Advanced Network Security

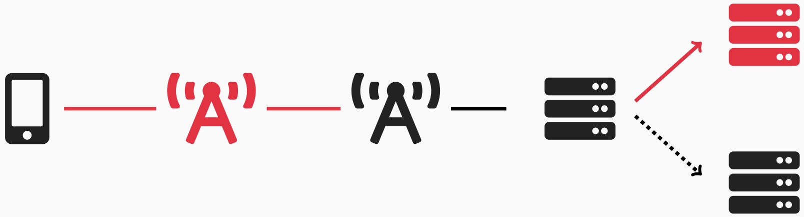
Lecture 9: Impersonation Attack

Harald Vranken, Katharina Kohls

April 26th, 2021

Open University Nijmegen
Radboud University Nijmegen

Last time. . .



- Website Fingerprinting and Identification attack
- User Data Redirection
  - No integrity protection for user plane
  - Malleable encryption
  - DNS spoofing through XOR manipulation

IMP4GT: IMPersonation Attacks in 4G NeTworks

IMP4GT: IMPersonation Attacks in 4G NeTworks

David Rupprecht
Ruhr University Bochum
david.rupprecht@rub.de

Katharina Kohls
Ruhr University Bochum
katharina.kohls@rub.de

Thorsten Holz
Ruhr University Bochum
thorsten.holz@rub.de

Christina Pöpper
NYU Abu Dhabi
christina.poepper@nyu.edu

***Abstract*—Long Term Evolution (LTE/4G) establishes mutual authentication with a provably secure Authentication and Key Agreement (AKA) protocol on layer three of the network stack. Permanent integrity protection of the control plane safeguards the traffic against manipulations. However, missing integrity protection of the user plane still allows an adversary to manipulate and redirect IP packets, as recently demonstrated.**

In this work, we introduce a novel cross-layer attack that exploits the existing vulnerability on layer two and extends it with an attack mechanism on layer three. More precisely, we take advantage of the default IP stack behavior of operating systems and show that combining it with the layer-two vulnerability allows an active attacker to impersonate a user towards the network and vice versa; we name these attacks IMP4GT (IMPersonation attacks in 4G neTworks). In contrast to a simple redirection attack as demonstrated in prior work, our attack dramatically extends the possible attack scenarios and thus emphasizes the need for user-plane integrity protection in mobile communication standards. The results of our work imply that providers can no longer rely on mutual authentication for billing, access control, and legal prosecution. On the other hand, users are exposed to any incoming IP connection as an adversary can bypass the provider's firewall. To demonstrate the practical impact of our attack, we conduct two IMP4GT attack variants in a live, commercial network, which—for the first time—completely break the mutual authentication aim of LTE on the user plane in a real-world setting.

I. Introduction

Long Term Evolution (LTE) is the latest widely deployed mobile communication standard and is used by hundreds of millions of people worldwide. The protocol offers high-speed Internet access and packet-based telephony services and has become an integral component of our daily communication. We fundamentally rely on the security of LTE for a variety of applications. The security goals of LTE include, amongst others, mutual authentication, traffic confidentiality, and location privacy; any attack vector undermining these security aims has far-reaching implications to the use of LTE as a communication medium.

In the context of mobile communication, mutual authentication is an important security aim since it ensures that both communication parties (i. e., the user equipment and the network) mutually verify their identities. As the wireless medium is accessible for everyone in the vicinity and identifiers can be easily forged, mutual authentication is essential for building trust between communication parties. Telecommunication providers rely on *user authentication* for accounting, authorization, and the association of data sessions to a legal person. The latter case is of particular importance in prosecution, in which a possible offender is accused of committing a crime via a mobile Internet connection. Additionally, users rely on *network authentication* for the confidentiality of their communication. One important example for missing network authentication is the second mobile network generation GSM (Global System for Mobile Communications): by faking the identity of a legitimate network, an attacker can impersonate the network in GSM and eavesdrop on the communication of the victim.

