Aalto University

Vulnerabilities in the eSIM download protocol

Presenters
Abu Shohel Ahmed, Aalto University
Tuomas Aura, Aalto University

Joint work with
Aleksi Peltonen, CISPA
Mohit Sethi, Kone and Aalto University

Who are we? our story

Shohel Ahmed,
security researcher

Hey, I am working on implementing eSIM download protocol

How do I know the protocol is secure ?

We could apply formal verification to find out

Let’s do it

Tuomas Aura,
Professor

Mohit Sethi

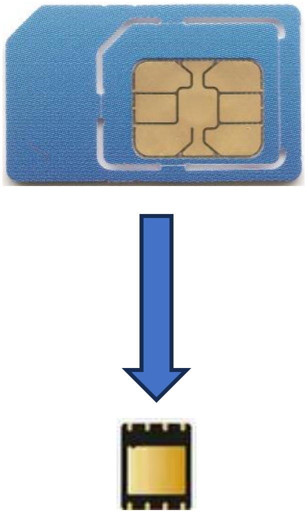
Aleksi Peltonen

Talk outline

1. eSIM and the Consumer Remote SIM Provisioning (RSP) protocol
2. Research methodology
3. Discovered vulnerabilities
   - What did we find
   - Why does it matter
   - What can we do about it

From SIM to eSIM

- SIM contains credentials for authenticating a mobile network subscriber



- eSIM replaces removable SIM with downloadable SIM profiles
  - Installed into an embedded secure chip (eUICC)
  - Managed from phone settings or an app

Consumer eSIM user experience

Activation code approach

- User inputs SM-DP+ server address and activation code
- Manual entry or QR code

LPA:1$sm-dp.example.com$
95A9CB26933E7f1C

SM-DP+ address

Secret one-time code

Default server approach

- eUICC or app has a default SM-DP+ server address
- d to know the order profile

EID:89049032000001000000

Consumer eSIM user experience

Activation code approach

- User inputs SM-DP+ server address and activation code
- Manual entry or QR code

LPA:1$sm-dp.example.com$95A9CB26933E7f1C

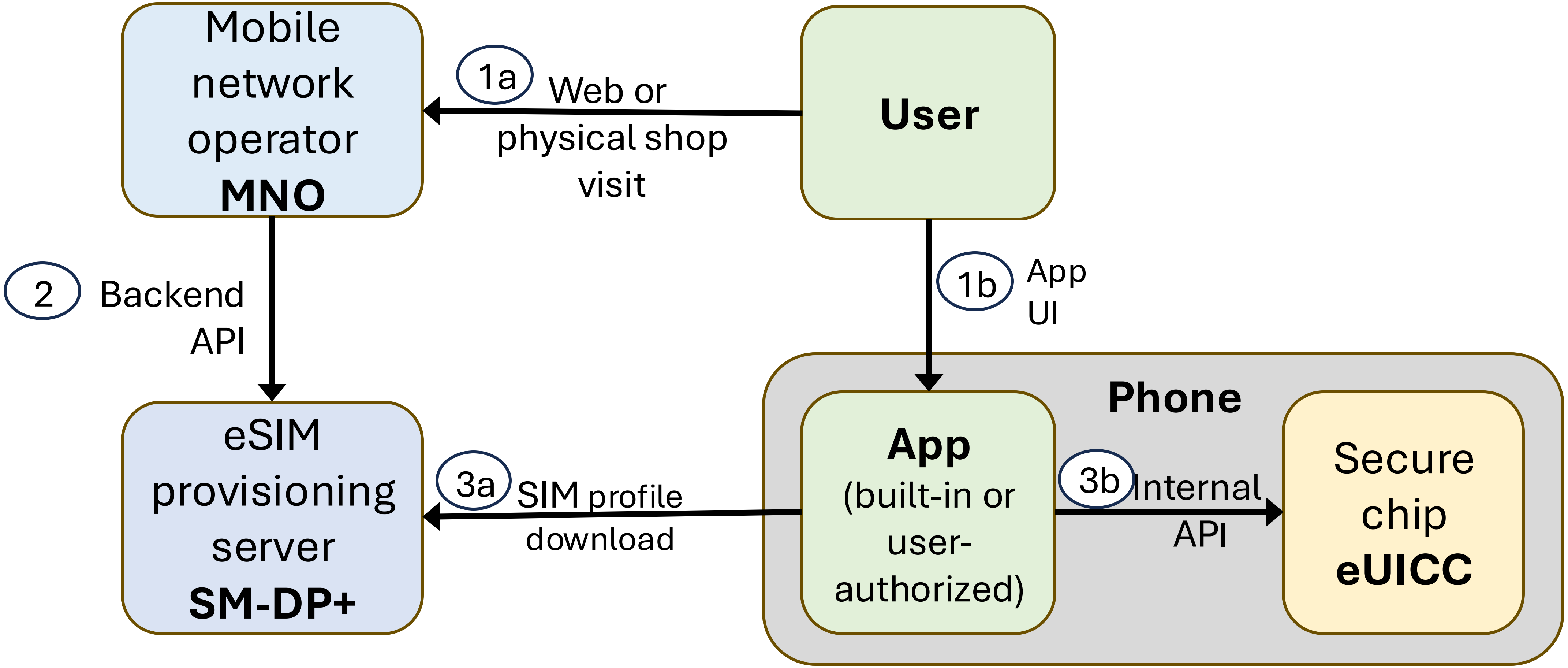
Default server approach

- eUICC or app has a default SM-DP+ server address
- Operator need to know the device EID to order profile

