Make a secure mobile payment

Yongjun Park

HITCON 2014

Profile

- **Yongjun Park(朴湧俊)**
- **Security testing of financial application, authentication method and security module**
- **Research and report new threats, vulnerabilities and security trends**
- **Participated in IT security audit of the ministry**

Agenda

- Mobile Payment Trend and Service
- Shadow of Mobile Payment
- Threat & Attack Surface
- Vulnerability Case
- Approach to Mobile Payment Test

Trend of mobile payment

- **Global businesses are entering into the mobile payment market**
- **ICT, Device manufactures, SNS…**
  - Kakao, one of the biggest mobile messenger will launch a new money transfer service with bank
  - Samsung and Visa make an alliance for mobile payment
- **Most financial institutions in Korea are supporting mobile platform**
  - Banking(balance check, money transfer, financial product purchase)
  - Payment(online, offline)
  - iOS, Android

Trend of mobile payment

- **Usage of mobile payment**

  - Increased 10 times in 2013 in Korea

Population = ***Mobile Card*** < ***Mobile Banking***

45,000,000

Type of services

- **Mobile payment is not a technology, a group of services**

  - Application
  - NFC (+ SIM or …)
  - Card reader
  - SE(Secure Element)
  - Mixed

Shadow of mobile payment

- **Good target for criminal**

  - Financial services = money, cash, $
    . Ex) Target
  - Malware spread out to payment, POS, shopping services
  - Phishing, Pharming

- **Case of illegal payment (May 2014)**

  - Mobile payment service was hacked by criminal
  - Credit cards were duplicated and user credential were stolen
  - Damage of illegal payments was more than $60,000

Shadow of mobile payment

- **The service registered credit card data into app**

  - Personal credential was issued for each user(Client certificate)

  1) malware spread out to target Android

  2) malware stealing PIN and credential

  3) install payment app and copy the stolen credential into iPhone

  4) load the credit card from server into installed app

✓ Some app was not adopted SIM authentication

✓ Copying credential was not restricted

Critical data and threat

- **Critical Data**

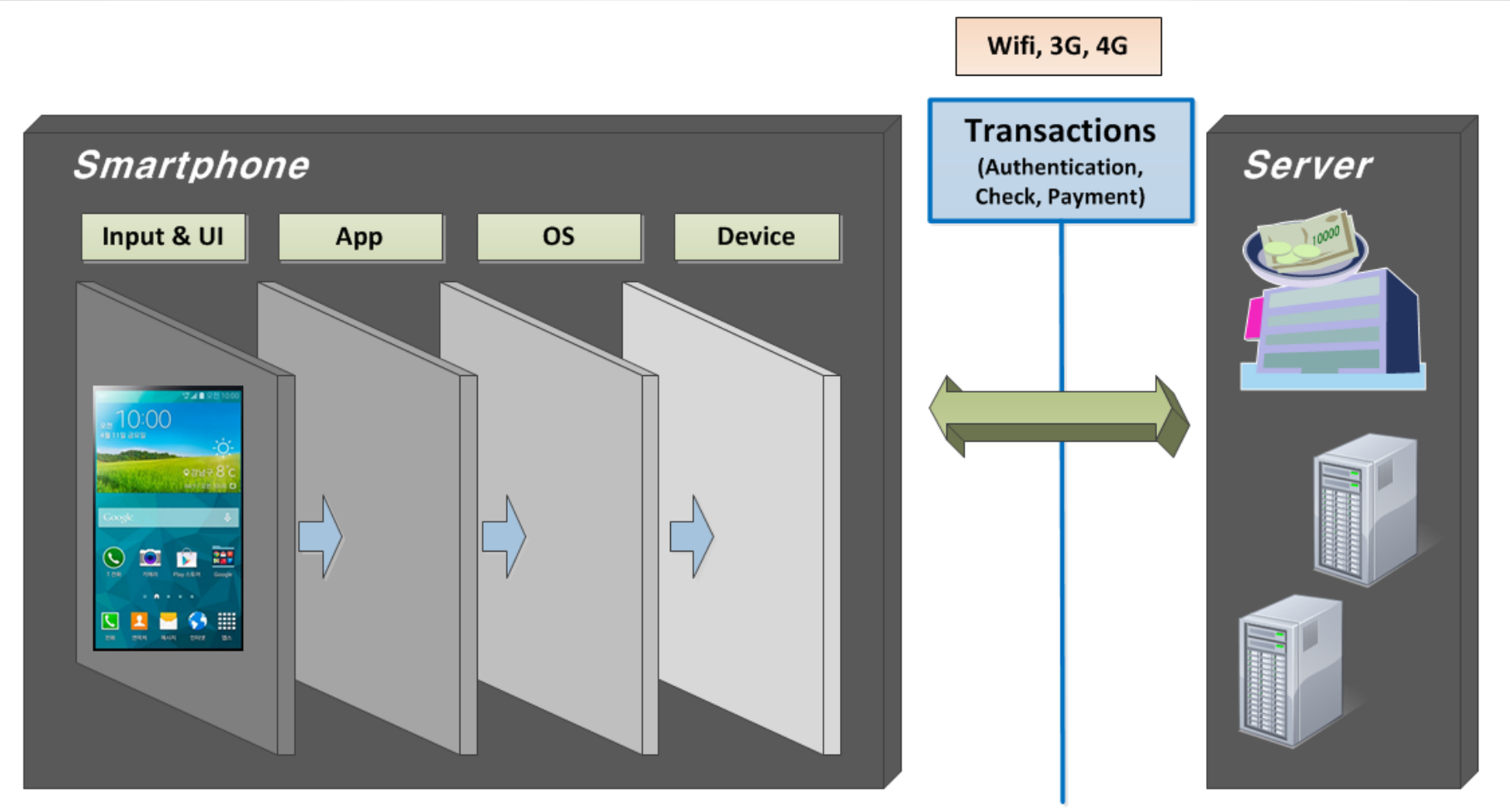
  - Transaction : account, card info(No, CVV, expiration date), receiver, amount, shop Info
  - User : user Identification, Privacy
  - Credential : card PIN, authentication, password

