International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

X.1401

(11/2019)

SERIES X: DATA NETWORKS, OPEN SYSTEM COMMUNICATIONS AND SECURITY

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

Security threats to distributed ledger technology

Recommendation ITU-T X.1401

International Telecommunication Union

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	X.1140–X.1149
Security protocols (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.1360–X.1369
Intelligent transportation system (ITS) security	X.1370–X.1389
Distributed ledger technology security	**X.1400–X.1429**
Distributed ledger technology security	X.1430–X.1449
Security protocols (2)	X.1450–X.1459
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
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	X.1700–X.1729

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

Recommendation ITU-T X.1401

Security threats to distributed ledger technology

Summary

Distributed ledger technology (DLT) provides a technical mechanism for achieving verifiable trust through consensus and collective decision-making. DLT also involves maintaining ledgers in a decentralized way and using crypto-mechanisms that can deliver some intrinsic security features. However, a distributed ledger system still has security limitations, for example, confidentiality must be added depending on use case, data sensitivity and applicable data privacy regulations.

Recommendation ITU-T X.1401 provides a structured and systematic threat analysis method to design, implement and operate a distributed ledger technology system and to evaluate its security.

History

Edition	Recommendation	Approval	Study Group	Unique ID*
1.0	ITU-T X.1401	2019-11-29	17	11.1002/1000/14092

Keywords

Distributed ledger technology, DLT.

* 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, 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 TSB patent database at http://www.itu.int/ITU-T/ipr/.

Table of Contents

Introduction

Distributed ledger technology (DLT) achieves trust through collective decision-making and maintaining a single global ledger in a decentralized way.

This Recommendation identifies possible threats to various functional components of a distributed ledger system, such as protocol, network and data. This Recommendation can be considered in the design or implementation of a DLT system as a reference baseline.

Recommendation ITU-T X.1401

Security threats to distributed ledger technology

1 Scope

This Recommendation provides guidance and categorization on security threats to distributed ledger technology (DLT) components. Each of the threats is described in four dimensions:

– targeted component;

– attacks;

– attack impact; and

– attack likelihood.

It also assigns an index to security threats with attack methods or vulnerabilities so that each threat can be referenced by other parts of this Recommendation and other Recommendations.

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.

None.

3 Definitions

3.1 Terms defined elsewhere

This Recommendation uses the following terms defined elsewhere:

3.1.1 account [b-ITU-T FG DLT D1.1]: Representation of an entity whose data is recorded on a distributed ledger.

3.1.2 address [b-ITU-T FG DLT D1.1]: Identifier for entity(ies) performing transactions or other actions in a blockchain or distributed ledger network.

3.1.3 block [b-ITU-T FG DLT D1.1]: Individual data unit of a blockchain, composed of a collection of transactions and a block header.

3.1.4 blockchain [b-ITU-T FG DLT D1.1]: 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.

3.1.5 consensus [b-ITU-T FG DLT D1.1]: Agreement that a set of transactions is valid.

3.1.6 distributed ledger [b-ITU-T FG DLT D1.1]: A type of ledger that is shared, replicated, and synchronized in a distributed and decentralized manner.

3.1.7 fork [b-ITU-T FG DLT D1.1]: Creation of two or more different versions of a distributed ledger.

3.1.8 guideline [b-ISO/IEC 27000]: Description that clarifies what should be done and how, to achieve the objectives set out in policies.

3.1.9 hashing [b-NISTIR 8202]: A method of calculating a relatively unique output (called a *hash digest*) for an input of nearly any size (a file, text, image, etc.). The smallest change of input, even a single bit, will result in a completely different output digest.

3.1.10 ledger [b-ITU-T FG DLT D1.1]: Information store that keeps final and definitive (immutable) records of transactions.

3.1.11 node [b-ITU-T FG DLT D1.1]: Device or process that participates in a distributed ledger network.

3.1.12 peer-to-peer [b-ISO 22739]: Relating to, using, or being a network of peers that directly share information and resources with each other without relying on a central entity.

3.1.13 permissioned distributed ledger system [b-ITU-T FG DLT D1.1]: Distributed ledger system in which permissions are required to maintain and operate a node.

3.1.14 permissionless distributed ledger system [b-ITU-T FG DLT D1.1]: Distributed ledger system where permissions are not required to maintain and operate a node.

3.1.15 public distributed ledger system [b-ISO 22739]: Distributed ledger system which is accessible to the public for use.

3.1.16 private distributed ledger system [b-ISO 22739]: Distributed ledger system which is accessible for use only to a limited group of DLT users.

3.1.17 proof of work [b-ITU-T FG DLT D1.1]: Consensus process to solve a difficult (costly, time-consuming) problem that produces a result that is easy for others to correctly verify.

3.1.18 risk [b-ITU-T X.1521]: The relative impact that an exploited vulnerability would have to a user's environment.

3.1.19 smart contract [b-ITU-T FG DLT D1.1]: Program written on the distributed ledger system which encodes the rules for specific types of distributed ledger system transactions in a way that can be validated, and triggered by specific conditions.

3.1.20 threat [b-ISO/IEC 27000]: potential cause of an unwanted incident, which may result in harm to a system or organization.

3.1.21 token [b-ITU-T FG DLT D1.1]: A digital representation of value on a shared distributed ledger that is owned and secured using cryptography to ensure its authenticity and prevent modification or tampering without the owner's consent.

3.1.22 transaction [b-ITU-T FG DLT D1.1]: 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 mining: A reward-seeking activity in some consensus mechanisms, which operates through a demonstration of proof of work.

3.2.2 consensus mechanism (CM): Rules and procedures by which 'consensus', meaning an agreement that a set of transactions is valid, is reached.

4 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

AA Analysis Attack

ACA Asymmetric Cryptographic Algorithm

ACAT Asymmetric Cryptographic Algorithm Threats

AN Authentic Node

ATD Account Data and Transaction Data

ATDT Account Data & Transaction Data Threats

BA	Backdoor Attack
BFCA	Brute Force Cracking Attack
BGP	Border Gateway Protocol
CA	Collision Attack
CAS	Component Attack Surface
CC	Component Classes
CHA	Cryptographic Hash Algorithm
CHAT	Cryptographic Hash Algorithm Threats
CM	Consensus Mechanism
CM-51A	51% Attack
CM-BA	Bribing Attack
CM-BWA	Block Withholding Attack
CM-CHA	Chain Hopping Attack
CM-DSA	Double-Spending Attack
CM-SMA	Selfish Mining Attack
CN	Compromised Node
CPA	Cryptographic Protocol Attack
CVSS	Common Vulnerability Scoring System
DA	DDoS Attack
DCA	Data Component Attacks
DCC	Data Component Class
DDoS	Distributed Denial of Service
DDOST	Network DDOS Threat
DLT	Distributed Ledger Technology
DLTN	DLT Network
DLTS	DLT System
DLTS-CC	DLTS Component Classes
DLTS-P	DLTS protections from the threat
DLTS-R	DLTS risks resulting from threats
DSA	Digital Signature Attack
DSEM	DLTS Security Expression Model
EA	Eclipse Attack
ECDSA	Elliptic Curve Digital Signature Algorithm
FNIA	Fraudulent Node Identity Attack
FUD	Forget Unlocking Data
ISP	Internet Service Provider
IP	Internet Protocol

MA	Malware Attack
MCCA	Mathematical Cryptanalysis Cracking Attack
MEA	Mishandled Exceptions Attack
MHR	Merkle Hash Root
MN	Malicious Node
NB	Node Buffer
NCA	Network Components Threats
NCC	Network Component Class
NI	Node Identity
NIM	Node Identity Management
NIT	Node Identity Threats
NNI	Network Node Identity
NR-DA	Network Routing Delayed Attack
NR-PA	Network Routing Partition Attack
NRT	Network Routing Threats
NRTT	Node Routing Table Threat
OIA	Integer Overflow Attack
P2P	Peer-to-Peer
PA	Physical Attack
PCC	Protocol Component Class
PCL	Paper Private Key Code Loss
PIN	Personal Identification Number
PKLeT	Private Key Leakage Threats
PKLoT	Private Key Loss Threats
PMMA	Protocol Message Manipulation Attack
PNC	P2P Network Connection
PoW	Proof of Work
PQC	Practical Quantum Computers
PQCT	Practical Quantum Computers Threats
PrA	Preimage Attack
PrK	Private Key
PRNA	Predictable Random Number Attack
PSD	Public Sensitive Data
PSDA	Public Sensitive Data Attack
SA	Sybil Attack
SBA	Spam Block Attack
SC	Smart Contract

SCA Software Client Attack
SCT Smart Contract Threats
SPA Second Preimage Attack
SPV Simple Payment Verification
TD Transaction Data
TDA Timestamp Dependence Attack
TMA Timestamp Manipulation Attack
TN Targeted Node
TT Transactions Threat
UAA Unauthorized Access Attack
UL Unlocking Loss
UT Unconfirmed Transaction
VM Virtual Machine
VM-EA Virtual Machine Escape Attack
VM-FHA Fault Handling Attack
VM-MCA Memory Corruption Attack
VMT Virtual Machine Threats
WKMA Weak Key Material Attack
ZKP Zero-Knowledge Proof

5 Conventions

None.

6 Threats to distributed ledger systems

Threats to distributed ledger systems are firstly categorized by the protocol, network or data layer which they affect, then further decomposed according to components of each layer. Each threat is specified by identifying the targeted component, acronyms of attacks specific on the component, impact and likelihood of the threat. Other Recommendations can refer to an attack with the acronym as an index.

6.1 Threats to protocols

Protocols are the configurations that drive network behaviour. Consequently, any unauthorized modification of protocols can have a significant impact on distributed ledger system behaviour.

The threats to protocols can be further decomposed into following six groups:

– Consensus mechanism;
– Smart contract;
– Virtual machine;
– Cryptographic hash algorithm;
– Asymmetric cryptographic algorithm; and
– Practical quantum computers.

6.1.1 Consensus mechanism threats

Targeted component: Consensus mechanism (CM)

The target under consideration is the consensus mechanism which is the rules and procedures by which consensus is reached.

Consensus mechanism attacks:

There are many different consensus mechanisms each focused on reaching different forms of consensus such as proof of work, stake, elapsed time, activity, importance, weight, capacity, etc. Weaknesses of consensus mechanism can be exploited by following attacks:

– **51% attack (CM-51A)**: In the distributed ledger system using competition to achieve consensus, if an attacker controls enough DLT nodes and therefore proof of competition, it may have the ability to revoke or rewrite the distributed ledger system Ledger by creating a new fork in the chain and replacing the authentic fork.