In contrast to earlier network generations, LTE establishes *mutual* authentication on layer three of the network stack using a provably secure Authentication and Key Agreement (AKA) protocol [6], [8]. Based on this protocol, subsequent encryption ensures the confidentiality of user and control data. Permanent integrity protection, however, is only applied to the control data. A recent study has revealed that missing integrity protection of the user plane on layer two allows to manipulate user data in a deterministic way [40]. More specifically, a layer-two attacker in a Man-in-the-Middle (MitM) position between the phone and the network can introduce undetectable bit flips due to malleable encryption and *redirect* traffic to another destination. While this attack demonstrates the potential consequences of traffic manipulation, it is solely limited to redirecting traffic to another destination.

In this work, we introduce a novel *cross-layer attack concept* that complements the known layer-two vulnerability (i. e., missing integrity protection on the user plane [40]) with exploiting the default IP stack behavior of operating systems on layer three. More precisely, we make use of the reflection mechanism of certain IP packets, which allows us to not only redirect user-plane traffic, but also to create an encryption and decryption oracle that enables an adversary to perform a *full impersonation* of the phone or network on the user plane. We call this concept IMP4GT (IMPersonation in 4G neTworks, pronounced [imˌpæk(t)]). IMP4GT completely breaks the mutual authentication property for the user plane on layer three, as an attacker can send and receive arbitrary IP packets despite any encryption.

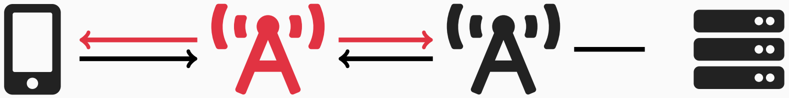
This attack has far-reaching consequences for providers and users. Providers can no longer assume that an IP connection originates from the user. Billing mechanisms can be triggered by an adversary, causing the exhaustion of data limits, and any access control or the providers' firewall can be bypassed. A possible impersonation also has consequences for legal pros-

Network and Distributed Systems Security (NDSS) Symposium 2020
23-26 February 2020, San Diego, CA, USA
ISBN 1-891562-61-4
https://dx.doi.org/10.14722/ndss.2020.24283
www.ndss-symposium.org

Today:
Impersonation Attacks

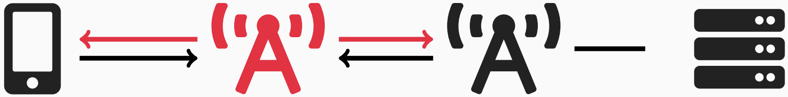
https://imp4gt-attacks.net/

This time. . .



- **Full Impersonation**
- Why is that different from last week?
  - Last time we only altered specific packets
  - Manipulation was limited
  - **Now: Inject and manipulate arbitrary packets**

Why is this worse?



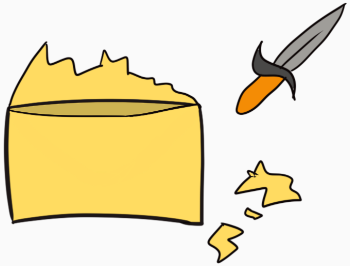
(1) Access a website with the identity (IP address) of the victim $\rightarrow$

(2) Circumvent the provider's firewall and directly access the phone $\leftarrow$

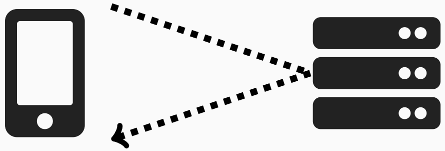
Breaks mutual authentication in both directions

What do we need this time?