- **Type of threat**

  - Spoofing
  - Tampering
  - Information Disclosure
  - Repudiation, DoS, Elevation of privilege

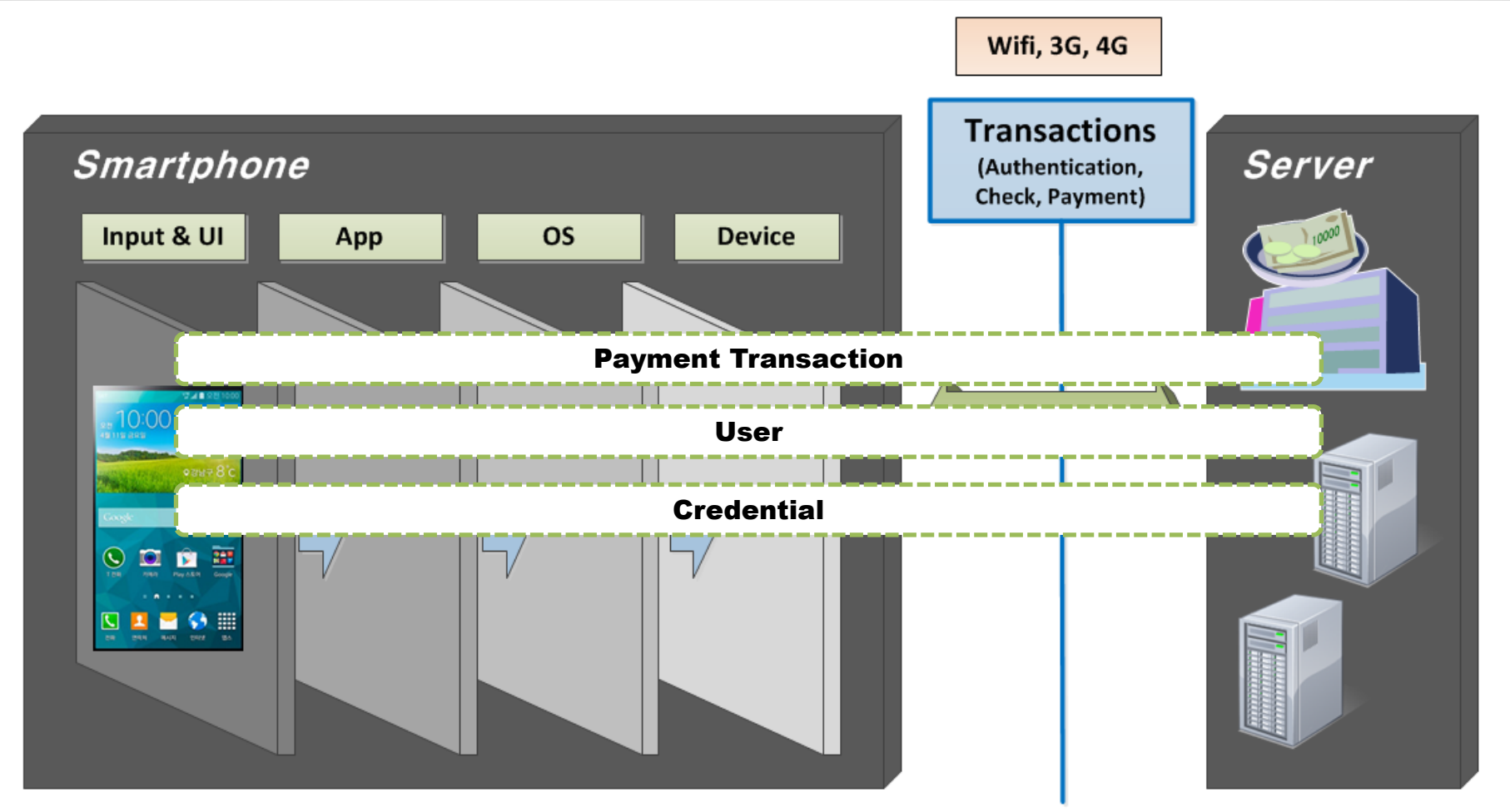
Attack Surface

- **System architecture**



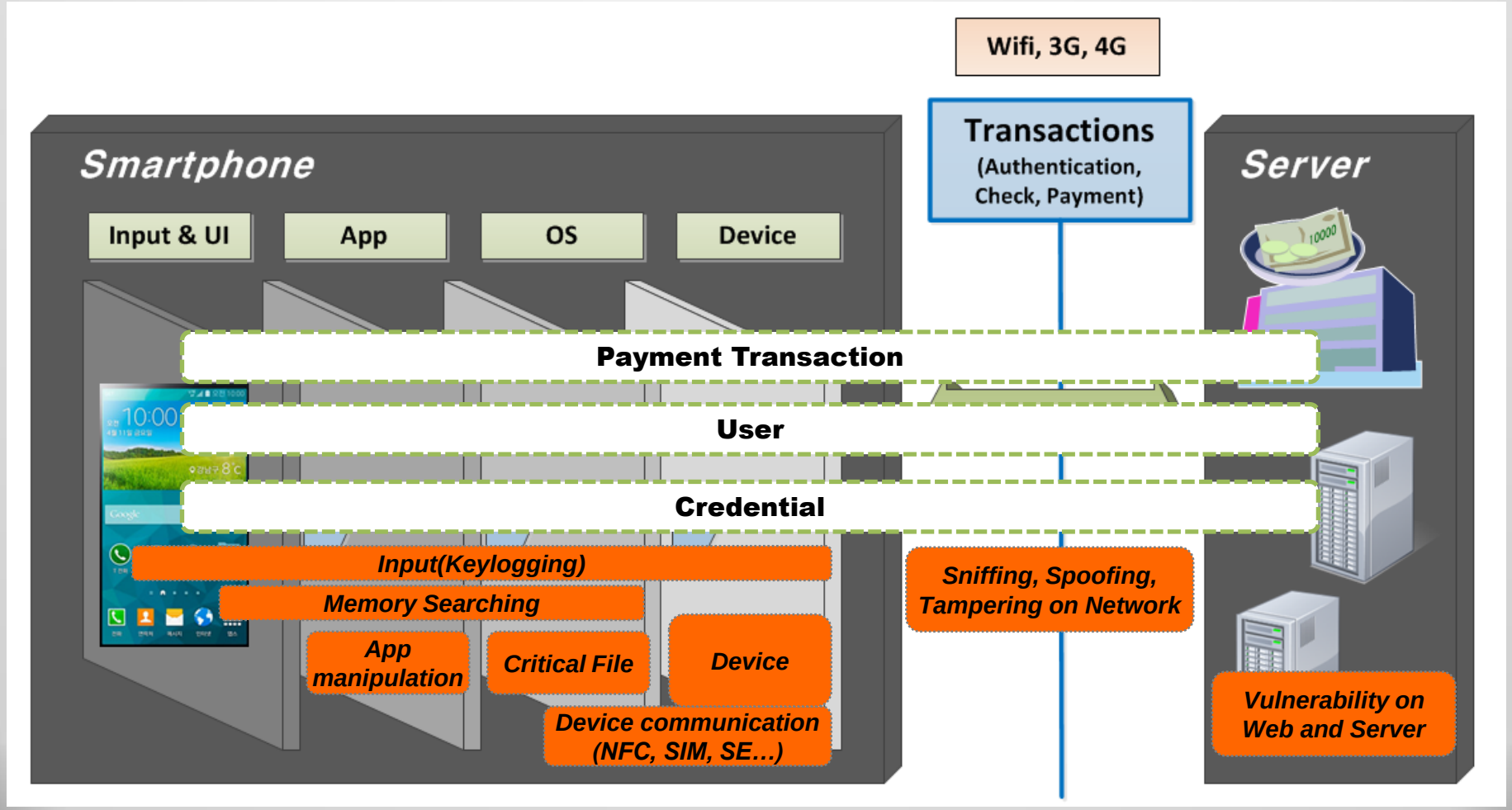
Attack Surface

- **Data flow on system architecture**



Attack Surface

- **Attack Surface and vulnerability**

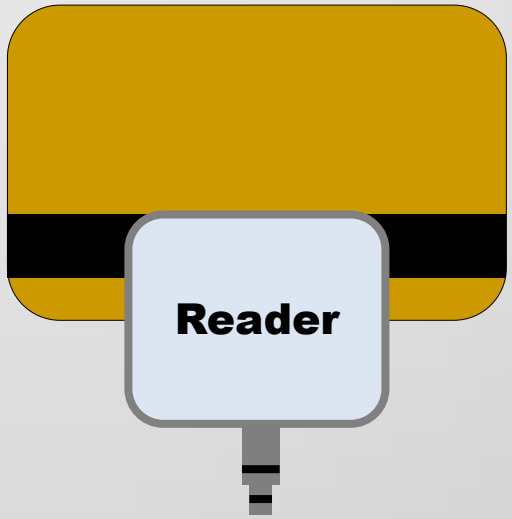


Vulnerability Case

Vulnerability – Case

- **Credit card reader**

  - One of the most popular payment in US
  - Put a small card reader in smartphone(or pad) using headphone jack
  - Reading card stripe and using it to check out
  - For a personal check out or as a POS(point of sale) of small business



Card reader manipulation

- **Data flow**

  - Make a payment using card data and PIN

  - Transaction encrypted on network, SSL

Card reader manipulation

PTRACE(2) Linux Programmer's Manual PTRACE(2)

NAME top

ptrace - process trace

SYNOPSIS top

```
#include <sys/ptrace.h>

long ptrace(enum __ptrace_request request, pid_t pid,
            void *addr, void *data);
```

DESCRIPTION top

The **ptrace**() system call provides a means by which one process (the "tracer") may observe and control the execution of another process (the "tracee"), and examine and change the tracee's memory and registers. It is primarily used to implement breakpoint debugging and system call tracing.

A tracee first needs to be attached to the tracer. Attachment and subsequent commands are per thread: in a multithreaded process, every thread can be individually attached to a (potentially different) tracer, or left not attached and thus not debugged. Therefore, "tracee" always means "(one) thread", never "a (possibly multithreaded) process". Ptrace commands are always sent to a specific tracee using a call of the form

```
ptrace(PTRACE_foo, pid, ...)
```

where *pid* is the thread ID of the corresponding Linux thread.

