WHITE PAPER

API Cybersecurity in Telecom Networks, Ecosystems, and Services

Executive Summary

Application programming interfaces (APIs) have always been an important part of telecoms networks and services life cycles, from activation, operations, administration, and management (OAM); billing; and third-party charging, integration, and more. Application programming interfaces are fast becoming the default communication method, such as within the 5G core and some of its service exposure points.

In the efforts and investments of telecoms providers to drive growth via service and market innovation, we can identify two major trends, both heavily relying on APIs for enablement and monetization:

1. Technology and service partners' increased scope and role in building beyond-connectivity value ecosystems for enterprises and consumers alike. These ecosystems are critical to delivering growth and value to the end customer and are heavily dependent on APIs for integration, communications, and overall life-cycle management.
2. Exposing telecom network capabilities and data to application developers, enterprises, and other third parties is a way to monetize and increase return on investment (ROI) on past and existing investments.

The growing API exposure to trusted and untrusted third parties, the sensitive nature of what may be exposed and its impact on services and operations, and the increasing regulatory landscape telecoms providers and their partners and customers operate in all drive the urgent need for API cybersecurity in modern telecom networks, ecosystems, and services.

Telecoms API Use Cases

From a high-level cybersecurity point of view, API use in telecom environments is in two categories:

1. **Internal use cases:** These use cases use APIs to communicate, integrate, and expose components that are relatively within the domain and control of the telecom operator. These APIs may expose very sensitive information and are key to sustaining operations.

   As telecoms environments contain a mix of modern and legacy systems and technologies that may not be very well known or documented, these use cases will also include shadow APIs. Although this environment is considered trusted and under the operator's control and management, these APIs are still exposed to intentional and non-intentional risks and need to be clearly identified and protected.
2. **External use cases:** These use cases are driven by new technologies such as 5G and the Internet of Things (IoT) and the search for monetization, automation, and orchestration. These APIs expose telecom's critical network and service components, capabilities, and data, to third parties and enterprise customers not within its domain or control.

API Threat Model

Application programming interfaces are vulnerable to a range of issues, such as:

- Misconfigured and inadequate authentication and authorization
- Information disclosure
- Classical web attacks, such as command execution
- Local and remote files
- Inadequate rate limiting and denial-of-service (DoS)

Often attackers have combined two or more vulnerabilities to execute an attack successfully. For example, exploiting authentication and authorization with the lack of rate limiting and insufficient logging to exfiltrate sensitive data.

To prioritize API-enabled service security, an understanding of the most exploited areas by attackers is required. Diagram 3 provides a high-level threat model related to APIs and the services they enable:

