Hacking 5G is no rocket science

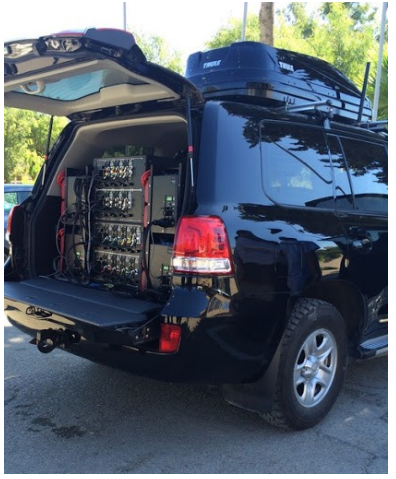
Dr. Altaf Shaik
& Matteo Strada

TU Berlin

Nullcon Goa 2022

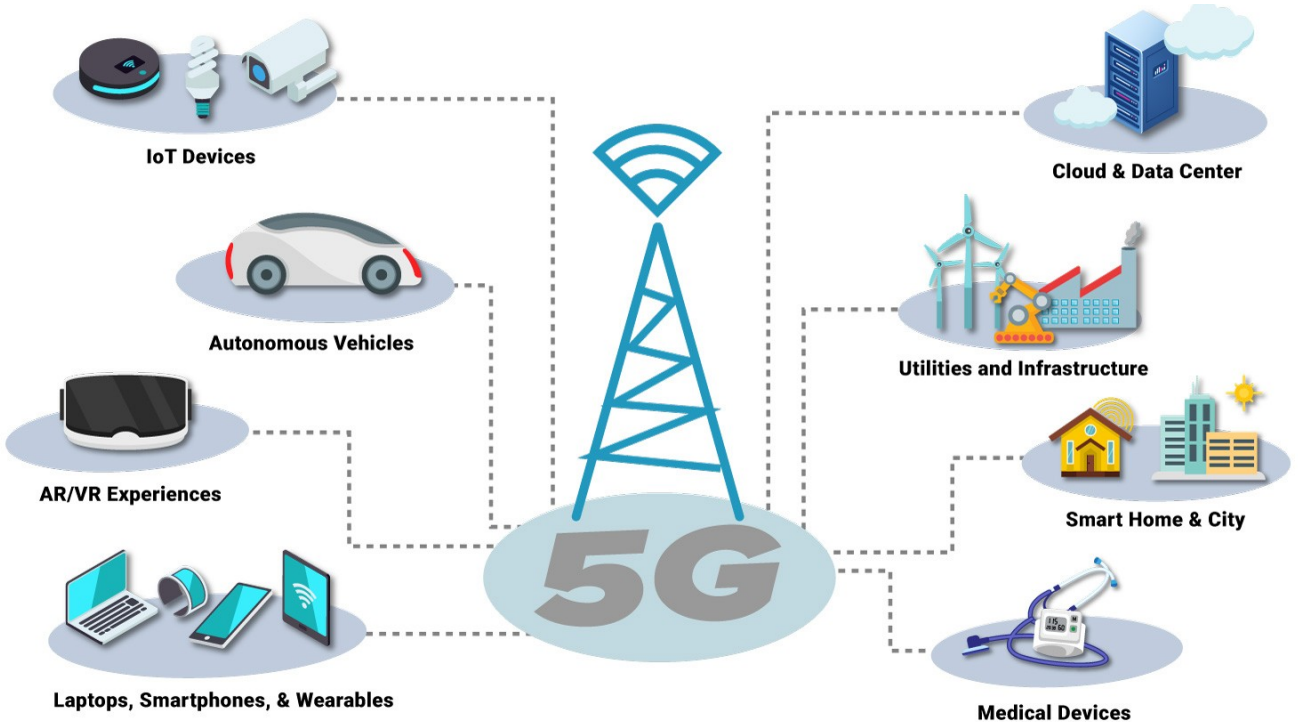
Attacks so far in mobile networks

- Radio access network – IMSI catchers, False base stations
  - Lack of sufficient authentication and security protocols
- Signaling interconnect – SS7, Diameter interfaces
  - Implicit trust between operators
- SIM attacks – authentication, SIM Jacker
  - SIM browser exploits
- SMS spam, SMShing
- Backdoor (wiretapping)



Classic Attacks (user-targeted)
Information extraction
Location Tracking
SMS and call Interception
Denial of Service
Fraud

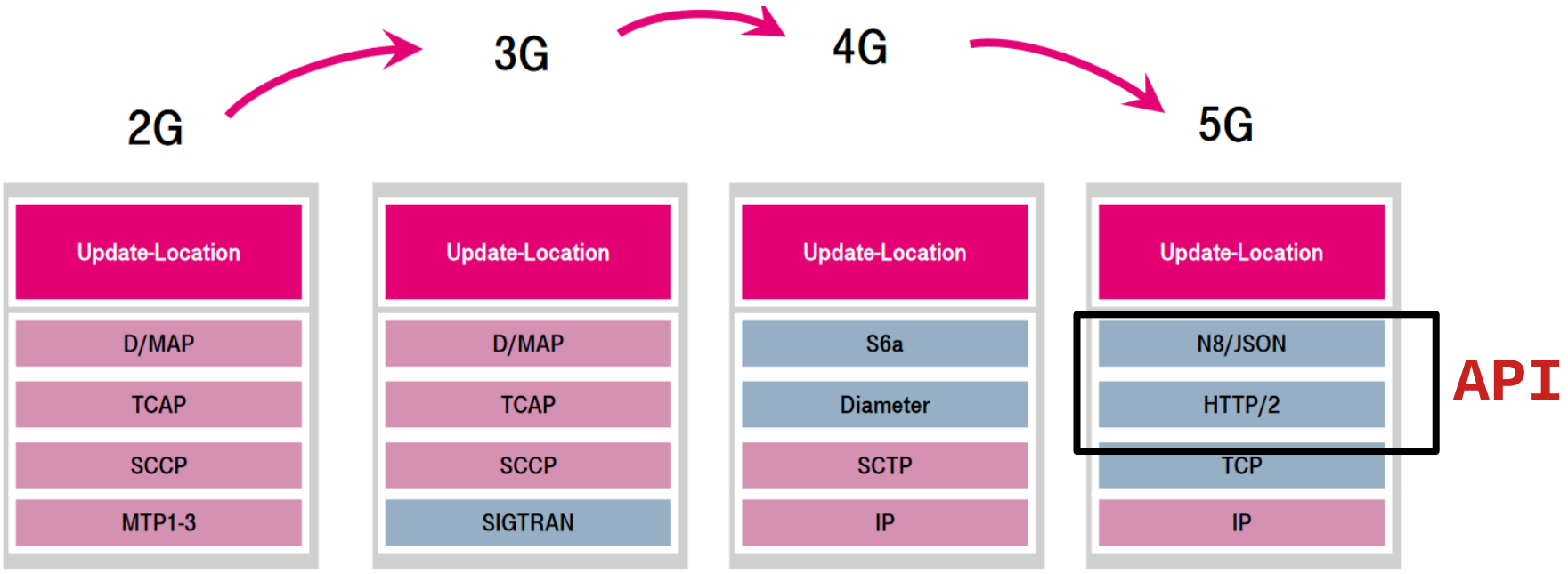
5G is for things



- Infrastructure targeted attacks
- Increased threat
- Enormous damage

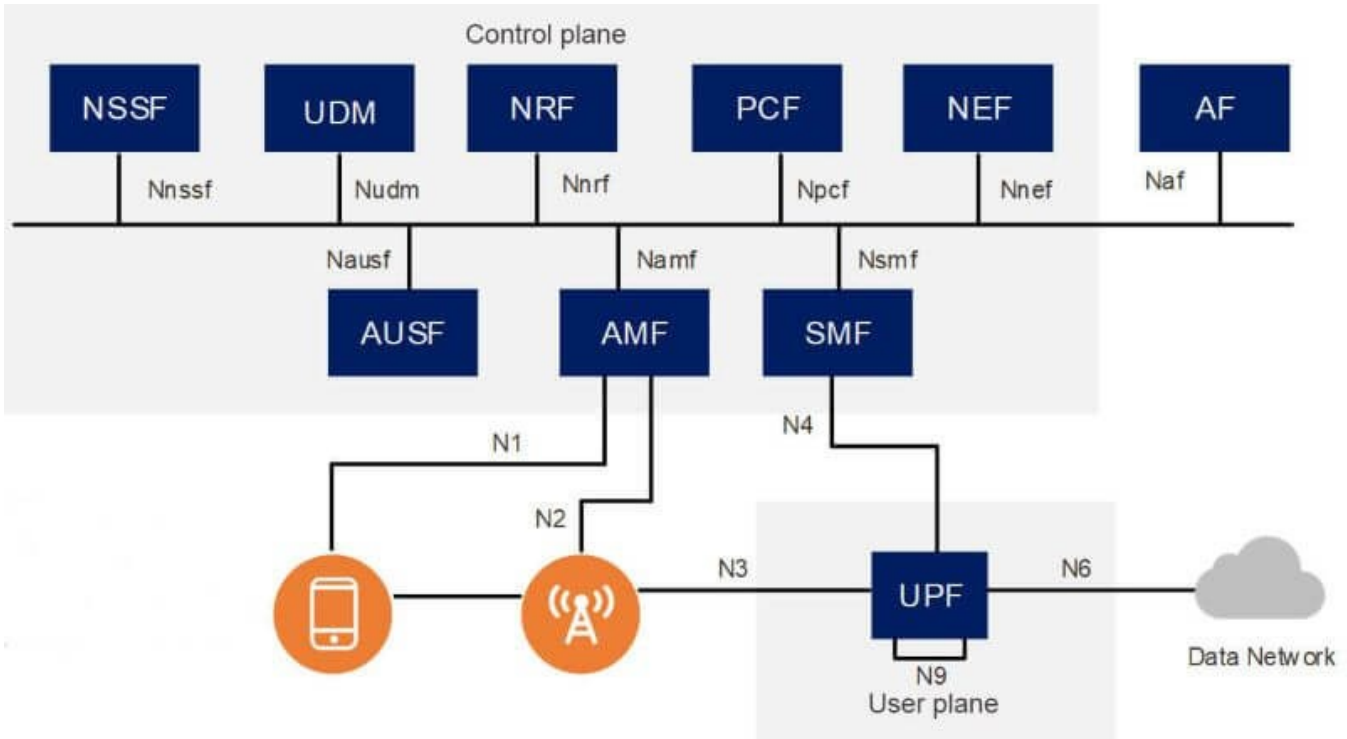
Img src: https://zvelo.com/impacts-5g-part-1-wireless-technology-device-growth/