Old: Missing Integrity Protection

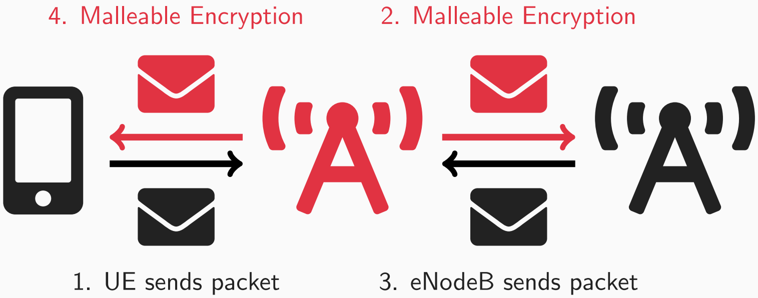


New: Ping Reflection



General Concept

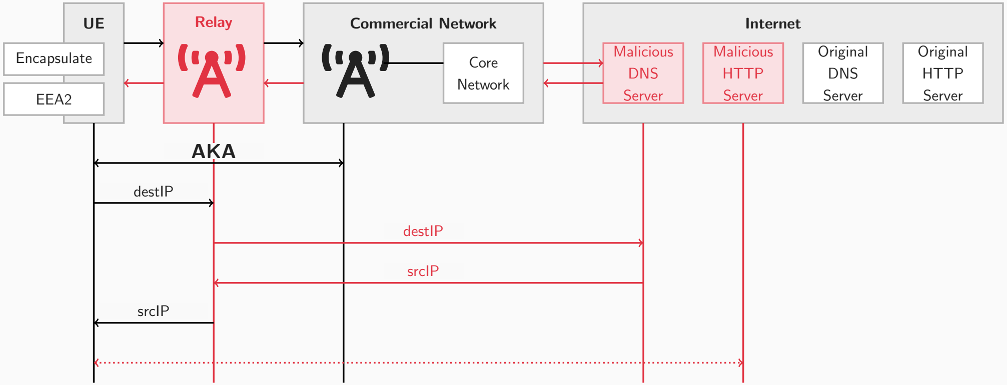
Simple Attack Concept



Can you imagine a problem here?

The adversary is not authenticated and does not have the keys!

Mutual Authentication

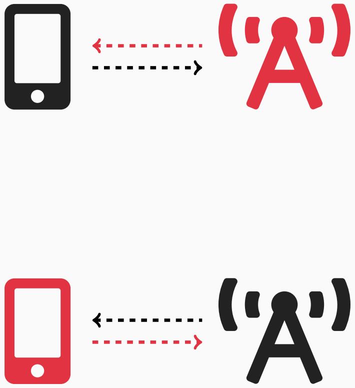


Why do we want Mutual Authentication?

Authentication 📱 ↔ (A)

- UE and eNodeB authenticate each other
- Can protect against Man-in-the-Middle, replay, spoofing attacks

Wait a second. Spoofing? Man-in-the-Middle? Heard that before!



Authentication and Key Agreement AKA

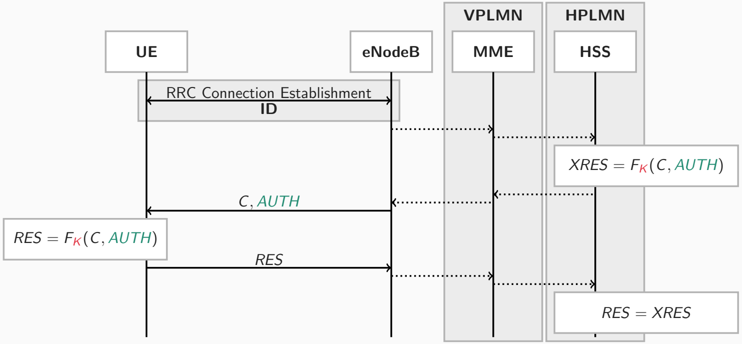
Mutual Authentication in LTE:

- LTE uses a challenge-response protocol to establish **mutual authentication** between the UE and the network
- The protocol uses symmetric key cryptography
- The UE has its secret K on the SIM card
- The operator stores their secrets K in the core network (HSS)

Authentication and Key Agreement AKA:

- Before the AKA, the RRC Connection Establishment takes place
- (Remember the Identity Mapping attack of last week, RNTIs, . . .)
- In this process, the UE sends its ID towards the network
- The ID is used to check the correct individual information