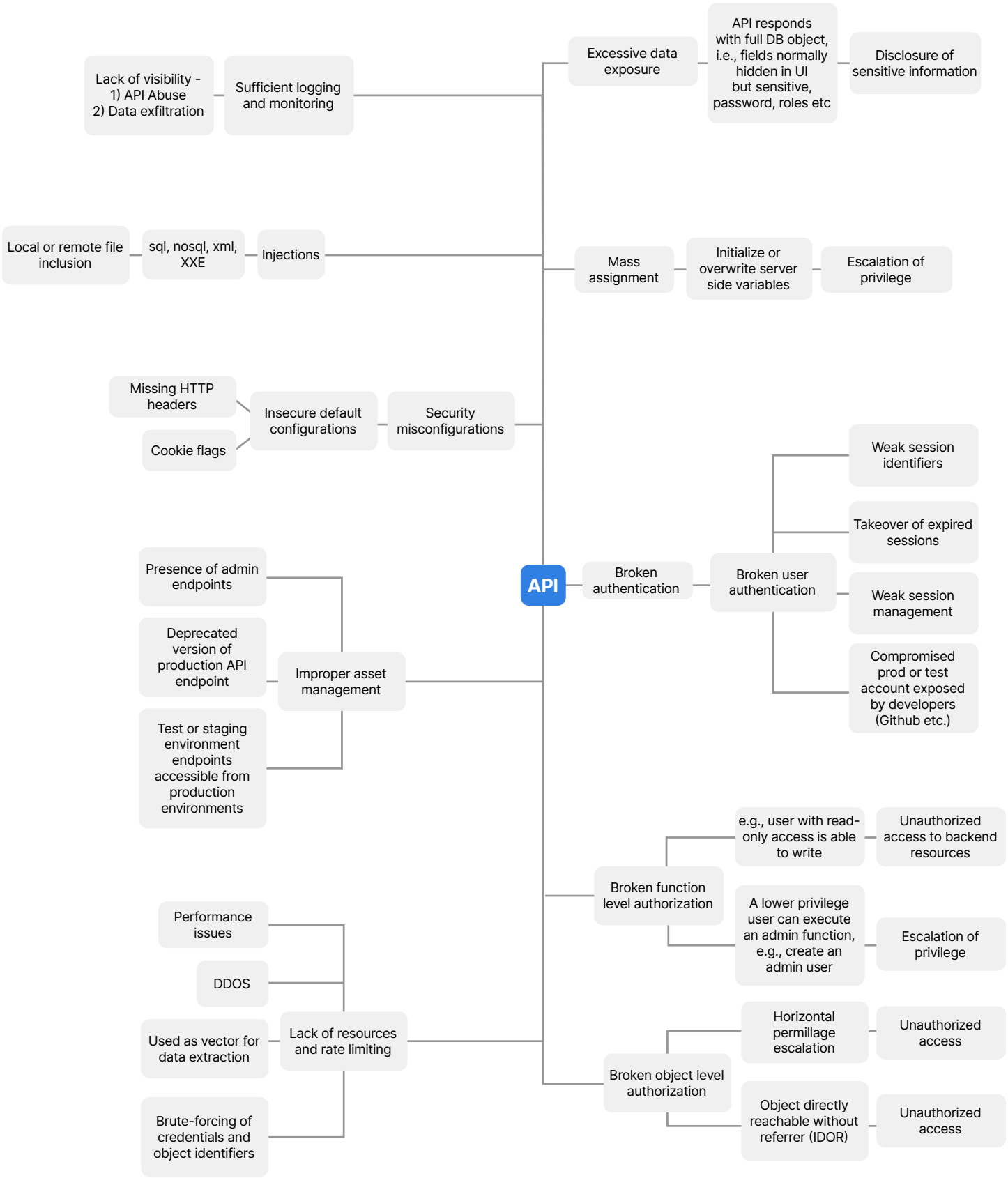

Figure 1: High-level API threat model

- **Broken authentication:** Application programming interfaces are vulnerable to broken user authentication when authentication mechanisms are lacking or not implemented correctly, allowing attackers to assume other users' identities temporarily or permanently. There are extreme cases where no authentication is required due to the perceived internal and trusted nature of the application and service.
- **Broken authorization** occurs when users (authenticated or anonymous) access and edit data or objects that are not related to their role or identities (profiles, account information). These issues could be further broken down into broken object–level identifiers and broken user–level identifiers (weak session identifiers, gaining access to expired sessions, weak session management, or even access to test accounts left exposed by developers). In most API service attacks, broken authentication and authorization are initial attack points.
- **Mass assignment:** Mass assignment threats are caused when the API endpoints save a request's body as-is instead of extracting request parameters individually. Malicious users typically exploit this flaw to initialize or overwrite server-side parameters the developers never intended to expose. A simple example is an attacker sending additional parameters in the request that tie their roles into "admin" functionality with no validation, escalating their role's privilege.
- **Excessive data exposure:** A database query response commonly includes the full database object with all properties, while only partial or limited information will be delivered based on the API call. Excessive data exposure happens when attackers avoid user interface filters that will limit the exposed data and gain access to all information in the response.
- **Lack of resource limitation:** Also known as lack of rate limitation, lack of resource limitation can lead to DoS attacks. This can also enable attackers to conduct brute-force attacks and high-speed data exfiltration.
- **Improper asset management:** Consumed APIs may be related to multiple endpoints mixing users and administrators and even depreciated versions of APIs that can be attacked and expose data unrelated to the user's role.
- **Security misconfiguration** is a broad group of insecure default configurations, such as missing HTTP headers, missing cookie flags, exposed API documentation in the WSDL files, or even exposed mobile application source code.
- **Classical web attacks** are mostly related to injection attacks such as SQL, no SQL, XML file Inclusions, and the like. These facilitate the attack by committing an action not intended by the application.
- **Insufficient logging and monitoring:** Failure to properly log, monitor, or report API-related security events makes suspicious behavior difficult to detect and significantly raises the likelihood that an attacker can successfully exploit API vulnerabilities. For example, an attacker may probe an API for known vulnerabilities over time. Allowing such probes to continue undetected increases the likelihood that the attacker finds and successfully exploits a vulnerability.