– **Timestamp manipulation attack (CM-TMA)**: The attacker chooses a malicious timestamp in a block producing instead of the system time to spoof the consensus mechanism to cause the block with the correct timestamp to be refused by other nodes, thereby gaining the advantage.

– **Bribing attack (CM-BA)**: A briber owning enough resource entices other participants to take special actions in order to attack the blockchain by providing extra profit such as purchasing computing power or tokens.

– **Selfish mining attack (CM-SMA)**: In the use case of proof of work (PoW) consensus mechanism, a malicious node attempts to waste other node's computing resources in following ways: When the malicious node finds a new effective block, the malicious node does not broadcast the block to other nodes immediately, but continues to mine proof of work blocks before a new block is produced by other nodes. When other authentic nodes mine a new block, the malicious node will broadcast the block immediately to the network causing the block mined by other authentic nodes to be considered invalid [b-ARXIV2013].

– **Chain hopping attack** (CM-CHA): In a chain hopping attack, the attackers (miners) switch between various blockchains to maximize their profits by taking advantage of the difficult adjustment algorithms of the chain. This will lead to the loss of honest miners' awards and make the block generating time become very unstable.

– **Block withholding attack** (CM-BWA): A malicious node withholds a newly found block from other miners' nodes. The malicious node attempts to mine in secret and let the rest of the other mining nodes work on a block that will end up rejected and orphaned. Another block withholding attack (BWA) is that mining nodes in a mining pool submit only partial PoW solutions (he/she drops the newly found valid block rather than report it to the mining pool). This will waste the resources of other authentic mining nodes.

– **Double-spending attack** (CM-DSA): A group of malicious nodes in a distributed ledger system may be able to change the Ledger transaction history by creating a new fraudulent fork and rolling back the transaction history such that an asset may be spent twice in two transactions, the first one of which being authentic.

Consensus mechanism attack impact:

Successful CM attacks can allow malicious nodes to control the distributed ledger system with the goal to control the propagation of the chain or weaken or destroy the decentralized consensus mechanism operating the chain. An un-authentic fork will allow the attacker to add fraudulent blocks in the fraudulent fork and make legitimate transaction data in the authentic fork invalid.

Consensus mechanism attack likelihood:

Most consensus mechanism attacks need a large amount of resources such as computing power, votes or tokens. The likelihood of launching attacks depends on the resource cost of a potential attack.

6.1.2 Smart contract threats

Targeted component: Smart contract (SC)

Any distributed ledger system which uses smart contract could be threatened. A smart contract (SC) is a program written on the distributed ledger system which encodes the rules for specific types of distributed ledger system transactions in a way that can be validated, and triggered by specific conditions. The smart contract is stored in the DLT, encoded as part of a "creation" transaction that introduces a contract to the DLT.

Smart contract attacks:

There are vulnerabilities in smart contract that can be exploited by malicious users. The vulnerabilities include but are not limited to:

– **Timestamp dependence attack (SC-TDA)**: A smart contract may use the timestamp as a triggering condition to execute some critical operations, e.g., transferring money between accounts. Attackers can select a specific time stamp which triggers a condition in their favour. This may cause severe economic losses.

– **Mishandled exceptions attack (SC-MEA)**: This is a possible vulnerability when a smart contract calls another as a premise. If there is an exception raised (e.g., exceeding call stack limit) in the callee contract but the exception is not handled properly so that the callee does not perform the premise condition and an exception in the callee contract may also not get propagated to the caller. Or the caller contract does not check the execution status (e.g., explicitly check the return value) to verify if the call has been executed properly, the caller may continue to execute the next steps such as sending money.

– **Integer overflow attack (SC-IOA)**: In computer programming, an integer overflow occurs when an arithmetic operation attempts to create a numeric value that is outside of the range that can be represented with a given number of bits – either larger than the maximum or lower than the minimum representable value. In smart contracts, integer overflow can result in a lot of tokens being created and owned by the attackers.

– **Predictable random number attack (SC-PRNA)**: Smart contracts such as gambling games always use random numbers to determine the outcome of the game. Developers often use block header information to generate "random numbers". However, this "random number" can be expected by a malicious miner, which allows a malicious miner to control the outcome of the game.

Smart contract attack impact:

A successful attack on a smart contract could have the following impact:

Vulnerabilities in smart contracts may be exploited by malicious users to gain profit without following the agreement of related parties. It is difficult to update the smart contract which makes the consequence more durable.

Smart contract attack likelihood:

Vulnerabilities in smart contracts prevail in distributed ledger systems. For example, several design bugs were found as reported by [b-ACM2016]. Lack of formal verification makes it difficult to find the vulnerabilities. The attack difficulty also depends on the condition of vulnerabilities exploitation.

6.1.3 Virtual machine threats

Targeted component: Virtual machine (VM)

Any distributed ledger system which uses smart contract VM could be threatened. The virtual machine (VM) is the platform running different kinds of smart contracts and provides an execution environment such as a sandbox environment.

VM attacks:

The smart contract VMs running on nodes of blockchain receive and deploy the code of smart contracts. There are vulnerabilities in the VMs that can be exploited by malicious users. Vulnerabilities in smart contract VMs exist because of inadequate design and implemented mechanisms. The vulnerabilities include but are not limited to:

– **Escape attack (VM-EA)**: The smart contract VM provides a sandbox environment for running byte codes. Users should only execute corresponding code within the sandbox. Attackers can exploit the VM escape vulnerability and seek more execution permissions outside the sandbox which may result in attackers executing other malicious code or impacting contracts in other VMs.

– **Fault handling attack (VM-FHA)**: The VM can implement some fault-tolerant handling when aware of malicious data or code. Vulnerabilities in handling may cause some logical problems like a "short address attack".

– **Memory corruption attack (VM-MCA)**: Memory corruption occurs in a computer program when the contents of a memory location are modified due to programmatic behaviour that exceeds the intention of the original programmer or program/language constructs; this is termed violating memory security. Although there are many mitigation techniques to memory corruption attacks to make a memory corruption bug harder to be exploited, memory corruption bugs are still very harmful to the distributed ledger system. A memory crash bug in block validation may result in a crash of the entire distributed ledger system. Attackers exploiting memory out of bound write bugs in smart contract VMs may control thousands of nodes across the network in seconds.

VM attack impact:

Vulnerabilities in smart contract VM may be exploited by malicious users to gain profit without following the agreement of related parties, which may cause resource consuming, denial of service, information disclosure or remote code execution and so on.

VM attack likelihood:

Vulnerabilities exploitability of the smart contract VMs varies according to the vulnerabilities. Attackers need to have knowledge of the mechanism and then invoke some specific contract with some specific transaction to exploit the vulnerabilities.

6.1.4 Cryptographic hash algorithm threats

Targeted component: Cryptographic hash algorithm (CHA)

A cryptographic hash algorithm $H:\{0,1\}^* \rightarrow \{0,1\}^L$ is a mathematical algorithm that takes a string of arbitrary length as input and produce a bit string of a fixed length L where L is a fixed non-negative integer. An output of the cryptographic hash algorithm is called the "hash value" (simply "hash") or "message digest".

Cryptographic hash algorithms are used in:

- the generation of the account address from account owner's public key,
- the construction of Merkle tree,

- the "data chaining process" in tamper evident proof ledger creation,
- the generation of a digital signature, and
- the construction of anonymous authentication based on a zero-knowledge proof.

Cryptographic hash algorithm attacks:

Attacker could launch the following hash algorithm attacks:

1) **Hash collision attack (CHA-HCA)**: It is to find two different messages, m_1 and m_2 such that $H(m_1) = H(m_2)$.
2) **Second preimage attack (CHA-SPA)**: Given an input message, m_1, it is to find different input message m_2 such that $H(m_1) = H(m_2)$. The attacker creates a document other than the authentic one and has the same Merkle hash root (MHR).
3) **First preimage attack (CHA-FPA)**: Given a hash value $h=H(m_1)$ of an authentic message m_1, attackers can find a fraudulent message FM such that h = Hash(FM).

Cryptographic hash algorithm attack impact:

A successful cryptographic hash algorithm attack means that a malicious adversary can replace or modify the input data without changing its digest. Undesirable consequences from collision attacks are described as follows:

– Account address collision:

  In some distributed ledger systems, the account address is generated from the hash value of the account owner's public key. Peer nodes can verify the relationship among the received signatures, the public key and the account address to make sure whether the account address is owned by the signature provider. The property related to the account address can be dealt by the signature provider. If the hash algorithms suffer from second preimage attacks and the attackers find an account address which is the same as a user's but with a different public/private key pair, then the attackers can manipulate the user's property related to the address.

– Invalid Merkle tree:

  A hash algorithm can be used for constructing a Merkle tree [b-M87] from a set of data (e.g., transactions, account states). By checking the root of a Merkle tree, it is easy to find whether the data set is tampered with. For example, it is easy to check whether transactions in the block have been tampered with. If the hash algorithms are insecure, the tampering or forging may not be discovered. A Merkle tree is also used to find whether one given data item is in the set by calculating the Merkle root from the data item and the Merkle branch hash values provided by the DLT network. If the Merkle root calculated is the same as that given or recorded locally, that means the given data item is included in the set. For example, it is easy to make sure whether one transaction has been written into a block using simple payment verification (SPV). If the hash algorithms are insecure, attackers may forge a transaction which can achieve the same Merkle root and the user who uses the SPV may accept the forged transaction.

– Invalid proof of work (PoW) consensus:

  In distributed ledger systems using PoW to achieve consensus, peers decide whether to accept received blocks according to the verification of working proof provided by blocks. The verification is based on whether the hash value of the block head is less than a given value. Providing a block with such a hash value needs lots of calculations, so that the hash value can be seen as working proof. If the hash algorithms are insecure, attackers can obtain the value which has to be tried one by one.

– Forgery of a digital signature or zero-knowledge proof:

A digital signature is used as a core primitive in DLT system to guarantee authenticity and integrity of a transaction message. To treat a long transaction message, many digital signature schemes are designed with cryptographic hash algorithms that map strings of arbitrary size to a short fixed-length string. If the cryptographic hash algorithms of the digital signature schemes do not provide the second preimage resistance then attackers are able to present a forged transaction message with a malicious intension.

