International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

X.1149

(05/2020)

SERIES X: DATA NETWORKS, OPEN SYSTEM COMMUNICATIONS AND SECURITY

Secure applications and services (1) – Web security

Security framework of an open platform for FinTech services

Recommendation ITU-T X.1149

ITU-T

ITU
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	
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
5G SECURITY	X.1800–X.1819

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

Recommendation ITU-T X.1149

Security framework of an open platform for FinTech services

Summary

Recommendation ITU-T X.1149 describes an open platform architecture for financial technology (FinTech) services. It also specifies threats and vulnerabilities of open platform, open application programming interface (API) usage procedure for FinTech services, and detailed security requirements to open platform of FinTech services from both financial company and FinTech company sides. The appendix to this Recommendation includes some use cases of the proposed open platform.

History

Edition	Recommendation	Approval	Study Group	Unique ID[*]
1.0	ITU-T X.1149	2020-05-29	17	11.1002/1000/14250

Keywords

FinTech, open API, open platform.

[*] 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 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

FinTech revolution has been one of increasing interest in the last few years. It has disrupted the status quo of financial industry, modernized legacy financial institutions, and changed the way consumers access financial products and services. As FinTech start-ups grow in number and sophistication, they are establishing an increasing number of links with traditional financial institutions. However, interfaces between systems are a common source of cybersecurity vulnerabilities, often caused by mismatched configurations and parameters of the systems being connected.

This Recommendation provides an open platform architecture to identify threats and vulnerabilities of interfaces among financial companies, FinTech companies and users, and specifies security requirements for FinTech services to enhance competitiveness of the financial industry to offer diversified users' options.

Recommendation ITU-T X.1149

Security framework of an open platform for FinTech services

1 Scope

This Recommendation provides a security framework for an open platform that supports financial technology (FinTech) services.

This Recommendation demonstrates the evolution of the FinTech service platform to an open platform, analyses threats and vulnerabilities of open platform and open application programming interface (API), describes an open platform architecture and an open API usage procedure for FinTech services, and specifies detailed security requirements for FinTech services. Appendix I of this Recommendation describes use cases of the open platform API.

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 open standards [b-ITU-T L.1370]: Standards made available to the general public and which are developed (or approved) and maintained via a collaborative and consensus driven process.

NOTE – Open standards facilitate interoperability and data exchange among different products or services and are intended for widespread adoption.

3.1.2 patch [b-ITU-T X.1206]: A broadly released fix for a product-specific, security-related vulnerability. A method of updating a file that replaces only the parts being changed, rather than the entire file.

3.2 Terms defined in this Recommendation

This Recommendation defines the following terms:

3.2.1 access key: A string issued by an authorization server to access the protected resources hosted by the resource server.

NOTE – In an open platform, the financial technology (FinTech) company requests the access key issued by the financial company in order to access the resources of the financial company.

3.2.2 financial technology (FinTech): ICT technologies used by a financial industry to improve financial services. FinTech includes the applications, processes, products, or technology models in the financial services industry, composed of one or more financial technologies that are provided as an end-to-end process via the Internet.

NOTE – This definition is based in part on the information found in [b-FinTech].

3.2.3 open application programming interface (API): A publicly available application programming interface (API) that provides developers with programmatic access to a proprietary software application or web service.

NOTE – This definition is based in part on the information found in [b-OAPI].

3.2.4 open platform: A software system which is based on open standards [b-ITU-T L.1370], such as published and fully documented external application programming interfaces (APIs) that allow using the software to function in other ways than the original programmer intended, without requiring modification of the source code. Using these interfaces, a third party could integrate with the platform to add functionality.

NOTE – This definition is based in part on the information found in [b-OPLTM].

3.2.5 platform: A hardware or software system that serves as a foundation or base for realizing a certain functionality.

4 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

API	Application Programming Interface
CAPTCHA	Completely Automated Public Turing test to tell Computers and Humans Apart
CEO	Chief Executive Officer
CISO	Chief Information Security Officer
CVE	Common Vulnerabilities and Exposures
DMZ	Demilitarized Zone
FinTech	Financial Technology
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
I/F	Interface
ICT	Information and Communication Technology
IT	Information Technology
PaaS	Platform as a Service
SEO	Search Engine Optimization
SMS	Short Message Service
SQL	Structured Query Language
URL	Uniform Resource Locator

5 Conventions

This Recommendation uses the following conventions:

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

In this Recommendation, the words "**shall**", "**shall not**" and "**should**" may sometimes appear, in which case they are to be interpreted, respectively, as "**is required to**", "**is prohibited**" and "**is recommended**". The appearance of such phrases or keywords in an appendix or in material explicitly marked as informative are to be interpreted as having no normative intent.

6 Overview

6.1 FinTech and security

Digitalization is transforming the financial industry into an ultra-competitive market. Traditional high-street financial companies are scrambling to implement digital transformation projects to become more agile to meet challenges like aggressive time-to-market, fierce competition for customers and shifting business models from product-centric to user-centric. Traditional financial companies have difficulties handling users' diverse needs and personalized services by themselves with their limited human and technology resources, thus there is a market vacuum for innovative FinTech services provided by FinTech companies. Start-up FinTech companies have to overcome obstacles to interconnect to various existing financial companies in order to provide comparable service capabilities with even smaller workforces and access to testbeds to validate their initial products based on real and sufficient data. Therefore, there is a need for an open platform to exchange and integrate data and services from existing financial companies to test FinTech products.

Developers from both financial and FinTech companies use open APIs to connect technologies (mostly apps, but also platforms and systems) to create digital financial innovations. APIs provide open connections between platforms. Failure to protect these connections will give hackers the opportunity to attack API services with either stolen or invalid credentials. Financial services must not overlook the security risk associated with the creation of financial apps in the open platform environment – in particular, API security.

6.2 Open platform

In computing, an open platform describes a software system that is based on open standards [b-ITU-T L.1370], such as published and fully documented external APIs that allow additional functions in ways other than the original programmer intended, without requiring modification of the source code. Using these interfaces, a third party could integrate with the platform to add functionality.

Using an open platform, a developer could add or alter features or functionalities that other platform vendors did not complete or did not envision, as its specifications are publicly available as open standards.

Generally, a platform is a hardware or software system that serves as a foundation or base for realizing a certain functionality on top of it. In case of software platforms, they offer a set of services that facilitate application development when used. According to the definition of 'open platform' in clause 3.2.4, a software platform could be called 'open' if it has one or more of the following characteristics, as highlighted in [b-ReAAL]:

– open API (ref. 3.2.2): An open API that is well documented and publicly available for use by application developers;

– open scope (extensibility): Extensibility refers to the capability of using the platform for purposes for which it was not originally planned;

– open source: Open source refers to the availability of software source code, in a given programming language, in which a license gives licensees freedom to study, change and distribute the software according to the license agreement. There are different open source licensing models with different degrees of freedom for change, usage, and distribution of the software in source code or binary forms. An open platform does not mean it must be open source, however most open platforms have APIs in open source implementations, resulting in open source platforms. Open source platforms do however always provide open APIs;

- open usage (adoptability): Adoptability refers to enabling others to use the open platform while bypassing specific business development negotiations. This does not necessarily mean that the usage has to be royalty-free;
- open adaptation (adaptability): Assuming that its specifications are publicly available, adaptability of an open platform refers to the possibility of 'changing' an existing functionality of the platform itself, as opposed to 'adding' a new functionality.

7 Reference architectures for FinTech services

7.1 Architecture of digital financial service provided by traditional financial companies

Figure 1 illustrates a traditional functional architecture for digital financial services. In this architecture every financial company has developed and operated financial services and provides its own apps, user can connect directly to financial companies and must use different apps of individual financial companies for same services.

