Computer Security Capstone

Project I: IPsec Session Hijacking

Chi-Yu Li (2023 Spring)
Computer Science Department
National Yang Ming Chiao Tung University

Goals

- Understand how to hijack IPsec sessions

- You will learn about
  - the IPsec operation
  - fabricating packets using raw socket
  - fabricating IPsec ESP headers and authentication data
  - fabricating TCP packets

What is IPsec?

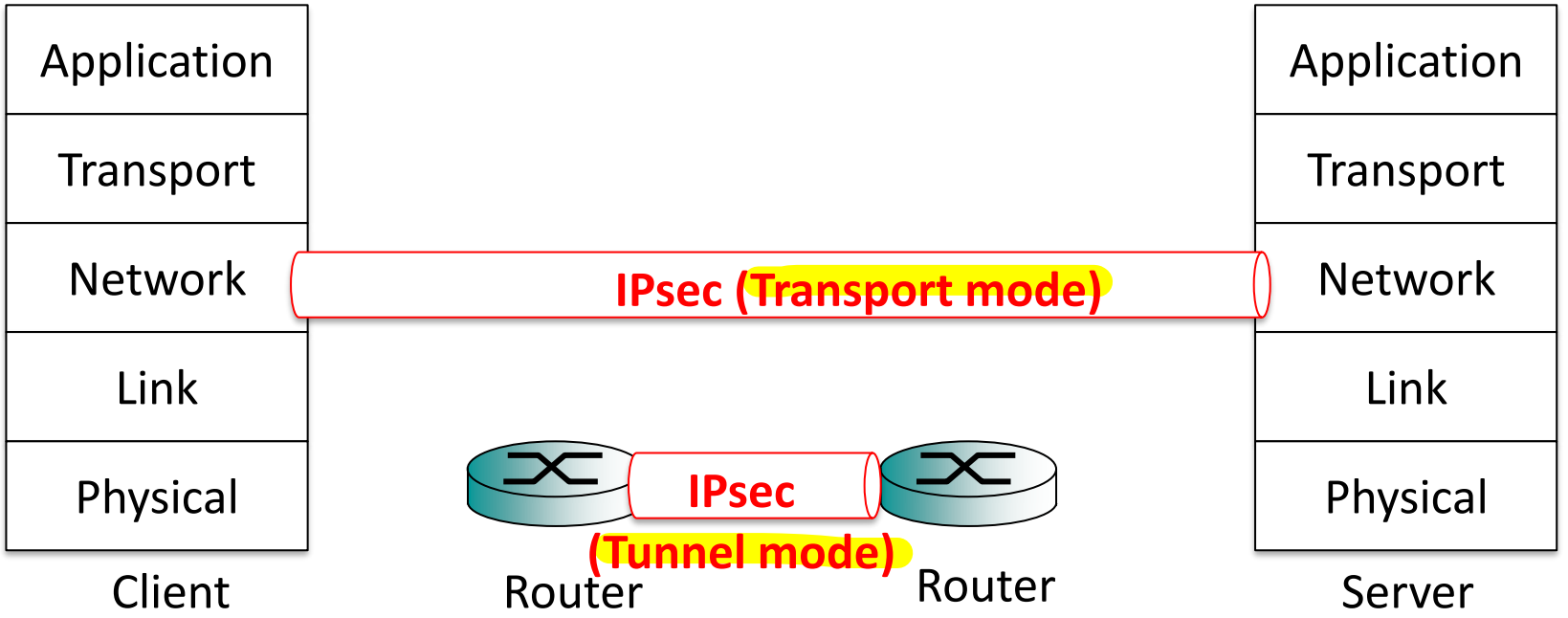
- Internet Protocol Security (IPsec) is a secure network protocol suite
  - It provides secure communication by authenticating and encrypting data packets
  - It ensures the confidentiality and integrity of the data
- Two main functions
  - Encapsulating Security Payload (ESP): a combined authentication/encryption function
  - A key exchange function: Internet Key Exchange standard (IKEv2)