A non-interactive zero-knowledge proof can be used in privacy protection mechanisms to guarantee that transactions are generated by a valid user while permitting selective disclosure of information about the user's identity. For example, we can consider anonymous authentication systems [b-C85] and [b-D88] such as Idemix credential system [b-CH02] in Hyperledger Fabric and Indy, and group signature schemes [b-BBS04] and [b-HCCN15]. To generate a zero-knowledge proof non-interactively, i.e., without involving a verifier, a zero knowledge protocol makes use of a cryptographic hash algorithm that is supposed to output a uniformly random value. However, if the cryptographic hash algorithm is vulnerable to the second preimage attack then a forged proof about the user's identity can be generated so that an attacker can obtain an illegal access to the user's resources in DLT systems.

Cryptographic hash algorithm attack likelihood:

The dependency on cryptographic hash algorithm prevails in distributed ledger systems. Some algorithms such as SHA-1[b-CRYPTO2017] are proven to suffer from collision attacks. Launching a collision attack on a hash algorithm needs an enormous amount of computing power. There is the possibility of the collided hash value as an account address. Most hash algorithms are resistant to the preimage attacks or only suffer preimage attacks with theoretical possibilities [b-EUROCRYPT2009], [b-CRYPTO2008], [b-ASIACRYPT2009] and [b-IACR2009].

6.1.5 Asymmetric cryptographic algorithm threats

Targeted component: Asymmetric cryptographic algorithm (ACA)

An asymmetric cryptographic algorithm (ACA) is a type of cryptographic algorithm that works with a pair of two different keys, a private key and a public key where the private key is kept secret by a user but the corresponding public key is publicly distributed so that it can be accessed. It can be used as an essential component in a distributed ledger system in order to provide intended security properties including secrecy, authenticity, integrity, non-repudiation and privacy.

A distributed ledger system can be constructed with various asymmetric cryptographic algorithms. However, if it is not well organized with secure asymmetric cryptographic algorithms, the resulting distributed ledger system could be subject to various threats.

Asymmetric cryptographic algorithm attacks:

The asymmetric cryptographic algorithms or cryptographic protocols based on them used in a distributed ledger system may not be secure enough. Attacks include but are not limited to:

– **Weak key material attack (ACA-WKMA)**: Unreliable generation of keys and parameters
  - Generation of a private key over a predicted distribution
  - Use of insufficient length for a private or public key
  - Unreliable generation of public parameters
  - Brute force attacks for a private key with low-entropy
– **Backdoor attack (ACA-BA)**
  - Misuse of pseudo-random number generators with backdoors [b-DPSW16]
– **Mathematical cryptanalysis cracking attack (ACA-MCCA)**

- Forgery of signatures in digital signature schemes
- Exposure of private messages

– **Malicious manipulation of protocol messages (ACA-PMMA)**
  - Reply, replay, injection, alternation, etc.
  - Impersonation of a legitimate user

Asymmetric cryptographic algorithm attack impact:

As the two main ASA, public key encryption and digital signature algorithms are able to provide two fundamental protection properties, i.e., message confidentiality and authenticity, respectively. A public key encryption algorithm such as Rivest-Shamir-Adleman (RSA) and ElGamal schemes [b-IEEE1985] can be used to encrypt a message on a distributed ledger system using a public key of a receiver. Upon receiving an encrypted message, the receiver who holds the corresponding secret key, could decrypt it to read the original message. A digital signature algorithm such as the elliptic curve digital signature algorithm (ECDSA) [b-ECDSA2001] is used to generate a signature of a transaction message by a signer holding a private signing key on a distributed ledger system. The signature can be verified using the public key corresponding to the signing key for authenticity of the message.

In addition, an asymmetric cryptographic algorithm can be used to prove ownership of the input property in a transaction even without exposure of input information. For example, a zero-knowledge proof is a protocol that allows one party called a prover to prove to another party called a verifier that a statement, for example, "The prover possesses knowledge of certain information" is true without revealing the information. In distributed ledger systems, zero knowledge protocols can be used to guarantee that transactions are valid while information about a sender, a recipient and other transaction details remain hidden as in Zcash and Ethereum.

Usage of insecure asymmetric cryptographic algorithms may cause the exposure of the private key and private information such as (user-sensitive) verifiable claims or generation of false signatures which can result in forgery of message endorsement or stealing of property related to the private key.

Asymmetric cryptographic algorithm attack likelihood:

Being prevalent in distributed ledger systems, asymmetric cryptographic algorithm efficacy is highly dependent on the correct and sufficiently strong cryptographic parameters such as key length and algorithm class.

However, there is the possibility of misusing maliciously designed asymmetric cryptographic algorithms, for example, pseudo-random number generators with backdoors to generate a predicted private key or inadequately constructed zero knowledge protocol to reveal private information such as transaction details. The development of quantum computers will make brute force attacks on asymmetric cryptographic algorithms easier by using Shor's algorithm [b-Shor97].

6.1.6 Threats from practical quantum computers

Targeted component: Hashing and cryptographic algorithms

Practical quantum computers threats are an acute and urgent pending risk to security targets, that is hashing and cryptographic algorithms that encrypt text, chain text blocks and form digital signatures.

Practical quantum computers attacks:

Practical quantum computers threats represent heightened risk to these targets as secured targets are protected one day and not the next once practical quantum computers capability is available.

The following lists possible practical quantum computers attacks.

– **Cryptographic protocol attack** (PQC-CPA) by a Shor's algorithm [b-Shor97] and superposition attacks [b-DFNS11].
– **Brute force cracking attack (PQC-BFCA)** attempt 1) cracking through crypto-analysis the private key derived from the corresponding public key or 2) decryption of an encrypted text resulting in unauthorized access.
– **Digital signature attack (PQC-DSA)** attempt to create fraudulent signatures resulting in identity impersonation.

Practical quantum computers attack impact:

The practical quantum computers will be able to weaken (and in some cases, render useless) existing cryptographic algorithms. The security of those cryptographic algorithms relies on the computational complexity of integer factorization (such as RSA) or those on solving discrete logarithms (such as DSA and Diffie-Hellman).

The hashing algorithms and Merkle trees (MT) are much less susceptible to practical quantum computers attacks, but are still weakened when practical quantum computers are available. The public key cryptographic algorithms used in distributed ledger systems should be enhanced to versions that are quantum resistant. Table II.1 shows the impact of quantum computers on the commonly used cryptographic algorithms [b-NISTIR 8105].

Practical quantum computers attack likelihood:

The dependency on cryptographic algorithms that are not resistant to practical quantum computers prevails in distributed ledger systems. The condition for this attack to be successful is that the practical quantum computers are available. The threat likelihood of practical quantum computers attacks to these susceptible hash and algorithm targets increases to 100%, and the specific algorithms will not withstand the brute force computing power. It may take some time for this attack to be successful. The development of quantum computation will make brute force attack easier.

6.2 Threats to networks

A distribute ledger network consists of nodes. The threats to networks can be further decomposed into following four groups:

– Node routing table
– Network DDoS
– Node identity
– Network routing

6.2.1 Node routing table threat (NRTT)

Targeted component: Node routing table

The target under consideration is the node routing table which contains IP routing addresses of peer nodes.

Distributed ledger systems with the following features could be threatened:

– Nodes connected using P2P protocol.
– There is no authentication between peer nodes.
– The victim node has a permanent IP address.
– Connections between nodes are dynamic.
– Each node propagates and stores addresses of other potential peer nodes in the network.

– Each node can accept unsolicited incoming connections from any IP address.
– The list of tried addresses is refreshed each time a new incoming connection from any other node is established.
– The list of new addresses is refreshed when unsolicited address messages are received from any other node.
– Each node selects peer nodes from the list of tried addresses or new addresses and forms a number of long-lived outgoing connections with them.

Node routing table attacks:

Attackers attempt to corrupt the tried DLT system authentic IP addresses by propagating error routing or outside distributed ledger system IP address or tampering with the node routing table IP address data.

An NRT eclipse attack (NRT-EA) which is a typical routing table attack would have the following launch characteristics:

Step 0: An attacker controls a pool of compromised nodes.

Step 1: The attacker repeatedly establishes connections from attack nodes to target node.

Step 2: attack node overwrites the new address table of the target node by sending unsolicited address messages (UAM) full of meaningless IP addresses outside the distribute ledger network.

Step 3: The attack of step1 and 2 continues until the target node restarts and chooses new outgoing connections from the tried and new tables in its persistent storage. The target node establishes all outgoing connections to attack node since the target node cannot connect to the normal nodes in the DLT network.

Step 4: The attacker occupies the target node's remaining incoming connections by maintaining the connections for a long time, at which point the target node's will lose all awareness of the DLT network.

An analysis of eclipse attacks to the Bitcoin network is described in [b-USENIX2015].

Node routing table attack impact:

A successful network node routing table attack could have the following impact:

All connections to and from the target node are monopolized by attack node. This can lead to the following consequences:

1) The attacker can selectively filter a target node's view of the DLT to make it inconsistent with the view of normal nodes and thus disrupt the DLT network.
2) The attacker can block the generated blocks from an eclipsed node and waste the target node's effort on orphan blocks.
3) The attacker can split the computing power in the distributed ledger system to decrease the difficulty of attacks such as 51% attack [b-bitcoin2008].
4) The attacker can double spend at an eclipsed merchant.

Node routing table attack likelihood:

The likelihood or degree-of-difficulty to achieve a successful attack can be described as follows:

An attacker with enough IP addresses and time can eclipse any target node with a permanent IP address.

There are a number of parameters affecting the success of the eclipse attacks, including:

– Number of connections: less connections increase the success probability.

– Connection duration: shorter connection duration increases the success probability.
– Size of tried/new address list: smaller size increases the success probability.
– Freshness of the tried list: infrequent updating increases the success probability.
– Mechanism to select IP address that forms a long-lived outgoing connection.
– Anomaly detection: absence of anomaly detection in the node increases the success probability.
– Limitation on unsolicited messages including address list: lack of limitation increases the success probability.

Knowledge of P2P protocol used by the DLT is the first step to launch an eclipse attack. Open documentation of the P2P protocol details or open source of the software implementation decreases the difficulty of launching this attack.

6.2.2 Network DDoS threats

Targeted component: Network node and network connection (N)

The target under consideration are the devices or processes that participate in a distributed ledger network and the communication link between them.

Network DDoS attacks (N-DDOSA): This attack would have the following characteristics.