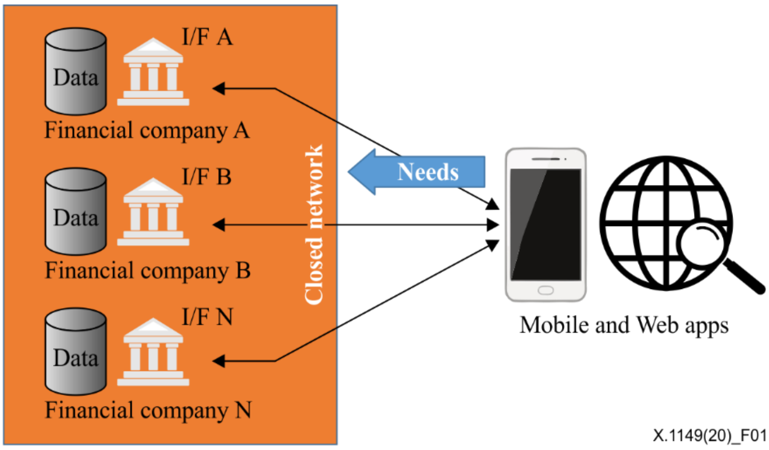
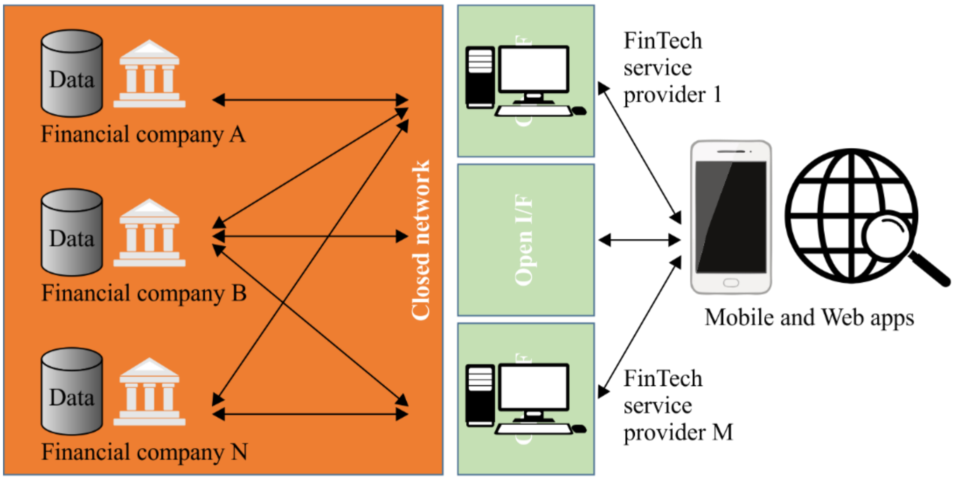


Figure 1 –Architecture of digital financial services provided by traditional financial companies

7.2 Architecture with FinTech service providers

Due to the rapid development of information and communication technology (ICT), it is difficult for financial companies to develop and operate on their own the financial services they provide to users. As FinTech companies have better ICT techniques and capabilities, financial companies want to use those FinTech companies to handle the difficulties and to obtain a competitive edge when compared to their competitors.

Figure 2 shows an architecture with emerged FinTech service providers, which are not in the architecture of traditional digital financial services. Financial companies hire different FinTech companies to develop different service apps and provide respective APIs for them. This architecture allows a financial company to provide users with respective apps for each of its services. This architecture causes inconvenience to users that a user has to connect to different apps of each financial company interface. This architecture has a characteristic of limited data sharing, which is the data of a financial company that is opened to a FinTech company so that the FinTech company has a dedicated authorization to access a user's data in this financial company, but the FinTech company has no authorization to access data of the same user in other financial companies. Whenever the FinTech company accesses the user's data beyond agreement with the origin financial company, it must obtain permission or authorization. Users interact differently and in a non-standardized manner with each of the FinTech companies. FinTech companies must secure each of its interfaces with each financial company and user.



NOTE – Interface (I/F)

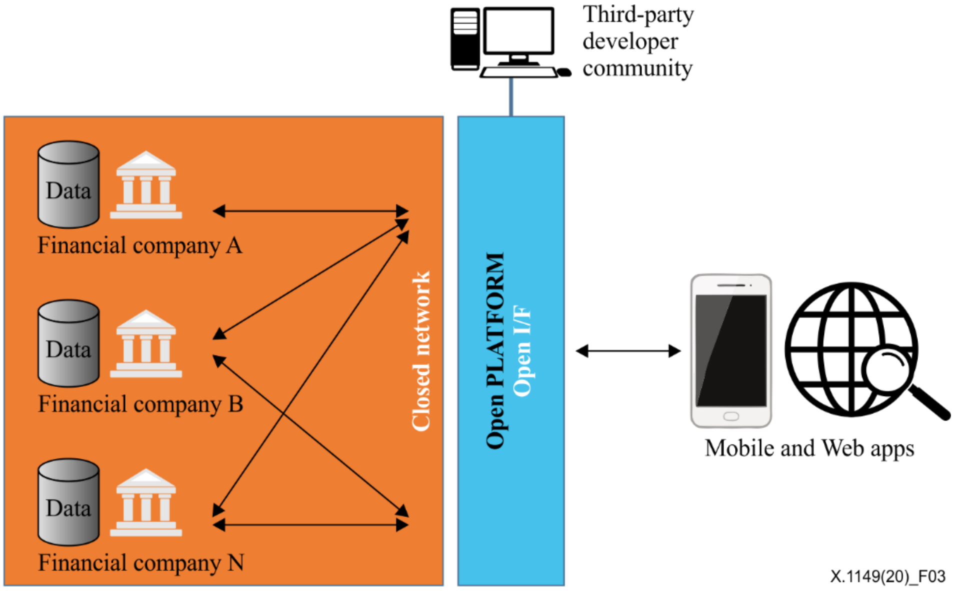
X.1149(20)_F02

Figure 2 –Architecture with FinTech service providers

7.3 Open platform for FinTech services

As discussed in the architecture with FinTech service providers (Figure 2), usability and efficiency to provide financial services are two problematic issues. On one hand, a FinTech company needs additional permissions across financial companies to integrate and create new data and services. On the other hand, the interfaces of the FinTech with different financial companies are different among themselves.

An open platform with open APIs in a standardized format provides an efficient interface for using open APIs, a common place to create and share value-added data and services using users' data located among financial companies, and serves as a testbed for new services and open APIs. Figure 3 shows an open platform architecture for FinTech services to streamline third-party data access and use secure open APIs to protect data and services for users.



X.1149(20)_F03

Figure 3 – Open platform architecture for FinTech services

8 Threats and vulnerabilities of open platform

An open platform based on open APIs provides new communication pathways through other financial companies on the open platform and thus enables new services to users. There are, therefore, threats and vulnerabilities that these communication pathways could be misused/abused, leading to leakage or falsification of data and unauthorized transactions. The anticipated threats and vulnerabilities associated with open platform (including open APIs) are listed in this clause.

8.1 Threats

A threat refers to a potential cause of an unwanted incident, which may result in harm to a system or a company [b-TBMC]. Threats to open API based applications in this clause are taken from [b-OAT2WA] and [b-JBA-API].

8.1.1 Account aggregation

Account aggregation is the compilation of credentials and information from multiple application accounts into another system. Account aggregation may be used to merge information of a single user from multiple applications, or alternatively to merge information of many users of a single application. This is commonly used for aggregating social media accounts, email accounts and financial accounts to obtain a consolidated user profile, provide integrated reporting and analysis, and to simplify usage and consumption by a user or their professional advisors. This may include making changes to account properties and interacting with the aggregated application's functionality.

8.1.2 Account creation

Bulk account creation, and sometimes profile population, is accomplished by using an application's account sign-up processes. These accounts may subsequently be misused for generating content spam, laundering cash and goods, spreading malware, affecting reputation and skewing search engine optimization (SEO), reviews and surveys.