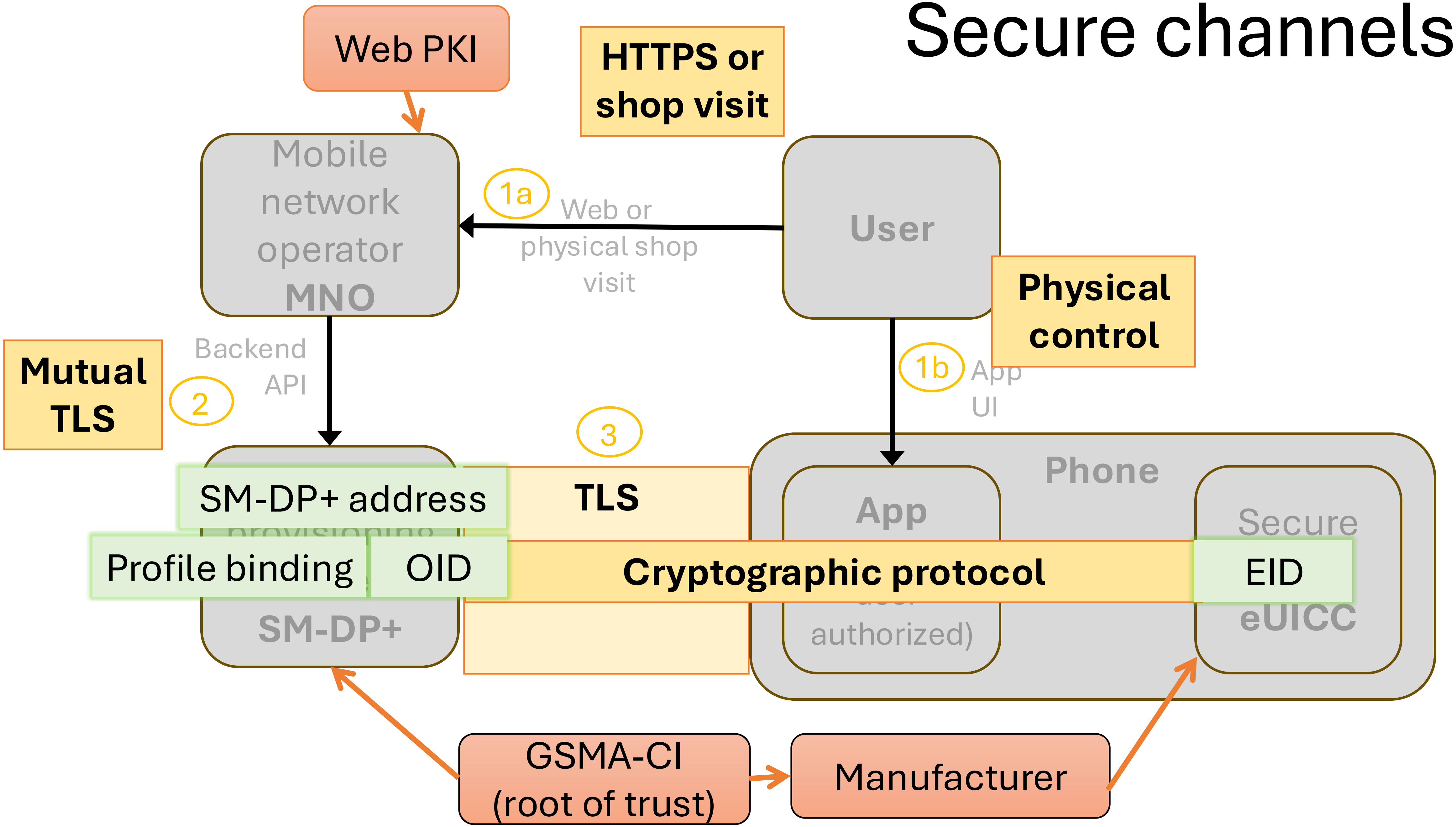
EID:89049032000001000000044883019442

Identifies the device, privacy sensitive data

How does it work under-the-hood?