The distributed ledger network consists of nodes. When a large number of unconfirmed transactions (UTs) are sent to the nodes in the distributed ledger network in a short time, each nodes needs to receive, validate, transmit and (at least temporarily) store the data. As a consequence, a distributed ledger system could be flooded with an overwhelming number of network packets (NP). The connection bandwidth will be oversaturated and the node resources (e.g., buffers storing transactions waiting list) is depleted. Attackers can send "useless" transactions that serve no purpose other than consume scarce resources. An example would be sending many tiny fractions of digital currency that cannot feasibly be used in an actual payment transaction.

Network DDoS attack impact:

Spam transactions lead to a significant burden on the distributed ledger network and nodes in the network. They may cause a temporary congestion or denial of service.

Network DDoS attack likelihood:

The attack difficulty will have a positive correlation with:

– Number of nodes in the distributed ledger system
– Bandwidth of the distributed ledger network
– Block capacity
– Node buffer size

The cost to initiate a transaction will have a negative correlation with the intention to launch the attack.

6.2.3 Node identity threat

Targeted component: Network node identity (NNI)

The target under consideration is the network node identity which is bound with and represents a node. The identity can be identified and authenticated.

Network node identity attacks:

This attack would have the following characteristics.

Due to weaknesses in network node identity practices, a malicious node may:

– **Sybil attack (NNI-SA)**: Having more than one identity, multiple backups of the data may be stored in one malicious node. This may weaken or destroy the redundant backup mechanism of DLT.

– **Fraudulent node identity attack (NNI-FNIA)**: Impersonate the identity of a legitimate node (LN): the non-repudiation and the confidentiality of the node may be impacted.

Network node identity attack impact:

Flawed identity allocation may result in non-unique and indistinguishable nodes and result in some security problems and even make the system unable to run.

Network node identity attack likelihood:

The possibility of this threat depends on whether the nodes are unique and distinguishable.

6.2.4 Network routing threats

Targeted component(s): Network routing of Internet service providers (ISP).

The target under consideration is a network routing function of an Internet service provider (ISP). An ISP is an organization that provides services for accessing, using or participating in the Internet.

Distributed ledger systems with the following features could be threatened:

– Most of the DLT nodes are hosted in just a few ISPs.

– Most of the traffic exchanged between nodes traverses just a few ISPs.

Network routing attacks:

An ISP may divert the traffic by advertising fake announcements in the routing system. There are two ways to launch such routing attacks [b-IEEE2017].

– **Partition attack (ISP-PA)**: An attacker can use a hijack to partition the network into two (or more) disjointed segments. By preventing nodes within a component to communicate with nodes outside of it, the attacker forces the creation of parallel ledgers.

– **Delayed attack (ISP-DA)**: An attacker can use a hijack to delay the delivery of a block to a victim node while staying completely undetected. During this period the victim is unaware of the most recently generated block and the corresponding transactions.

Network routing attack impact:

After the partition attack stops, all blocks generated within the smaller component will be discarded together with all included transactions and the revenue. Delayed attack may make the nodes unable to propagate the last version of the ledger which will make the merchant susceptible to double spending attacks and may waste computational power.

Network routing attack likelihood:

Routing attacks are frequent. Whether the DLT nodes are spread uniformly and whether the traffic is traversed among few nodes impacts on whether a malicious ISP can intercept a lot of DLT traffic.

6.3 Threats to data

Data is generated in many places and many events of a distributed ledger system. This can be account data, transaction data, audit data, operations data, etc.

The threats to data can be further decomposed into following four groups:

– Account data and transaction data;

– Private key leakage;

– Private key loss; and
– Transactions.

6.3.1 Account data and transaction data threats

Targeted component: Account data and transaction data (ATD)

The target under consideration is the account and transaction data. Account is the representation of an entity whose data is recorded on a distributed ledger. Transaction is the whole of the exchange of information between nodes.

Account data and transaction data attacks:

There are different ways that may cause an unauthorized disclosure of sensitive account data and transaction data.

– **Public sensitive data attack (ATD-PSDA)**:
  - Sensitive transaction data stored in a distributed ledger system is public.
  - Account identity is account data and therefore participant identity data would be publicly available.
  - A smart contract may contain and process some account data or transaction data. Since the smart contract is running on every distributed ledger network, all public sensitive data (PSD) will be accessible by any node.
– **Analysis attack (ATD-AA)**:
  - Sensitive data can be analysed from the transaction data stored in a distributed ledger system.
  - Connections and sensitive data exchanged between nodes are monitored and/or traced.
  - Relationships between transaction accounts and nodes can be captured. It could be possible to identify or locate participants through big data analysis.
– **Unauthorized access attack (ATD-UAA)**
  - Account data and transaction data in distributed ledger system nodes may be searched or accessed by unauthorized and anonymous entities.

Account data and transaction data attack impact:

Successful attacks can cause sensitive unauthorized disclosure.

Account data and transaction data attack likelihood:

The likelihood of a successful attack depends on the value and the hidden degree of the sensitive data from account data and transaction data stored in the distributed ledger system.

6.3.2 Private key leakage threats

Targeted component: Private key of a public/private cryptographic key pair (PrK)

The private key(s) can be managed in the software client. The user (key owner) usually uses some method to protect the private key(s), for example, a personal identification number (PIN) code, a password, a gesture, or a fingerprint, etc.

The private key(s) can also be printed on paper or other objects.

Private key leakage attacks:

– **Software client attack (PrK-SCA)**: When the private key is stored in a software client,
  - the client device is malware infected and the key is exposed.
  - the participant loses his/her client device, and

- either the key is not adequately protected with PIN, fingerprint, etc.
- or the client software is reverse engineered and the private key is extracted.

– **Physical attack (PrK-PA)**: When the private key is printed on paper or other objects,
  - The user loses his/her printed private key and the unprotected key is exposed to others.

Private key leakage impact:

The private key is critical in establishing who has control and therefore ownership of a value token involved in a transaction. Control over the private key is the foundation of authentication of the source of a message and whether it has been modified since it was signed. Unauthorized access to a participant's private key can result in unauthentic messages and transactions. It can result in the theft of property, both digital and physical related control over the token. Unauthorized control of the token can result in its transfer from legitimate participant to attacker. In cases where significant value is concentrated in one value token, the impact will be greater. Also, in cases where multiple private keys are stored in one place or managed by a single person, these single points of failure can have significant consequences.

Private key leakage likelihood:

Private keys are widely used in distributed ledger systems. Given the critical nature of their use, private keys will always be a primary target. The development of new attack vectors is ongoing. Additionally the probability of losing the private key is not insignificant and special care must be taken.

6.3.3 Private key loss threats

Target components: Private key of a public/private cryptographic key pair (PrK)

The private key(s) can be managed in the software client. The user (key owner) usually uses some method to protect the private key(s), for example, a PIN code, a password, a gesture, or a fingerprint, etc.

The private key(s) can also be printed on paper or other objects.

Private key loss attacks:

Loss of a private key may be caused by several reasons:

– **Malware attack (PrK-MA)**: The client device is infected with malware and the key is rendered invalid.

– **Forget unlocking data (PrK-FUD)**: The user forgets his/her PIN code, password, or gesture, etc. to unlock the private key.

– **Unlocking loss (PrK-UL)**: The user losses his/her biometric characteristics to unlock the private key, e.g., injured finger(s).

– **Paper private key code loss (PrK-PCL)**: The user loses his/her printed private key and there is no backup.

Private key loss impact:

The loss of private keys will make it impossible to control the corresponding property (digital or physical). Lost property remains in the account, and no one else can control it, but the original key owner himself might not be able to reclaim it either.

In some circumstances, e.g., too much value is bound to a single private key, or keys for multiple signatures are stored in one place or managed by a single person, the consequences of the private key loss will be even worse.

Private key loss likelihood:

Probability of forgetting the unlocking secret or losing a client device or the printed private key is not insignificant.

The loss of one's biometric characteristics rarely happens, so private key(s) is less likely to be lost if it is protected by fingerprints, iris scan, etc.

6.3.4 Transactions threats

Targeted component: Transaction data (TD)

The target under consideration is transaction data which is the exchange of information between nodes.

A distributed ledger system with the following features could be threatened:

– The nodes can choose which valid transactions can be packed into a block when generating a block.

Transaction data attacks:

Threat to transaction data is the spam block attack (TD-SBA). Spam block refers to the block that has no transactions or is only composed of valid spam transactions. Nodes with the right to generate a valid block may generate spam blocks ignoring the transactions that are valuable and need to be dealt with.

Transaction data attack impact:

Spam blocks lead to a waste of block capacity and storage. Transactions that are valuable may not be dealt with in time and the ability of DLT to record will be weakened.

Transaction data attack likelihood:

The scale of a DLT network and the method to consensus the block will impact the attack difficulty. The profit to generate a block will impact the intention for the attackers to launch the attack.

7 How stakeholders can use this Recommendation

Below are some examples of primary stakeholders and corresponding usage suggestions:

1) DLT threats interested parties: This Recommendation provides non-ambiguous titles and descriptions to the threats to DLT so that stakeholders share at the same context.
2) DLT security framework developers: The common security threats described in this Recommendation shall be fully considered for the design of the general DLT security framework. Based on the identified threats, general security requirements, countermeasures and security framework can be provided.
3) Particular DLT application developers: In the design or implementation of a specific DLT application, developers can start by considering threats listed in this Recommendation as a baseline of high-level threats and design security mechanisms for these common threats as a start to security development.
4) Customers: The customers of the distributed ledger systems can have knowledge of what threats their data or assets will suffer and then claim security service level based on their service requirement.
5) Threat assessment: These identified common threats can be assessed. The common vulnerability scoring system (CVSS) model is used as an example (see Appendix I).
    a) On one hand, threats can be assessed independently of any particular system and prioritized relative to each other. Based on the generalized assessment results,

development managers can create strategies for prioritizing and mitigating threats, or at least the portion that is deemed to be most at risk, possibly by defining custom "Top-N" lists. Security service developers can focus on studying security solutions for the threats that are the most at risk.

b) On the other hand, for the particular distributed ledger systems, assessment can be adjusted in accordance with the needs of a specific context that may integrate business/mission priorities, threat environments, risk tolerance, etc. Developers are unable to investigate and fix every reported weakness due to limited time frames, release cycles and limited resources. They may choose to concentrate on the worst and the easiest-to-fix problems.

Annex A

Security features of DLT

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

The DLT architecture and key technologies can achieve some security features, for example, anti-DDoS, tamper-resistant, non-repudiation, etc. Thus, the platform and service system based on DLT naturally have the capabilities of these security protections.