8.1.3 Advertisement fraud

Advertisement fraud occurs when an advertisement is fraudulently displayed. It may be promulgated by the owners of web sites displaying ads, competitors, and vandals.

8.1.4 CAPTCHA defeat

The 'Completely Automated Public Turing test to tell Computers and Humans Apart' (CAPTCHA) challenge is used to distinguish human users from "bots". Automation is used in an attempt to analyse and determine answers to visual or aural CAPTCHA tests and related puzzles. Apart from conventional visual and aural CAPTCHA, puzzle-solving mini games or arithmetical exercises are sometimes used. Some of these may include context-specific challenges.

The process that determines answers may utilize tools to perform optical character recognition or matching against a prepared database of pre-generated images, or using other machine reading or human click farms.

8.1.5 Card cracking

Card cracking is a brute force attack against application payment card processes to identify missing values for a start date, expiry date or the card security code. It may also be referred to in other ways, such as including card validation number 2, card validation code, card verification value and card identification number.

8.1.6 Carding

Carding is the process of taking lists of full credit or debit card data and testing them against a merchant's payment processes to identify valid card details. The quality of stolen data is often unknown, and carding is used to identify good data of higher value. Payment cardholder data may have been stolen from another application, stolen from a different payment channel, or acquired from a criminal marketplace.

8.1.7 Cashing out

Cashing out refers to obtaining currency or high-value merchandise via an application using stolen, previously validated payment cards or other account login credentials. Cashing out sometimes may be undertaken in conjunction with product return fraud. For financial transactions, this is usually a transfer of funds to a mule's account. For payment cards, this activity may occur following carding of bulk stolen data, or card cracking, and the goods are dropped at a re-shipper's address. Refunding of payments via non-financial applications (e.g., tax refunds, claims payment) is also included in cashing out.

8.1.8 Credential cracking

Credential cracking involves brute force, dictionary (word list) and guessing attacks used against authentication processes of the application to identify valid account credentials. This may utilize common usernames or passwords or involve initial username evaluation.

8.1.9 Credential stuffing

Credential stuffing attacks use lists of authentication credentials stolen from elsewhere which are then tested against an application's authentication mechanisms to identify whether users have re-used the same login credentials. The stolen usernames (often email addresses) and password pairs could have been sourced directly from another application by the attacker, purchased in a criminal marketplace, or obtained from publicly available breach data dumps.

8.1.10 Denial of inventory

Denial of inventory attacks involve the selection and holding of items from a limited inventory or stock, but which then are never actually purchased, paid for, or confirmed. This attack prevents other users from legitimately buying, paying for, or confirming these items themselves. This attack differs from scalping (clause 8.1.15) in that the goods or services are never actually acquired by the attacker.

Denial of inventory is most commonly thought of as taking e-commerce items out of circulation by adding many of them to a cart/basket; the attacker never actually proceeds to checkout to buy them but contributes to a possible out-of-stock condition. A variation of this automated threat event is making reservations (e.g., booking hotel rooms, restaurant tables, holiday bookings, flight seats), or click-and-collect without payment. This exhaustion of inventory availability also occurs in other types of web applications such as in the assignment of non-goods like service allocations, product rations, availability slots, queue positions and budget apportionments.

8.1.11 Denial of service

Denial of service attacks resemble legitimate application traffic but lead to the exhaustion compute of resources such as file systems, memory, processes, threads and central processing units, and human or financial resources. These resources might be related to web, application or databases servers or other services supporting the application, such as third-party APIs, including third-party hosted content, or content delivery networks. In a denial of service attack, the application may be affected as a whole, or the attack may be against individual users causing account lockout.

8.1.12 Expediting

Expediting attacks use speed to violate explicit or implicit assumptions about the application's normal use to achieve unfair individual gain. This is often associated with deceit and loss to some other party.

8.1.13 Fingerprinting

Fingerprinting attacks send specific requests to an application, eliciting information in order to profile the application. This probing typically examines hypertext transfer protocol (HTTP) header names and values, session identifier names and formats, contents of error page messages, uniform resource locator (URL) path case sensitivity, URL path patterns, file extensions, and whether software-specific files and directories exist. Fingerprinting is often reliant on information leakage and this profiling may also reveal some network architecture/topology. Fingerprinting may be undertaken without any direct usage of the application, e.g., by querying a store of exposed application properties such as those held in a search engine's index.

8.1.14 Footprinting

Footprinting involves information gathering with the objective of learning as much as possible about the composition, configuration, and security mechanisms of the application. Unlike scraping (clause 8.1.16), footprinting is an enumeration of the application itself, rather than the data. It is used to identify all URL paths, parameters and values, and process sequences, e.g., to determine entry points, also collectively called attack surfaces. As the application is explored, additional paths will be identified which in turn need to be examined.

8.1.15 Scalping

Although scalping may include monitoring awaiting availability of goods or services and then taking rapid action to beat normal users to obtain these, it is not a "last minute" action like sniping (clause 8.1.18), nor is it simply related to automation on behalf of the user such as in expediting (clause 8.1.12). Rather, scalping includes the additional concept of limited availability of sought-after goods or services and is most well known in the ticketing business where tickets acquired are then resold later at a profit by the scalpers/touts. This can also lead to a type of user denial of service, since the goods or services become rapidly unavailable.

8.1.16 Scraping

Scarping involves collecting accessible data or processed output from an application. Some scraping may use fake or compromised accounts, or the information may be accessible without authentication. The scraper may attempt to read all accessible paths and parameter values for web pages and APIs, collecting the responses and extracting data from them. Scraping may occur in real time or may be more periodic in nature. Some scraping may be used to gain insight into how it is constructed and operates, or perhaps for cryptanalysis, reverse engineering, or session analysis.

8.1.17 Skewing

Skewing using automated repeated clicking or requesting or submitting content, affecting application-based metrics such as counts and measures of frequency or rate. These metrics or measurements may be visible to users (e.g., betting odds, likes, market/dynamic pricing, visitor count, poll results, reviews) or hidden (e.g., application usage statistics, business performance indicators). Metrics may affect individuals as well as the application owner, e.g., user reputation, influence others, gain fame, or undermine someone else's reputation.

8.1.18 Sniping

The defining characteristic of sniping is an action undertaken at the last opportunity to achieve a particular objective, leaving insufficient time for another user to bid/offer. Sniping can also involve the automated exploitation of system latencies in the form of timing attacks. Careful timing and prompt action are both necessary parts. Auction sniping is the most well-known case, but the same threat event can be used in other types of applications. Sniping normally leads to some dis-benefit for other users, and sometimes may be considered a form of denial of service.

8.1.19 Spamming

Malicious content used in spamming can include malware, inline frame (IFRAME) distribution, photographs and videos, advertisements, referrer spam and tracking/surveillance code. This content may be less overtly malicious than other attacks, but nonetheless is still used as an attempt to cause SEO skewing or to dilute/hide other posts.

The mass abuse of broken form-to-email and form-to-short message service (SMS) that serves to send messages to unintended recipients is not included in this threat, or any other in this ontology, since those are considered to be the exploitation of implementation flaws alone.

8.1.20 Token cracking

Token cracking involves the identification of valid token codes, thus providing some form of user benefit within the application. The benefit may be a cash alternative, a non-cash credit, a discount, or an opportunity such as access to a limited offer.

8.1.21 Vulnerability scanning

Vulnerability scanning involves the systematic enumeration and examination of identifiable, guessable, and unknown content locations, paths, file names, parameters, in order to find weaknesses and points where a security vulnerability might exist. It includes both malicious scanning and friendly scanning by an authorized vulnerability scanning engine. It differs from scraping (clause 8.1.16) in that its aim is to identify potential vulnerabilities.