Authentication and Key Agreement



Authentication and Key Agreement

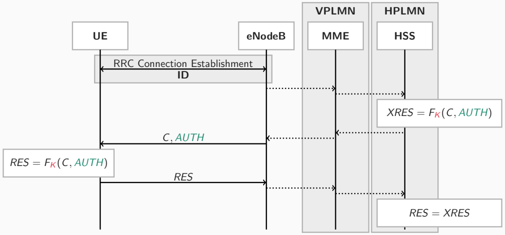
(1) After connection was established, network sends the challenge C and authentication token $AUTH$
(2) Network generates individual $XRES$
(3) UE uses secret K to generate RES
(4) Send RES towards network, where it's compared to $XRES$

Important:

- The authentication token $AUTH$ authenticates the network towards the UE
- $RES = XRES$ authenticates the UE towards the network
- The eNodeB only does the communication. All important computations are done in the *core network*.

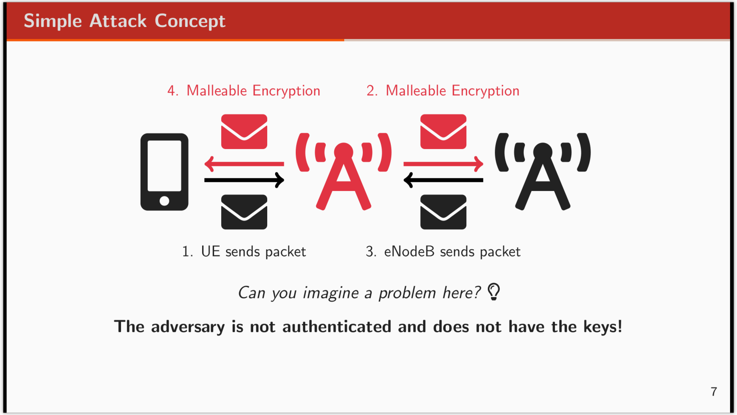
AKA Core Components

- Challenge C: Like a nonce
- Authentication Token AUTH: ID-specific
  - Sequence number, receives updates whenever used
  - In sync between HSS and UE
  - Authenticates network to UE
- Cryptographic function F: Generate tokens RES and $XRES$
- Secret K: Symmetric key

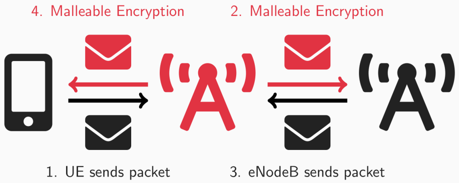


Why all this trouble?

Because Mutual Authentication does not pair well with an **impersonation**.



Simple Attack Concept



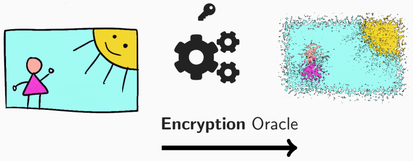
Can you imagine a problem here?

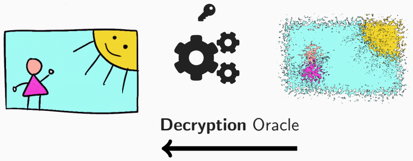
The adversary is not authenticated and does not have the keys!

Encryption and Decryption Oracle

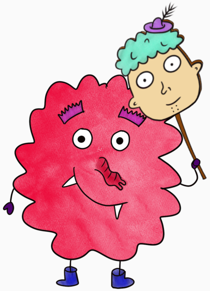
How to impersonate in both directions:

- Use an encryption oracle in uplink direction
  - Use an arbitrary plaintext packet
  - Encrypt it with the correct keys
- Use a decryption oracle in downlink direction
  - Receive an arbitrary encrypted packet
  - Decrypt it with the correct keys