Secure channels



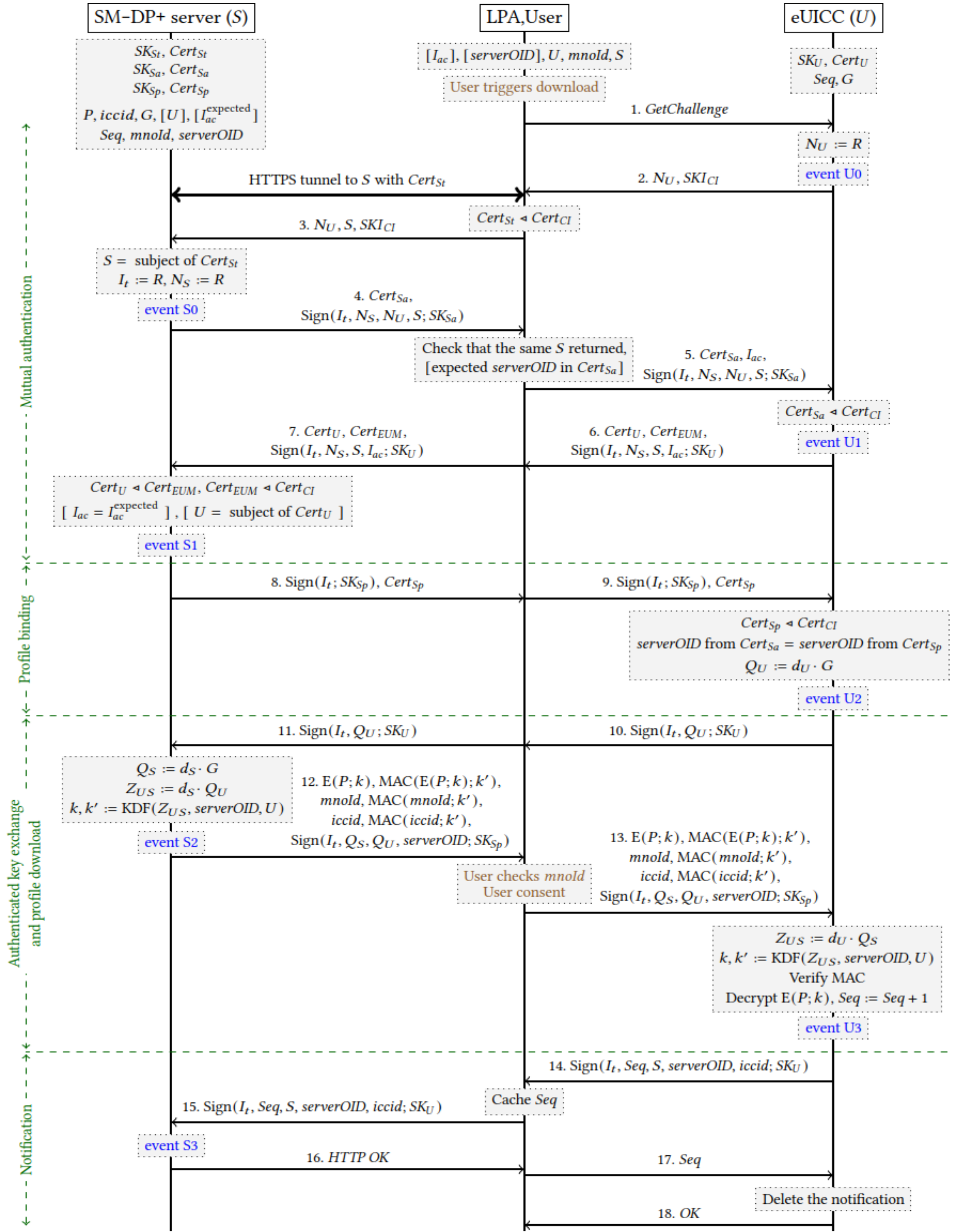
Research methodology

Is the eSIM download protocol secure?

How does the eSIM download protocol work?
What are the security goals?
Does the protocol meet the security goals?
...

Research methodology

1. Protocol description as message sequence chart



Research methodology

1. Protocol description as message sequence chart
2. Formal model of the protocol

Participants of the protocols

```
(* ===== MAIN PROCESS ===== *)
process
  (** == CA == **)
  let PK_CI = pk(SK_CI) in
  out(c, PK_CI);

  (** == Honest processes == **)
    !MNO(PK_CI)
  | !SMDP(PK_CI)
  | !(new U:Id_t; out(c, U);
      new LPA2EUICC:channel;
      LPA(LPA2EUICC,PK_CI,U) |
EUICC(LPA2EUICC,PK_CI,U)
     )

  (** == Base attacker model == **)
  | A_ORDER(PK_CI)
  | !A_TLS()
  | (new U:Id_t; out(c, U);
     event OWNER(AttackerUserId,U);
     new LPA2EUICC:channel; out(c, LPA2EUICC);
     A_EUICC(LPA2EUICC,PK_CI,U)
    )
```

Research methodology

1. Protocol description as message sequence chart
2. Formal model of the protocol
3. Partial compromise scenarios

- Base-case: all participants are honest, network is the adversary
- Partial compromise scenarios
  - Compromised participants
  - Compromised outsiders
  - Compromised channels

Research methodology

1. Protocol description as message sequence chart
2. Formal model of the protocol
3. Partial compromise scenarios
4. Test the security goals with model checker

Result summary

- 600 verification targets
- No failures when all design assumptions hold

Default-server approach

Partial compromise scenario	Authentication goals											Secrecy goals			
	A	B	B′	C	D	E	F	G	I	J	K	W	X	Y	Z
1: —	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
2: server	X^{2}	$X^{2,c}$	✓	X^{2}	X^{c}	X^{2}	X^{2}	X^{2}	X^{2}	X^{2}	✓	✓	X^{2}	✓	X^{2}
3: eUICC	✓	X^{4}	✓	O^{d}	X^{4}	✓	✓	X^{4}	✓	✓	✓	X^{4}	✓	X^{4}	✓
4: LPA	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
5: 2nd server	O^{3}	O^{c}	✓	O^{3}	O^{c}	O^{3}	O^{3}	✓	O^{3}	O^{3}	✓	✓	O^{3}	✓	O^{3}
6: 2nd eUICC	✓	✓	✓	O^{d}	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
7: 2nd MNO	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
8: order as user	✓	✓	X^{7}	✓	✓	✓	✓	X^{7}	✓	✓	X^{7}	✓	✓	✓	✓
9: order for eUICC	✓	✓	X^{a}	✓	✓	✓	✓	X^{a}	✓	X^{a}	X^{a}	✓	✓	✓	✓

Attacker owns some eUICCs in all the scenarios 1–9. Client-side goals are gray. No security is expected in Scenarios 2-3.

Activation-code approach

Partial compromise scenario	Authentication goals											Secrecy goals			
	A	B	B′	C	D	E	F	G	I	J	K	W	X	Y	Z
1: —	✓	✓	O^{1}	✓	✓	✓	✓	O^{1}	✓	✓	O^{1}	✓	✓	✓	✓
2: server	X^{2}	$X^{2,c}$	$X^{1,f}$	X^{2}	X^{c}	X^{2}	X^{2}	$X^{1,2,f}$	X^{2}	X^{2}	$X^{1,f}$	✓	X^{2}	✓	X^{2}
3: eUICC	✓	X^{4}	$X^{1,6}$	O^{d}	X^{4}	O^{e}	O^{e}	$X^{1,4,6}$	O^{e}	O^{e}	$X^{1,6}$	X^{4}	✓	X^{4}	✓
4: LPA	✓	✓	$X^{1,9}$	✓	✓	✓	✓	$X^{1,9}$	✓	X^{9}	$X^{1,9}$	✓	✓	✓	✓
5: 2nd server	O^{3}	O^{c}	O^{1}	O^{3}	O^{c}	O^{3}	O^{3}	O^{1}	O^{3}	O^{3}	O^{1}	✓	O^{3}	✓	O^{3}
6: 2nd eUICC	✓	O^{5}	O^{1}	O^{d}	O^{5}	✓	✓	$O^{1,5}$	✓	✓	O^{1}	O^{5}	✓	O^{5}	✓
7: 2nd MNO	✓	✓	O^{1}	✓	✓	✓	✓	O^{1}	✓	✓	O^{1}	✓	✓	✓	✓
8: order as user	✓	✓	$X^{1,7}$	✓	✓	✓	✓	$X^{1,7}$	✓	✓	$X^{1,7}$	✓	✓	✓	✓
10: code leaks	✓	✓	$X^{1,8}$	✓	✓	✓	✓	$X^{1,8}$	✓	✓	$X^{1,8}$	✓	✓	✓	✓
11: code spoofed	✓	✓	$X^{1,b}$	✓	✓	✓	✓	$X^{1,b}$	✓	X^{b}	$X^{1,b}$	✓	✓	✓	✓

Attacker owns some eUICCs in all the scenarios 1–11. Client-side goals are gray. No security is expected in Scenarios 2-3.