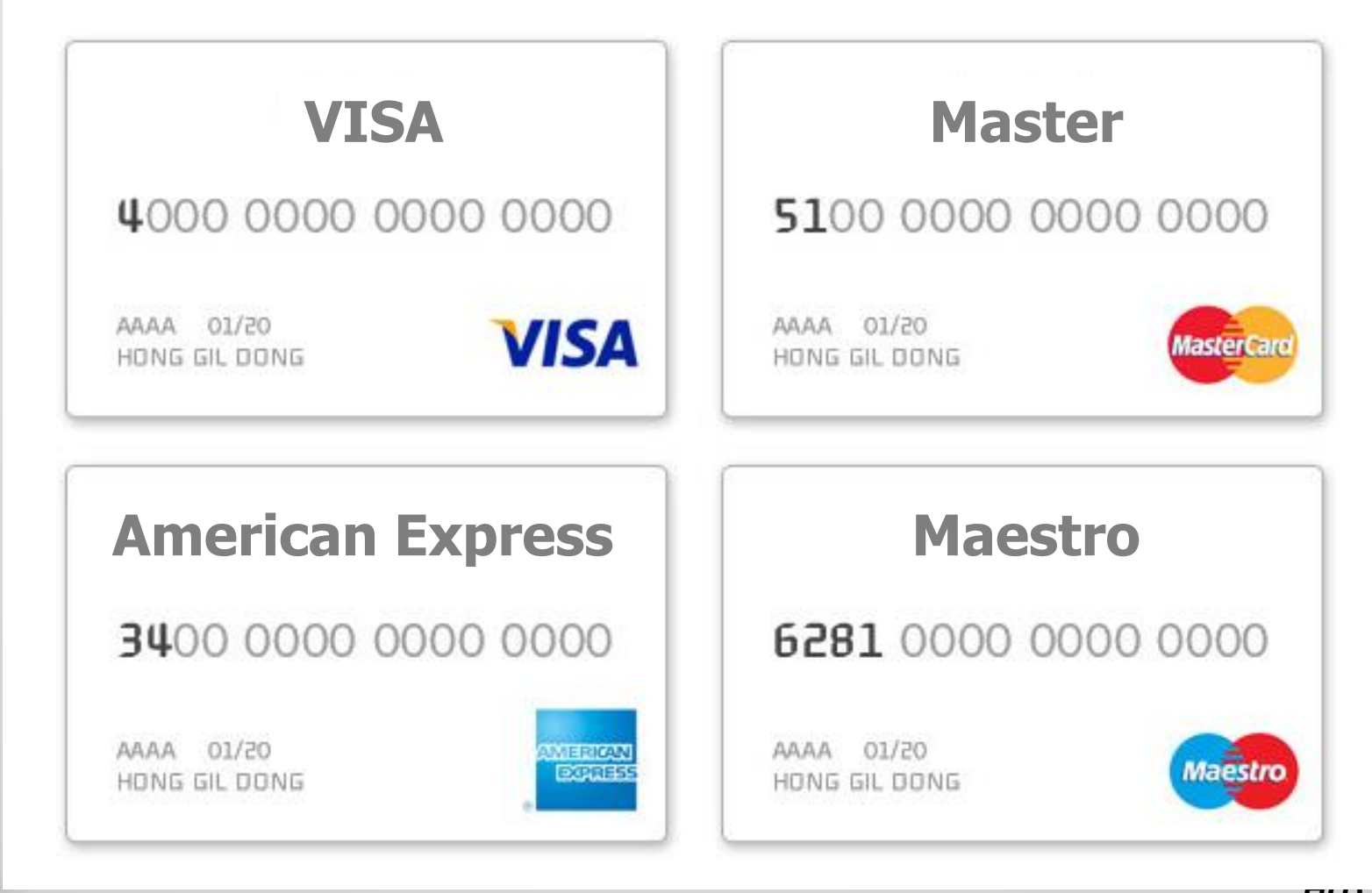
PTRACE_PEEKTEXT, **PTRACE_PEEKDATA**
Read a word at the address *addr* in the tracee's memory, returning the word as the result of the **ptrace**() call. Linux does not have separate text and data address spaces, so these two requests are currently equivalent. (*data* is ignored; but see NOTES.)

PTRACE_PEEKUSER
Read a word at offset *addr* in the tracee's USER area, which holds the registers and other information about the process (see *<sys/user.h>*). The word is returned as the result of the **ptrace**() call. Typically, the offset must be word-aligned, though this might vary by architecture. See NOTES. (*data* is ignored; but see NOTES.)

