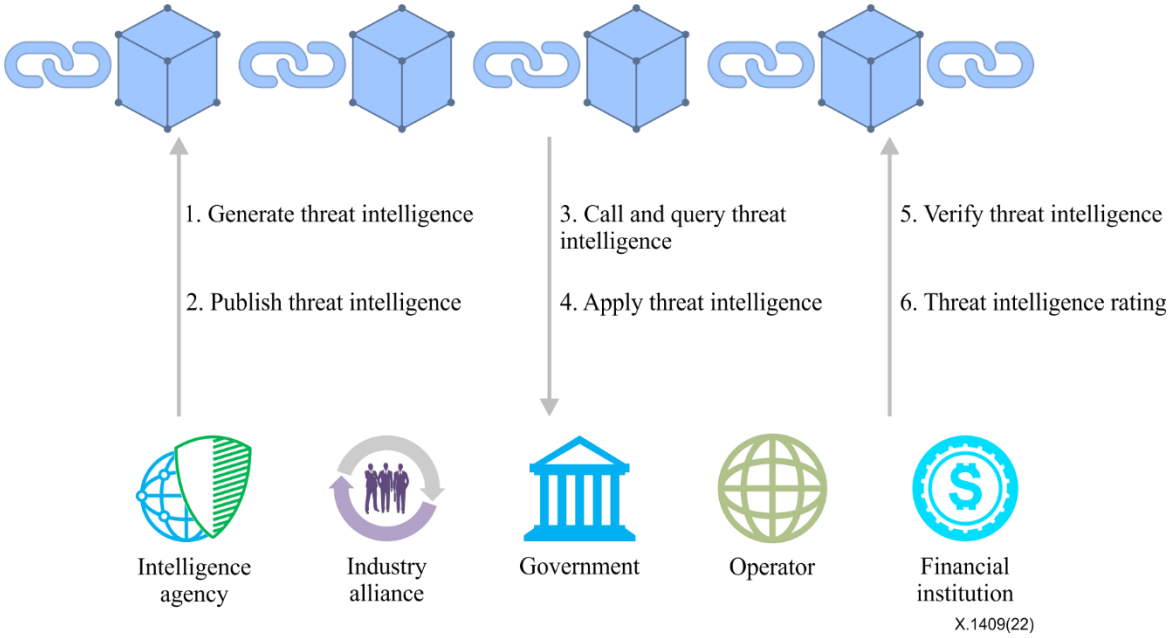

Figure IV.2 – DLT-based threat intelligence sharing and rating

The entity nodes consist of the threat intelligence providers and threat intelligence users. The providers could submit threat intelligence to the DLT-based platform, and then the threat intelligence and its source would be recorded into the ledger. The participants could query and use the threat intelligence. The authorized participants could feedback and evaluate the threat intelligence.

The data recorded in the ledger includes node role, threat intelligence, threat intelligence usage record, information source credibility and contribution rate.

The data is verified by threat intelligence users, such as governments, operators and financial institutions. They cross-validate the value of intelligence information.

As for updating data, all the verification nodes participate together. For valuable threat intelligence, they consistently update its usage records and the credibility and contribution rate of threat intelligence sources. Meanwhile, corresponding point rewards are subsequently established.

The smart contract automatically calculates the credibility and contribution rate of threat intelligence sources so as to achieve threat intelligence source ratings.

Appendix V

Architecture and use case of DLT-based security audit platform

(This appendix does not form an integral part of this Recommendation.)

V.1 DLT-based security audit platform architecture

Figure V.1 illustrates a DLT-based security audit platform architecture. The resource layer, the protocol layer and the management function are the same as those of the DLT-based threat intelligence sharing platform described in Appendix IV. The function layer contains the functions to record the logs into the DLT ledger and the functions to implement audit.

- Log submission: This module deals with the submitted logs. It is used when the specific logs need to be recorded into the ledger and to be audit decentralized.
- Log hash value submission: This module is used if the hash values of the logs need to be recorded into the ledger. It is recommended to verify the hash value.
- Permission control: Permission control is recommended when a participant in the platform queries the logs.
- Log record query: The logs and the hash values of the logs recorded in the ledger are queried to implement audit.

The API layer defines the interfaces to submit the logs and hash values, and the interface to query the logs for audit.