8.1.22 Unauthorized access

This threat occurs when an illegal entity gains the privilege to access the open platform by masquerading as an authorized entity. It may launch further attacks through malicious code injection, system/application parameters modification, etc.

8.2 Vulnerabilities

A vulnerability is a known or unknown flaw or weakness in a system that could result in the loss of the system's integrity, availability, or confidentiality. An internal vulnerability could be a lack of security awareness training or no documentation for a critical process. This clause details a list of vulnerabilities related to open source code. There may be other vulnerabilities on ICT services and the common vulnerabilities and exposures (CVE) list [b- ITU-T X.1520] would be useful for system and service developers.

See [b-OSSV] for a list of the top open source security vulnerabilities.

8.2.1 Glibc vulnerability

The glibc vulnerability is an open source security vulnerability affected by all servers and web frameworks such as Python, PHP hypertext pre-processor (PHP), Rails, as well as API web services that use the GNU C library. This vulnerability enables hackers to compromise apps via a man-in-the-middle attack, with the possibility of taking control of a user's system that has accessed a hacker-controlled domain name system host.

8.2.2 Quadrooter

The quadrooter vulnerability requires the user to first install a malicious app. Unlike other malware, however, the installed app requires no special permissions; this means the user's suspicions typically are not raised.

After the malware's installation, the attacker can gain root access to the device by exploiting any of the four Quadrooter vulnerabilities. This puts all system contents and controls (including sensitive data, microphone, , and system changes) at risk of exploitation.

8.2.3 Drown attack

One point of the hypertext transfer protocol secure (HTTPS) web protocol is to make Internet transmissions more secure. However, the drown vulnerability gives users a false sense of security even when using HTTPS.

The drown vulnerability exploits a flaw in the secure sockets layer version 2 (SSL v2) protocol, allowing attackers to break HTTPS' encryption, and consequently steal sensitive communications such as usernames, passwords, credit card numbers, sensitive documents, etc.

8.2.4 Zero-day attack vulnerability

This open source security vulnerability impacted all systems running the Linux versions 3.8 and higher operating systems, as well as 66% of all Android devices. Once the bug is exploited, the attacker can gain root access to the user's operating system.

8.2.5 Database vulnerability

By injecting malicious settings into structured query language (SQL) configuration files, this vulnerability allows attackers to gain full access to the server on which the affected SQL was running. This means hackers can view, change, and even erase any entries they wish.

Not only can this open source security vulnerability be exploited both locally and remotely, but the attacker has the choice of using either valid access credentials (e.g., passwords), or an SQL injection attack as vectors.

As the SQL injection attack is one of the most common issues in web applications, the last vector makes this vulnerability particularly challenging.

8.2.6 Operating system kernel vulnerability

The operating system kernel vulnerability exploits a weakness in the transmission control protocol of all relevant systems. Once exploited, the vulnerability enables an attacker to degrade the privacy of anonymous networks (e.g., Tor browser), track users' online activities, hijack a conversation between hosts and even forcibly terminate a conversation.

8.2.7 OpenJDK vulnerability

The Open Java Development Kit (OpenJDK) security vulnerability can be remotely exploited without any need for authentication details, such as passwords or usernames. This means a single visit to a malicious web page can allow an attacker to degrade the availability, integrity, and confidentiality of a user's system.

9 Open API usage procedure for FinTech services

The open API usage structure may vary depending on the type of API, open API system design, and service characteristics. Figure 4 shows an example of open API usage structure and a procedure of using open APIs for FinTech services in a simplified representation of communication flow.

– Registration process (A)

  A1) A user requests a FinTech company to register a FinTech service (e.g., account balance inquiry, transaction history inquiry, etc.);

  A2) FinTech company requests the operator (financial company) to grant an open API access key related to the user;

  A3) The operator performs user authentication prior to granting the open API access key;

  A4) The operator grants the open API access key to the user.

– Usage process (B)

  B1) The registered user requests an open API-enabled FinTech service (e.g., account balance inquiry);

  B2) FinTech company submits the open API access key granted at the registration stage to the operator (financial company) and requests the open API service;

B3) The operator provides the corresponding open API service result to FinTech company;

B4) FinTech company provides the service to the user based on the information received in step B3.

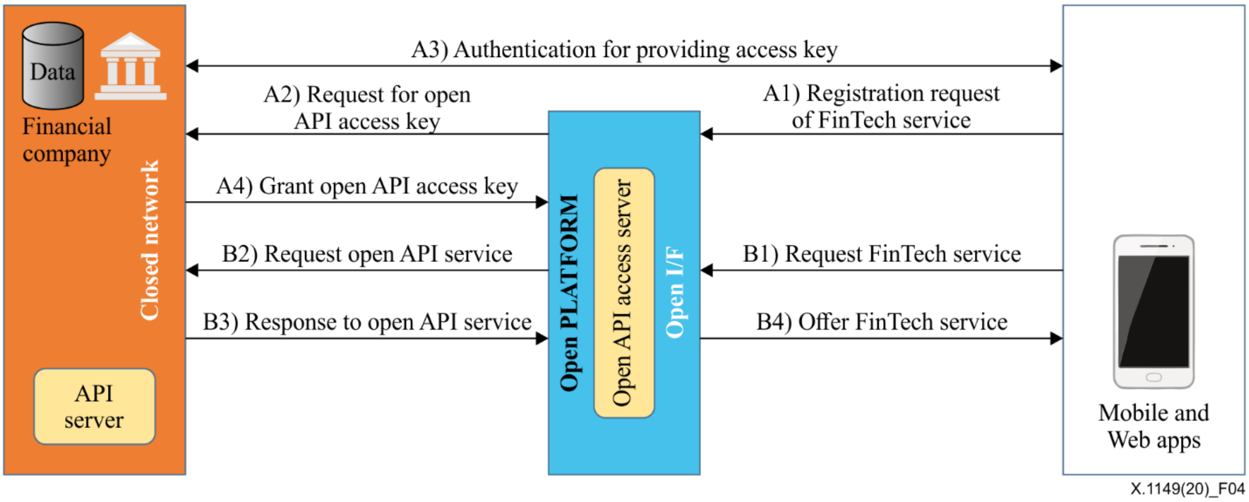

Data
Financial company
Closed network
API server
A3) Authentication for providing access key
A2) Request for open API access key
A1) Registration request of FinTech service
A4) Grant open API access key
B2) Request open API service
B1) Request FinTech service
B3) Response to open API service
B4) Offer FinTech service
Open PLATFORM
Open API access server
Open I/F
Mobile and Web apps
X.1149(20)_F04

Figure 4 – A procedure of FinTech services based on open platform and open API

10 Security requirements for open platform of FinTech services

This clause specifies security requirements to financial companies and FinTech companies for FinTech services.

10.1 Security requirements to financial companies

This clause specifies security requirements to services (e.g., data transmission, update, and deletion, etc., provided to FinTech companies) and facilities (e.g., server, network devices, computing room, power, etc., to operate services) of financial companies.

10.1.1 Authentication to use open APIs

This clause specifies requirements for authentication of the entity using FinTech services.

10.1.1.1 Appropriateness of entity authentication

– It is recommended to apply appropriate user authentication in consideration of the permission of the access key to prevent authentication by stealing the identity of another person.

10.1.1.2 Appropriateness of FinTech company authentication

– It is recommended to apply a combination of basic authentication mechanisms and additional mechanisms for secure authentication in order to properly respond to requests for issuance of access keys by unauthorized user and spoofing attack by users.

10.1.1.3 Appropriateness of high risk electronic financial transaction authentication

– It is recommended that a financial company strengthens authentication such as additional user authentication to verify the access key when FinTech company requests access to high-risk electronic financial transaction APIs (e.g., high-value transfer).

10.1.1.4 Appropriateness of authorization when handling API access request

– It is recommended to restrict the permission given by the access key by verifying the authorization of the access key.