General API Security Requirements

Regardless of where they are used, APIs need certain base security requirements to be protected:

Enforce centralized authentication and authorization to ensure all API endpoints are authenticated, regardless if they are accessed by human or machine users. The method of authentication is dependent on the target services protected. In addition to enforcing authentication, additional controls to protect safe storage and key rotation is essential. Upon successful authentication, authorization mechanisms must be enforced to ensure access is strictly based on zero-trust permissions principles to ensure least-privilege access and fine-grained roles and verification methods per service.

Concerning mobile networks, 3GPP specifications have mandated certain security requirements for exposure services, mainly in documents such as TS 129 522 and TS 33 501. These include protecting the exposure function using OAuth 2.0 with token-based authorization for application function (AF) clients.

- **Validation of user requests (schema validation)** is critical to protect against multiple threats, as mentioned in the OWASP API Top 10 lists, including mass assignment, injection attacks, disclosure of sensitive information, and more. It is critical to enforce validation of user requests with a combination of schema validation (ensuring user requests conform to input rules expected by the service) and a set of predefined rules.
- **Response filtering** is required to avoid the disclosure of sensitive information, such as DNNs, S-NSSAI, and SUPI. Sensitive information needs to be filtered out in case of unsecure exposure.

- **Early vulnerability detection** while services and their related APIs are being developed and deployed will help early detection of vulnerabilities and reduce the overall attack surface. These can include security scanning of the code and additional third-party libraries and components.
- **Rate-limitation enforcement** needs to be set for the maximum rate of requests on exposed APIs based on the applicable use case. This is to avoid DoS or brute-force attacks. Additionally, rate limiting can reduce the rate at which data is retrieved from the exposed services.
- **Maintaining API inventory**, both current and previous versions, is important to track the spread of exposed APIs. Detecting new API requests is also recommended to detect new exposed services or endpoints. These inventories help reduce shadow APIs and ensure all exposed endpoints are adequately secured and accounted for.
- **Protecting underlying platforms** with adequate security mechanisms, including firewalls, endpoint security, and underlying infrastructure vulnerability protection, are all components in protecting the hosted services (for example, securing and monitoring container orchestration platforms in a containerized environment).

Requirements Specific to Network Exposure API

In addition to generic API security requirements, exposed function-specific security requirements must be defined and maintained. In the example of 5G Network Exposure Function (NEF), these are broadly categorized as:

- Secure translation of internal to external information (maps AFs service identifiers to internal networks DNN, S-NSSAI)
- Masking of sensitive internal network information (DNN, S-NSSAI, UE sensitive information)
  In accordance with the AF trust level accredited by the operator
- Forwarding sensitive traffic information and service identifiers from AF to respective NFs within the internal network securely (SMF, PCF)
- SLA enforcement (number of requests/second agreed)
- Always enforce authentication and authorization
- Whitelisting source SCS/AS or AFs
- Maintain specific enforcement policies for each SCS/AS or AFs based on agreed-upon criteria
- Enforce transport security
- Enforce protection against malformed or oversized requests from SCS/AS or AFs
- API request discovery (assists in monitoring changes in incoming requests, benchmarking expected traffic, and later identifying anomalies within these requests)
- Implement filtering based on specific parameters within the incoming request (allowed MSISDN range, External ID)

The above set of NEF security requirements is an additional layer of security on top of common sense, generic API-related security requirements:

- Restrict service reachability
- Enforce allowed methods for allowed source AFs
- Protect of underlying web server and overall infrastructure
- Protect against generic API-based attacks (insecure direct object reference, mass assignment, injection, manipulation of hidden parameters)

Securing the API Ecosystem with Fortinet

Fortinet's application security suite enables telecoms to protect services enabled by APIs throughout their life cycles, from development to deployment and maintenance, throughout the telecom domains.