Protocol evolution

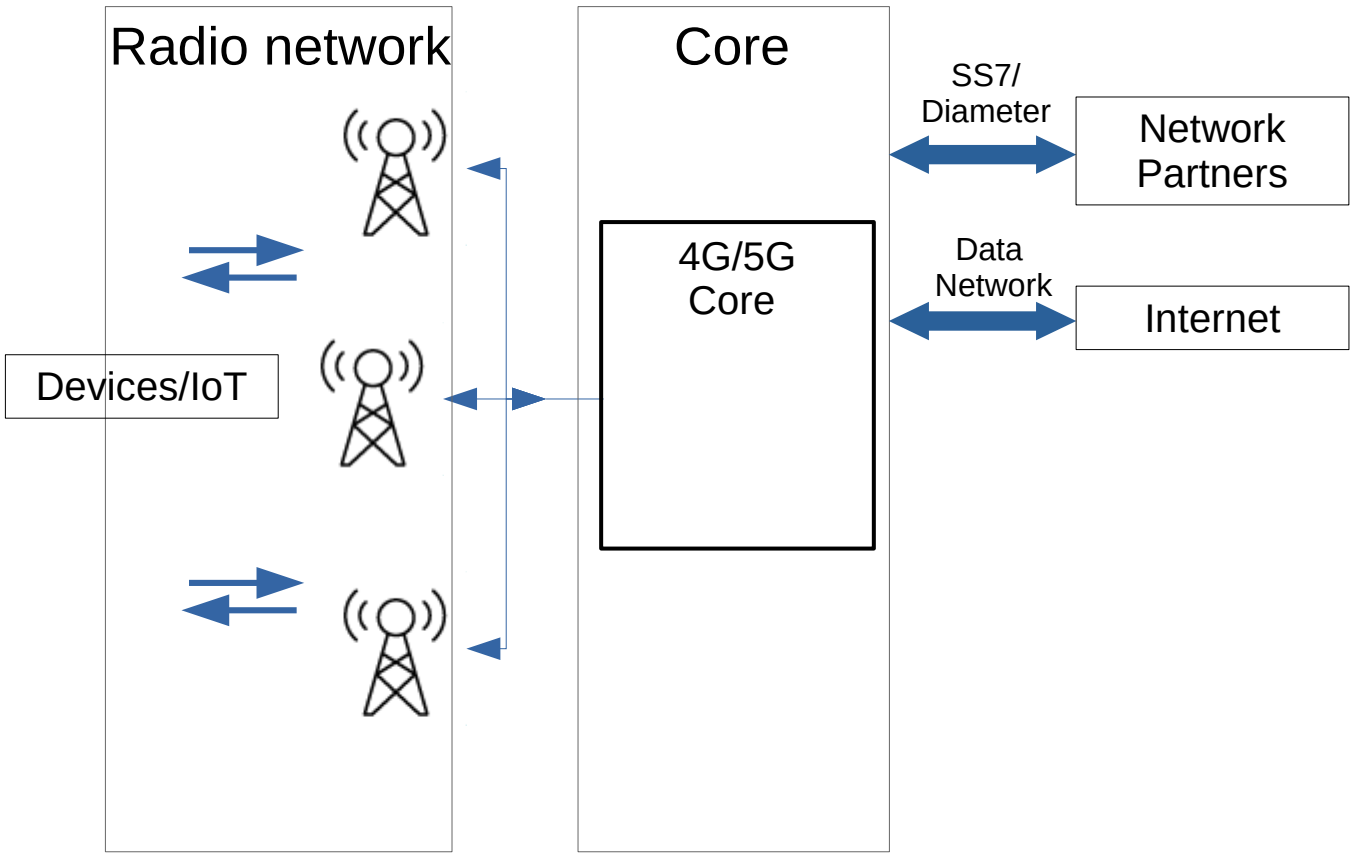


Source: Deutsche Telekom

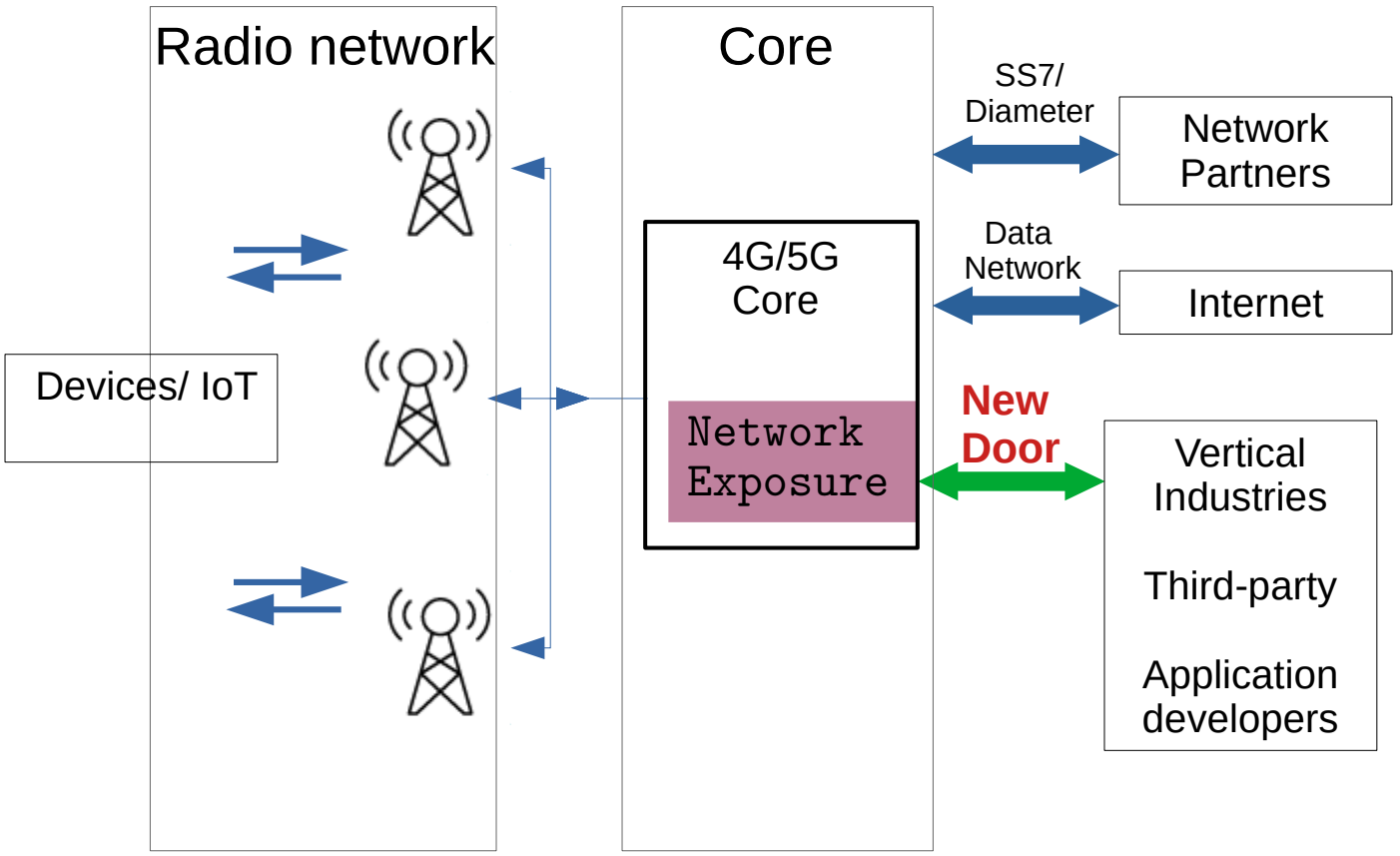
5G network


Control plane
NSSF
UDM
NRF
PCF
NEF
AF
Nnssf
Nudm
Nnrf
Npcf
Nnef
Naf
Nausf
Namf
Nsmf
AUSF
AMF
SMF
N1
N2
N3
N4
N6
UPF
N9
User plane
Data Network