1) Resistant to single-point failure: The distributed ledger system is resistant to single-point failure compared with a centralized system because of the following DLT technology features:
   - The control architecture of the distributed ledger system is decentralized or distributed which means the decision making relating to the distributed ledger is controlled by a number of architecture elements or all architecture elements based on a consensus.
   - The storage architecture of the distributed ledger system is decentralized or distributed which means that there are some nodes or each node store(s) a replica of the distributed ledger.
2) Tamper-resistant: The distributed ledger system is tamper-resistant because of technologies such as following DLT technology features:
   - The transactions are transmitted with signatures which avoids tampering with transactions in transmission.
   - Data are copied in many nodes, so that tampering with the data in one node cannot change the copy in other nodes.
   - A Merkle tree is constructed from the transactions in a block. The tampering of any transaction can make the root of the Merkle tree different. It is easy to find whether the transactions are tampered with by checking the Merkle root.
   - Blocks are chained by recording the head hash value of the former block in every block. Tampering of any data in one block's head can make its hash value different. It is easy to find whether the blocks are tampered with by checking the hash value. In distributed ledger systems using computing force competition for consensus, the hash value also needs to meet the target difficulty which requires lots of computing force.

Appendix I

DLT threat assessment

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

Table I.1 provides an analysis and assessment of the threats listed from multiple metrics according to CVSS [b-ITU-T X.1521].

Table I.1 – DLT threat assessment of the DLT

Threat	Exploitability metrics					Impact metrics		
	Attack vector	Attack complexity	Privileges required	User interaction	Scope	Confidentiality impact	Integrity impact	Availability impact
Node routing table threat	Network	High	Low	None	Changed	None	High	High
Network DDoS threat	Network	Low	None	None	Changed	None	None	High
Node identity threats	Network	High	Low	None	Changed	Low	Low	High
Network routing threats	Network	High	Low	None	Changed	None	None	High
Account and transaction data threats	Network	Low	Low	None	Unchanged	High	None	None
Transaction threat	Network	Low	None	None	Changed	None	None	High
Cryptographic hash algorithm threats	Network	High	High	Required	Changed	High	High	High
Asymmetric cryptographic algorithm threats	Network	High	Low	Required	Changed	High	Middle	Low
Threats from practical quantum computers	Network	High	Low	Required	Changed	High	High	High
Consensus mechanism threat	Network	High	None	None	Changed	None	High	High
Smart contract threats	Network	Low	Low	Required	Changed	Middle	Middle	High
Virtual Machine threats	Network	High	Low	Required	Changed	Middle	Middle	High
PrK leakage threats	Network	Low	Low	Required	Changed	High	Middle	Low
PrK loss threats	Network	Low	Low	Required	Unchanged	None	None	High

Appendix II

Impact of quantum computing on common cryptographic algorithms

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

Table II.1 shows the impact of quantum computers on the commonly used cryptographic algorithms according to [b-NISTIR 8105].

Table II.1 – Impact of quantum computers on commonly used cryptographic algorithms

Cryptographic algorithm	Type	Use	Impact
AES	Symmetric	Encryption	Large key sizes
SHA-2, SHA-3	Hash	Hash function	Larger output
RSA	Public key	Signature, key transport	No longer secure
ECDSA, ECDH	Public key	Signature, key exchange	No longer secure
DSA	Public key	Signature, key exchange	No longer secure

Appendix III

Risk | DLT system | Protection security expression model

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

III.1 Abbreviations and acronyms

For a list of the abbreviations and acronyms used in this appendix please refer to clause 4.

III.2 Risk | DLT system | Protection security expression model

A DLT system (DLTS) is composed of DLTS component classes (DLTS-CC) representing core types of components types such as applications, network elements and protocols. Each DLTS-CC component represents a specific component attack surface (CAS), herein after generally referred to as the "target" under specific consideration.

The "target under consideration" at the DLTS highest level is illustrated in the centre of Figure III.1, the DLTS security expression model (DSEM). The model represents the threats to the target on the left side as DLTS Risk (DLTS-R) and protections from those threats on the right side as DLTS-protection (DLTS-P).

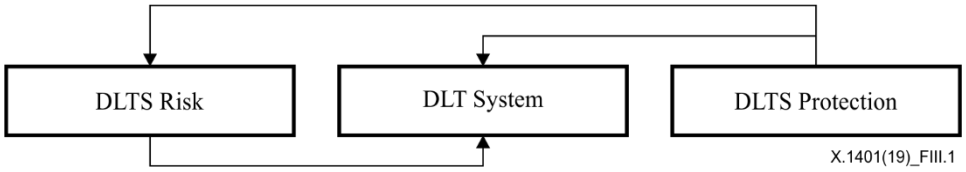

Figure III.1 – DLTS security expression model: System risk to DLT system and DLT protection

The model in Figure III.1 remains consistent as it decomposes and represents smaller and smaller targets. Figure III.2 illustrates the hierarchical structure between a DLTS through increasingly smaller and smaller targets from component class, to component attack surface and to CAS vulnerability and corresponding threats and security applied to the same level.

Starting from the highest aggregated level corresponding to DLTS as a target to the intermediary component class target level, to the lowest target level corresponding to component and its vulnerabilities, the expression architecture naming is as follows:

- **Risk | System | Protection** level 1 System Expression
- **Threat Class | Component Class | Security Class** level 2 Class Expression
- **Threats | Component Attack Surface | Security Technique** level 3 Threat Expression
- **Attack| Component Vulnerability | Countermeasure** level 4 Attack Expression

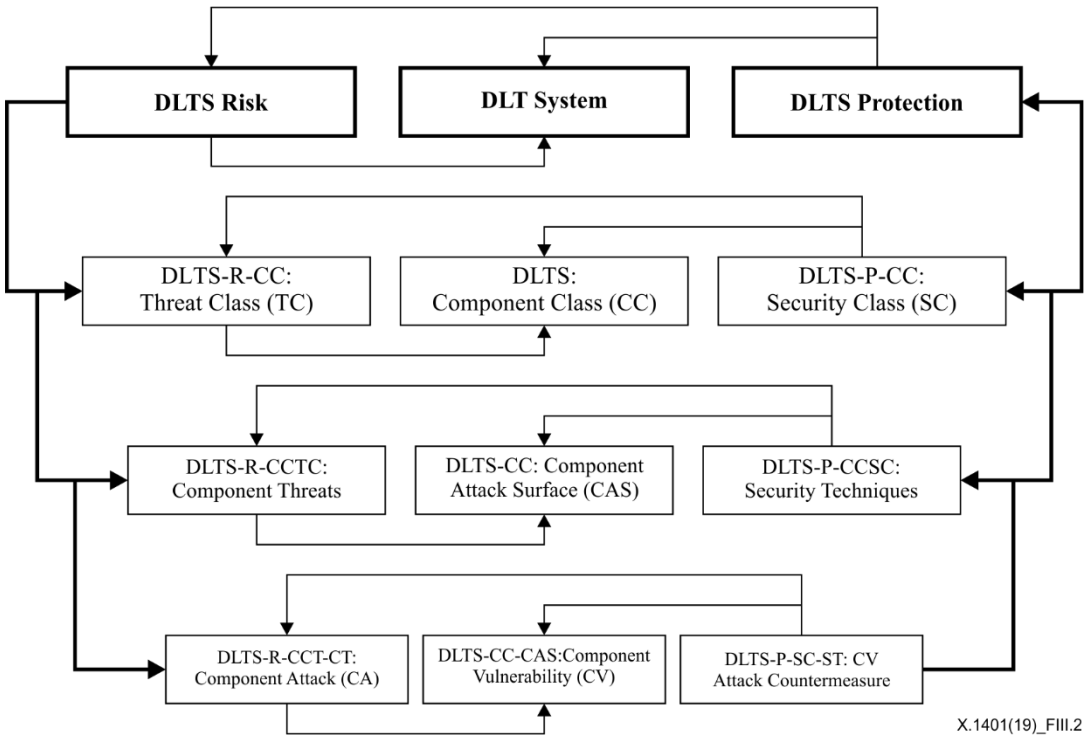

Figure III.2 – DLTS security expression model hierarchy

Figure III.1 and III.2 represent an expression architecture agnostic to component class (CC). There are three CCs that will be considered more focused on infrastructure.

1) **Protocol** Component Class (PCC), represents protocol components as targets.

2) **Network** Component Class (NCC), represents network components as targets.

3) **Data** Component Class (DCC), represents data components as targets.

That can be applied consistently across all CCs as follows and as is illustrated in Figure III.3.