Result summary

- 600 verification targets
- No failures when all design assumptions hold
- Found failures in partial compromise scenarios

Default-server approach

Partial compromise scenario	Authentication goals											Secrecy goals			
	A	B	B′	C	D	E	F	G	I	J	K	W	X	Y	Z
1: —	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
2: server	X[2]	X[2,c]	✓	X[2]	X[c]	X[2]	X[2]	X[2]	X[2]	X[2]	✓	✓	X[2]	✓	X[2]
3: eUICC	✓	X[4]	✓	O[d]	X[4]	✓	✓	X[4]	✓	✓	✓	X[4]	✓	X[4]	✓
4: LPA	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
5: 2nd server	O[3]	O[c]	✓	O[3]	O[c]	O[3]	O[3]	✓	O[3]	O[3]	✓	✓	O[3]	✓	O[3]
6: 2nd eUICC	✓	✓	✓	O[d]	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
7: 2nd MNO	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
8: order as user	✓	✓	X[7]	✓	✓	✓	✓	X[7]	✓	✓	X[7]	✓	✓	✓	✓
9: order for eUICC	✓	✓	X[a]	✓	✓	✓	✓	X[a]	✓	X[a]	X[a]	✓	✓	✓	✓

Attacker owns some eUICCs in all the scenarios 1–9. Client-side goals are gray. No security is expected in Scenarios 2-3.

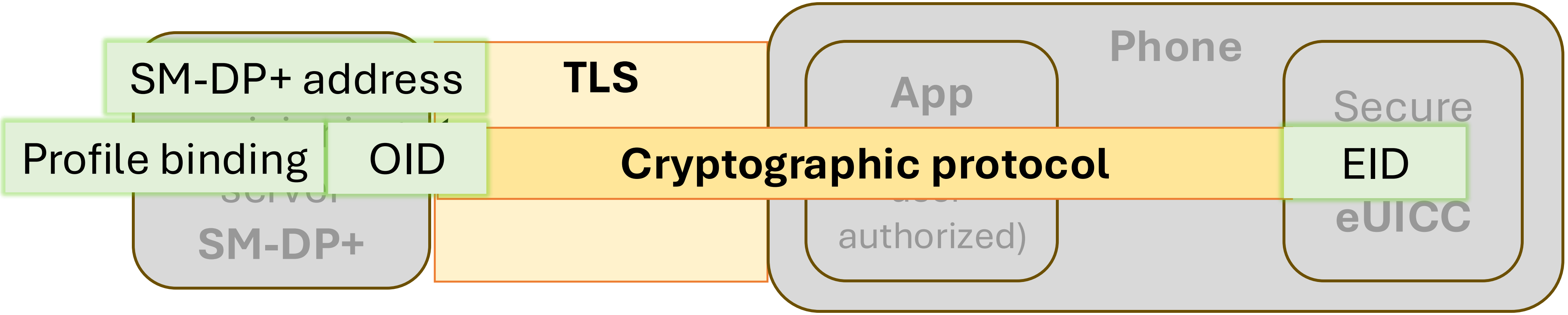
Activation-code approach

Partial compromise scenario	Authentication goals											Secrecy goals			
	A	B	B′	C	D	E	F	G	I	J	K	W	X	Y	Z
1: —	✓	✓	O[1]	✓	✓	✓	✓	O[1]	✓	✓	O[1]	✓	✓	✓	✓
2: server	X[2]	X[2,c]	X[1,f]	X[2]	X[c]	X[2]	X[2]	X[1,2,f]	X[2]	X[2]	X[1,f]	✓	X[2]	✓	X[2]
3: eUICC	✓	X[4]	X[1,6]	O[d]	X[4]	O[e]	O[e]	X[1,4,6]	O[e]	O[e]	X[1,6]	X[4]	✓	X[4]	✓
4: LPA	✓	✓	X[1,9]	✓	✓	✓	✓	X[1,9]	✓	X[9]	X[1,9]	✓	✓	✓	✓
5: 2nd server	O[3]	O[c]	O[1]	O[3]	O[c]	O[3]	O[3]	O[1]	O[3]	O[3]	O[1]	✓	O[3]	✓	O[3]
6: 2nd eUICC	✓	O[5]	O[1]	O[d]	O[5]	✓	✓	O[1,5]	✓	✓	O[1]	O[5]	✓	O[5]	✓
7: 2nd MNO	✓	✓	O[1]	✓	✓	✓	✓	O[1]	✓	✓	O[1]	✓	✓	✓	✓
8: order as user	✓	✓	X[1,7]	✓	✓	✓	✓	X[1,7]	✓	✓	X[1,7]	✓	✓	✓	✓
10: code leaks	✓	✓	X[1,8]	✓	✓	✓	✓	X[1,8]	✓	✓	X[1,8]	✓	✓	✓	✓
11: code spoofed	✓	✓	X[1,b]	✓	✓	✓	✓	X[1,b]	✓	X[b]	X[1,b]	✓	✓	✓	✓

Attacker owns some eUICCs in all the scenarios 1–11. Client-side goals are gray. No security is expected in Scenarios 2-3.