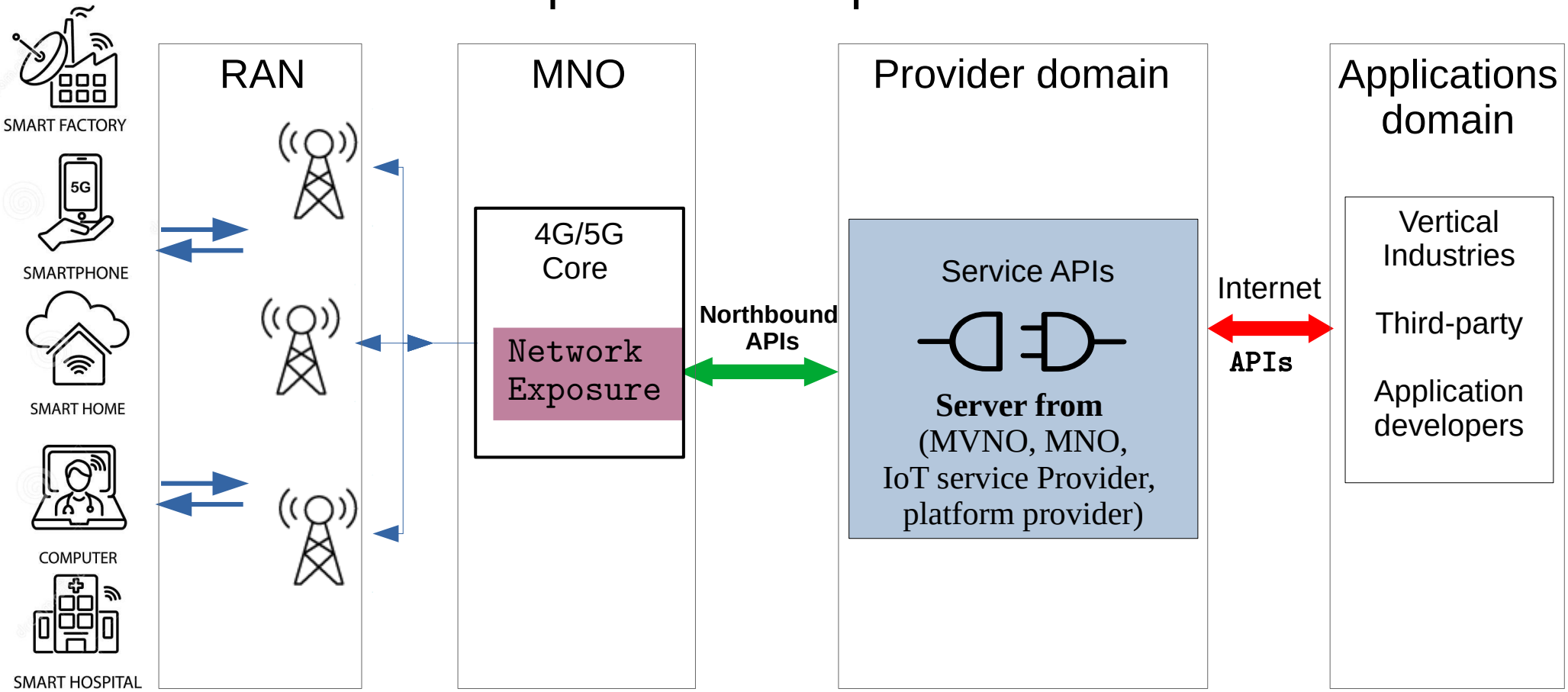
General mobile network



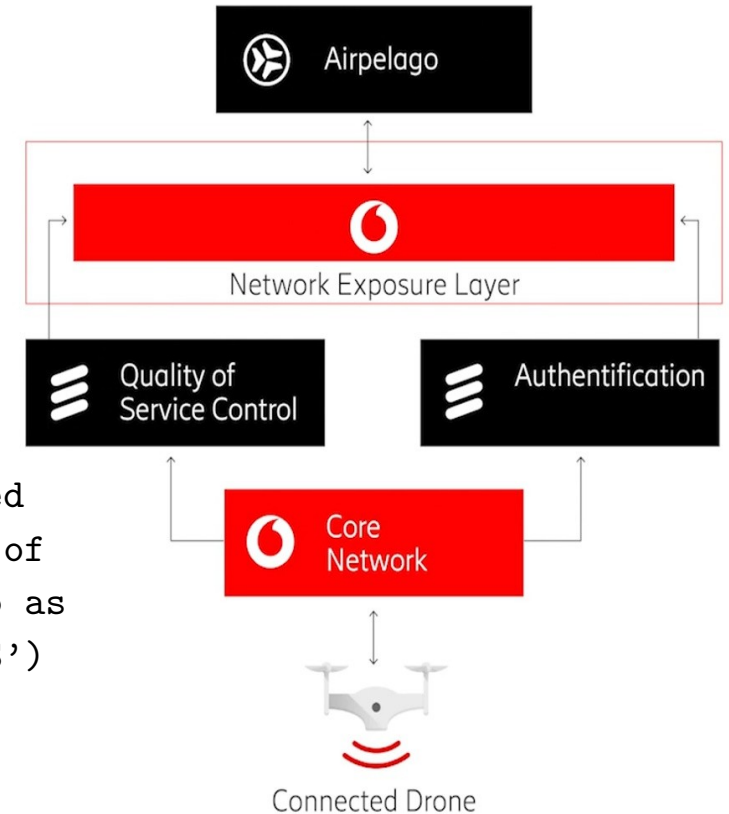
API interface: Network Exposure



Exposure via a provider



Drone control via network exposure



Cellular-connected Drones – example of Vodafone's 'Telco as a Service' ('TaaS') Model

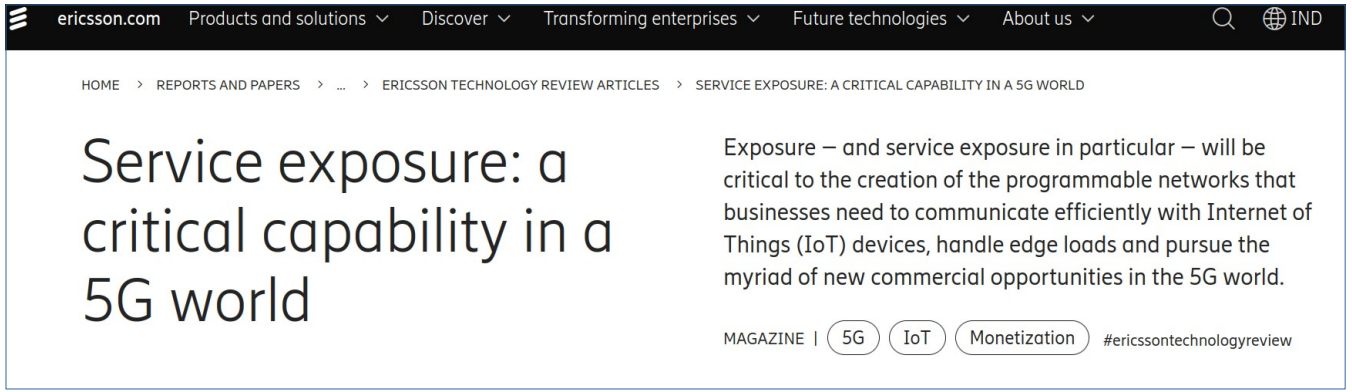
Vodafone provides to trusted third parties via APIs

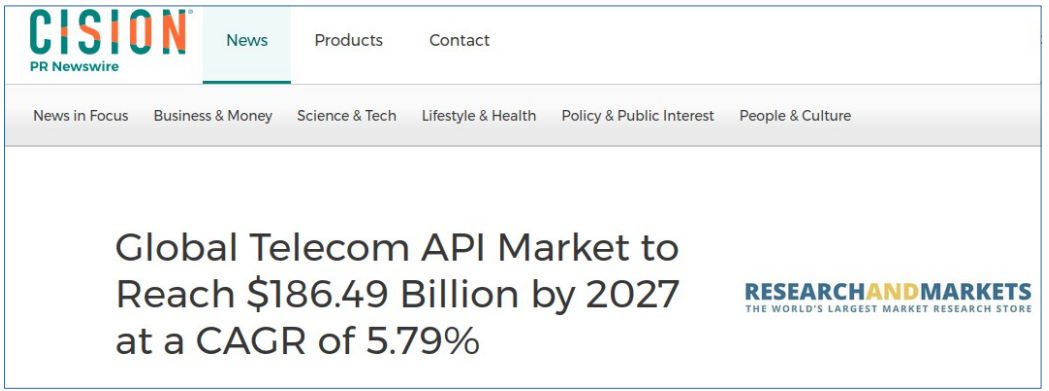
- Network intelligence to produce coverage maps
- Anonymized mobile user information to find crowds
- Ensure constant contact with the control center, even when out of sight.

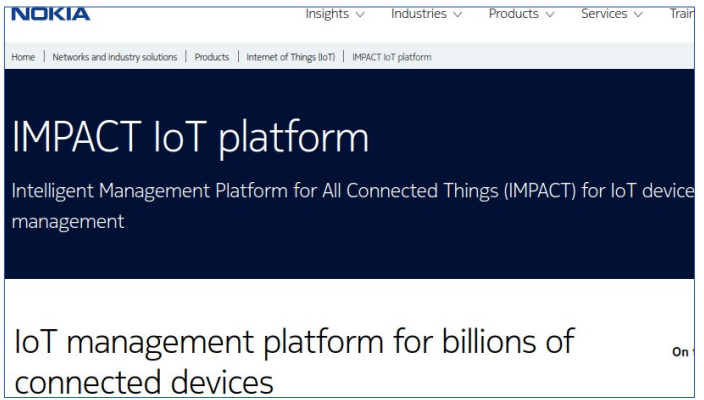
Vodafone's 5G Mobility Lab in Aldenhoven, Germany

Source: https://www.connectivity.technology/2021/01/cellular-connected-drones-to-form-part.html