IPsec Primer: Security Associations

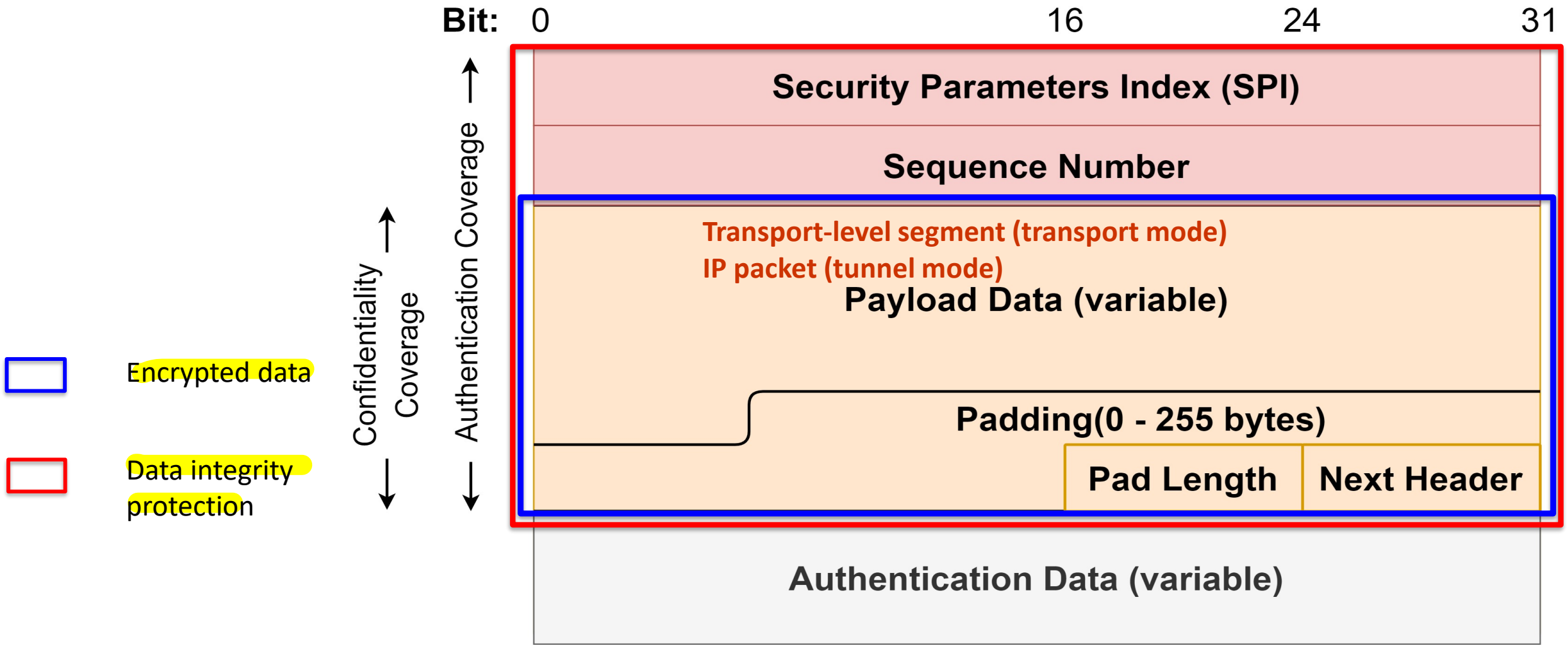
- A key concept of IPSec
  - One-way relationship between a sender and a receiver
  - Two-way secure exchange: two SAs are required
- Uniquely identified by three parameters
  - Security parameter index (SPI)
  - IP destination address
  - Protocol identifier: AH or ESP

IPsec Primer: Two IPsec Operation Modes

- Transport and Tunnel modes



IPsec Primer: Encapsulating Security Payload (ESP)



IPsec Primer: Transport and Tunnel Modes

Transport Mode

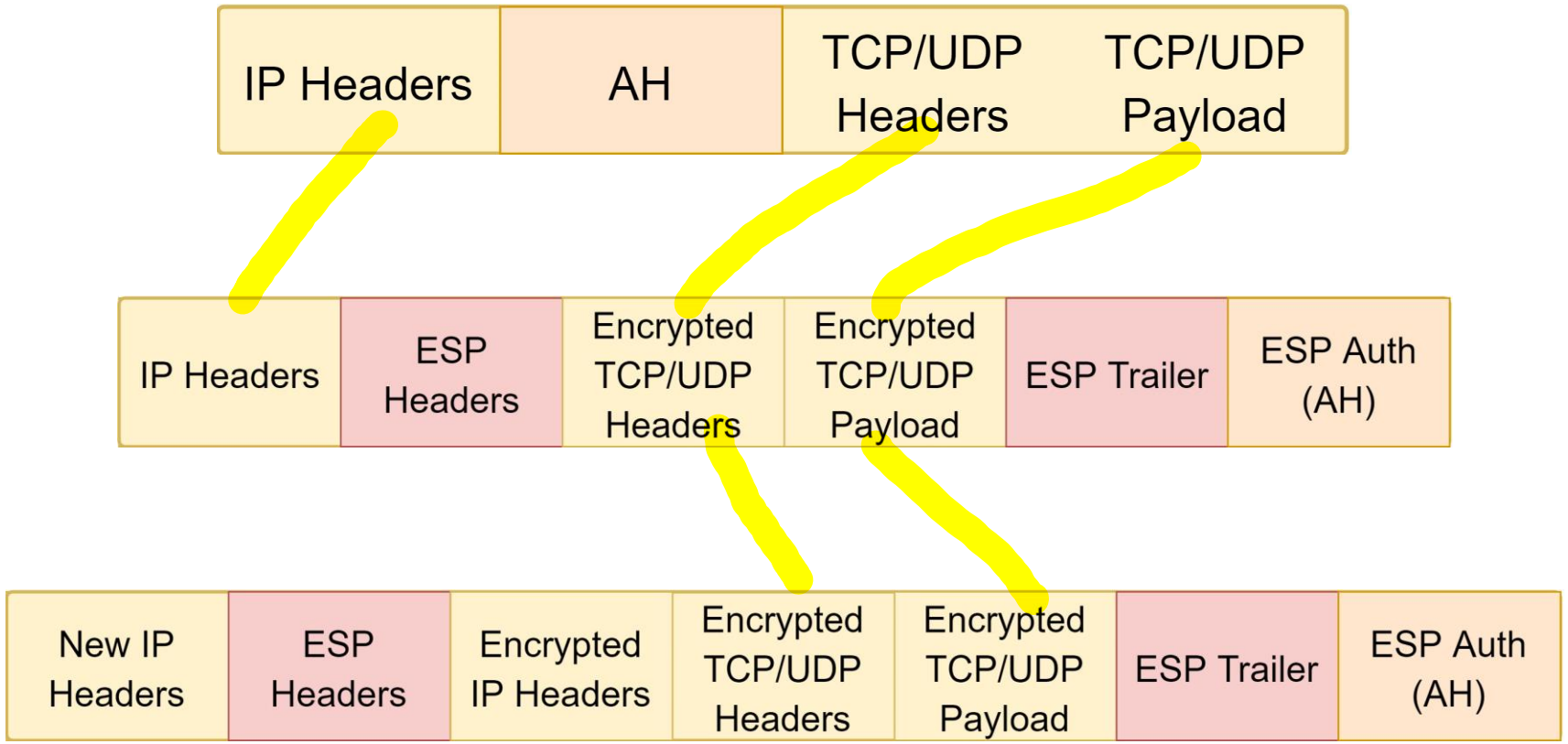
- Protection: the payload of an IP packet
- Typically used for end-to-end communication between two hosts
- ESP protects the IP payload but not the IP header