What did we find

Observation 1: dependence on TLS



- TLS is great. What is the problem?
  - Defense in depth or privacy layer vs critical component
  - Front-end API server or TLS gateway is less secure than we expect from the provisioning server
  - Trust anchor should be GSMA-CI, but vendors prefer web PKI
- Ok, what if TLS fails?

Vulnerability 1: server OID not known

Activation code: LPA:1$sm-dp.example.com$
95A9CB26933E7f1C$1.3.6.1.4.1.31746
Default server EID: 89049032000001000000044883019442

Unique SM-DP+
server identifier

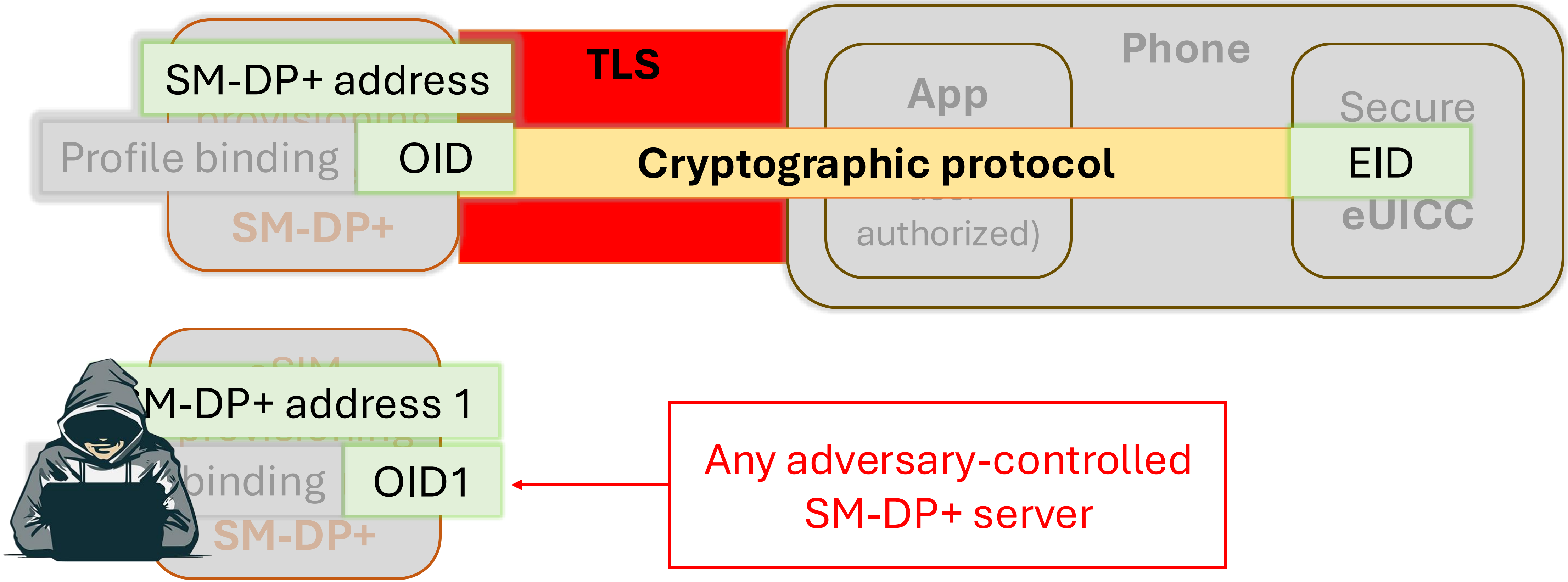
Vulnerability 1: server OID not known

Activation code: LPA:1$sm-dp.example.com$
95A9CB26933E7f1C$1.3.6.1.4.1.31746
Default server EID:89049032000001000000044883019442

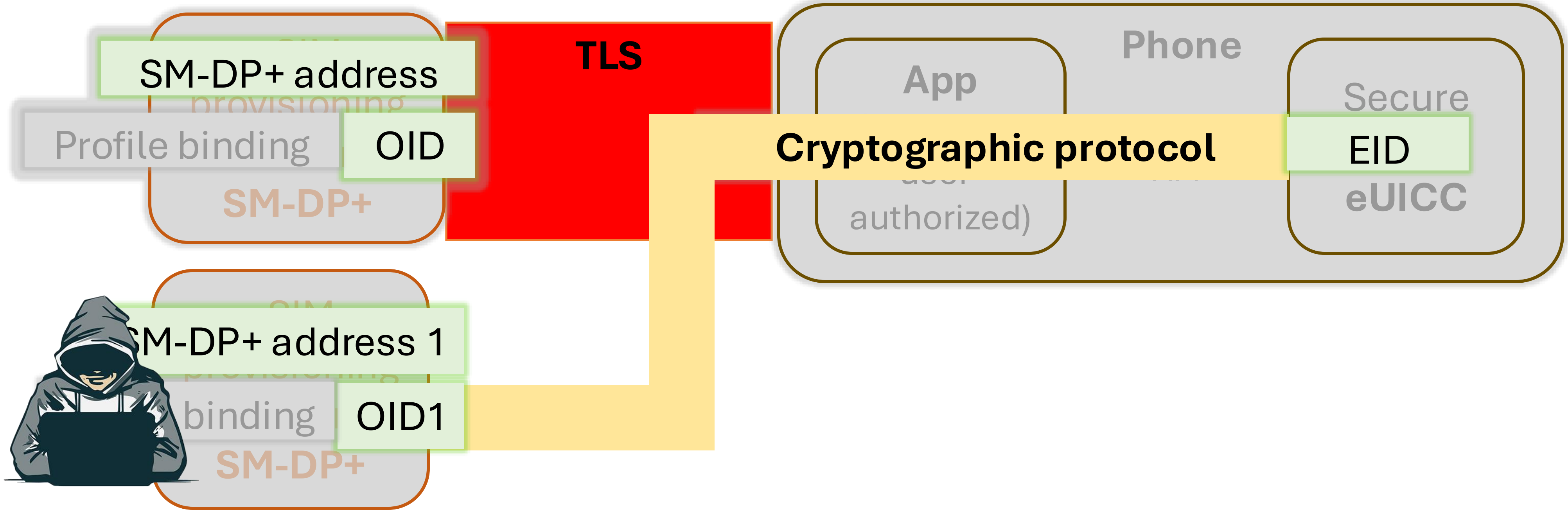
App and eUICC may lack knowledge of the SM-DP+ server OID