Future is APIs in Telecom

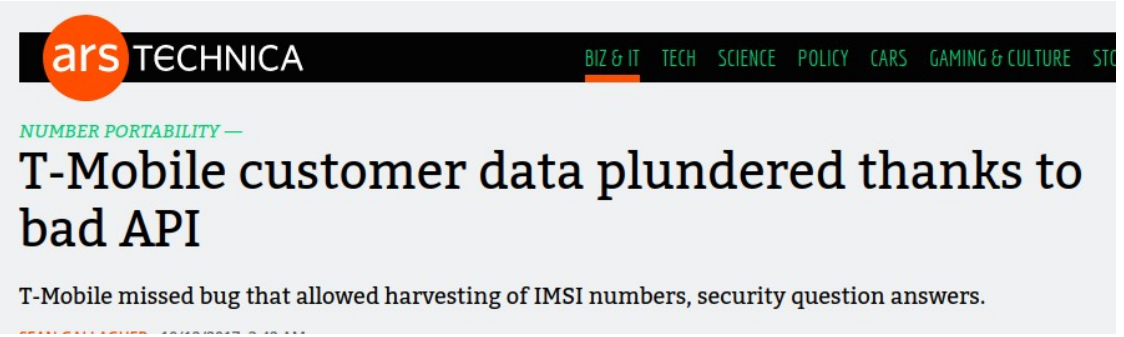






Different from API attacks known in Telecom

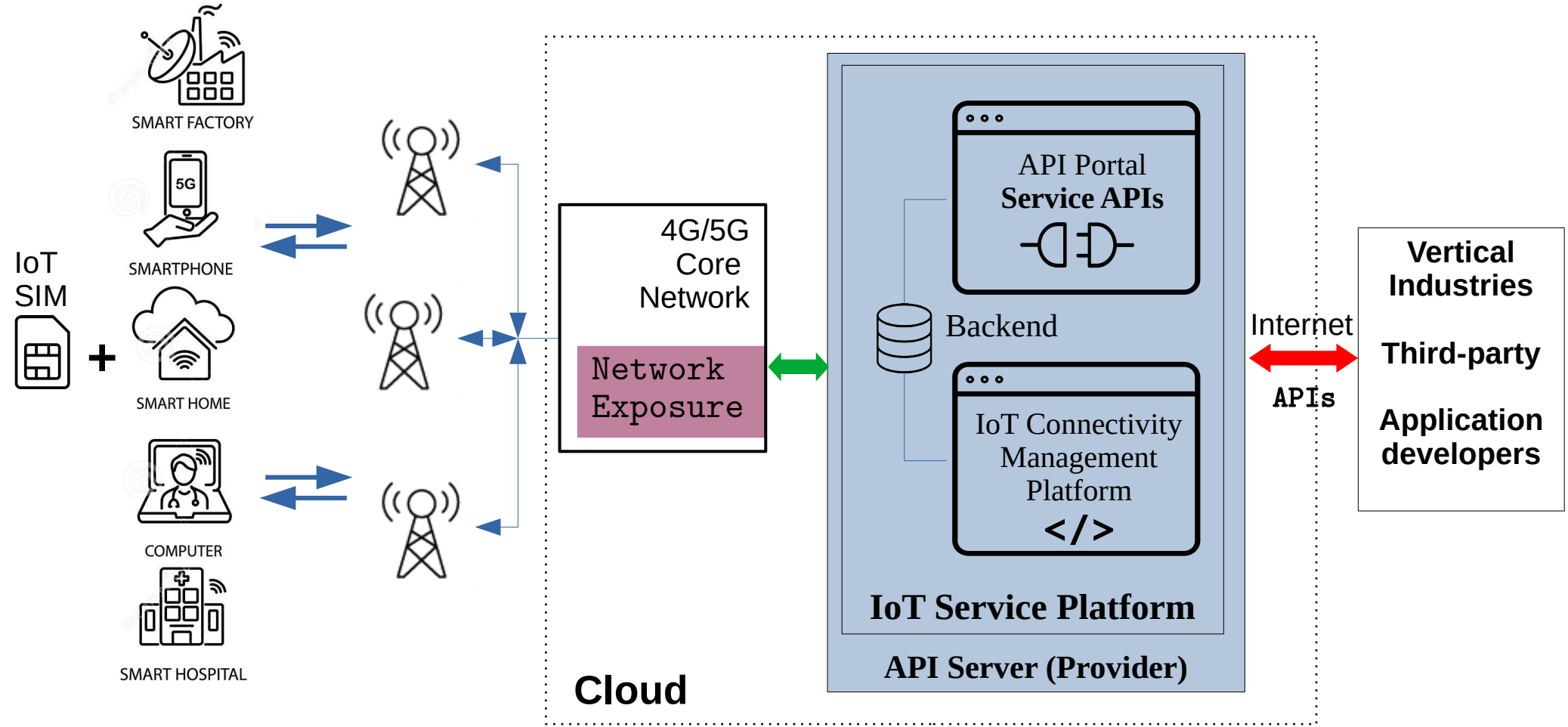
- not supposed to be exposed and hidden from end-users →



It goes like this..

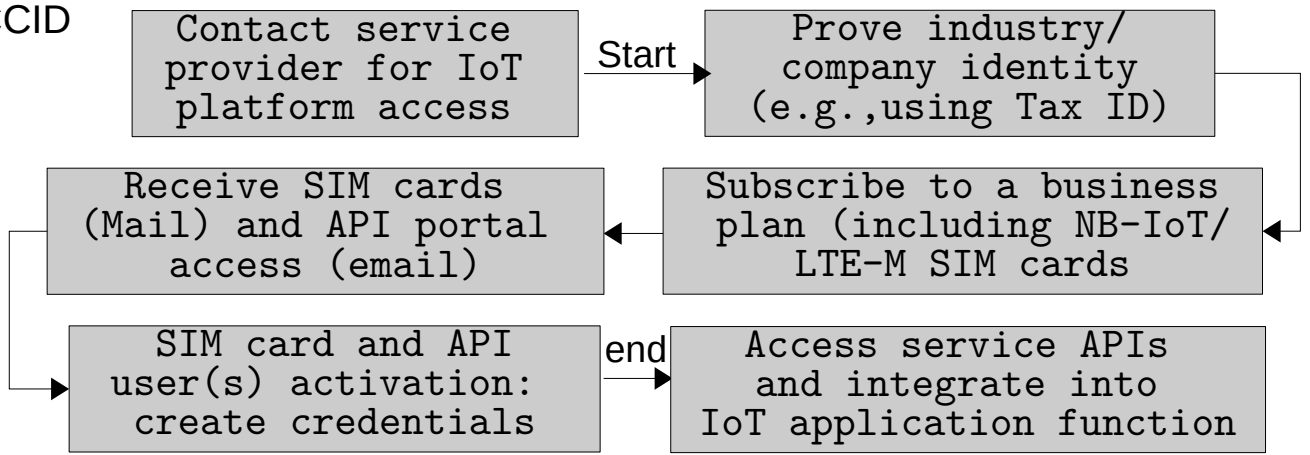
- Commercial network exposure (for IoT)
- Features and configurations
- Security investigation
- Common API risks
- Attacks and findings (vulnerabilities)
- Responsible Disclosure
- Takeaways

Control IoT with 4G and 5G networks



Buy IoT SIM cards

- IoT SIM cards (with IP-data and SMS tariff)
  - e.g., 750MB, 250 SMS, 10 year lifetime, roaming free, 10 $$
- Only available to business entities
- Radio connectivity: 4G networks (NB-IoT, LTE-M, 2G)
- Identification in APIs: IMSI/ICCID

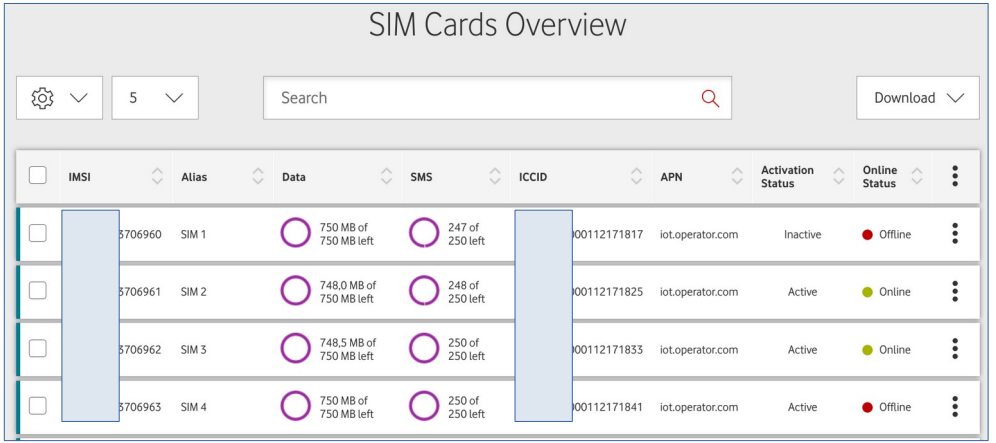


Flow diagram: obtaining access to exposure services

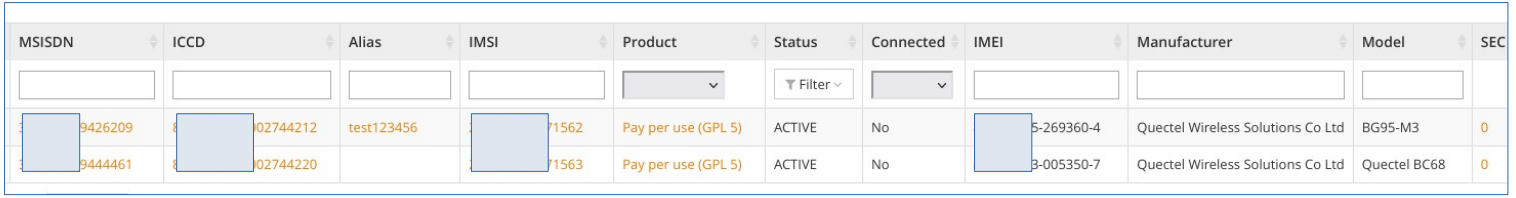
Control and configure the SIMs

After business agreement, access is granted to

- **IoT connectivity management platform**
  - User/SIM management web application
  - Create API user/developer
  - Activate and deactivate SIM
  - Purchase data volume, SMS etc.



IoT connectivity management platform



Access service APIs

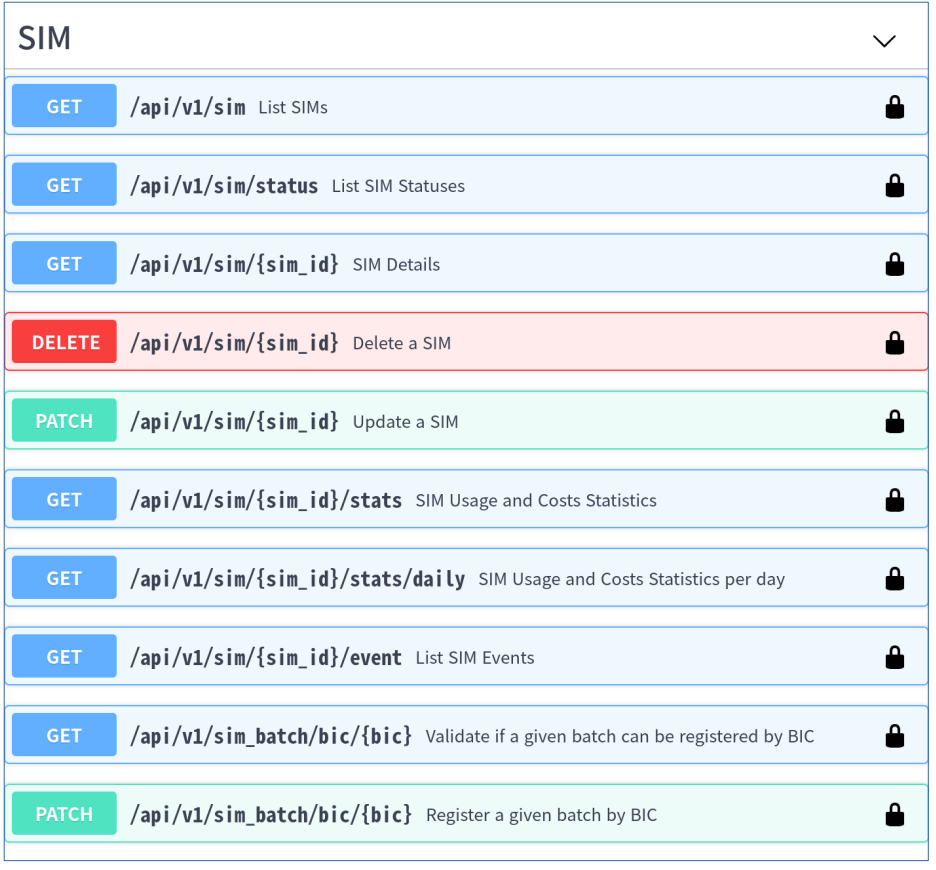
IoT service platform

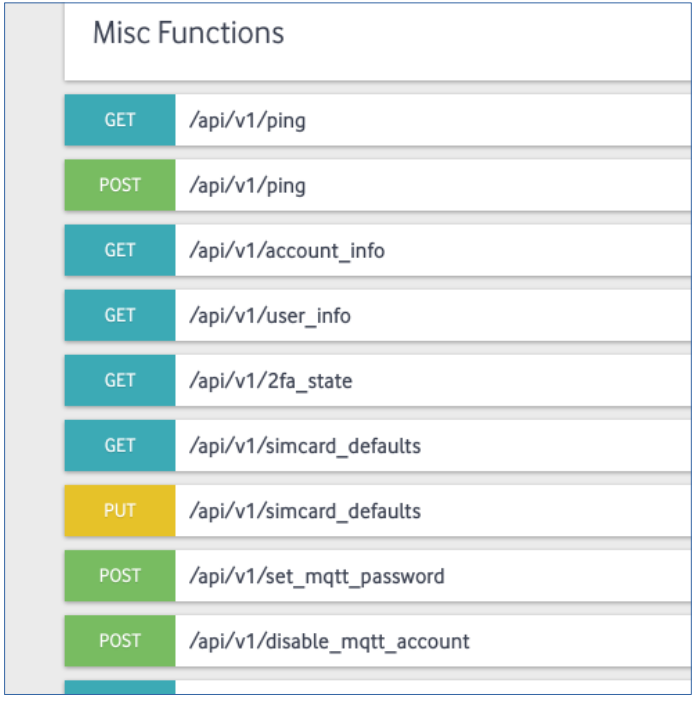
- Service APIs portal (swagger/OpenAPI interface)
- Authenticate and authorize API users
- APIs for location-based services through GPS information, payment integration, voice, messaging and video capabilities, SMS and WebRTC-based features
- Service Level Agreement (SLA) to define and access and API management
- **Core configuration control** - device IP address management, roaming policy control, data-rate, bandwidth, set sleep modes
- **Admin control** - billing and data plan management, SIM & credential management