10.1.1.5 API server authentication

- It is recommended to check whether the API server in the financial company that the user accesses for authentication is a legitimate system and can be authenticated.
- It is recommended to verify the API server's certificate by checking the server address, validity period, issuer, and owner in the server certificate when performing open API system authentication in the case of using a server certificate.

10.1.1.6 User authentication bypass prevention

- It is recommended to check whether the access key can be granted to the FinTech company by bypassing the authentication step for user verification.
- It is recommended to check whether authentication is implemented to prevent bypassing without going through the authentication phase normally.

10.1.1.7 Mitigating risks of leakage of access keys

- It is recommended to mitigate the risk of leaking access keys or code values used to issue access keys (e.g., authorization codes).
- It is recommended that the financial company securely encrypts important information such as issued access keys (including renewal keys) and store them in a secure location.

10.1.1.8 Avoid guessing FinTech company's authentication information

- It is recommended to take measures to prevent unauthorized user using guessed or falsified authentication information of FinTech companies to attack the API server.
- It is recommended to counter attacks in which an attacker tries to authenticate repeatedly by guessing authentication information (access key, authentication key, authentication code, etc.) of a legitimate FinTech company.

10.1.1.9 Authentication and transaction record management

- It is recommended that a financial company keeps records of authentications and transactions and protect database or logs where the transaction records through authentication and open API are stored.

10.1.1.10 Authentication key management

- It is recommended to have a lifecycle security management plan regarding issuance, registration, distribution, and disposal of authentication keys.

10.1.2 Confidentiality and integrity of transaction information

This clause specifies requirements for confidentiality, integrity, cryptographic algorithm adequacy and key management for transaction information management.

10.1.2.1 Confidentiality of transaction information

- It is recommended to secure the confidentiality of information by encrypting important information such as electronic financial transaction details stored and transmitted between the FinTech company and the financial company.

10.1.2.2 Integrity of transaction information

- It is recommended to verify the integrity of main protection targets such as electronic financial transaction details (including transaction text) against forgery and alteration.

10.1.2.3 Use a secure cryptographic algorithm

- It is recommended that cryptographic operations should use proven cryptographic algorithms with sufficient key length to support confidentiality and integrity.

10.1.2.4 Secure key management

- It is recommended that cryptographic keys used for encryption should be securely managed throughout the cryptographic key's lifecycle (generation, distribution, access, destruction, etc.).

10.1.2.5 Secure cryptographic program management

- It is recommended to prevent illegal distribution and illegal use of cryptographic program by designating a person in charge of it, controlling its usage and using a separate storage for its source code.

10.1.3 Information processing system protection measures

This clause specifies requirements to appoint a manager, execute a patch, check vulnerabilities, respond to infringement incidents, etc., for the security of information processing systems (API server).

10.1.3.1 Designation and operation of managers and administrators

- It is recommended that the information processing system's administrator and manager are designated and operated.

10.1.3.2 Perform critical patches

- It is required that immediate patching be taken for urgent and important patches such as the operating system and system utilities of the information processing system.
- It is required to support a one-way dynamic application layer forward error correction mechanism to ensure the reliability of data transmission for security patches to low-level security facilities.

10.1.3.3 Additional authentication for operating system account

- It is recommended to perform additional authentication process to log into the operating system account of the information processing system, in addition to account username and password.

10.1.3.4 Key terminal protection

- It is recommended to designate the terminal connecting to the information processing system as a key terminal and apply appropriate protective measures.

10.1.3.5 Prevention of server hacking

- It is recommended that the API server takes measures to prevent exposure to hacking (electronic infringement) attacks, such as stealing storage data, forgery, and denial of service attacks.

10.1.3.6 Security verification and vulnerability check

- It is recommended to execute self-security verification such as open API system establishment, vulnerability check before adding new API, static/dynamic test, and to check whether related system performs periodic vulnerability check using mock hacking and inspection tool.

10.1.3.7 Open server installation and access control

- Open API servers, such as open web servers, are recommended to be installed in a separate communication network, a so-called Demilitarized Zone (DMZ), between the internal and external communication networks and to be checked for access control.

10.1.3.8 Response to FinTech company's infringement incidents

– It is recommended for a financial company to check whether the FinTech company has an infringement incident response system and mechanism to share the infringement incident information.

10.1.4 User terminal protection

This clause specifies requirements for input information protection, terminal protection, and certificate management of users.

10.1.4.1 Input information protection

– It is recommended that a financial company protects the important information that a user enters from being exposed and leaked.

10.1.4.2 FinTech company's user terminal protection

– It is recommended that a financial company verifies that the FinTech company has user terminal protection measures when implementing service applications using open API.

10.1.4.3FinTech company's certificate management

– It is recommended that a financial company verifies that the FinTech company keep the certificate securely for signing electronic financial transactions.

10.1.5 Information leakage prevention

This clause specifies the requirements for account management, logging, and leakage prevention.

10.1.5.1 Access account management

– With regard to user account and password, it is recommended to manage them thoroughly through the lifecycle of the user account (registration, change, retirement, etc.).

10.1.5.2 Information system log preservation and analysis

– It is recommended to preserve logs such as operation records of information processing systems for more than one year and check whether regular log analysis is performed.

10.1.5.3 Information leakage prevention

– It is recommended for a financial company to guide FinTech company to prepare and manage leakage prevention measures for sensitive information when storing and processing user-related information (login name, password, etc., and transaction details acquired through API).

10.1.6 Countermeasures against abnormal financial transactions

This clause specifies the requirements for anomaly detection, response, and notification.

10.1.6.1 Monitoring and detection of abnormal financial transactions

– It is recommended that an appropriate system to judge abnormal financial transactions is in place and verify that abnormal financial transactions are detected.

10.1.6.2 Respond when anomalous financial transactions are detected

– It is recommended to verify that appropriate countermeasures are in place when abnormal financial transaction attempts are detected.

10.1.6.3 Notification of important transactions to user

– It is recommended that users are properly notified of important transaction information to prevent electronic financial accidents.

10.1.7 System availability and emergency measures

This clause specifies the requirements for continuity, redundancy and backup of data and equipment.

10.1.7.1 Establishment of work continuity plan

- In order to ensure that open API operations are not interrupted in the event of an emergency such as a failure, disaster, strike or terrorism, it is recommended to establish and comply with measures to secure business continuity, and check the effectiveness and adequacy of these security measures at least once a year to ensure that they are kept and managed up to date.

10.1.7.2 Redundancy of main computer equipment

- It is recommended that redundancy or spare devices are guaranteed for the main computer equipment of an open API.

10.1.7.3 Backup and dispersion management

- It is recommended that the operating system, configuration, and main data of the information processing system are regularly backed up and dispersed in the remote and safe area according to the importance, and the backup data is recorded and managed for more than one year.

10.1.8 Physical access control to FinTech company's facilities

This clause specifies the requirements for physical access control to FinTech facilities.

10.1.8.1 Physical access control to FinTech company's facilities

- It is recommended that a financial company provides guidance to a FinTech company for the installation of facilities such as open API access server and access key storage in a safe protected area and to control unauthorized access to them.

10.2 Security requirements to FinTech companies

This clause specifies security requirements to services and facilities (server, network devices, computing room, and power) of FinTech companies.

10.2.1 Information security policy and organization

This clause specifies requirements to establish information security policy and organization.

10.2.1.1 Appointment of chief information security officer and his supporting team

- The chief executive officer shall appoint a chief information security officer (CISO) to manage the organization's information security.
- A supporting team with expertise to support the CISO and to systematically implement the organization's information security activities should be established.
- The CISO is required to check compliance with the information security checklist at appropriate intervals (e.g., at least once per quarter).
- The inspection result should be approved by the CISO and reported to the chief executive officer (CEO).