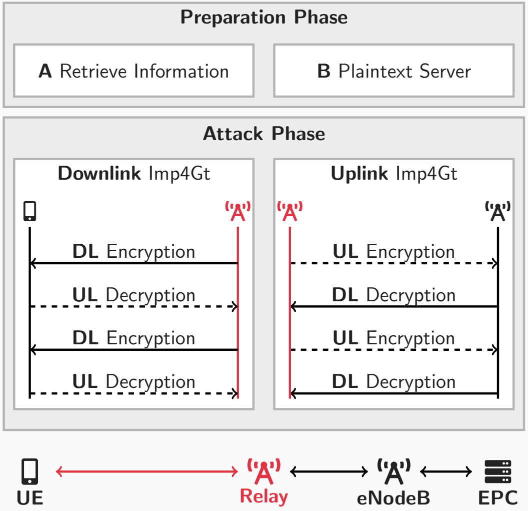
Impersonation versus Man-in-the-Middle



What is the difference?

- Man-in-the-Middle
  - Authentication with the User Equipment (UE)
  - Authentication with the Evolved NodeB (eNodeB)
  - $\rightarrow$ Establish own keys with both parties
- (Our) Impersonation
  - Relay the traffic
  - Encrypt arbitrary new packets
  - Decrypt incoming packets
  - $\rightarrow$ Does not interfere with the keys!

Attack Concept



Attack Concept

Downlink Impersonation

- eNodeB impersonates legitimate base station towards UE
- Encrypt packets in downlink direction to make it look like original traffic
- Decrypt packets in uplink direction to get access to UE's traffic

Uplink Impersonation

- eNodeB impersonates legitimate UE towards the network
- Encrypt packets in uplink direction to make it look like original traffic
- Decrypt packets in downlink direction to get access to UE's traffic

Same same but different!

General Concept: Summary

- **Challenge**: Impersonate the UE towards the Evolved NodeB (eNodeB), impersonate the Evolved NodeB (eNodeB) towards the UE.
- **Problem**: Long Term Evolution (LTE) uses an Authentication and Key Agreement (AKA) to establish mutual authentication.
- **Solution**: Relay traffic, use an encryption and decryption oracle.
- **Uplink**: Encryption oracle injects arbitrary packets and encrypt.
- **Downlink**: Decryption oracle receive packet and successfully decrypts it.
- **Result**: Full impersonation in both directions.

Mutual Authentication is important for the exam!

General Concept: Exam Examples

- What security feature does the Authentication and Key Agreement (AKA) introduce?
- What entity stores the shared secret?
- What information in the Radio Resource Control (RRC) connection establishment is important for the Authentication and Key Agreement (AKA)?
- Sketch the Authentication and Key Agreement (AKA).
- What is the purpose of the authentication token *AUTH*?
- Explain *RES* and *XRES*.

Encryption and Decryption Oracle

What do we need the oracles for?

Encryption Oracle

We use the encryption oracle to learn the *keystream* of a connection. We use this keystream to *encrypt* arbitrary packets and inject them in the connection.

🔓 → 🔒

Decryption Oracle

We use the decryption oracle to decrypt packets of the connection and access their *plaintext*.

🔒 → 🔓

Encryption Oracle

Goal: Learn the *keystream* of the connection.

Reason: Encrypt *our own* packets with the *original* keystream!

(1) Oracle injects a known plaintext

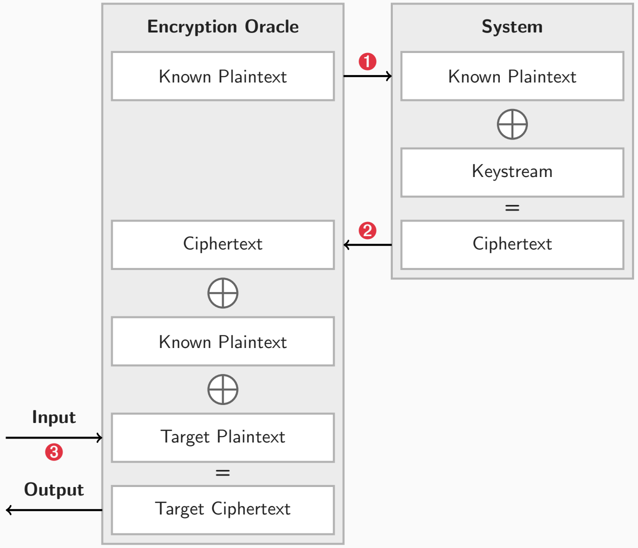
(2) System encrypts it with the original keystream