Example service APIs

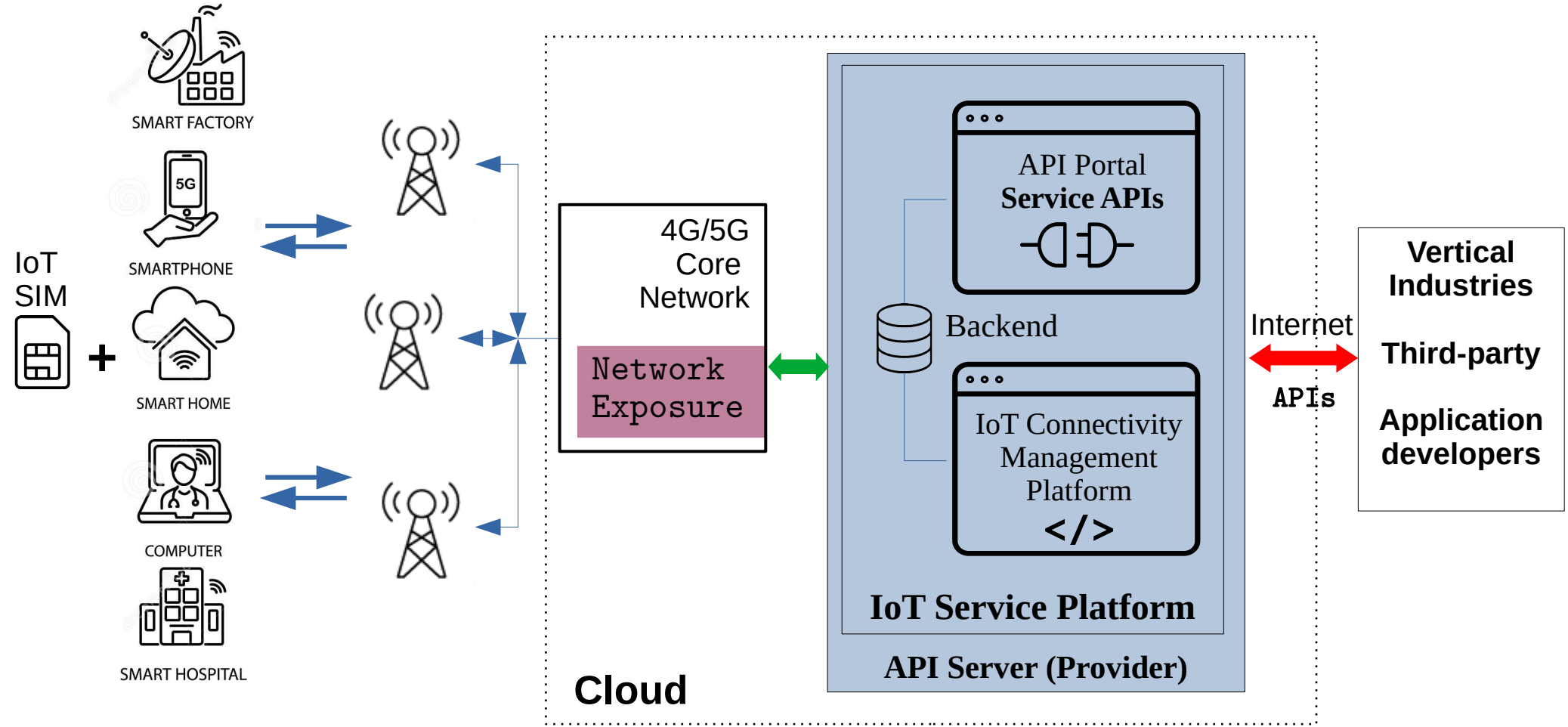
Service APIs inside IoT Service platform

e.g., Swagger interface





Control IoT with 4G and 5G networks



API security for Network Exposure

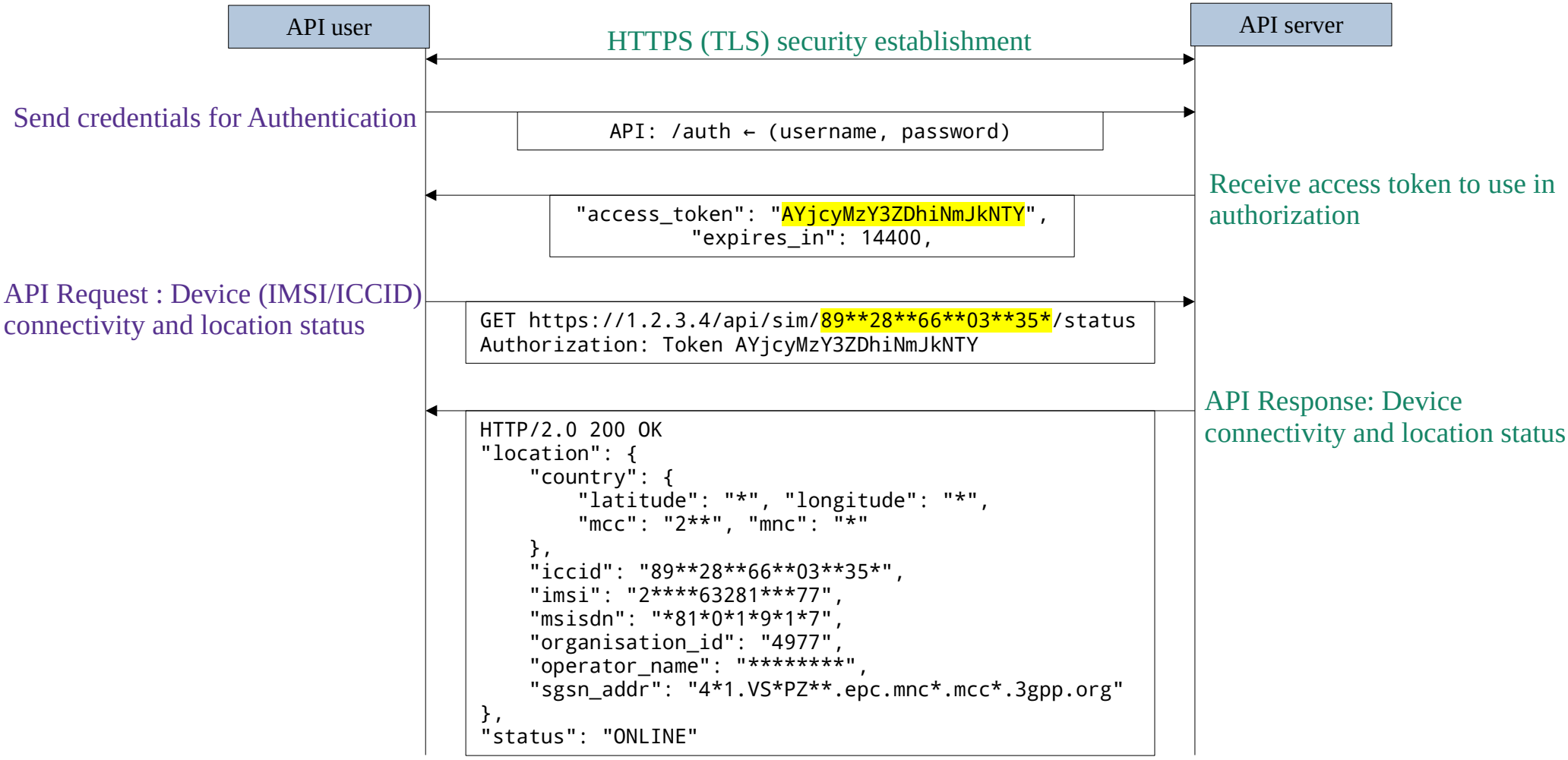
3GPP Standard (recommended) fundamental security mechanisms for exposure services

- Authentication & Authorization (OAuth 2.0)
- Confidentiality and integrity protection (TLS)
- Privacy
- Rate limiting*
- Logging and Monitoring*
- Firewalls/IDS*
- Guidelines from GSMA[1,2]