- Communicating the OID out-of-band with activation-code is optional
- Input not supported by app user interfaces
- Not specified for the default-server approach

Vulnerability 1: server OID not known



Vulnerability 1: server OID not known



Becomes a problem if TLS to the SM-DP+ server is compromised

→ Adversary who controls any SM-DP+ server in the world can issue fake SIM profiles to any subscriber of any MNO

Vulnerability 2 : EID not known

Activation code: LPA:1$sm-dp.example.com$
95A9CB26933E7f1C
EID:89049032000001000000044883019442

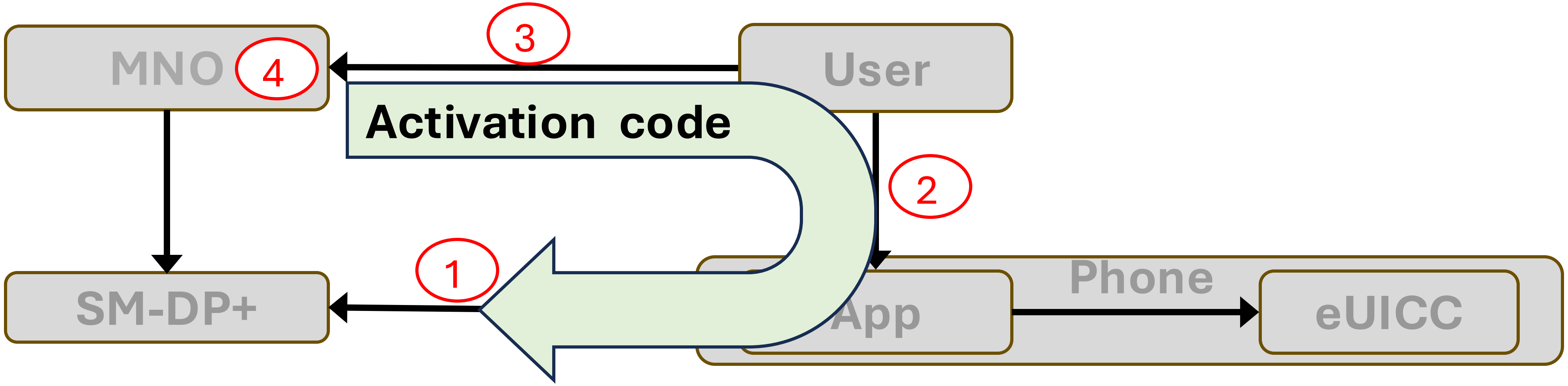
Profile bound to one-time secret

In the activation code approach, SM-DP+ server usually lacks a-priori knowledge of the EID

Theft of activation codes

Ways activation code can leak:

1. TLS from mobile to SM-DP+ path
2. User to App path (e.g., sloppy user, insecure app)
3. User to MNO path
4. MNO processes



Vulnerability 2 : EID not known

Activation code: LPA:1$sm-dp.example.com$
95A9CB26933E7f1C$1.3.6.1.4.1.31746
EID:89049032000001000000044883019442

- Activation code leaks ➔ adversary can steal the SIM profile
- If adversary has the private key of any eUICC in the world, adversary can also get the profile and the secret key in it

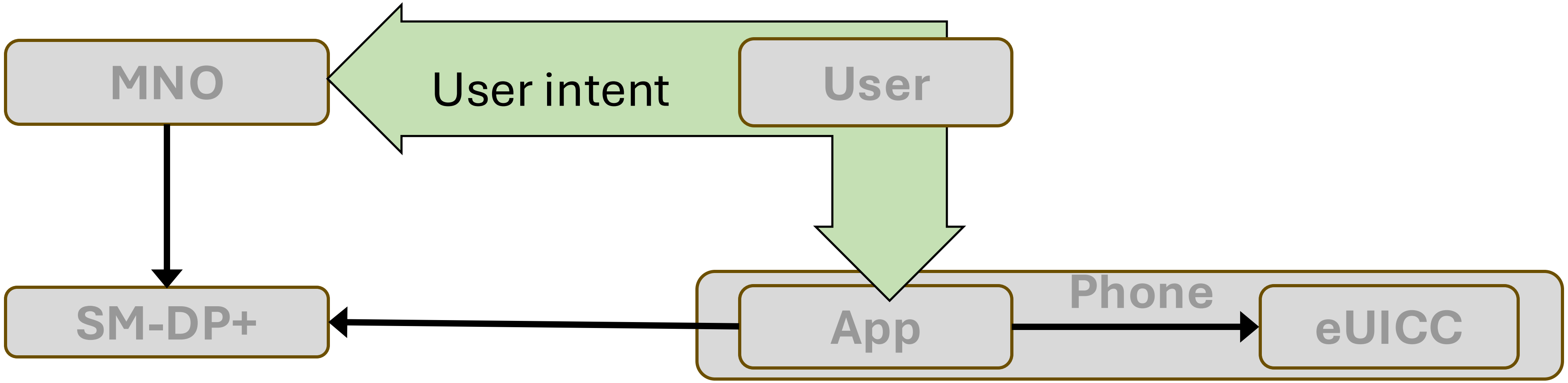
Lessons for protocol design

- Authentication without a-priory knowledge of the identifier
  - Certificate proves the entity class (SM-DP+ or eUICC) but not the individual identity ➔ Attacker can substitute a different one

- Dependence on the TLS tunnel leads to vulnerabilities when combined with other weaknesses
  - Dependency is easy to remove in the default server approach
  - Major redesign required in the activation code approach.