(3) Send the ciphertext to the oracle

(4) It derives the keystream because we know the plaintext

(5) From now on we can encrypt arbitrary packets with the original plaintext

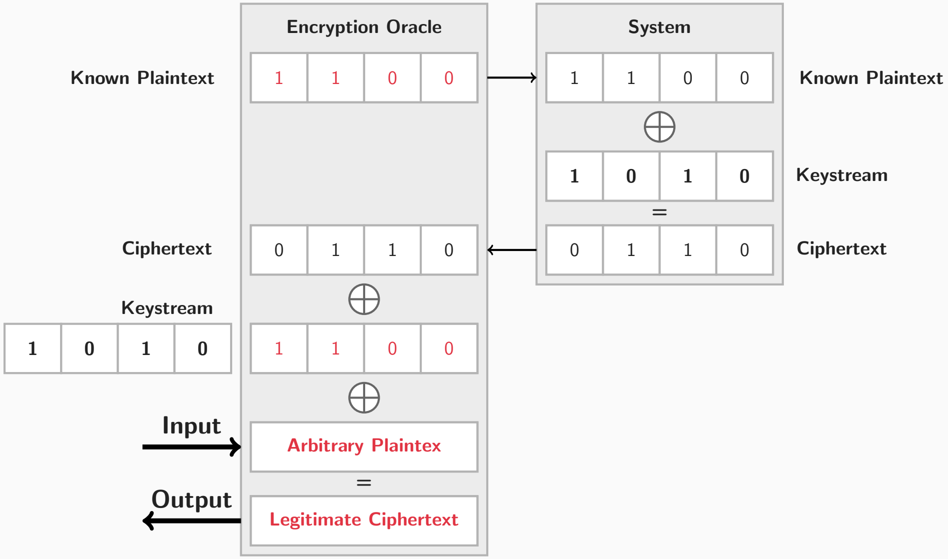
Encryption Oracle



Encryption Oracle

1. Create a known plaintext and send it to the system
   - Plaintext is received as a normal packet
   - System encrypts the packet with the keystream
   - As a result, we get the ciphertext
2. Send the ciphertext back to the oracle
   - Again, `XOR` the ciphertext with a known plaintext
   - The result of this is the keystream!
3. Inject a target plaintext
   - This is the packet that we want to inject
   - Because we have the keystream, we can encrypt it
   - The target ciphertext can be now used

Encryption Oracle: Example



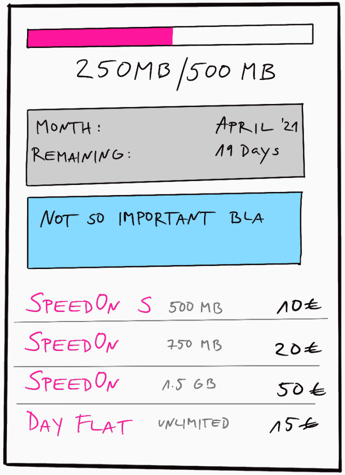
Why do we do this again?

“Inject arbitrary packets”

- **Inject**
  - Send a packet to the Evolved NodeB (eNodeB)
  - Encrypt it correctly, otherwise the connection fails
- **Arbitrary**
  - Packet goes through the core network…
  - *…and arrives where we want, requesting what we want!*

What could an adversary do with this ability?