10.2.1.2 Establishment and announcement of information security policy

- In order to provide a basis for all information security activities carried out by the organization, a top-level information security policy including the following items should be established and approved by the CEO:
  - commitment of the organization's management team to protect information;
  - the organization's information security objectives, scope, responsibilities;
  - the rationale for information security activities performed by the organization;

  - statutes and related provisions that the organization must comply with, i.e., information security regulations.

- A policy implementation document containing details of security management subjects, method, and procedure for implementing the information security policy should be established and approved by the CISO:
  - it is recommended to include sanctions in case of violation of information security regulations so as to raise the awareness of information protection;
  - information security policy, regulation and other policy implementation documents are recommended to be announced and the latest documents should be provided to employees.

10.2.2 Outsider management

This clause specifies the requirements for managing consignment company.

10.2.2.1 Selection and management of consigner

- When entrusting information processing tasks to outsiders or allowing access to information assets, or when using external services such as cloud services for business purposes, it is recommended to identify security requirements and specify relevant contents in contracts and agreements with these consigners.
- Regular checks should be made for compliance with the security requirements specified in the contract.
- Risks associated with using cloud service and establish and manage countermeasures must be identified.
- It is recommended to mitigate security risks by using cloud services that have been certified for security.

10.2.3 Information asset management

This clause specifies requirements to appoint manager, identify and rate information assets.

10.2.3.1 Identification and rating of information assets

- It is recommended that all information assets subject to information security management related to services using an open API should be identified, and the identified information assets should be organized into a list (system or document) and managed systematically.
- Considering the confidentiality, integrity, availability and legal requirements of identified information assets, it is recommended to prepare own criteria of the importance based on the impact on financial company, and evaluate and assign a security rating according to the criteria.

10.2.3.2 Designation of a responsible person for an information asset

- It is required to designate a person in charge of an identified information asset that is responsible for the introduction, modification, disposal, or transfer of the information asset.
- It is recommended to mark the information asset with the identification of the person in charge.

10.2.4 Information security education

This clause specifies the requirements to plan and execute education plans of information security.

10.2.4.1 Establishment and execution of information security education plan

- The CISO (or CEO) is required to establish and implement an annual information security education plan. including the timing, period, target, contents, and methods of information security education.

10.2.4.2 Information security education for practitioners

- Executives and employees within information technology (IT) and information security organizations should complete the specialized education necessary to enhance their professional skills on information security.
- Open API-enabled applications are open to public and are easily exposed to attacks, so IT developers are strongly recommended to complete application secure coding education.

10.2.5 Human resource security

This clause specifies the requirements for confidentiality of job information, separation of duties and retirement management.

10.2.5.1 Confidentiality agreement

- Employees and outsiders who are granted access to open API-related information assets shall sign a pledge of information protection, including responsibility for and compliance with information protection.
- Upon the time of retirement of employees and termination of outsider's work, they shall be reminded of the signed confidentiality agreement in order to prevent any leakage of important information of the organization that they has learned from their work and also that they have a legal responsibility in case of leakage.
- It is recommended that information protection and confidentiality agreements be kept securely and managed in order to be easily located, if necessary, because they can be used as evidence of legal liability in the event of a legal dispute.

10.2.5.2 Separation of duties

- In order to prevent misuse of authority of duties, key job separation criteria related to information protection should be established and implemented, and roles and responsibilities for each job should be clearly defined.
- Development, operation, and information security management tasks are recommended to be separated to reduce unauthorized modification, access, insertion, and deletion.
- When task separation is impossible due to organizational size or other reasons, complementary controls such as cross-reviewing between the employees of high-risk tasks such as changes in critical information assets, monitoring and approving changes by high-level managers, and ensuring accountability of the employees should be implemented.

10.2.5.3 Retirement and job change management

- Upon retirement of internal and external employees and job changes, procedures for adjusting or retrieving access permission to open API-related information assets shall be established and operated.
- Personnel department is required to promptly share personnel changes (such as change of department and job, leave of absence, retirement, etc.) to the information protection department and the information processing system operation department, so that measures such as change of access authority to open API related information assets can be taken swiftly.

10.2.6 Risk management

This clause specifies requirements for vulnerability inspection.

10.2.6.1 Establishing vulnerability inspection policy and carrying out inspection

- To prevent open API-related information processing systems from known vulnerabilities, a vulnerability check policy should be established, and a vulnerability check (at least once a year) should be carried out on a regular basis to manage risks.
- Any vulnerabilities found as a result of the vulnerability check shall be removed or corresponding remedial action shall be taken.

10.2.7 Infringement incident response

This clause specifies the requirements for infringement incident response and record management.

10.2.7.1 Procedures for responding to infringement and education

- It is required to establish procedures for responding to encroachment incidents, including procedures for classification and reporting of importance by type of encroachment and recovery, emergency contact system, etc.
- Relevant education shall be provided to the employees so that they can understand the procedures for responding to encroachment.
- In order to assess the employees' knowledge of the response procedures and their adequacy and effectiveness in responding to the encroachment accident, it is recommended that the employees periodically conduct simulated education in response to the encroachment accident in accordance with the scenario and supplement the response procedures.

10.2.7.2 Preservation and monitoring of logs related to incident response

- It is required to preserve the required logs for a certain period of time and review them periodically for analysis of encroachment accidents.
- For high importance or risk task, it is recommended to carry out real-time incident detection and response.

10.2.8 Fault response

This clause specifies requirements for fault response.

10.2.8.1 Establishing backup policy and establish recovery procedures

- Recovery procedures for key information should be established to enable recovery against failures caused by IT hazards (hacking, system failure, fire, etc.).
- A backup policy should be established considering legal requirements for the data to be recovered, and the backup records and backup data should be maintained for a certain period of time.
- For critical tasks, it is recommended to set a recovery target time to ensure continuity of work, and to establish and operate backup and recovery procedures to the extent possible not only for information but also for system and service recovery.

10.2.9 User protection

This clause specifies requirements for personal information processing and user grievances.

10.2.9.1 Protection of users related to personal information processing

- When a user is registered or personal information is entered in an open API service, a FinTech company is required to prepare a privacy policy and disclose it in a location that can be easily identified by the user (e.g., on the main screen of the website).
- The required and optional options are recommended to be appropriately distinguished and agreed to so that the personal information to be collected is minimal for the purpose.

10.2.9.2 Notification to accessing personal and credit information and trading orders

- If it is necessary to access personal and credit information and electronic financial transaction instructions for the user's financial company account through the open API, the FinTech company is required to fully explain the contents and obtain the user's consent.
- In addition to the time of agreement, the FinTech company is required to provide a means for users to easily check relevant information.

10.2.9.3 Establishing and disclosing policy for handling user grievance

- A policy shall be prepared to respond to various inquiries from users, such as counselling, inquiry, complaints, and reporting accidents, and shall be made public on the website for easy identification by the users.
- In particular, matters concerning the handling of personal information-related user grievances are recommended to reflect relevant legal requirements.

10.2.9.4 Notification of user security precautions

- User-side precautions should be guided so that users can understand and pay attention to security risks when using the service and should be reflected in service development to reduce such risks.

10.2.10 Physical security

This clause specifies requirements to protect area and office, access control, and asset import and export.

10.2.10.1 Designation of protected areas and access control

- In order to protect main facilities and systems from non-authorised physical access and various physical and environmental disasters, physical protection zones shall be designated and implemented.

  Protected areas are categorized into three areas as follows:

  - Reception area: an area accessible to outsiders without a pass;
  - Restricted area: a place where separate access control devices and monitoring systems are installed to prevent unauthorized persons from accessing, and where a pass, such as an employee card, is required (for example, office);
  - Controlled area: a place that includes all the control items of the restricted area and requires additional procedures for access if the access qualification is maintained to the minimum number of people (e.g., computer, communication equipment, control, power rooms).
- Locations where critical open-API-related systems operate shall be designated as controlled areas with tight access control.
- Entry and exit records of restricted and controlled areas shall be preserved for a certain period of time (more than one month) and reviewed periodically.