Tunnel Mode

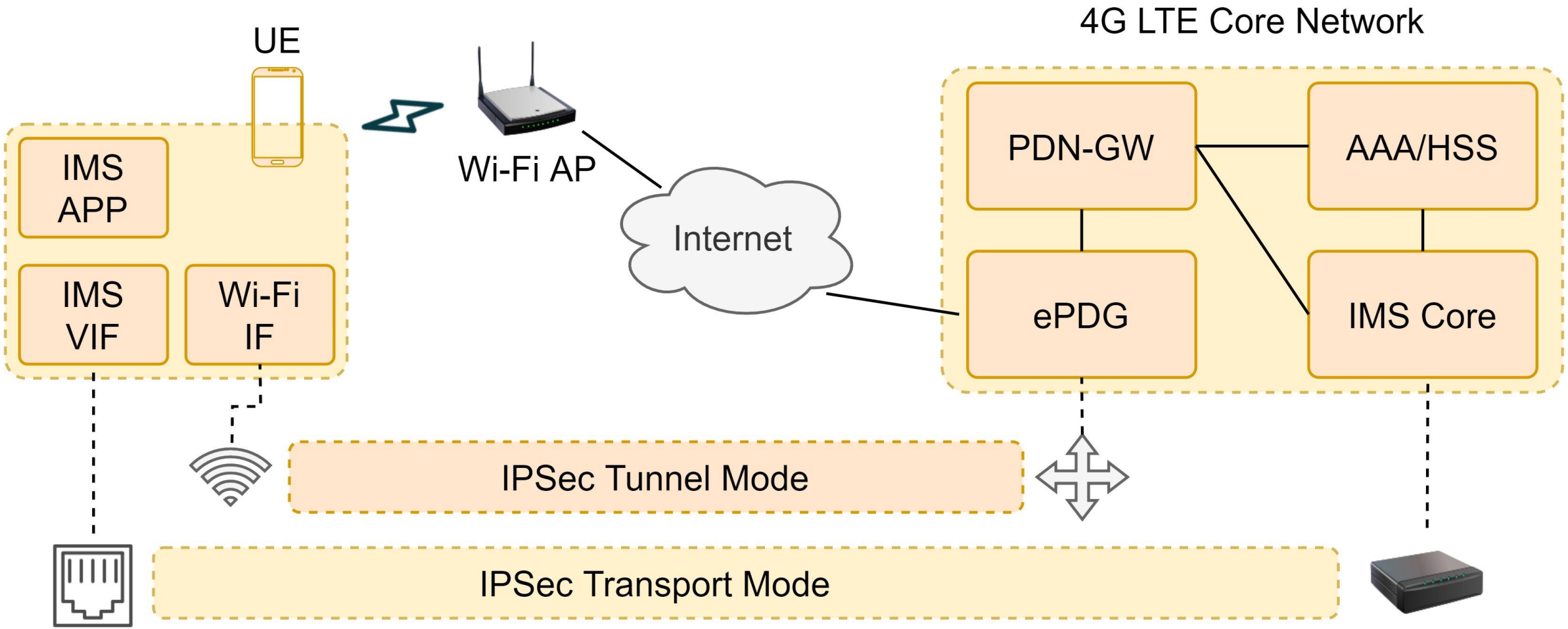
- Protection: the entire IP packet
- Entire original packet travels through a tunnel from one point to another
- Used when one or both ends of a security association are a security gateway
- Hosts on networks behind firewalls may engage in secure communications without implementing IPsec