Attack Examples

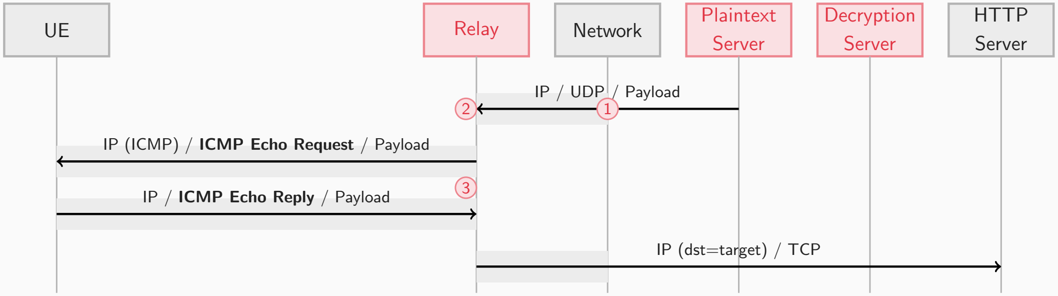


Order several data passes

- Visit the mobile data plan site of the provider
- Order several data passes
- User has to pay for this
- Network receives legitimate requests

But how exactly does this happen?

HTTP Example



Encrypted for radio layer

HTTP Example

1. The plaintext server creates a payload and sends it to the network.
   Why can we send it to the network?
   We opened a port in the NAT before and now make it look like a packet for the UE.

2. Intercept the packet in the Relay. We now have a ciphertext for our known plaintext.
   Why aren't we done here?
   We want to inject a packet in *uplink* direction.

We need the ping reflection to prepare a packet for the uplink direction!

HTTP Example

3 **Relay:** Prepare for the ping reflection. The relay changes the ICMP field to `echo request` and the IP protocol field to `ICMP`. It also sets the correct checksum.

3 **UE:** Receives the `echo request` and prepares an `echo reply`. It first *decrypts* the incoming packet, then adjusts the IP addresses, and finally encrypts it again.

We now have a legitimate packet in the required network context. This can be sent in uplink direction, and it contains the payload we injected!

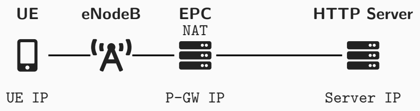
Encryption Oracle: Summary

So far we looked at the *encryption oracle* and (once more) at a known-plaintext attack. Before diving into details with the *ping reflection*, let's wrapup this first part:

- **Challenge**: Inject arbitrary packets in uplink direction.
- **Problem**: We don't know the keystream.
- **Solution**: Encryption oracle + Malleable encryption.
- **Result**: We act as UE and request a server we want.

So far: Known-Plaintext + Encryption Oracle.
Next: Ping Reflection + Header Information

Header Information

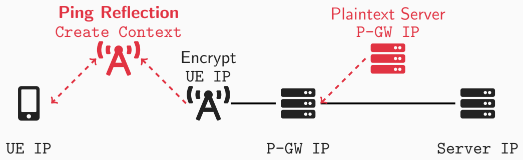


- **HTTP Server:** IP address of server
- **P-GW:** (External) IP address of the P-GW
- **UE:** (Internal) IP address of the UE

Internal vs. External

- The PDN Gateway (P-GW) is the *gateway* to the Internet.
- The P-GW is a NAT:
  - GW has its own IP address
    $\rightarrow$ Outside the LTE network
  - Users get individual internal IPs
    $\rightarrow$ Inside the LTE network

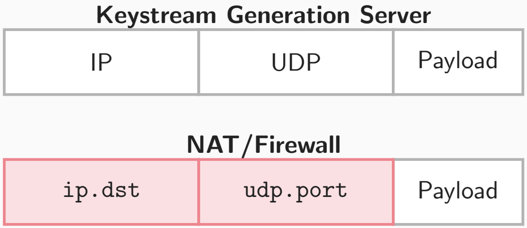
Inject, Intercept, Encrypt, Send



Using an open port:

- The plaintext server prepares the *payload we want to send in uplink direction.*
- We use an open port in the P-GW to inject the packet.
- We depend on the ping reflection to add the correct *context* and *source addresses*.