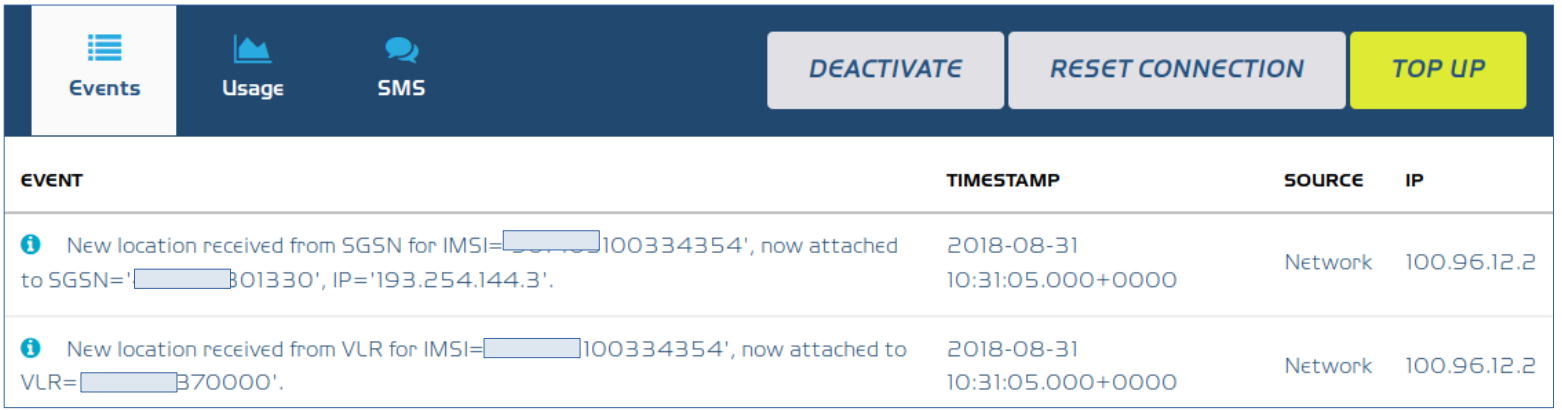
*additional security best-practices

1. GSM Association. Iot security guidelines for network operators version 2.2
https://www.gsma.com/iot/wp-content/uploads/2020/05/CLP.14-v2.2-GSMA-IoT-Security-Guidelines-for-Network-Operators.pdf
2. GSM Association. IoT SECURITY GUIDELINES for IoT Service Ecosystems
https://www.gsma.com/iot/wp-content/uploads/2016/02/CLP.12-v1.0.pdf

How it works: Get device location



Device location updates from VLR and HSS



EVENTS:

Refresh | Export As CSV

Message	Severity	Data Type	Type
SUCCESS HSS ULA for Thing name = "ICCID 8999911240071102	Info	HSS_ULA	EVENT
Thing location history for Thing Name: ICCID 899991124007110	Info	LOCATION_HISTORY	LocationHistory
HSS ULR for Thing name = "ICCID 8999911240071102 4830", MM	Info	HSS_ULR	EVENT
SUCCESS HSS ULA for Thing name = "ICCID 8999911240071102	Info	HSS_ULA	EVENT
Thing location history for Thing Name: ICCID 899991124007110	Info	LOCATION_HISTORY	LocationHistory
HSS ULR for Thing name = "ICCID 8999911240071102 4830", MM	Info	HSS_ULR	EVENT
SUCCESS HSS ULA for Thing name = "ICCID 8999911240071102	Info	HSS_ULA	EVENT

```
"pdp_context": {
    "ggsn_ip_addr": "10.70.4.17",
    "rat_type": { "description": "NB-IoT" },
    "sgsn_control_plane_ip_addr": "10.73.4.5",
    "ue_ip_address": "100.96.15.132"
},
```

Misc functions

Send downlink message

PUT https://api.scs[redacted]n.com/m2m/endpoints/{serialNumber}/downlinkMsg/0/data

Basic API_CON_[redacted]_dbf32 : 7WM63LtS9.9C26I5

CURL

REQUEST Examples

```
curl --request PUT \
     --url https://api.scs[redacted]n.com/m2m/endpoint
     --header 'Accept: application/json' \
     --header 'Authorization: Basic QVBJX0NPTl8wMDAwMDA
     --header 'Content-Type: application/json' \
     --data '
{
     "resourceValue": "Hello world"
}
'
```

Try It!

RESPONSE 202 Try It

```
{
  "requestId": "59148c45-5231-4eaf-8256-d2c6157a474c"
  "msg": "Accepted",
  "code": 1002
}
```

Headers

POST /sim

Load Authentication Center with SIM secret keys. Upload given CSV file (expected format is ICCID,IMSI,KI,OPC)

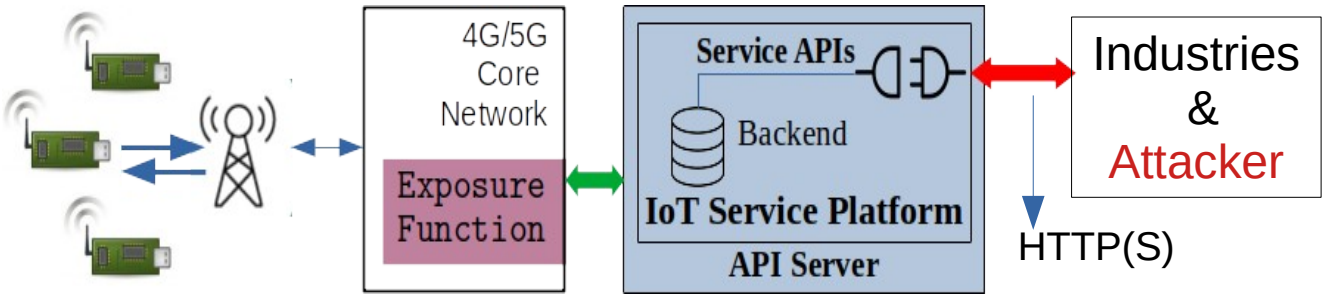
Parameters

Upload a new key to HSS/HLR

Name	Description
simuploadfile * required file *(formData)*	CSV file (expected format is ICCID,IMSI,KI,OPC) upload
authalgo * required string *(formData)*	2G Authentication Algorithm 3
algo3G string *(formData)*	3G Authentication Algorithm: Milenage/TUAK. Default Milenage algo3G - 3G Authentication Algorithm: Milen
amf * required string *(formData)*	It must be 4 characters long, they need to represent hexadecimal digits 8000
MobileSubscriberType * required string *(formData)*	Type of the uploaded mobile identities. Possible values: 1. **Regular** - This is the default type. 2. **External HLR** - Mobile identity that is provisioned to an external HLR. Thi 3. **Bootstrap** - Bootstrap mobile identity. This option is relevant for DSA syst Regular

Attack model for network exposure

- **Requirements**
  - business relationship with the operator or service provider (can forge a tax ID)
    - External, Insider, malicious developer
    - authentication credentials to get authenticated and authorized
    - access to all service APIs, platform and connectivity management platform
- **Goals**: obtain data of arbitrary IoT service platform users (industries), compromise server and penetrate into mobile core network via the exposure function
- **Privileges**: Web/API knowledge Internet, using HTTP(S), remotely-located, use VPN or tor.

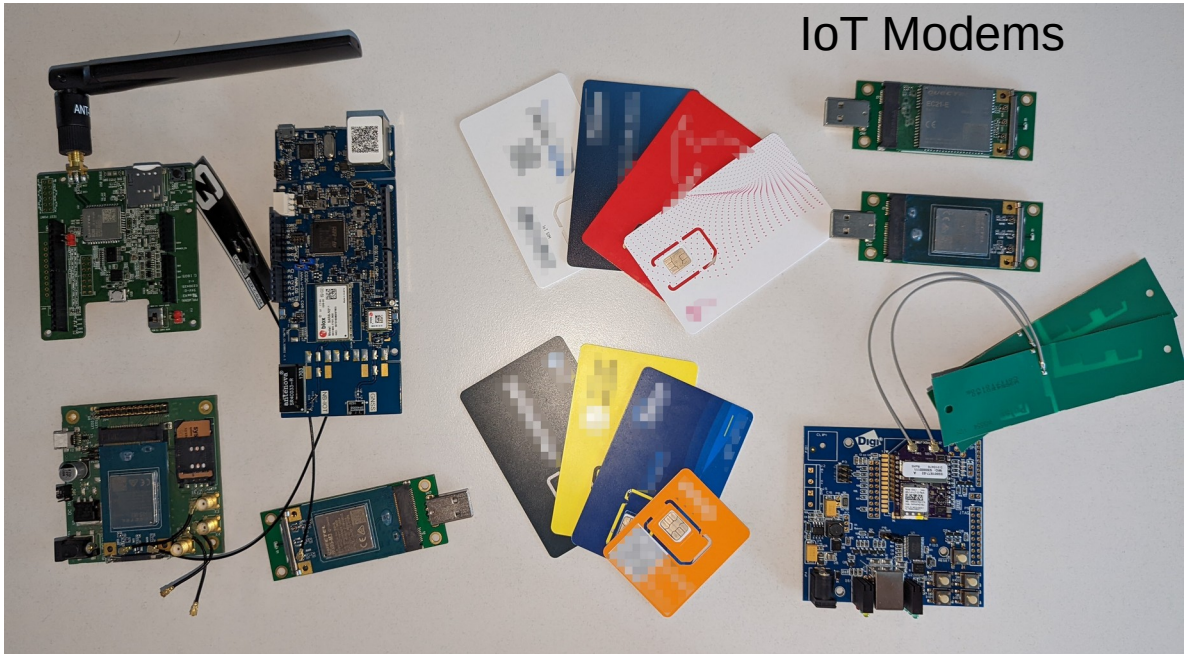


Security questions with IoT platforms?

- Standard security mechanisms. Are they sufficient
- Business logic flaws targeting IoT applications
  - Require manual intensive testing
- Web/API Firewalls or security-by-design
- Security scanners and automated testing
  - Existing are unsuitable for Telecom and IoT applications
- Limited knowledge on attacks on IoT service platforms
- Insecure API – access to API server, customer data, core network & IoT devices

Hands on: Playground

IoT SIM cards +
IoT Modems



\+ **Service APIs**

Commercial IoT service platform security configurations

SP	Type	Authentication	Authorization	TLS [HSTS]	Cloud
1	MVNO	HTTP Basic	OAuth2 + UUID	1.2, 1.3 [✓]	Amazon
2	MVNO	✗	Shared token per platform	1.0–1.3 [✗]	Cloudflare
3	MVNO	HTTP Basic	OAuth2 + JWT HS512	1.2, 1.3 [✗]	Cloudflare
4	MVNO	HTTP Basic	OAuth2 + JWT HS256	1.0–1.2 [✗]	awselb 2.0
5	MVNO	HTTP Basic	OAuth2 + JWT HS256	1.2, 1.3 [✓]	Amazon
6	MNO	HTTP Basic	OAuth2 + JWT RS256	1.2, 1.3 [✓]	✗
7	MNO	HTTP Basic	Static token per user	1.2 Only [✓]	Amazon
8	MNO	HTTP Basic	Static token per user	1.1, 1.2 [✓]	Oracle
9	MVNO	HTTP Basic	Static token per user	1.0–1.2 [✓]	✗

HSTS: HTTP Strict-Transport-Security

- SP: Service platform
- Type of exposure: See document by NGMN
- Authentication: Username + Password
- Current network exposure using 4G core (SCEF)

Platform analysis(1)

To find vulnerabilities in

- API configuration
- Input validation
- Business flow
- Authentication
- Access-control
- Encryption, integrity and replay protection

Hundreds of APIs available in each platform for functionalities. Key functions:

- Exchange SMS/IP with IoT devices, get real-time location
- Update configurations in the core network (sleep, bandwidth, data rate)
- Control and track billing and charging operations
- Upload new Master key for SIM card into the HLR or AuC

Platform analysis(2)

Select APIs that have high impact on business, operation and reputation aspects to do

- Code injection and execution, and device hijacking
- Auth bypass for sending SMS or IP messages to arbitrary IoT devices
- Billing fraud, Reset billing and charging counters and CDRs to zero
- APN manipulation, location tracking, device blacklisting
- Custom IP addresses, VPN setup, malware injection

Modeling a set of attacks:

- Study reactions to malicious actions, payloads, strings, characters, files
- Parameters from 3GPP specifications, API design files, etc.
- Guidelines from OWASP web security testing, REST security cheat sheets
- Tools: Burp suite, ZAP and developed other tools for API analysis

Ethical considerations

- Only access or manipulate API data corresponding to our own user/admin accounts.
- Only key API parameters (like IMSI,ICCID, APN, Tariff, topup, MSISDN, SMS) per platform are analyzed for vulnerabilities – to avoid traffic towards API platform
- GET/POST/PUT operations are carried out into our own accounts
- We took measures neither to damage the exposure platform nor interrupt the ongoing API services for other verticals/users.
- Clear guessing strategy is applied rather than a random penetration/function testing
- Noisy attacks such as DoS or bruteforce are ignored

Platform design and forged access?