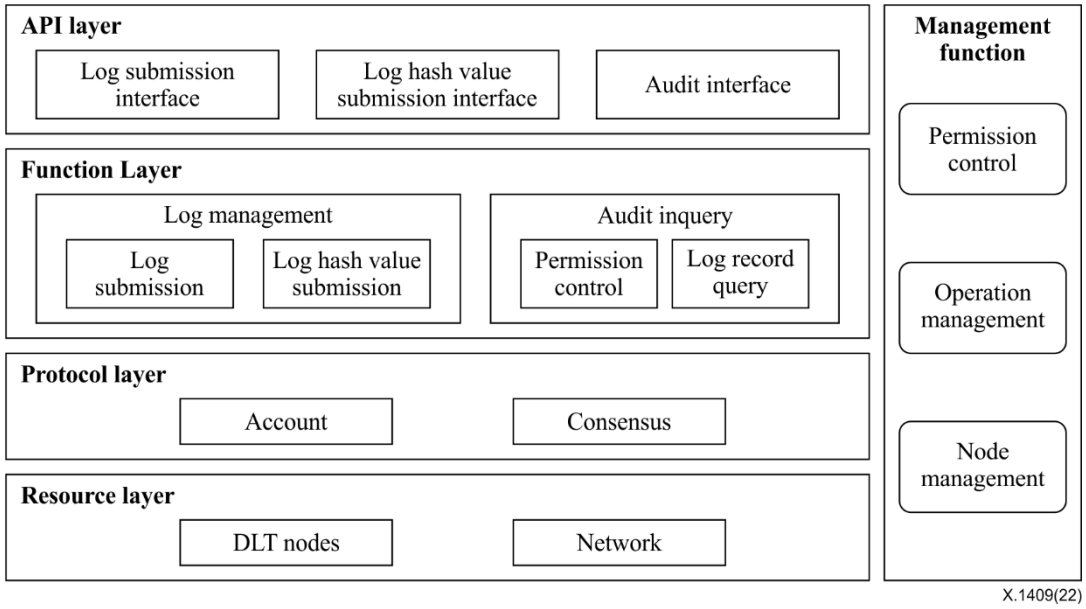

Figure V.1 – A DLT-based security audit platform architecture

V.2 Public-key certificate audit

The core of public key infrastructure is the ecosystem of a certification authority which is responsible for issuing and maintaining certification authority public-key certificates. The security issues in the certification authority audit (i.e., whether it could effectively and efficiently resist public-key certificate forgery and tamper attacks) is pivotal in this procedure. Log-based public-key certificate audit allows website users and domain owners to identity mistakenly or maliciously issued public-key certificates and identify certification authorities that have gone rogue through a system of public-key certificate logs, monitors and auditors. In existing centralized solutions, the data security still depends on an individual log server that is chosen to synchronize public-key

certificates. Figure V.2 illustrates an overview of a DLT-based public-key certificate audit. In this case, there are three types of operations on public-key certificates, i.e., certificate registration, certificate update and certificate revocation.