IPsec Primer: AH + ESP

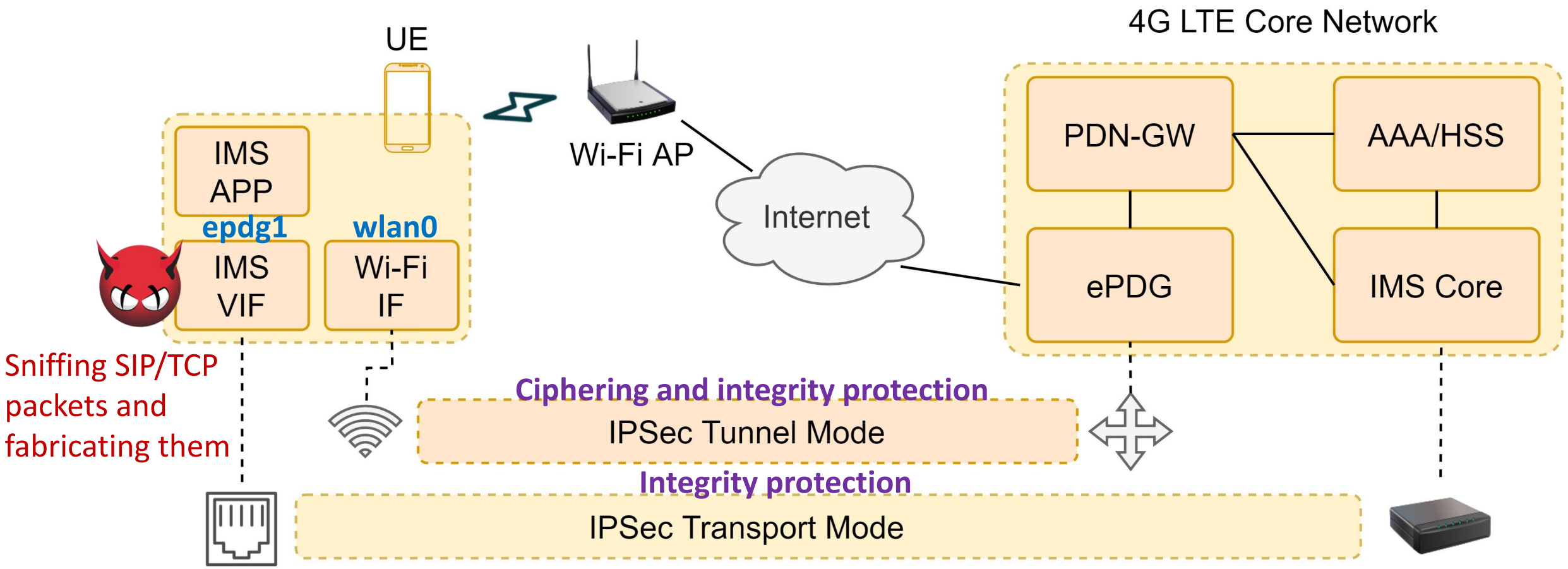
- IP AH only
- IP AH + ESP
  - Transport mode
  - Tunnel mode