10.2.10.2 Import and export management of protected areas

- Control procedures for portable devices (laptop, tablet PC, etc.) shall be established and controlled in each protected area to prevent security accidents such as leakage of important information through the transfer of unapproved devices and infection of malicious codes in the internal network.
- In principle, portable devices, and storage media in the controlled area where the critical system for open API is located is prohibited.
- Transport of terminals, portable devices, and storage media to restricted areas (e.g., offices) shall be controlled.

10.2.10.3 Establish and execute an office environmental security policy

- Terminal protection measures shall be established and executed to prevent unauthorized access.
- For the protection of important information, it is prohibited to leave important documents or storage media on the desk for more than a certain period of time or at the time of work.
- A drawer or cabinet in which important documents are stored shall be locked.

10.2.11 Security of system development

This clause specifies requirements to secure system design, coding, and testing along with system lifecycle.

10.2.11.1 Identify and reflect security requirements in system design stage

- Security requirements should be derived, and countermeasures reflected in the design stage, taking into account relevant laws for information protection, the latest security vulnerabilities, and the basic elements for information protection in the event of new developments and changes.
- Open API service shall be developed safely by familiarizing developers with the FinTech company's terms and conditions and API development guides.
- For electronic financial transaction services, measures to prevent forgery of financial information, exposure of important information (secret numbers, unique identification numbers, etc.) and bypass of authentication should be reflected in the design.

10.2.11.2 Secure coding and security vulnerability check and correction

- The information processing system shall be implemented according to secure coding methods, and the scenarios and checklists shall be prepared and tested to confirm whether the security requirements derived from the system analysis and design process have been applied to the information processing system.
- Upon completion of coding, the source code shall be verified in accordance with secure coding methods, and technical security vulnerability checks (or mock-hacking) in the same environment as the operating environment shall be performed. The source code shall be corrected immediately upon detection of vulnerabilities.

10.2.11.3 Restrictions on use of user personal and credit information in testing

- Test data is recommended not to use operational data to prevent the leakage of sensitive information, including personal and credit information, during development and testing.

10.2.11.4 Access and change control for source programs and computer ledgers

- Change control procedures should be established and implemented so that only authorized users can access the source program and electronic financial ledgers.
- Details of access and changes to the electronic financial ledgers should be recorded so that responsibility traceability can be secured by preserving them for a certain period of time.

10.2.12 Password control

This clause specifies requirements to protect sensitive information related to open APIs.

10.2.12.1 Establish and execute a sensitive information encryption policy

- Policies related to encryption should be established and implemented to protect critical information related to open API. Policies should reflect legal requirements related to encryption, such as application of encryption when storing and transmitting sensitive information.

- Cryptographic keys used for encryption should be generated and used according to their intended use, managed safely to avoid exposure, not stored in plaintext in personal information handling system, or not used by hard-coding inside the program of the system.

10.2.13 Access control

This clause specifies requirements to manage access control to open API-related information processing systems.

10.2.13.1 Management of critical information asset accounts and access rights

- Access right to open API-related information processing system should be minimized under the control of the responsible person and should be accessed in a secure manner.
- Connection time to main information processing system should be restricted.
  - if there is no use for a certain period of time after a user is connected to a server, the connection should be terminated (set session timeout time, etc.) to prevent unauthorized access.
- When accessing server's operating system account in an open API related information processing system, it is highly recommended to apply additional authentication in addition to password.
- Access control should be limited to its designated manager only for program that manage users' personal and credit information or electronic financial transactions.

10.2.13.2 Key terminal assignment and access control

- Key terminals related to open API based services should be designated and managed, and enhanced protection measures should be applied.

10.2.14 System security

This clause specifies requirements to manage malware infection, information leakage, remote access, open web server protection and periodic security patch on open API-related information processing system.

10.2.14.1 Prevention of malware infection and information leakage of critical systems

- In order to prevent the malicious code infection of an open API related information processing system, protection measures such as prevention, detection, and countermeasure of malicious code should be established and implemented.
- In order to prevent the malicious code import and information leakage through the Internet, access to the Internet and groupware should be controlled in the information processing system.
- Antivirus program should be installed on a critical server, updated regularly, and configured to allow real-time test.

10.2.14.2 Remote management control through Internet

- In principle, it is recommended to control remote management through an external network such as Internet for open API-related information processing system.

10.2.14.3 Removal of functions, programs, and ports other than critical system purpose

- Only minimal service port and function should be available to open API related computer system and important terminals, and functions and programs other than business purpose should be removed.

10.2.14.4 Independent operation of critical server and application of information protection system

- The server providing open API access should be operated as a separate server.
- When using cloud or server virtualization, it is recommended to use independent virtual machine without using public web server or database server as a public virtual machine, and it is recommended to have a separate protection against threats to server virtualization.
- It is recommended to operate an appropriate information security system to protect critical servers, set up a warning function to notify the information security system of abnormal symptoms to respective security administrator as soon as possible, and periodically (more than once a month) to check the normal operation of security functions.

10.2.14.5 Public web server protection measures

- When operating a public web server, a separate protection measure should be prepared and applied.
- Storage and management of important information such as personal information and credit information should be prohibited on the public web server.

10.2.14.6 Establishment and implementation of security patch application guidelines

- It is recommended to establish and implement operating system and software patch management policies and procedures according to asset importance or characteristics.
- Patch should be implemented by collecting and reviewing threat information from security experts and security patch information from the system manufacturer.
- If an emergency patch is needed, action should be taken without delay to minimize business impact.
- It is recommended to restrict patch through real-time access from Internet to open API related information processing system and key terminal.
- It is recommended to integrate and minimize functions of file distribution in internal network, such as utilizing patch management system within an organization and take sufficient protection measures such as conducting integrity verification and access control prior to patch distribution.
- It is recommended to manage system version, software version, and final patch version installed in major servers, network equipment, information protection systems, etc., so that they can be reviewed smoothly when collecting patch information.
- It is recommended to support a one-way dynamic application layer forward error correction mechanism to ensure reliability of data transmission for security patches to low-level security facilities.

10.2.15 Network security

This clause specifies requirements to manage communications with demilitarized zone (DMZ), wireless networks and external organization.

10.2.15.1 DMZ segment configuration

- An intrusion prevention system should be used to create a DMZ between external network and internal network, and to protect the internal network.

10.2.15.2 Internal network private IP utilization and main system deployment

- To control unauthorized access to the network, management procedure for network access control lists, network identifiers (IPs), among others should be established, and internal and external networks should be separated according to the importance of services, user groups and information assets.

- It is recommended to use a private IP address system in the internal network, ensure that the internal IP address system is not externally leaked, and apply network address translation function at location of connection with external network.
- Database server, which stores important information related to open API, is recommended not to be located in DMZ that provides services to the outside world, but in the internal network area protected by information protection systems such as intrusion prevention systems.

10.2.15.3 Minimize use of wireless networks and establish security measures for applications

- In the establishment of a wireless network environment that can be connected to an organization's internal network, internal approval procedures should be developed to prevent unauthorized wireless network equipment from operating, and the following protective measures should be applied by conducting a prior security review:
  - Necessary accreditation procedures should be in place to prevent outsiders from accessing internal network over the wireless network and to allow only executives and employees to use the wireless network.
  - Internal network penetration and internal information leakage through wireless networks should be prevented by separating internal network from the wireless network.

10.2.15.4 Secure communication with external organizations

- Secure communication shall be applied and connected to physically separated external agencies (including open API operators) to protect confidentiality and integrity of data transmitted during communication.