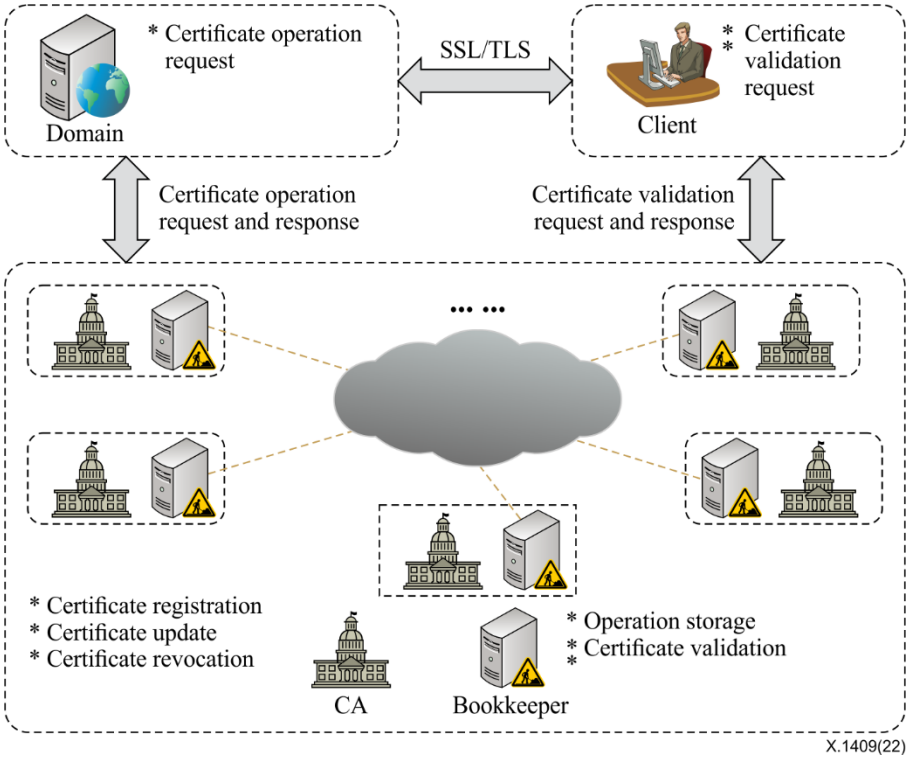

Figure V.2 – Overview of a DLT-based public-key certificate audit

In Figure V.2, a client is the entity who intends to establish TLS [b-IETF RFC 8446] connections with a domain. A domain usually refers to a website, which obtains a public-key certificate from a certification authority for secure connections. Certification authorities generate and sign public-key certificates. Bookkeepers store the operations in blocks and maintain the ledger. The DLT works in a permission mode which means that only authorized nodes can participate in public-key certificate management.

The public-key certificate audit procedure is as follows:

1) A domain requests a public-key certificate operation from a certification authority, such as public-key certificate registration, update or revocation.

2) After the public-key certification authority finishes the requested certification operation, it signs the operation and broadcasts it to all bookkeepers.

3) A client validates a public-key certificate with the assistance of bookkeepers.

Bibliography

[b-ITU-T X.1217] Recommendation ITU-T X.1217 (2021), *Guidelines for applying threat intelligence in telecommunication network operation*.

[b-ITU-T X.1252] Recommendation ITU-T X.1252 (2021), *Baseline identity management terms and definitions*.

[b-ITU-T X.1400] Recommendation ITU-T X.1400 (2020), *Terms and definitions for distributed ledger technology*.

[b-ITU-T FG DLT D3.1] ITU-T Focus Group on Application of Distributed Ledger Technology (FG DLT), Technical Specification FG DLT D3.1. *Distributed ledger technology reference architecture*.

[b-3GPP TS 33.320] 3GPP TS 33.320 V17.0.0 (2022), *Security of Home Node B (HNB) / Home evolved Node B (HeNB)*.

[b-IETF RFC 6749] IETF RFC 6749 (2012), *The OAuth 2.0 authorization framework*.

[b-IETF RFC 7522] IETF RFC 7522 (2015), *Security assertion markup language (SAML) 2.0 profile for OAuth 2.0 client authentication and authorization grants*.

[b-IETF RFC 8446] IETF RFC 8446 (2018), *The Transport Layer Security (TLS) Protocol Version 1.3*.

[b-ISO 22739] ISO 22739 (2020), *Blockchain and distributed ledger technologies — Vocabulary*.

[b-OpenID] OpenID Foundation (2014), *OpenID Connect Core 1.0.* Available [viewed 2022-04-22] from: https://openid.net/specs/openid-connect-core-1_0.html.

[b-SDP Spec 1.0] Software Defined Perimeter Working Group (2014), *SDP Specification 1.0*.

SERIES OF ITU-T RECOMMENDATIONS

Series A	Organization of the work of ITU-T
Series D	Tariff and accounting principles and international telecommunication/ICT economic and policy issues
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	Transmission systems and media, digital systems and networks
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	Cable networks and transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Environment and ICTs, climate change, e-waste, energy efficiency; construction, installation and protection of cables and other elements of outside plant
Series M	Telecommunication management, including TMN and network maintenance
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Telephone transmission quality, telephone installations, local line networks
Series Q	Switching and signalling, and associated measurements and tests
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	**Data networks, open system communications and security**
Series Y	Global information infrastructure, Internet protocol aspects, next-generation networks, Internet of Things and smart cities
Series Z	Languages and general software aspects for telecommunication systems

Printed in Switzerland
Geneva, 2022