VoWi-Fi Security

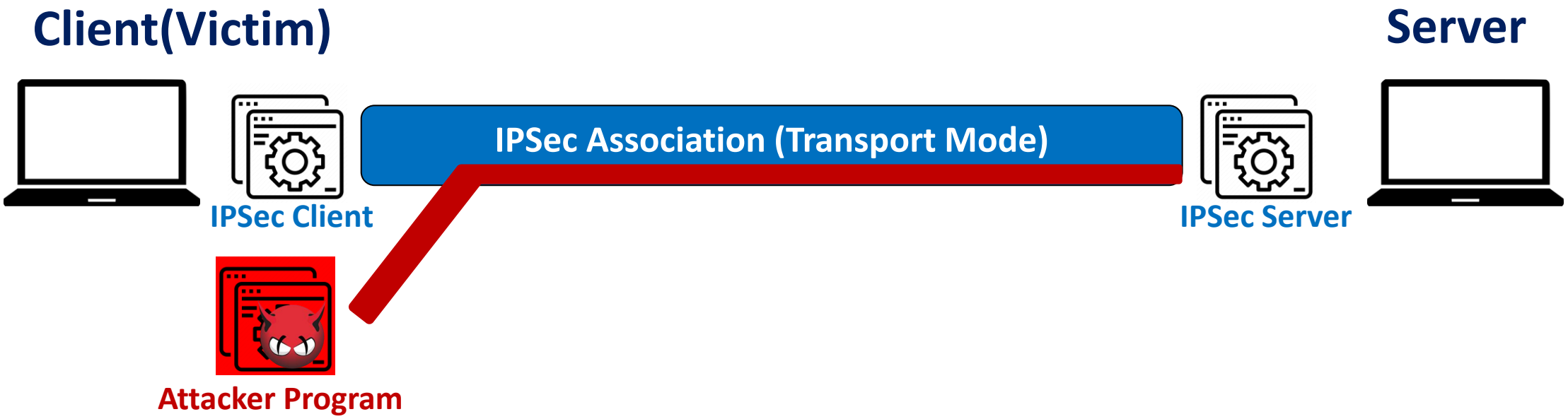


IPSec Hijacking Attack



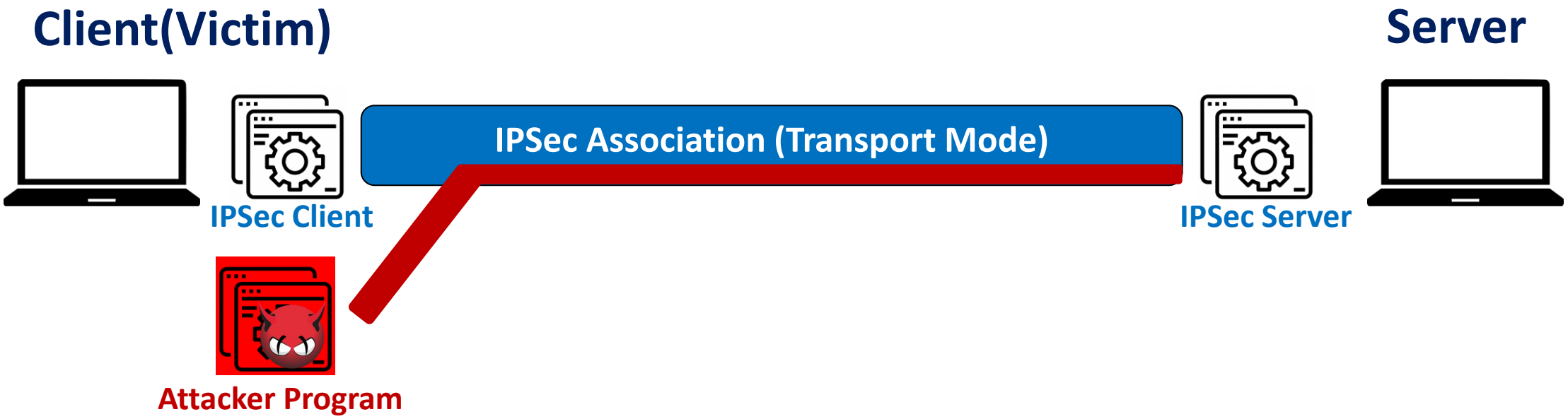
Attack Scenario in this Project

- Scenario: The TCP client has set up IPsec associations in transport mode for secure communication with a TCP server
- Attacker: Executing a malicious program to hijack the IPsec/TCP session



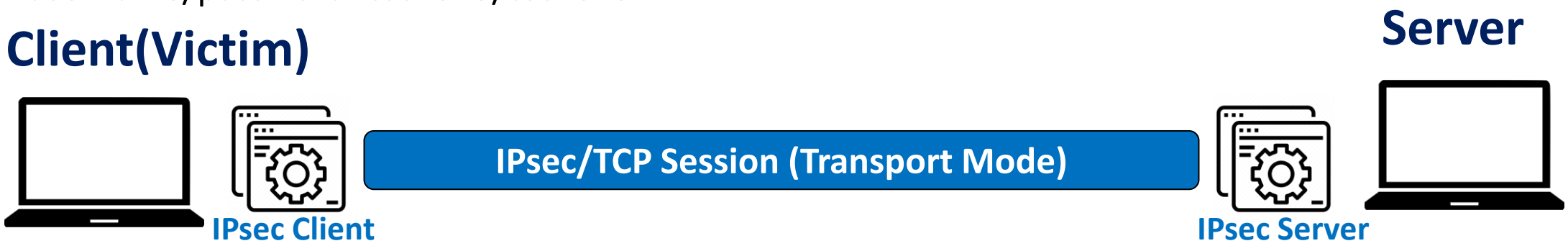
How to Proceed?

- Executing provided programs to establish the IPsec/TCP session
- Developing an attacker program on Client to hijack the IPsec/TCP session
- Sending specific flags to the server using the attacker program
  - With the successful hijacking, the server can reply to the flags with correct responses



Environment Setup

- Using two devices, designated as the client and the server, and establishing the IPsec/TCP session between them.
- Please download a VM image, including all the programs and sample codes in the Home directory
  - username/password: csc2023/csc2023



Step 1: Run	sudo sh ipsec_victim.sh	sudo sh ipsec_server.sh

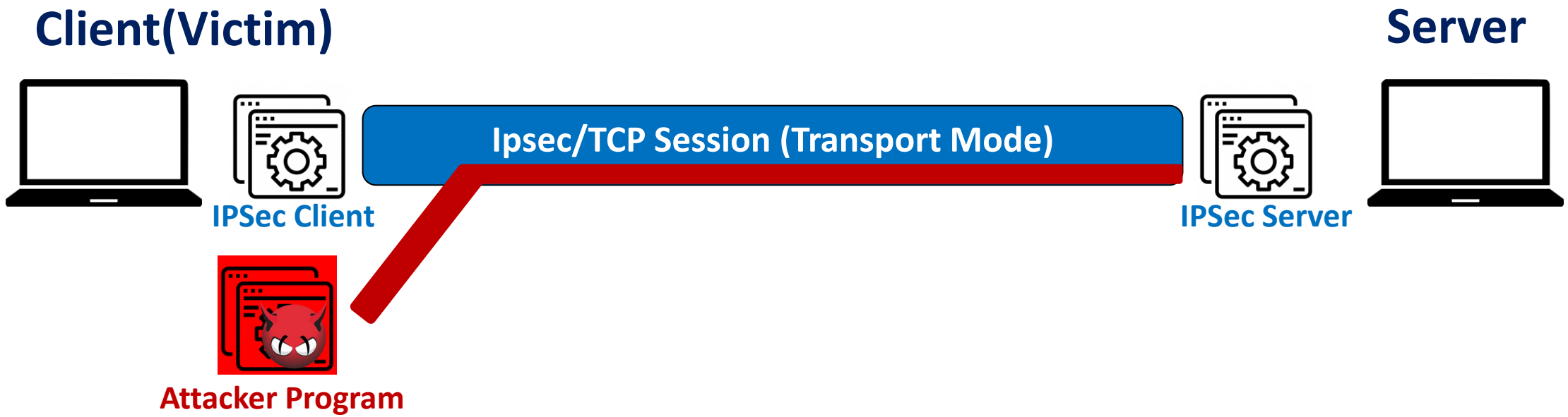
(Using sudo; replacing the IP address and port with yours in the sh file)

Step 2: Run	./tcp_client <server_ip> <server_port> -bp <victim_port>	./tcp_server <server_port>

(Putting the file answer.txt into the same directory as tcp_server)

Attack: IPSec/TCP Session Hijacking

- Developing a program, ipsec_hijack, to hijack the IPsec/TCP session




```
sudo ./ipsec_hijack <interface>
```

What should the attacker program do?