Observation 2: difficulty in verifying user intent

- User goes to the operator (web) shop, receives a QR code, and scans it with the eSIM app
- What is (or should be)communicated between the user and MNO?
- What if the secrecy or integrity is compromised?



Vulnerability 3: verifying user identity

Often, no reliable method for verifying user identity when subscribing

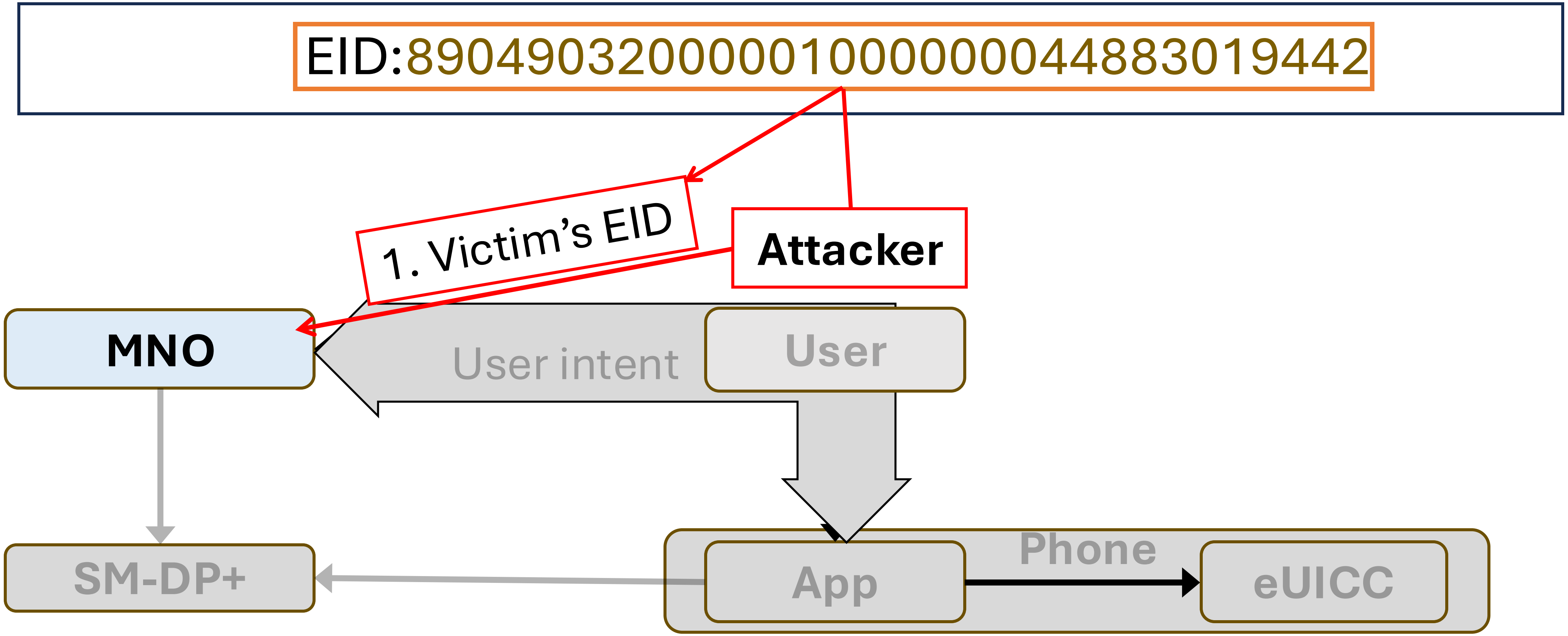
Identity fraud in ordering ➔ Adversary can steal the victim's SIM profile