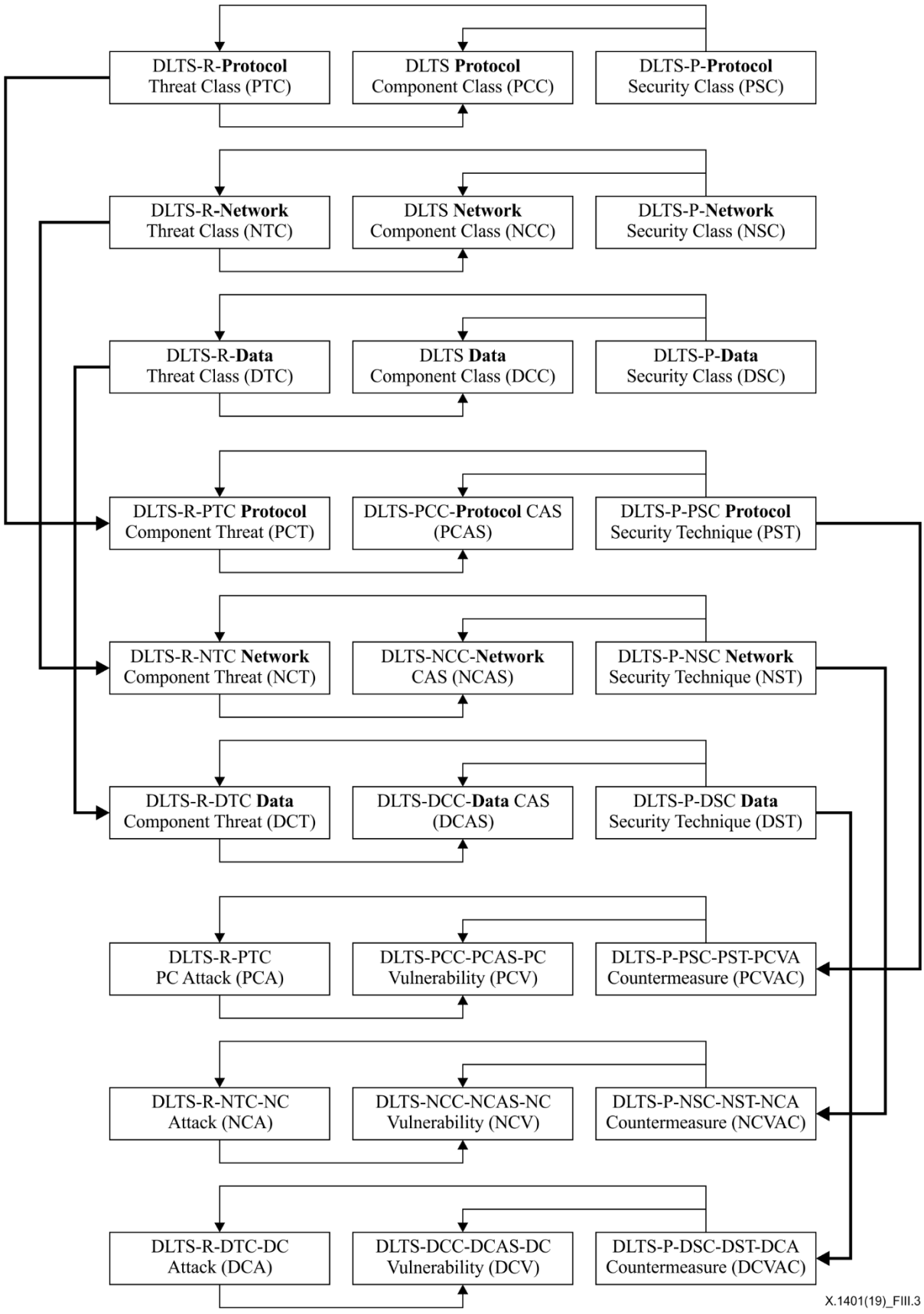

Figure III.3 – DLTS Level 2, 3, 4 architecture for each component class

Figure III.3 lays out the architecture across a component class at a given level, alternatively it can be arranged as a decomposition ally for each component class as illustrated in Figure III.4. Based on the architecture presented, the following are the DLTS component class decompositions.

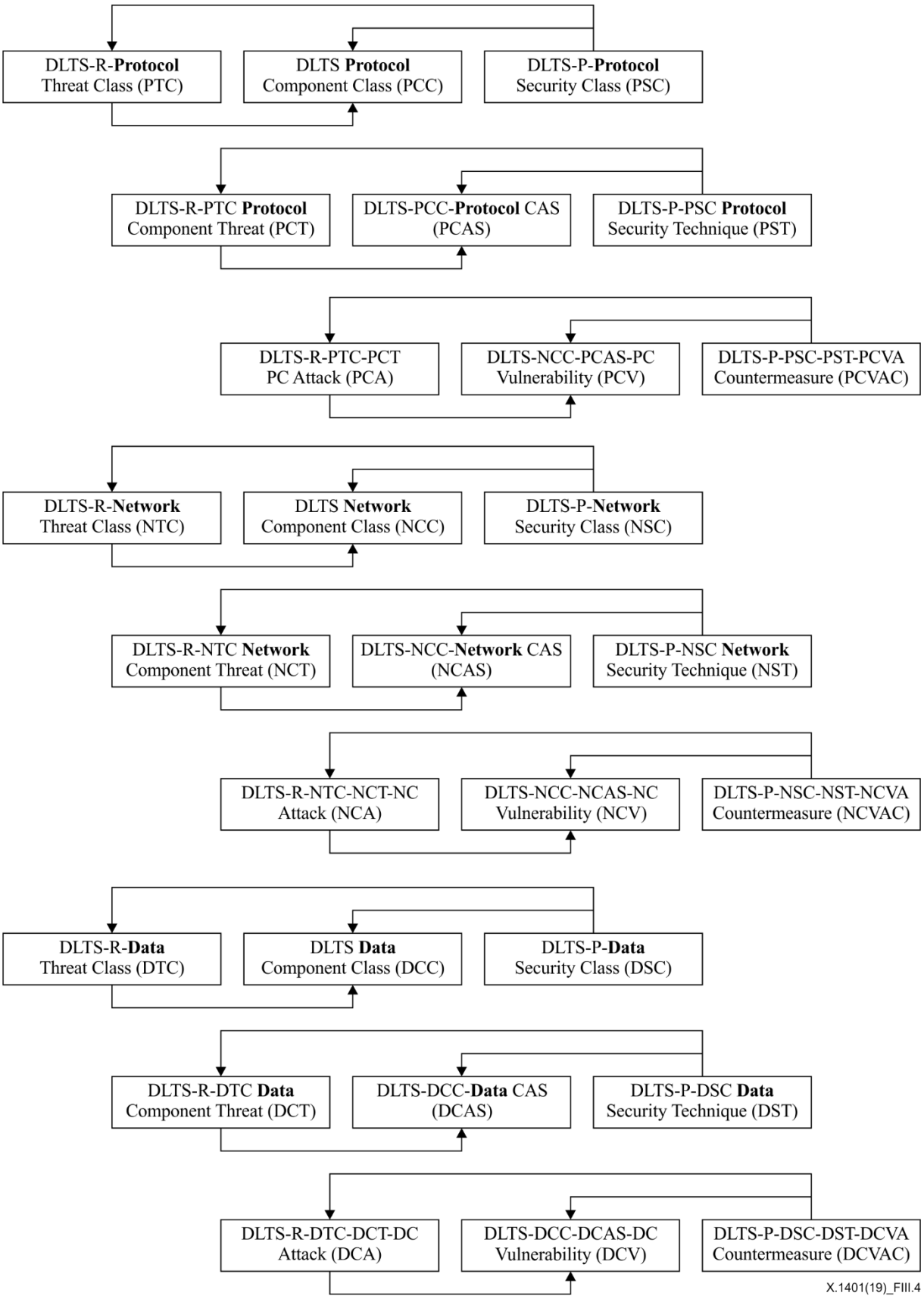

Figure III.4 – For each component class, DLTS Level 2, 3, 4 interconnected architecture

The structure and data illustrated in Figure III.4 can be listed as in the format of Table III.1. As the nomenclature indicates, as the target is smaller and smaller (lower in Table III.1), the inherited parentage is embedded in the nomenclature.

Table III.1 – DLTS security expression model nomenclature

DLTS SECURITY EXPRESSION MODEL NOMENCLATURE BY COMPONENT CLASS							
ID	**LEVEL**	**RISK**		**TARGET**		**PROTECTION**	
1.0	1.0	**SYSTEM RISK**		**DLT SYSTEM**		**SYSTEM PROTECTION**	
1.1	PND	DLTS Risk	DLTS-R	DLT System	DLTS	DLTS-Protection	DLTS-P
2.0	2.0	**CC THREAT CLASS (TC)**		**COMPONENT CLASS (CC)**		**CC SECURITY CLASS (SC)**	
2.1	Protocol	Protocol Threat Class	DLTS-R-PTC	Protocol Component Class	DLTS-PCC	Protocol Security Class	DLTS-P-PSC
2.2	Network	Network Threat Class	DLTS-R-NTC	Network Component Class	DLTS-NCC	Network Security Class	DLTS-P-NSC
2.3	Data	Data Threat Class	DLTS-R-DTC	Data Component Class	DLTS-DCC	Data Security Class	DLTS-P-DSC
3.0	3.0	**COMPONENT THREATS**		**CC COMPONENT ATTACK SURFACE (CAS)**		**COMPONENT SECURITY TECHNIQUE (ST)**	
3.1	Protocol	Protocol Component Threat	DLTS-R-PTC-PCT	Protocol Component Attack Surface	DLTS-PCC-PCAS	Protocol Security Technique	DLTS-P-PSC-PST
3.2	Network	Network Component Threat	DLTS-R-NTC-NCT	Network Component Attack Surface	DLTS-NCC-NCAS	Network Security Technique	DLTS-P-NSC-NST
3.3	Data	Data Component Threat	DLTS-R-DTC-DCT	Data Component Attack Surface	DLTS-DCC-DCAS	Data Security Technique	DLTS-P-DSC-DST
4.0	4.0	**CC COMPONENT ATTACK**		**CC COMPONENT VULNERABILITY**		**CC COMPONENT SECURITY COUNTERMEASURE (SC)**	
4.1	Protocol	Protocol Component Attack	DLTS-R-PTC-PCA	Protocol Component Vulnerability	DLTS-PCC-PCAS-PCV	Protocol Component Vulnerability Attack Countermeasure	DLTS-P-PSC-PST-PCVAC
4.2	Network	Network Component Attack	DLTS-R-NTC-NCA	Network Component Vulnerability	DLTS-NCC-NCAS-NCV	Network Component Vulnerability Attack Countermeasure	DLTS-P-NSC-NST-NSVAC
4.3	Data	Data Component Attack	DLTS-R-DTC-DCA	Data Component Vulnerability	DLTS-DCC-DCAS-DCV	Data Component Vulnerability Attack Countermeasure	DLTS-P-DSC-DST-DSVAC

The following clauses will discuss specific attacks to specific component vulnerabilities. These attacks are categorized by threat class to component class and threat to components as listed in Table III.1.

III.3 Threats to DLT system

DLTS threat class to component classes is further decomposed into the threats to components and attacks to component vulnerability. Table III.2 below provides threat class, threats, and attacks on protocol, network and data DLTS components. It provides a complete and integrated threat nomenclature and numbering system to tie all target and threat elements together. The general threats identified below can be referred and utilized from other Recommendations by using the ID.