- Realtime information monitoring and collection
  - Getting session information from SIP and TCP headers, e.g., TCP sequence number and ESP SPI
  - Retrieving IPsec security context (e.g., ESP authentication key) from Security Association Database (SAD)
- IPsec/TCP packet fabrication
  - Fabricating TCP/IPsec/IP headers, including all the fields and checksum
  - Generating ESP padding
    - The Pad Length and Next Header fields must be right aligned with a 4-byte word (RFC4303 Section 2.4)
  - Generation ESP Authentication data
    - Using hmac_sha1_96

Todo Check List for Sample Codes

File	Description
./src/dev.c	Fill up struct sockaddr_ll addr which will be used to bind in function set_sock_fd
./src/dev.c	store the whole frame into self->frame
./src/transport.c	Finish TCP checksum calculation
./src/transport.c	Collect information from segm
./src/transport.c	Fill up self->tcphdr
./src/net.c	Finish IP checksum calculation
./src/net.c	Collect information from pkt.
./src/net.c	Fill up self->ip4hdr
./src/esp.c	Dump authentication key from security association database (SAD)
./src/esp.c	Fill up self->pad and self->pad_len (Ref. RFC4303 Section 2.4)
./src/esp.c	Put everything needed to be authenticated into buff and add up nb
./src/esp.c	Collect information from esp_pkt.
./src/esp.c	Fill up ESP header and trailer

Three Verification Steps

- Step I: The server can receive fabricated IPsec packets belonging to the existing IPsec session (40%)
- Step II: The attacker program can correctly exchange TCP packets (data and ACK) with the server through the fabricated IPsec packets (30%)
- Step III: The attacker program can interact with the server with multiple handshakes (30%)

Step I: the server can receive fabricated IPsec packets belonging to the existing IPsec session

- Using Wireshark
  - Client/Attacker program: 172.17.1.1
  - Server: 172.17.100.254

	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	172.17.1.1	172.17.100.254	ESP	90	ESP (SPI=0x0000c6f8)
2	0.001400812	172.17.100.254	172.17.1.1	ESP	90	ESP (SPI=0xfb170e3f)
3	0.001441966	172.17.1.1	172.17.100.254	ESP	78	ESP (SPI=0x0000c6f8)
4	0.001633348	172.17.1.1	172.17.100.254	ESP	146	ESP (SPI=0x0000c6f8)
5	0.002686062	172.17.100.254	172.17.1.1	ESP	78	ESP (SPI=0xfb170e3f)
6	1.002215880	172.17.1.1	172.17.100.254	ESP	146	ESP (SPI=0x0000c6f8)
7	1.003546560	172.17.100.254	172.17.1.1	ESP	78	ESP (SPI=0xfb170e3f)
8	2.002884014	172.17.1.1	172.17.100.254	ESP	146	ESP (SPI=0x0000c6f8)
9	2.004261232	172.17.100.254	172.17.1.1	ESP	78	ESP (SPI=0xfb170e3f)

Step II: the attacker program can correctly exchange TCP packets with the server through the fabricated IPsec packets

- Using Wireshark
  - Client/Attacker program: 172.17.1.1
  - Server: 172.17.100.254
  - Modify the Wireshark Preferences to enable dissecting of raw data

	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	172.17.1.1	172.17.100.254	TCP	90	2222 → 1111 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_
2	0.001400812	172.17.100.254	172.17.1.1	TCP	90	1111 → 2222 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS
3	0.001441966	172.17.1.1	172.17.100.254	TCP	78	2222 → 1111 [ACK] Seq=1 Ack=1 Win=64512 Len=0
4	0.001633348	172.17.1.1	172.17.100.254	TCP	146	2222 → 1111 [PSH, ACK] Seq=1 Ack=1 Win=64512 Len=67
5	0.002686062	172.17.100.254	172.17.1.1	TCP	78	1111 → 2222 [ACK] Seq=1 Ack=68 Win=64512 Len=0
6	1.002215880	172.17.1.1	172.17.100.254	TCP	146	2222 → 1111 [PSH, ACK] Seq=68 Ack=1 Win=64512 Len=67
7	1.003546560	172.17.100.254	172.17.1.1	TCP	78	1111 → 2222 [ACK] Seq=1 Ack=135 Win=64512 Len=0
8	2.002884014	172.17.1.1	172.17.100.254	TCP	146	2222 → 1111 [PSH, ACK] Seq=135 Ack=1 Win=64512 Len=67
9	2.004261232	172.17.100.254	172.17.1.1	TCP	78	1111 → 2222 [ACK] Seq=1 Ack=202 Win=64512 Len=0

Step III: Multiple Handshake Tests with Three Flags

- An example with an invalid flag and two valid flags

```
smartphone# ./ipsec_hijack wlan0
you can start to send the flag...
abc123
flag1
get secret:
secret1
flag2
get secret:
secret2
```

Invalid flag

Valid flag

Valid flag

Important: How to Prepare Your Attack Program?