Consequences **similar to SIM swapping**

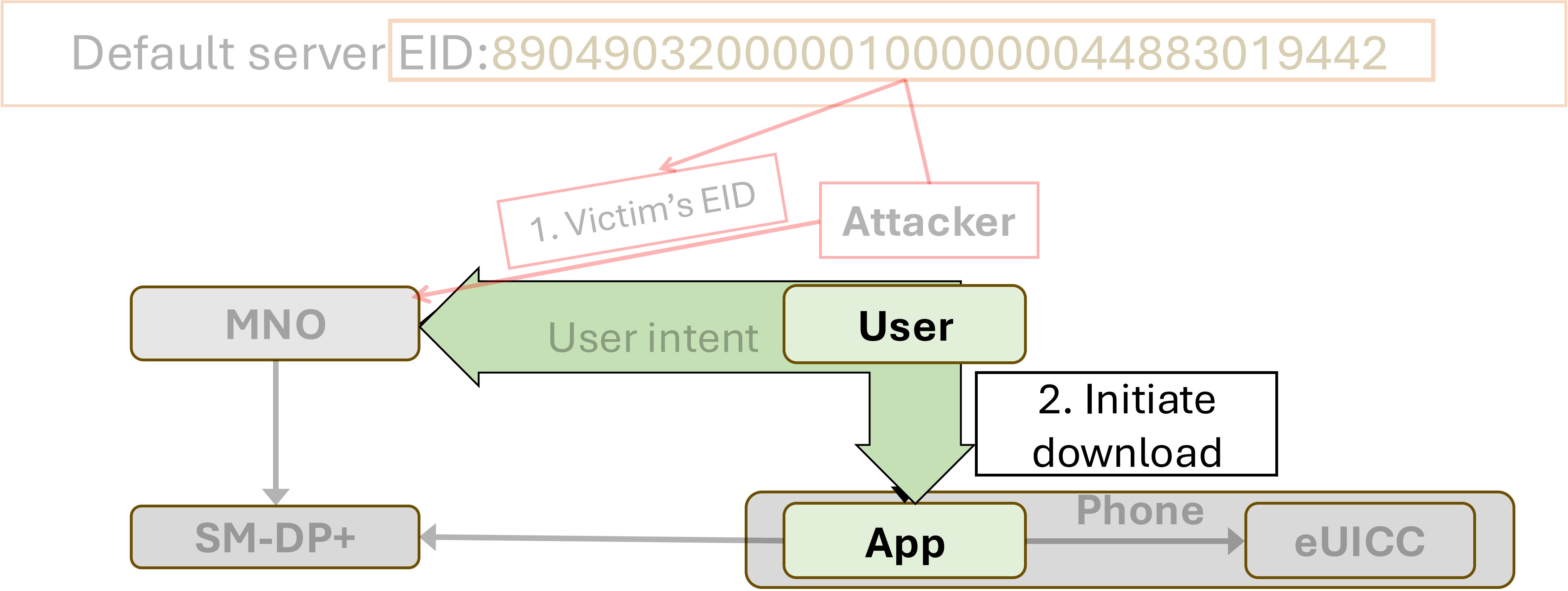
- May breaks 2FA, enables further fraud

Vulnerability 4: verifying eUICC ownership

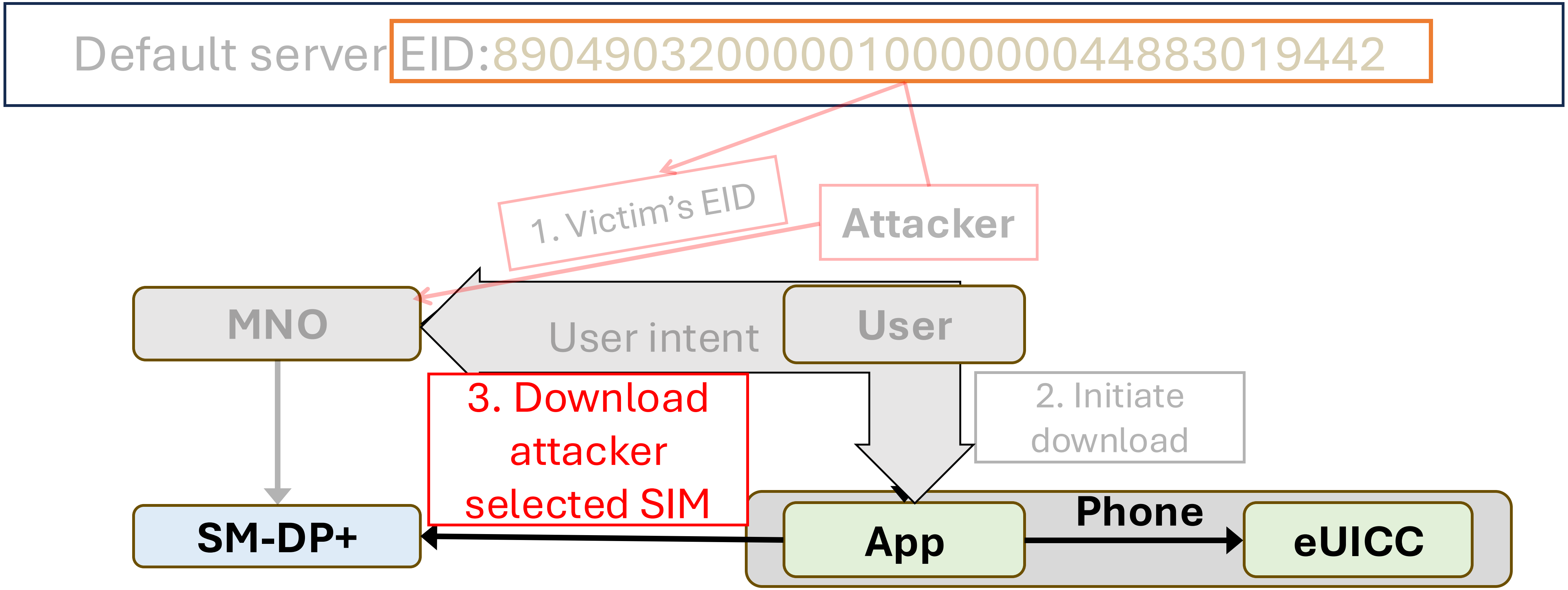
- How does MNO verify the eUICC ownership/possession in the Default server approach?



Vulnerability 4: verifying eUICC ownership



Vulnerability 4: verifying eUICC ownership



→ Victim tricked into using the adversary's mobile subscription

Potential consequences

Adversary's SIM profile is in the victim's phone. So what?

- Leakage of mobile metadata
  - Call and message logs, billing information, roaming history, location services
- Text and call capture with multi-SIM
  - Adversary has a multi-SIM subscription and gets one of the SIM profiles into the victim's phone ➔ Receives copies of text messages and can answer calls
- Data capture with home routing
  - Spies can use this to divert all mobile data from the device to their country

Lessons: what the operator should check

1. User identity check: make the order for the correct subscriber
2. Ownership verification: make the order for the correct eUICC (EID)

- Not easy to implement in practice

Notifying GSMA

- We notified GSMA's eSIM working group
- GSMA acknowledges our finding that the RSP protocol is secure between honest entities against network adversary
- For attacks performed with compromised endpoints, (e.g., SM-DP+ server and eUICC), GSMA places importance on eSIM certification process as mitigation control
- For attacks performed by compromising user intent, GSMA points these are out of specification scope

Key Takeaways: why should you care

- Protocol designer: Formal verification is an effective way to identify security weakness
- Red teams: Don’t just target products or websites – also target specifications as they affect all products based on them
- Specification body: Telco is not a closed world! Don’t assume everyone in the world is a good guy.

Questions ?

- AS Ahmed, A Peltonen, M Sethi, T Aura. Security Analysis of the Consumer Remote SIM Provisioning Protocol. ACM Transactions on Privacy and Security 27 (3), https://dl.acm.org/doi/pdf/10.1145/3663761
- Model in GitHub: https://github.com/peltona/rsp_model
- Contact
  - abu.ahmed@aalto.fi https://www.linkedin.com/in/shohel
  - tuomas.aura@aalto.fi https://www.linkedin.com/in/tuomas-aura-94749aa4/