Table III.2 – List of DLTS threats by Component Class

Ind #	ID	COMPONENT CLASS THREATS	COMPONENT THREATS	COMPONENT VULNERABILITY ATTACKS	ACRONYM
1.0	P	Protocol Threat Class (PTC)	PTC Threats	Protocol Component Attacks (PCA)	
1.1	P-CMT		**Consensus Mechansim Threats**		CM
1.1.1	P-CM-51			**51% Attack**	CM-51A
1.1.2	P-CM-TM			**Timestamp Manipulation Attack**	CM-TMA
1.1.3	P-CM-B			**Bribing Attack**	CM-BA
1.1.4	P-CM-SM			**Selfish Mining Attack**	CM-SMA
1.1.5	P-CM-CH			**Chain Hopping Attack**	CM-CHA
1.1.6	P-CM-BW			**Block Withholding Attack**	CM-BWA
1.1.7	P-CM-DS			**Double-Spending Attack**	CM-DSA
1.2	P-SCT		**Smart Contract Threats**		SC
1.2.1	P-SC-TD			**Timestamp Dependence Attack**	SC-TDA
1.2.2	P-SC-ME			**Mishandled Exceptions Attack**	SC-MEA
1.2.3	P-SC-IO			**Integer Overflow Attack**	SC-OIA
1.2.4	P-SC-PRN			**Predictable Random Number Attack**	SC-PRNA
1.3	P-VMT		**Virtual Machine Threats**		VM
1.3.1	P-VM-E			**Escape Attack**	VM-EA
1.3.2	P-VM-FH			**Fault Handling Attack**	VM-FHA
1.3.3	P-VM-MC			**Memory Corruption Attack**	VM-MCA
1.4	P-CHAT		**Cryptographic Hash Algorithm Threats**		CHA
1.4.1	P-CHA-C			**Collision Attack**	CHA-HCA
1.4.2	P-CHA-SP			**Second Preimage Attack**	CHA-SPA
1.4.3	P-CHA-PE			**Preimage Attack**	CHA-FPA
1.5	P-ACAT		**Asymmetric Cryptographic Algorithm Threats**		ACA
1.5.1	P-ACA-WKMA			**Weak Key Material Attack**	ACA-WKMA
1.5.2	P-ACA-BA			**Backdoor Attack**	ACA-BA
1.5.3	P-ACA-MCCA			**Mathematical Cryptanalysis Cracking Attack**	ACA-MCCA
1.5.4	P-ACA-PMMA			**Protocol Message Manipulation Attack**	ACA-PMMA

Table III.2 – List of DLTS threats by Component Class

1.6	P-PQCT		**Practical Quantum Computers Threats**		PQC
1.6.1	P-PQC-CPA			**Cryptographic Protocol Attack**	PQC-CPA
1.6.2	P-PQC-BFCA			**Brute Force Cracking Attack**	PQC-BFCA
1.6.3	P-PQC-DSA			**Digital Signature Attack**	PQC-DSA
2.0	**N**	**Network Threat Class (NTC)**	**Network Components Threats**	**Network Component Attacks (NCA)**	
2.1	N-NRTT		**Node Routing Table Threat**		NRT
2.1.1	N-NRT-EA			**Eclipse Attack**	NRT-EA
2.2	N-DDOST		**Network DDOS Threat**		DDOST
2.2.1	N-DDOS-DA			**DDoS attack**	N-DDOSA
2.3	N-NIT		**Node Identity Threats**		NNI
2.3.1	N-NI-SA			**Sybil Attack**	NNI-SA
2.3.2	N-NI-FNIA			**Fraudulent Node Identity Attack**	NNI-FNIA
2.4	NRT		**Network Routing Threats**		NRT
2.4.1	NR-PA			**Partition Attack**	ISP-PA
2.4.2	NR-DA			**Delayed Attack**	ISP-DA
3.0	**D**	**Data Threat Class (DTC)**	**Data Component Threats**	**Data Component Attacks (DCA)**	
3.1	D-ATDT		**Account Data & Transaction Data Threats**		ATD
3.1.1	D-ATD-PSDA			**Public Sensitive Data Attack**	ATD-PSDA
3.1.2	D-ATD-AA			**Analysis Attack**	ATD-AA
3.1.3	D-ATD-UAA			**Unauthorized Access Attack**	ATD-UAA
3.2	D-PKLeT		**Private Key Leakage Threats**		PKLeT
3.2.1	D-PKLe-SCA			**Software Client Attack**	PrK-SCA
3.2.2	D-PKLe-PA			**Physical Attack**	PrK-PA
3.3	D-PKLoT		**Private Key Loss Threats**		PKLoT
3.3.1	D-PKLo-MA			**Malware Attack**	PrK-MA
3.3.2	D-PKLo-FUD			**Forget Unlocking Data**	PrK-FUD
3.3.3	D-PKLo-UL			**Unlocking Loss**	PrK-UL
3.3.4	D-PKLo-PCL			**Paper Private Key Code Loss**	PrK-PCL

Table III.2 – List of DLTS threats by Component Class

3.4	D-TT		**Transactions Threat**		TT
3.4.1	D-TT-SBA			**Spam Block Attack**	TD-SBA

III.3.1 Threats to protocol components

Table III.3 shows the protocol security expression model nomenclature. The targets under consideration fall under protocol component class (PCC). Protocols are the configurations that drive network behaviour. Consequently, any unauthorized modification of protocols can have a significant impact on DLTS behaviour.

Table III.3 – Protocol security expression model nomenclature

DLTS PROTOCOL SECURITY EXPRESSION MODEL NOMENCLATURE							
ID	**LEVEL**	**PROTOCOL RISK**		**PROTOCOL TARGET**		**PROTOCOL PROTECTION**	
1.0	1.0	**SYSTEM RISK**		**DLT SYSTEM**		**SYSTEM PROTECTION**	
1.1	PND	DLTS Risk	DLTS-R	DLT System	DLTS	DLTS-Protection	DLTS-P
2.0	2.0	**CC THREAT CLASS (TC)**		**COMPONENT CLASS (CC)**		**CC SECURITY CLASS (SC)**	
2.1	Protocol	Protocol Threat Class	DLTS-R-PTC	Protocol Component Class	DLTS-PCC	Protocol Security Class	DLTS-P-PSC
3.0	3.0	**COMPONENT THREATS**		**CC COMPONENT ATTACK SURFACE (CAS)**		**COMPONENT SECURITY TECHNIQUE (ST)**	
3.1	Protocol	Protocol Component Threat	DLTS-R-PTC-PCT	Protocol Component Attack Surface	DLTS-PCC-PCAS	Protocol Security Technique	DLTS-P-PSC-PST
4.0	4.0	**CC COMPONENT ATTACK**		**CC COMPONENT VULNERABILITY**		**CC COMPONENT SECURITY COUNTERMEASURE (SC)**	
4.1	Protocol	Protocol Component Attack	DLTS-R-PTC-PCA	Protocol Component Vulnerability	DLTS-PCC-PCAS-PCV	Protocol Component Vulnerability Attack Countermeasure	DLTS-P-PSC-PST-PCVAC

The list of protocol threats in Table III.4 below is of specific attacks under Level 4 of the left or threat side of Table III.3 above.

Table III.4 – List of protocol component threats

Ind #	ID	COMPONENT CLASS THREATS	COMPONENT THREATS	COMPONENT VULNERABILITY ATTACKS	ACRONYM
1.0	P	**Protocol Threat Class (PTC)**	**PTC Threats**	**Protocol Component Attacks (PCA)**	
1.1	P-CMT		**Consensus Mechansim Threats**		CM
1.1.1	P-CM-51			**51% Attack**	CM-51A
1.1.2	P-CM-TM			**Timestamp Manipulation Attack**	CM-TMA
1.1.3	P-CM-B			**Bribing Attack**	CM-BA
1.1.4	P-CM-SM			**Selfish Mining Attack**	CM-SMA
1.1.5	P-CM-CH			**Chain Hopping Attack**	CM-CHA
1.1.6	P-CM-BW			**Block Withholding Attack**	CM-BWA
1.1.7	P-CM-DS			**Double-Spending Attack**	CM-DSA

Table III.4 – List of protocol component threats

1.2	P-SCT		**Smart Contract Threats**		SC
1.2.1	P-SC-TD			**Timestamp Dependence Attack**	SC-TDA
1.2.2	P-SC-ME			**Mishandled Exceptions Attack**	SC-MEA
1.2.3	P-SC-IO			**Integer Overflow Attack**	SC-OIA
1.2.4	P-SC-PRN			**Predictable Random Number Attack**	SC-PRNA
1.3	P-VMT		**Virtual Machine Threats**		VM
1.3.1	P-VM-E			**Escape Attack**	VM-EA
1.3.2	P-VM-FH			**Fault Handling Attack**	VM-FHA
1.3.3	P-VM-MC			**Memory Corruption Attack**	VM-MCA
1.4	P-CHAT		**Cryptographic Hash Algorithm Threats**		CHA
1.4.1	P-CHA-C			**Collision Attack**	CHA-HCA
1.4.2	P-CHA-SP			**Second Preimage Attack**	CHA-SPA
1.4.3	P-CHA-PE			**Preimage Attack**	CHA-FPA
1.5	P-ACAT		**Asymmetric Cryptographic Algorithm Threats**		ACA
1.5.1	P-ACA-WKMA			**Weak Key Material Attack**	ACA-WKMA
1.5.2	P-ACA-BA			**Backdoor Attack**	ACA-BA
1.5.3	P-ACA-MCCA			**Mathematical Cryptanalysis Cracking Attack**	ACA-MCCA
1.5.4	P-ACA-PMMA			**Protocol Message Manipulation Attack**	ACA-PMMA
1.6	P-PQCT		**Practical Quantum Computers Threats**		PQC
1.6.1	P-PQC-CPA			**Cryptographic Protocol Attack**	PQC-CPA
1.6.2	P-PQC-BFCA			**Brute Force Cracking Attack**	PQC-BFCA
1.6.3	P-PQC-DSA			**Digital Signature Attack**	PQC-DSA

III.3.2 Threats to network components

The target under consideration is the network component class (NCC). The NCC nomenclature is shown in Table III.5.

Table III.5 – Network component class nomenclature

DLTS NETWORK SECURITY EXPRESSION MODEL NOMENCLATURE							
ID	**LEVEL**	**RISK**		**TARGET**		**PROTECTION**	
1.0	1.0	**SYSTEM RISK**		**DLT SYSTEM**		**SYSTEM PROTECTION**	
1.1	PND	DLTS Risk	DLTS-R	DLT System	DLTS	DLTS-Protection	DLTS-P
2.0	2.0	**CC THREAT CLASS (TC)**		**COMPONENT CLASS (CC)**		**CC SECURITY CLASS (SC)**	
2.2	Network	Network Threat Class	DLTS-R-NTC	Network Component Class	DLTS-NCC	Network Security Class	DLTS-P-NSC
3.0	3.0	**COMPONENT THREATS**		**CC COMPONENT ATTACK SURFACE (CAS)**		**COMPONENT SECURITY TECHNIQUE (ST)**	
3.2	Network	Network Component Threat	DLTS-R-NTC-NCT	Network Component Attack Surface	DLTS-NCC-NCAS	Network Security Technique	DLTS-P-NSC-NST
4.0	4.0	**CC COMPONENT ATTACK**		**CC COMPONENT VULNERABILITY**		**CC COMPONENT SECURITY COUNTERMEASURE (SC)**	
4.2	Network	Network Component Attack	DLTS-R-NTC-NCA	Network Component Vulnerability	DLTS-NCC-NCAS-NCV	Network Component Vulnerability Attack Countermeasure	DLTS-P-NSC-NST-NSVAC

The list of network component threats in Table III.6 below is of specific attacks under Level 4 of the left or threat side of Table III.5 above.