- Must provide a **Makefile** which compiles your source codes into one executable file, named **ipsec_hijack** **(Missing: -20%)**
- Your developed attacker program shall be run in the provided VM which serves as the client
- Recommended development language: C/C++
- Using the given program framework is not necessary

Project Submission

- Due date: 3/15 11:55pm
- Makeup submission & demo (75 points at most): TBA (After the final)
- Submission rules
  - Put all your files into a directory and name it using your student ID(s)
    - If your team has two members, please concatenate your IDs separated by “-”
  - Zip the directory and upload the zip file to E3
  - A sample of the zip file: 01212112-02121221.zip
    - Makefile
    - ….
- Teamwork is allowed
  - Up to two members for each team

Project Demo

- Date: 3/17
- TA will prepare two VMs to run as the client and the server, respectively
  - Your zip file will be put into the client
- You will
  - be asked to launch an IPsec/TCP hijacking attack
  - be only allowed to “make” to compile all your files, and run your attack binary programs or scripts
  - be not allowed to modify your codes or scripts
  - be not allowed to install any programs or libraries in the VM
  - be asked some questions
  - be responsible to show the outcome to TA and explain why you have successfully achieved the goals

Hint 1: How to Get Key from SAD?

- The message format from SAD

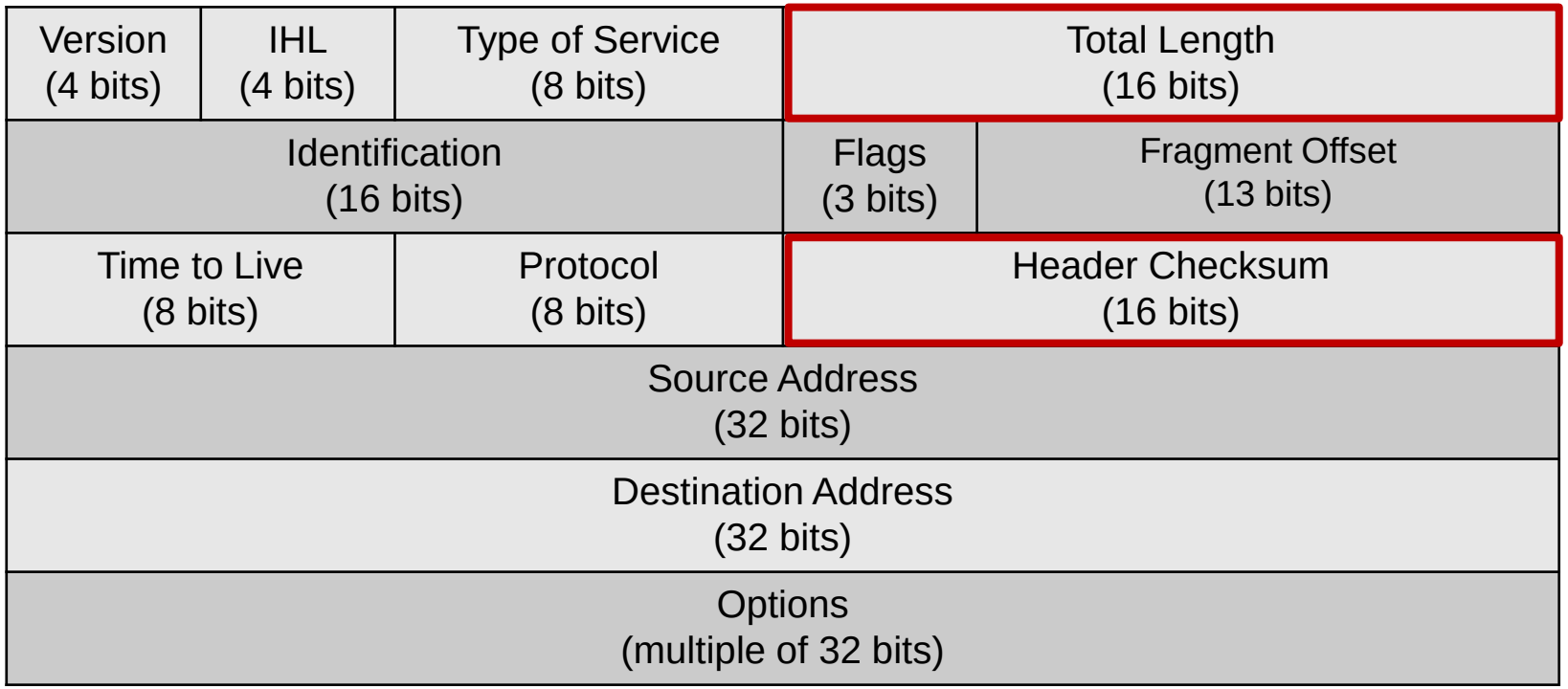
struct sadb_msg	sadb_ext	sadb_ext	…

- Each extension begins with a 16-bit ext_len and a 16-bit ext_type field

- Getting the key from the extension with sadb_ext_type "SADB_EXT_KEY_AUTH"