APIs available for unlimited use-cases and sensitive functions open even to simple demo users

Procedure to access IoT service platforms is vulnerable to a social engineering attack

- Attacker registers using a forged company (tax) ID and spoofed email address.
- Receives SIM cards to a private(arbitrary) address and also access to service APIs
- Can access IoT platform cloud and data resources hosted on it
- Attacker now masquerades a target company/industry while using the platform

Advantages:

- Limitless API operations – many lack rate-limits
- Lack of (strict) monitoring and logging facilities

Relaxed customer verification found with many providers

- **A strict KYC procedure should be implemented by both providers and operators.**

Common API weaknesses in IoT service platforms (9)

(access-control, authentication, backend exposure)

Guessable username and password policies for API authentication

Password creation, update, management are not compliant with GSMA guidelines[1,2]:

- Weak passwords are allowed (such a *root, admin, iotadministrator*) as credentials
  - only a "few dictionary passwords" are prohibited by some and have shortcomings
- Some restrict dictionary passwords during account creation, but allow them during password update

* asdf1234, qwer1234, qwerty1234 -> weak password, not allowed
* 1qaz2wsx -> top 100 weak password
* iotadmin1 -> Set password error : This is similar to a commonly used password
* iotuser1 -> Set password error : Add another word or two. Uncommon words are better.

*** iotuser10, Password1234, Administrator1 -> allowed**

Fix: comply to best password practices[1,2]

1. GSM Association. Iot security guidelines for network operators version 2.2, Section 5.8.4- Secure IoT Connectivity Management Platform
https://www.gsma.com/iot/wp-content/uploads/2020/05/CLP.14-v2.2-GSMA-IoT-Security-Guidelines-for-Network-Operators.pdf
2. Referring to section 6.11 of GSMA CLP.12 - Never allow a user to utilize a default, weak, or poorly designed password.
https://www.gsma.com/iot/wp-content/uploads/2016/02/CLP.12-v1.0.pdf

Token management

OAuth based authentication not found in several platforms

Token expiry

- Static API token (does not expire), should be revoked for every API user
- Token validity periods from 24 hours to 1 week

Fix: Use standard approach of Oauth and JSON web tokens for authorization and custom validity periods for each type of IoT use-case

1. 3GPP. Security aspects of Machine-Type Communications (MTC) and other mobile data applications communications enhancements. Technical Specification (TS) 33.187. Section 4.7 Requirements on T8 reference point https://www.etsi.org/deliver/etsi_ts/133100_133199/133187/16.00.00_60/ts_133187v160000p.pdf

2. 3GPP. Security aspects of Common API Framework (CAPIF) for 3GPP northbound APIs. Technical Specification (TS) 33.122, 3rd Generation Partnership Project.

Lack of rate limiting for API requests

Only 2 platforms have rate-limits for API requests

- Test: Sending 400 valid GET/POST requests in short period
  - Using same IP address and user account for all requests
- No backoff period or IP ban was observed from the API gateway
  - Did not receive any HTTP response like : `429 Too Many Requests`
- Some providers specify rate-limits in user manuals, but in practice they are unavailable
- **Fix: Rate limiting policies with random/exponential back-off timers**

Endpoint	API Rate Limit
Authorization: /oauth	no rate limit
SIM Management: /sims	no rate limit
Order Management: /orders	100 requests per IP address per 5 minutes
Product Information: /products	100 requests per IP address per 5 minutes
Support Management: /support	100 requests per IP address per 5 minutes

Verbose error messages

Easy user enumeration via probing with IMSI/ICCID/IMEI

- Attacker can find existing and non-existing IMSIs registered on the platform/database from the different API error responses
- Fix: The error can be very generic, such as, *unauthorized*.

Script Injection

Code Injection successful into 6 platforms

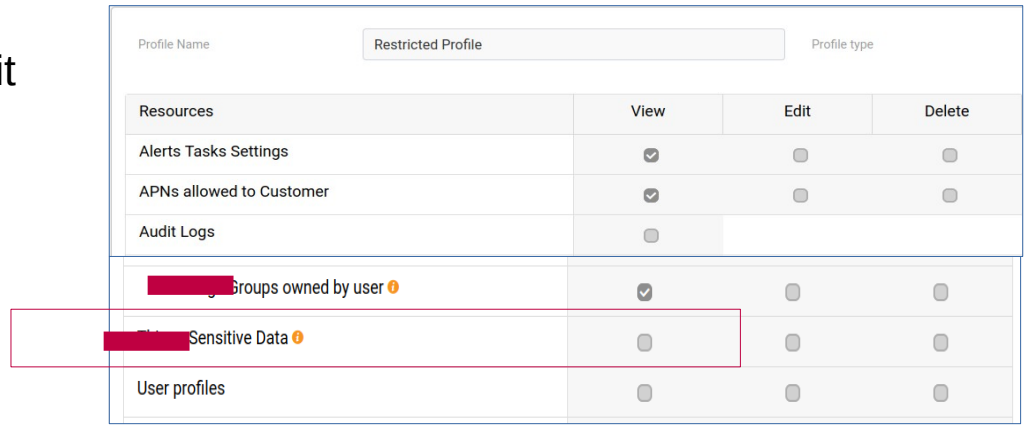
- Many APIs accept malicious strings, characters
- Accepts SQL commands and scripts
  - <script>Alert(123)</script>
- Causes a persistent XSS and execution attacks
- The injected values gets stored in backend DB
  - Command called by another backend process
  - Used in the customer management web application
- **Fix: strict input sanitization for each and every parameter**

ICCID 89999112400711024772	<script>alert(1);</script>a	default network for AP
ICCID 89999112400711024780	<script>alert(1);</script>a	default network for AP
ICCID 89999112400711024798	<script>alert(1);</script>a	default network for AP
ICCID 89999112400711024806	<script>alert(1);</script>a	default network for AP
<script>alert(1;)</script>ICCID 8999911240071102	<script>alert(1);</script>a	default network for AP
ICCID 89999112400711024830	<script>alert(1);</script>a	default network for AP
ICCID 89999112400711024848	<script>alert(1);</script>a	default network for AP
ICCID 89999112400711024855	<script>alert(1);</script>a	default network for AP
ICCID 89999112400711024863	<script>alert(1);</script>a	default network for AP

Access control misconfiguration

Sensitive data and functions misconfigured

- Discrepancies between API documentation and software implementation.
- **Admin-only** API/functions like send-binary-data, update billing information are made available to *API user*
- Malicious insider or employee can exploit
- Restricted profile failed in practice
  - (even though view permissions unchecked by administrator)



Firewall vs secure API-by-design

Database and software information exposed via error messages: Couchbase, Jboss

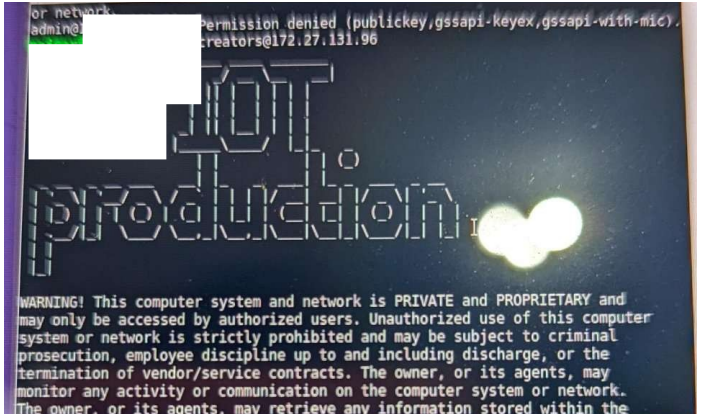
- Platform deployment details can be identified such as cloud provider and firewall name etc.
- Error responses from both firewalls and API parsing framework
- Firewall overlooks detecting injection on certain user-controlled parameters (trusted user)
  - Injection in IMSI, ICCID detected, whereas other like Alias and organization name stealthy
  - Inconsistent security setting: Injection over APIs failed – don't worry there is web interface portal to inject

Vulnerabilities in IoT service platforms (5)
(authorization, data leak, injection and code execution)

Internal node exposure

Device-side open issues

- IP scan from IoT devices exposes other user's internal SSH ports/interface
- Lateral movement allowed by the IoT gateway node firewall
- SSH Login attempt are made to an internal IoT gateway node
- Forged attacker can launch a bruteforce
- **Fix: configuration control and reduce exposure**



Private identifiers used in apps domain

ICCID, IMEI, and IMSI exposed outside of 3GPP domain (can be SUPI in 5G)

- To access/indicate the SIM cards and IoT devices; convenient for developers and API users
- Violates 3GPP privacy requirement [1] for Machine type communications using exposure services
- Enables user/device enumeration
- Fix: an identifier like General Purpose Subscriber Identifier (GPSI[2]) or custom identifier.
  - An alphanumeric proprietary id and its mapping to IMSI/ICCID is known only to the provider/operator.