API-exposed services parameter defense with FortiWeb web application and API protection

FortiWeb provides a rich set of applications and API security functionalities, from delivering core security requirements like authentication and authorization of AFs, to protecting exposed API services and enforcing third-party SLA agreements. Below are a few high-level features of FortiWeb that assist in securing exposed services:

- Certificate-based mutual authentication
- Enforcing OAuth authorization for third-party AFs
- Topology hiding (allowing only exposed services intended for the consumption of third-party AFs)
- Load balancing
- API syntax validation
- API schema validation: XML, JSON, OpenAPI (RESTful)
- Masking of information elements and request body checks
- API content, call rewrite
- Protection from zero-day attacks
- Lightweight API security (FortiWeb deployed as a black box reverse proxy)
- Enhanced with ML-based threat detection and API discovery:
  - Continuous learning adapts to API schema changes
  - API data classification
  - False positives elimination
  - Real-time threat intelligence via FortiGuard Labs

When the operator does not wish to deploy TLS inspection, the FortiWeb can be deployed as a transparent proxy to inspect encrypted traffic and enforce certain basic security requirements, such as Certificate SNI validation, DoS and DDoS protection (including rate limiting), delivering IP reputation enforcement and other threat intelligence services.

Life-cycle vulnerabilities detection with FortiDevSec continuous application security testing

FortiDevSec orchestrates and automates continuous app security testing. It allows developers to detect and remediate security vulnerabilities in app source code, open-source/third-party libraries, secret container images, IaC files, and live web app URLs.

FortiDevSec allows the testing, detection, and remediation of security vulnerabilities within the DevOps continuous integration, delivery, and deployment (CI/CD) life cycle:

- SAST on source code to identify application vulnerabilities (injection, XSS, command execution)
- Software composition analysis to identify vulnerable software components, libraries
- DAST scanner to identify vulnerable discoverable only during the runtime of the application
- Identification of hardcoded passwords, tokens, or secrets within the source code
- Integration with CI platforms to integrate security into the development process

Protecting the infrastructure

Security segmentation, perimeter protection, and securing the management layer needs to be part of the first layer of security in any telecom domain and service. Unauthorized access to physical and virtual infrastructure components and services can act as a major vector of attack on the infrastructure, services, and applications and must therefore be protected and monitored.

The below Fortinet solutions provide security visibility and enforcement for telecoms' physical and management layers:

- **FortiGate Next-Generation Firewalls (NGFWs)** provide security segmentation, perimeter protection, and management, administration, and orchestration (MAO) layer firewalling to protect the underlying infrastructure.
- **FortiWeb web application and API protection (WAAP) firewalls** protect management API traffic and any web services used for provisioning, operations, maintenance, and third-party or partner access. Securing these is critical as most MAO activities are executed using web or API access.
- **FortiCNP cloud-native protection** natively integrates with private and public cloud environments to deliver actionable security insights and real-time threat and misconfiguration protection in container environments and storage repositories while simplifying compliance.
- **FortiAuthenticator** enforces identity management and Single Sign-On (SSO).
- **FortiToken** provides two-factor authentication, an OATH-compliant OTP generator application on mobile devices, and support for time-based (TOTP) and event-based (HOTP) tokens.

Conclusion

The role of APIs in telecoms is rapidly growing in complexity and importance. These APIs are key to unleashing Network-as-a-Service (NaaS) opportunities, enabling better customization and agility, and creating value ecosystems to drive innovation and growth. They are the gateways to critical components, capabilities, and services, exposing these for consumption by trusted and untrusted third parties, application developers, and AFs.

Fortinet empowers telecom operators to protect their APIs and the critical components and platforms that they expose and interact with. The Fortinet Security Fabric provides multi-layer security solutions that protect any API-based exposure with single-pane-of-glass management and automation throughout the telecom's API life cycle and ecosystem.

August 13, 2023 9:11 PM

2299141-0-0-EN