Appendix I

Use cases of open application programming interface

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

An application programming interface (API) is a smart, programmable doorway for developers to access data from a digital organization. Things like using a social account to authenticate on a website, having weather forecasts on a phone, being able to access maps from a separate application, or triggering Internet of things devices - all rely on APIs to function.

Since APIs are base level, directly tied into the core architecture, and typically work over hypertext transfer protocol secure (HTTP), any device with Internet connectivity can use them, regardless of operating system or programming language, making them an extremely alluring way to construct a platform. Public APIs are often free or structured as a freemium plan that developers can purchase to use at a certain limit of calls per month. It is estimated that 15,000 + public API programs currently exist. Spanning hundreds of industries, behind these APIs sit a myriad of different providers; individual developers, start-ups and established businesses that have decided that they need a public API program to better serve their audience. Not all APIs are open sourced to the public for anyone to use. Many APIs are used strictly internally such as private API which powers video streaming on the internal network.

I.1 Useful points about APIs

When planning a new API or managing APIs, reviewing the following can increase the utilization of APIs:

- **APIs are strategic for most companies**: APIs can enable a business to open up new revenue streams, extend their branding, unite how their service is exposed, expand their end consumer base, and create more productive internal workflows;
- **APIs enable a business to become more "niche"**: App developers can leverage existing APIs instead of building out their own solutions; meaning they can specialize in what they do best;
- **There are good and bad ways of providing an API**: Doing so takes tact. The initial strategy, as well as the quality of design, security, documentation, and support can make or break a developer program;
- **APIs affect all industries**: Thousands of companies with a significant online presence have created developer programs to open some amount of data for third-party consumption, regardless of industry sector;
- **APIs are now an opportunity for entrepreneurs**: Micro-services API-first companies focus their attention on marketing to a new audience of developers, designers, entrepreneurs, architects, etc.;
- **An economy has formed**: APIs are a new type of product, and a complex offering at that, an economy of businesses and related tooling has arisen to help API providers excel.

I.2 Use cases (business models) using open APIs

Constructing a business that relies heavily on APIs is a model that has brought success to many companies. Sometimes success comes in the form of diversified revenue streams, other times it means increasing user value within the existing ecosystem. API-driven businesses also allow for more third-party innovation, which in turn boosts the success of the business. Several successful models are introduced in the clauses I.2.1 to I.2.6.

I.2.1 Platform as a service

APIs as platform as a service (PaaS) are a cloud communications platform as a service. This means developers can use their APIs to programmatically make and receive text messages and phone calls.

APIs as PaaS includes a simple API for making phone calls, an API for text messages, and APIs for SMS short codes, voice over Internet, 2-factor authentication, and more – all via API. Throughout the development of their business, the model has demonstrated the importance of starting small, focusing, and iterating based on user needs.

The merit from this model is learning to balance the needs of API consumers with the needs of business owners and key (non-technical) decision makers. This model has a great reputation amongst the development community for being easy to use and implement.

I.2.2 Online payment

Online payment is one of the most successful and best-known API-driven businesses. From the very beginning this piqued the interest of many investors and received a good deal of seed funding and venture capital investments. Prior to API-based online payment, developers were in limbo when it came to e-commerce payment processing. They were forced to either use a self-hosted gateway (e.g., Authorize.net), which came with a lot of set-up required, or go for a branded third-party option such as Paypal or others. These options were all very limited. They did not offer API and did not care about the experience of the developer. Payment processing is not exciting or ground-breaking, but so many companies have to do it and so many developers have to implement it, that there were major business opportunities for innovators.

Taking a developer-first approach has also been favourable, making it incredibly easy to set up and start operating, so that payments can be accepted almost immediately. This ease of use has resulted in many third-party integrations, which diversifies a company's ecosystem without removing their means of profit.

I.2.3 E-commerce

The original API was only released to a few selected official partners, at first, but today extensive APIs are open to all. With their API, users can submit items, sell items, list items, display listings, etc. It enables things like automated auction monitoring and improvements to the auction management process.

With all this ability to build upon the e-commerce API, there are many ways for third-party services to monetize their systems. This has resulted in a flourishing ecosystem offering far more than a company could or would alone.

More recently, companies have expanded into new API territory. They have added expanded commerce APIs which allow businesses to use an API provider as a platform for on-site ecommerce. This is a unique and exciting approach to e-commerce development.

I.2.4 User relationship management

From experience using XML APIs, managers realized early on that, particularly at the enterprise level, sales data need to be shared across multiple company platforms. In this case, an API was the perfect solution and allowed a company to fully integrate with existing systems.

The same APIs used by third-parties are also used internally to power salesforce.com. The APIs and web interface were launched at the same time and provided what is now recognized as the first enterprise-level web software as a service.

One major selling point of a company is their marketplace of third-party applications, available directly within the company. Companies actively work to support these app developers, recognizing that they also drive revenue, even if it is not direct.

I.2.5 Metadata for electronic media

In 1985, a company offered a database of metadata that was directly encoded with videos. From there, they expanded to other physical media, like digital video discs and digital video recorders, and with time most major movie studios utilized their encoding service. From there, they expanded to digital rights management and distribution management of electronic media.

In 2009, the company pivoted to offer their metadata via API, rather than encoding with media directly. They are most known for their music metadata, which is used by major companies. They also cover areas like TV, music, movies, and games.

I.2.6 Connectivity between apps and devices

All the models above focus on an API-driven approach that appeals primarily to developers. The APIs are highly technical, requiring prior knowledge to understand how to implement them. While these (and many, many more) APIs have much to offer, they are not available to the masses.

A few companies came on the scene attempting to change this. Instead of focusing on developers, they targeted a less technical audience. Their products offer a graphical user interface -based approach to API consumption.

One company offers a simple system of triggers and actions, available in a brightly coloured user interface. A user signs up for an account and selects a trigger. They authorize connections to the service in question, and then pick the action that follows the initial trigger. These triggers and actions are saved as recipes in the system and are run automatically once enabled.

Bibliography

[b-ITU-T L.1370] Recommendation ITU-T L.1370 (2018), *Sustainable and intelligent building services*.

[b-ITU-T X.1206] Recommendation ITU-T X.1206 (2008), *A vendor-neutral framework for automatic notification of security related information and dissemination of updates*.

[b-ITU-T X.1520] Recommendation ITU-T X.1520 (2014), *Common vulnerabilities and exposures*.

[b-FinTech] Webpage: Wikipedia – Financial Technology.
<https://en.wikipedia.org/wiki/Financial_technology>

[b-JBA-API] Website: Japanese Bankers Association (JBA) – Report of Review Committee on Open APIs: Promoting Open Innovation.
<https://www.zenginkyo.or.jp/fileadmin/res/news/news290713_3.pdf>

[b-OAPI] Webpage: Wikipedia – Open API.
https://en.wikipedia.org/wiki/Open_API

[b-OAT2WA] Webpage: Open Web Application Security Project – OWASP Automated Threats to Web Applications.
<https://www.owasp.org/index.php/OWASP_Automated_Threats_to_Web_Applications>

[b-OPLTM] Webpage: Wikipedia – Open platform.
<https://en.wikipedia.org/wiki/Open_platform>

[b-OSSV] Website: WhiteSource – Top Open Source Security Vulnerabilities.
<https://resources.whitesourcesoftware.com/blog-whitesource/top-open-source-security-vulnerabilities>

[b-ReAAL] Website: ReAAL evaluation Report – European Comminssion – Europa EU.
<https://ec.europa.eu/eip/ageing/sites/eipaha/files/results_attachments/20160803_d5-3_evaluation-validation-and-evidence-report.pdf>

[b-TBMC] Website: bmc blog – IT Security Vulnerability vs Threat vs Risk: What are the Differences?
<https://www.bmc.com/blogs/security-vulnerability-vs-threat-vs-risk-whats-difference/>

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