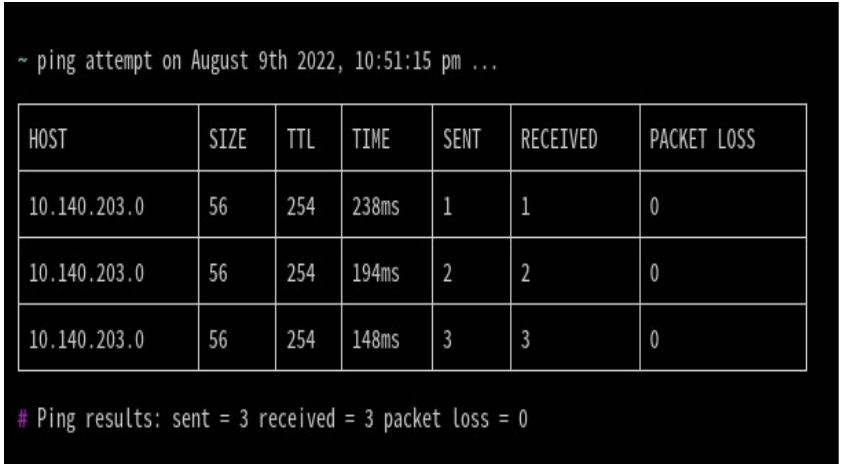
IMSI	ICCID
853428291819393	482012832923284480
853428291819394	482012832923284482
853428291819395	482012832923284484
853428291819396	482012832923284486

1. 3GPP. Security aspects of Machine-Type Communications (MTC) and other mobile data applications communications enhancements. Technical Specification (TS) 33.187. Section 4.7 Requirements on T8 reference point https://www.etsi.org/deliver/etsi_ts/133100_133199/133187/16.00.00_60/ts_133187v160000p.pdf
2. 5G; Procedures for the 5G System (5GS) (3GPP TS 23.502 version 15.4.1 Release 15)

Broken authorization while sending downlink message

IP address not validated for "*`send-downlink-data`*"

- Attacker can talk to arbitrary IoT devices in the network
  - e.g., in *`/ping`* API
- IoT device responds to ping operation (IPV4) with a ping reply. (up to 200 devices available)
- Port scans can be performed on target device and inject malicious IP packets into the device.
- increase data consumption over radio interface, and charge to victim's account
- energy drain for low-powered IoT devices, and eventually a DoS.
- **Fix: Strict authorization checks for every API parameter/object level.**



~ ping attempt on August 9th 2022, 10:51:15 pm ...

HOST	SIZE	TTL	TIME	SENT	RECEIVED	PACKET LOSS
10.140.203.0	56	254	238ms	1	1	0
10.140.203.0	56	254	194ms	2	2	0
10.140.203.0	56	254	148ms	3	3	0

Ping results: sent = 3 received = 3 packet loss = 0

Private details of SIM and customer are exposed over webhook

SIM PIN, PUK and subscriber details exposed

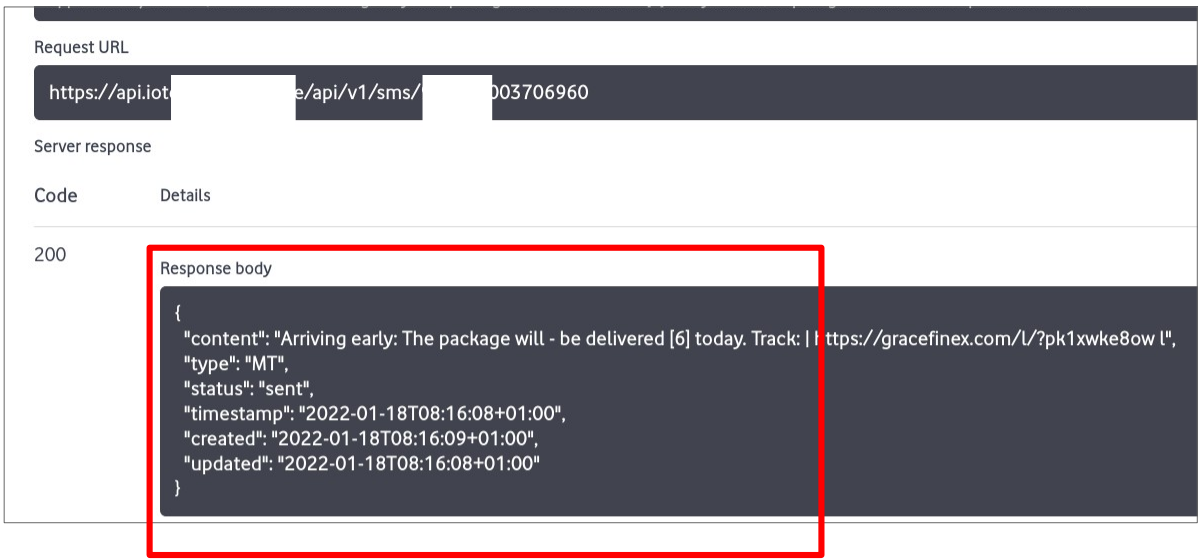
- While sending SMS using API, the HTTP response sent to a user-defined Webhook (URL) exposes user's private information
  - Private info: Billing details, subscriber plan and many other sensitive details linked to SIM card (identities, PIN1,PIN2, PUK, Opc, SQN, location, HLR ID).
  - Providers argue that some business cases require such sensitive information in the response
- BGP hijacking[1] to steal all the data exposed over a HTTP Webhook
- **Fix: use only HTTPS webhook, and eliminate sending SIM card private info to customer over the APIs**

1. What is bgp hijacking? https://www.cloudflare.com/ko-kr/learning/security/glossary/bgp-hijacking

Malware propagation inside user plane

Allows malicious data[1] (popular malware and binaries)

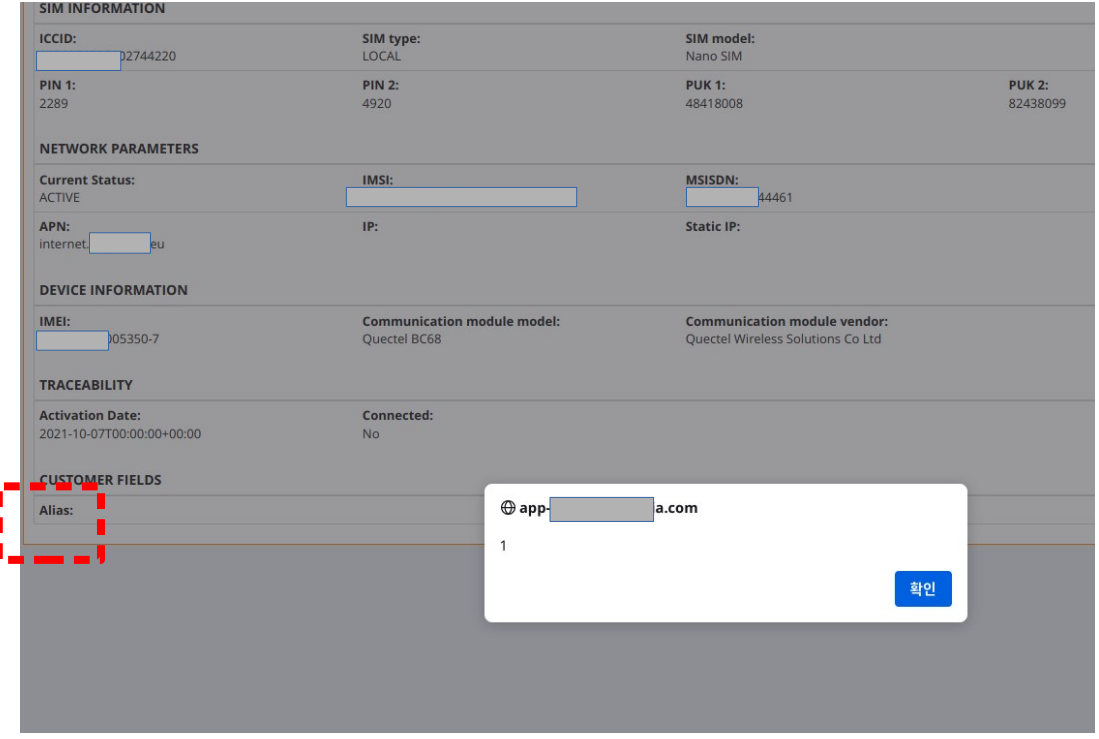
- Inside 100 SMS, and IP payload
- malware, spam and phishing content is allowed to propagate inside the mobile network and delivered to IoT devices
- No spam detection filters
- Malware[1] can be sent to arbitrary IoT devices with authorization bypass
- Operators argue that SMS and data inspection is against law in some countries



1. https://www.kaspersky.com/resource-center/threats/sms-attacks

XSS execution

- Code Injection
  - Via API on the service platform
  - e.g., the *Alias* is an alternate name of the SIM card and can be given as input from the user
  - Allows script and arbitrary code
- Code Execution
  - via the *IoT connectivity management platform*
  - *Alias* parameter is shared between both platforms and inject script is triggered on the web interface leading to code execution
  - With authorization bypass, attacker can inject code into another customer's platform and trigger it



Summary of security analysis

- Oauth and TLS is not widely practiced among platforms (5/9)
- Only 2 out of 9 IoT platforms are not affected with serious vulnerabilities and API risks
- Attacker can easily obtain access to IoT service platforms and service APIs with forged identity
- IMSI is exposed outside of 3GPP network, same practice may apply for 5G IMSI (SUPI)
- Lack of rate-limits, strong password policies
- Script/code injection vulnerability found in many platforms, and is missed in a internal pen-testing
- SMS and IP content inspection is not present in mobile and IoT networks
- Authorization vulnerabilities have serious consequences

Responsible disclosure

- Responsibly disclosed our findings to the affected IoT service providers and operators
- Received positive acknowledgments and confirmation of the vulnerabilities, and appreciation for our efforts to make the exposure services more secure.
- Operators confirmed that our testing methods never caused any damage to their services and infrastructure.
- Three of the tested service providers indicated that, injection vulnerabilities discovered in our findings remained hidden during their internal penetration testing exercise.
- We do not disclose any of the API and provider/operator names

Key takeaways

- 5G > 4G > 3G > 2G. Closed gardens shift towards a generalized, commoditized technology – clouds, APIs, SDN, VMs, Dockers
  - Attracts more bad and powerful adversaries, plenty of tools/resources to attack
- Standard Oauth and TLS mechanisms wont help achieve full API security
- Insecure API Design/Configuration **=** risk for mobile core, IoT devices and industries
- Firewalls won't always help – need security-by-design and testing into CI/CD
  - Inconsistent security settings in among APIs and web apps
- Telecom exposure API risks are new: application **logic flaws** – require rigorous application specific tests (not using general API security scanners)
- **Telecom API top 10** to help developers understand risks : Information entering & leaving the network

Questions? Concerns? Comments?

Can also write me on:

(altaf.shaik@fastiot.org)