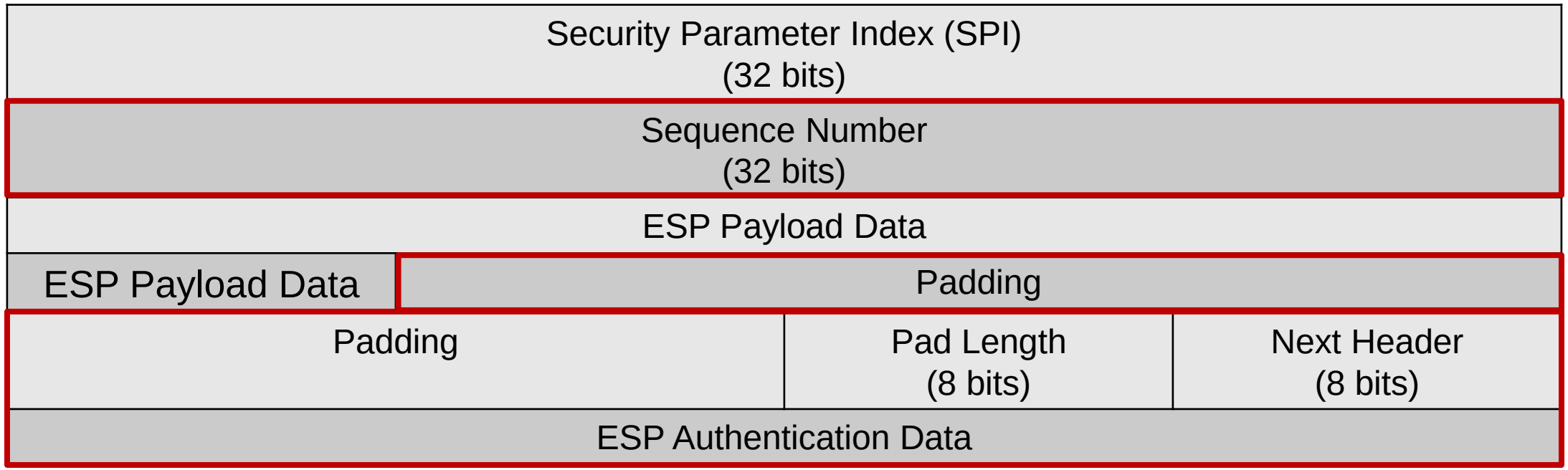
Hint 2: Which Packet Fields Need be Modified?

- IP header: struct iphdr in <netinet/ip.h>



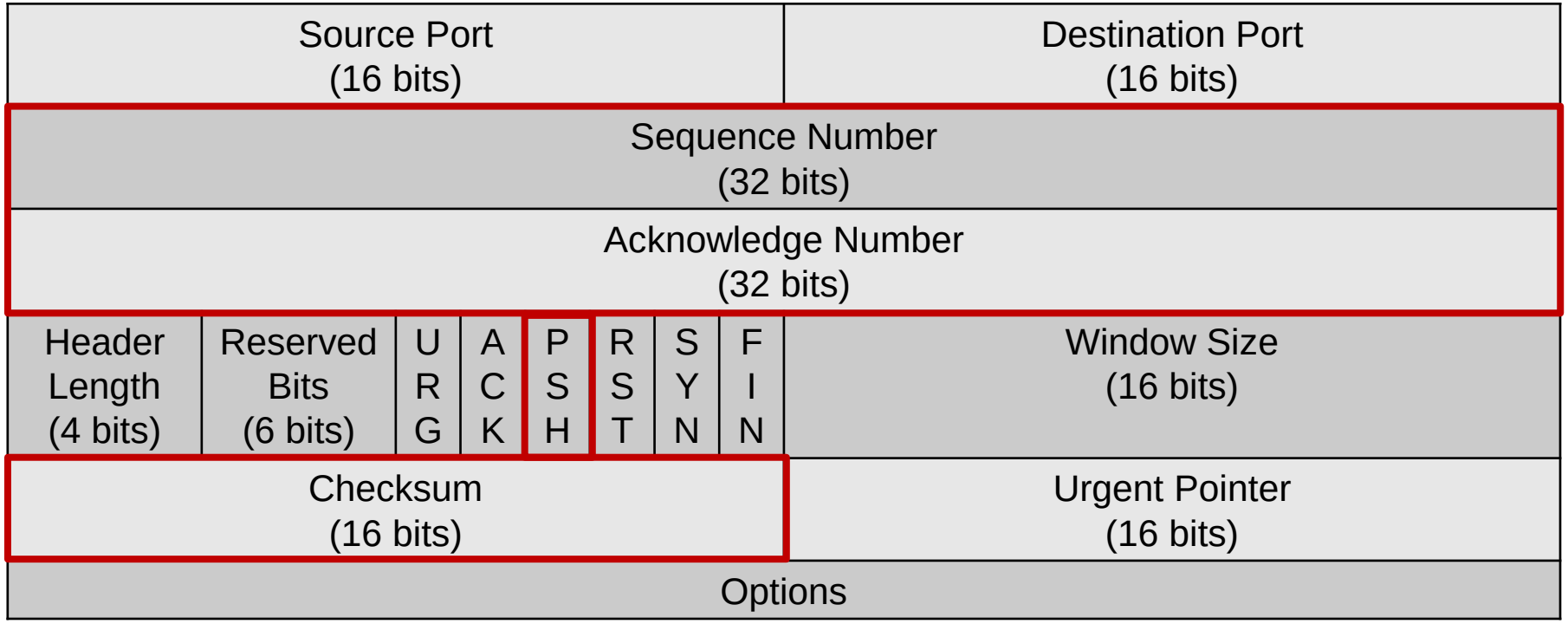
Hint 2: Which Packet Fields Need be Modified? (cont.)

- ESP header



Hint 2: Which Packet Fields Need be Modified? (cont.)

- TCP header: “struct tcphdr” in < netinet/tcp.h>



Questions?