Table III.6 – List of network component threats

Ind #	ID	COMPONENT CLASS THREATS	COMPONENT THREATS	COMPONENT VULNERABILITY ATTACKS	ACRONYM
2.0	N	**Network Threat Class (NTC)**	**Network Components Threats**	**Network Component Attacks (NCA)**	
2.1	N-NRTT		**Node Routing Table Threat**		NRT
2.1.1	N-NRT-EA			**Eclipse Attack**	NRT-EA
2.2	N-DDOST		**Network DDOS Threat**		DDOST
2.2.1	N-DDOS-DA			**DDoS attack**	N-DDOSA
2.3	N-NIT		**Node Identity Threats**		NNI
2.3.1	N-NI-SA			**Sybil Attack**	NNI-SA
2.3.2	N-NI-FNIA			**Fraudulent Node Identity Attack**	NNI-FNIA
2.4	NRT		**Network Routing Threats**		NRT
2.4.1	NR-PA			**Partition Attack**	ISP-PA
2.4.2	NR-DA			**Delayed Attack**	ISP-DA

III.3.3 Threats to data components

The target under consideration is the data component class (DCC).

Data is generated in many places and many events of a DLTS. This can be account data, transaction data, audit data, operations data, etc. Table III.7 shows data component class nomenclature.

Table III.7 – Data component class nomenclature

DLTS DATA SECURITY EXPRESSION MODEL NOMENCLATURE							
ID	**LEVEL**	**DATA RISK**		**DATA TARGET**		**DATA PROTECTION**	
1.0	1.0	**SYSTEM RISK**		**DLT SYSTEM**		**SYSTEM PROTECTION**	
1.1	PND	DLTS Risk	DLTS-R	DLT System	DLTS	DLTS-Protection	DLTS-P
2.0	2.0	**CC THREAT CLASS (TC)**		**COMPONENT CLASS (CC)**		**CC SECURITY CLASS (SC)**	
2.3	Data	Data Threat Class	DLTS-R-DTC	Data Component Class	DLTS-DCC	Data Security Class	DLTS-P-DSC
3.0	3.0	**COMPONENT THREATS**		**COMPONENT ATTACK SURFACE (CAS)**		**COMPONENT SECURITY TECHNIQUE (ST)**	
3.3	Data	Data Component Threat	DLTS-R-DTC-DCT	Data Component Attack Surface	DLTS-DCC-DCAS	Data Security Technique	DLTS-P-DSC-DST
4.0	4.0	**CC COMPONENT ATTACK**		**CC COMPONENT VULNERABILITY**		**CC COMPONENT SECURITY COUNTERMEASURE (SC)**	
4.3	Data	Data Component Attack	DLTS-R-DTC-DCA	Data Component Vulnerability	DLTS-DCC-DCAS-DCV	Data Component Vulnerability Attack Countermeasure	DLTS-P-DSC-DST-DSVAC

The list of data threats in Table III.8 below is of specific attacks under Level 4 of the left or threat side of Table III.7 above.

Table III.8 – List of data component threats

Ind #	ID	COMPONENT CLASS THREATS	COMPONENT THREATS	COMPONENT VULNERABILITY ATTACKS	ACRONYM
3.0	D	**Data Threat Class (DTC)**	**Data Component Threats**	**Data Component Attacks (DCA)**	
3.1	D-ATDT		**Account Data & Transaction Data Threats**		ATD
3.1.1	D-ATD-PSDA			**Public Sensitive Data Attack**	ATD-PSDA
3.1.2	D-ATD-AA			**Analysis Attack**	ATD-AA
3.1.3	D-ATD-UAA			**Unauthorized Access Attack**	ATD-UAA
3.2	D-PKLeT		**Private Key Leakage Threats**		PKLeT
3.2.1	D-PKLe-SCA			**Software Client Attack**	PrK-SCA
3.2.2	D-PKLe-PA			**Physical Attack**	PrK-PA
3.3	D-PKLoT		**Private Key Loss Threats**		PKLoT
3.3.1	D-PKLo-MA			**Malware Attack**	PrK-MA
3.3.2	D-PKLo-FUD			**Forget Unlocking Data**	PrK-FUD
3.3.3	D-PKLo-UL			**Unlocking Loss**	PrK-UL
3.3.4	D-PKLo-PCL			**Paper Private Key Code Loss**	PrK-PCL
3.4	D-TT		**Transactions Threat**		TT
3.4.1	D-TT-SBA			**Spam Block Attack**	TD-SBA

Bibliography

[b-ITU-T X.1521] Recommendation ITU-T X.1521 (2016), *Common vulnerability scoring system*.

[b-ITU-T FG DLT D1.1] ITU-T Focus Group on Application of Distributed Ledger Technology (FG DLT), Technical Specification FG DLT D1.1 (2019), *Distributed ledger technology terms and definitions*.

[b-IEEE2017] M. Apostolaki, A. Zohar, L. Vanbever. *Hijacking Bitcoin: Routing Attacks on Cryptocurrencies*. In Proceedings of IEEE Symposium on Security and Privacy 2017.

[b-IEEE1985] T. E. Gamal. *A public key cryptosystem and a signature scheme based on discrete logarithms*. In Proceedings of IEEE Transactions on Information Theory, 1985, 31:469-472.

[b-ISO/IEC 27000] ISO/IEC 27000:2018, *Information technology – Security techniques – Information security management systems – Overview and vocabulary*.

[b-ISO 22739] Recommendation ISO/TC307 22739, *Blockchain and Distributed Ledger Technologies – Terminology and Concepts*.

[b-ACM2016] L. Luu, DH. Chu, H. Olickel, P. Saxena, A. Hobo. *Making Smart Contracts Smarter*. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2016.

[b-ARXIV2013] I. Eyal, E.G. Sirer. *Majority Is Not Enough: Bitcoin Mining Is Vulnerable.* Eprint Arxiv, 2013.

[b-ASIACRYPT2009] K. Aoki, J. Guo, K. Matusiewicz, Y. Sasaki, L. Wang. *Preimages for Step-Reduced SHA-2*. In Proceedings of Asiacrypt 2009.

[b-BBS04] D. Boneh, X. Boyen, H. Shacham. *Short Group Signatures. In Proceedings of Crypto'04*, vol. 3152 of LNCS, pp.41–55. Springer-Verlag, 2004.

[b-BITCOIN2008] S. Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. https://bitcoin.org/bitcoin.pdf.

[b-C85] D. Chaum. Security without identification: *Transaction systems to make big brother obsolete. Communications of the ACM*, 28(10):1030–1044, 1985.

[b-CH02] J. Camenisch and E. V. Herreweghen. *Design and Implementation of the idemix Anonymous Credential System. In Proceedings of the 9th ACM conference on Computer and communications security*, pp. 21–30, 2002.

[b-CRYPTO2008] C. D. Cannière, C. Rechberger. *Preimages for Reduced SHA-0 and SHA-1*. In Proceedings of Crypto 2008.

[b-CRYPTO2017] M. Stevens, E. Bursztei, P. Karpman, A. Albertini, Y. Markov. *The first collision for full SHA-1*. In Proceedings of Crypto 2017.

[b-D88] I. B. Damgard. *Payment systems and credential mechanism with provable security against abuse by individuals*. In Proceedings of Crypto'88, vol. 403 of LNCS, pp.328–335. Springer-Verlag, 1990.

[b-DFNS11] I. Damgard, J. Funder, J. B. Nielsen, L. Salvail. *Superposition Attacks on Cryptographic Protocols*. Cryptology ePrint archive, report 2011/421. http://eprint.iacr.org/ (2011).

[b-DPSW16] J. P. Degabriele, K. G. Paterson, J. C. N. Schuldt, J. Woodage. *Backdoors in Pseudorandom Number Generators: Possibility and Impossibility Results*. In Proceedings of Crypto 2016.

[b-ECDSA2001] D. Johnson, A. Menezes, and S. Vanstone. *The elliptic curve digital signature algorithm (ecdsa). International Journal of Information Security*, vol. 1, no. 1, pp. 36–63, 2001.

[b-EUROCRYPT2009] Y. Sasaki, K. Aoki. *Finding Preimages in Full MD5 Faster Than Exhaustive Search*. In Proceedings of Eurocrypt 2009.

[b-HCCN15] J. Y. Hwang, L. Chen, H. S. Cho, D. Nyang. *Short Dynamic Group Signature Scheme Supporting Controllable Linkability. IEEE Transactions on Information Forensics and Security*, 10(6), pp.1109–1124, 2015.

[b-IACR2009] Y. Sasaki, L. Wang, K. Aoki. *Preimage Attacks on 41-Step SHA-256 and 46-Step SHA-512. IACR Cryptology Eprint Archive, 2009*.

[b-INFOCOM2006] A. Singh, T. Ngan, P. Druschel, and D. Wallach. *Eclipse Attacks on Overlay Networks: Threats and Defenses*. In Proceedings of IEEE INFOCOM 2006.

[b-M87] R. Merkle. *A Digital Signature Based on a Conventional Encryption Function*. In Proceedings of Crypto'87, vol. 293 of LNCS, pp.369–378. Springer-Verlag, 1987.

[b-NISTIR 8202] NISTIR 8202:2018, *Blockchain Technology Overview*.

[b-NISTIR 8105] NISTIR 8105:2016, *Report on Post-Quantum Cryptography*.

[b-Shor97] Peter W. Shor. *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. SIAM J. Comput.*, 26 (5): pp. 1484–1509, 1997.

[b-USENIX2015] E. Heilman A. Kendler, A. Zohar, and S. Goldberg. *Eclipse Attacks on Bitcoin's Peer-to-Peer Network*. In Proceedings of the 24th USENIX Security Symposium, Washington D. C., USA, 2015. Available: https://www.usenix.org/system/files/conference/usenixsecurity15/sec15-paper-heilman.pdf

[b-Shor97] Peter W. Shor. *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. SIAM J. Comput.*, 26 (5): pp. 1484–1509, 1997.

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, 2020