Changing Header Information



From P-GW to Evolved NodeB (eNodeB)

(1) Plaintext server sends packet to IP of the gateway

(2) Destination IP `ip.dst` and port `udp.port` change at the gateway: NAT and Firewall

Ping Reflection

ICMP Echo Request and Reply

- Tests if a host is reachable.
- In response to an echo request, the target sends an echo reply.
- This copies the payload of the request.
- **Ping Reflection!**

```
$ ping google.com
PING google.com (172.217.16.142) 56(84) bytes of data.
64 bytes from zrh04s06-in-f142.1e100.net (172.217.16.142): icmp_seq=1 ttl=116 time=12.1 ms
64 bytes from fra15s46-in-f14.1e100.net (172.217.16.142): icmp_seq=2 ttl=116 time=12.3 ms
64 bytes from fra15s46-in-f14.1e100.net (172.217.16.142): icmp_seq=3 ttl=116 time=12.2 ms
```

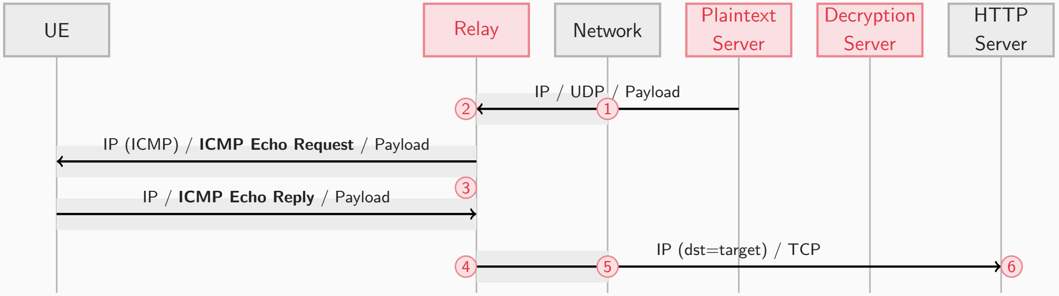
Ping Reflection

Exploiting the Ping Reflection

1. Relay
   - Relay changes the protocol type to ICMP
   - Set the ICMP header and correct ICMP checksum
2. UE
   - UE reflects the packet
   - Swaps source with destination IP in header
3. Relay
   - Receives the reflected packet
   - Forwards it in uplink direction

Our injected payload now travels in uplink direction and has a legitimate encryption and network context.

HTTP Example



Encrypted for radio layer

HTTP Example

4 The relay receives the encrypted packet and forwards it in uplink direction.

5 The commercial network receives the packet, decrypts it, and forwards it to the Internet.

6 **The packet arrives at the target server.**
It contains whatever we put in there!

Summary

Summary

- **Full Impersonation**
    - Act as phone towards the network
    - Act as network towards the phone
- **Authentication and Key Agreement (AKA)**
- **Oracles**
    - Encryption oracle: Apply legitimate encryption to arbitrary packets
    - Decryption oracle: Decrypt incoming packets
- **Ping Reflection**

Acronyms

AKA	Authentication and Key Agreement
C-RNTI	Cell Radio Network Temporary Identity
eNodeB	Evolved NodeB
EPC	Evolved Packet Core
E-UTRAN	Evolved Universal Terrestrial Radio Access
EPLMN	Equivalent PLMN
GUTI	Globally Unique Temporary Identifier
HPLMN	Home PLMN
HSS	Home Subscriber Service
IMSI	International Mobile Subscriber Identity
LTE	Long Term Evolution
MAC	Medium Access Control
MCC	Mobile Country Code
MME	Mobility Management Entity
MNC	Mobile Network Code
NAS	Non-Access Stratum
P-GW	PDN Gateway
PDCP	Packet Data Convergence Protocol
PDN	Packet Data Network
PHY	Physical Layer
PLMN	Public Land Mobile Network
RAP	Random Access Preamble
RA-RNTI	Random Access RNTI
RLC	Radio Link Control
RNTI	Radio Network Temporary Identity
RRC	Radio Resource Control
S-GW	Serving Gateway
S1AP	S1 Application Protocol
SCTP	Stream Control Transmission Protocol
VPLMN	Visiting PLMN
SDR	Software Defined Radio
TMSI	Temporary Mobile Subscriber Identity
UE	User Equipment