PTRACE_POKETEXT, **PTRACE_POKEDATA**
Copy the word *data* to the address *addr* in the tracee's memory. As for **PTRACE_PEEKTEXT** and **PTRACE_PEEKDATA**, these two requests are currently equivalent.

Card reader manipulation

- **Secret of card number**



Card reader manipulation

- **That's all?**

  - User can register a credit card on server and make a payment using it after Login

  - The user of transaction is identified by a serial number of card reader

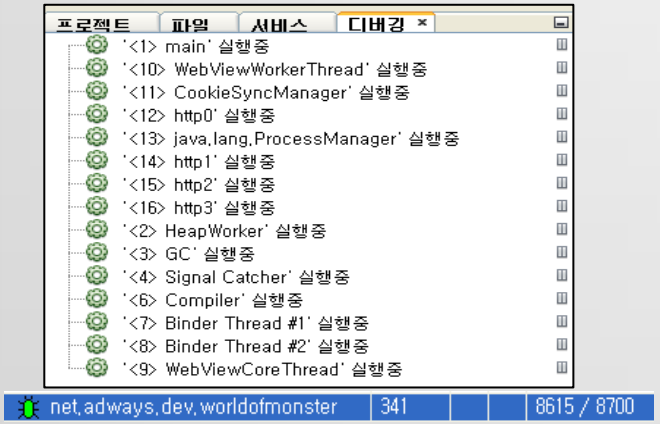
  - If you can find it, and if you are able to change it…. What happen?

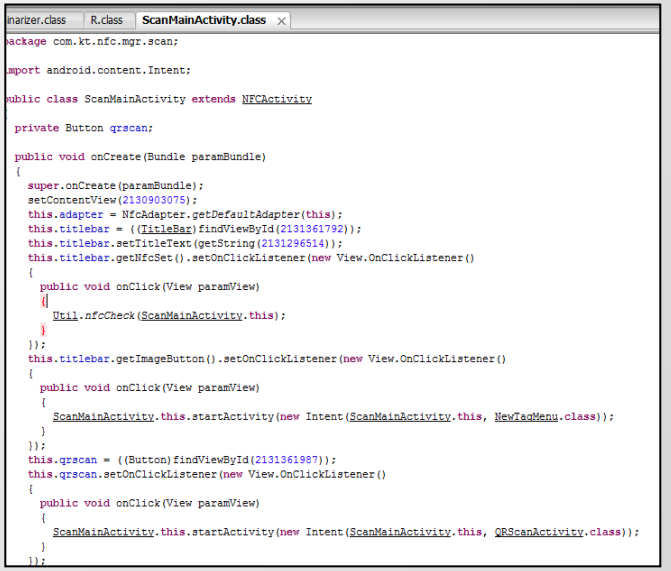
Vulnerability – Case

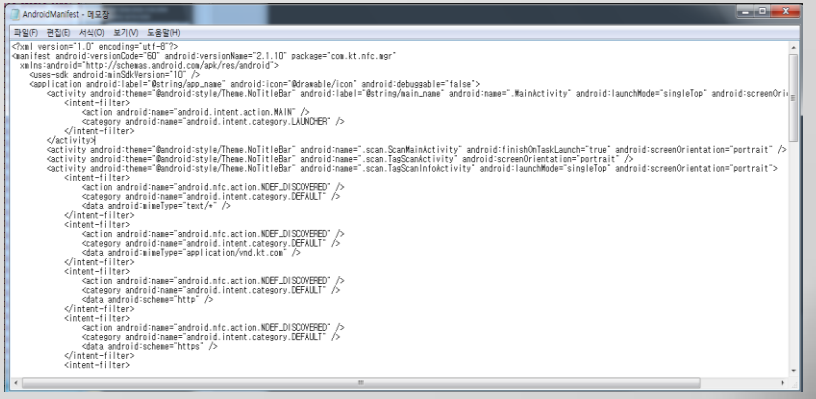
- **Binary protection for financial app**

  - Code obfuscation
  - Binary obfuscation
  - Binary integrity check
  - Debugger detection and …

Binary Protection

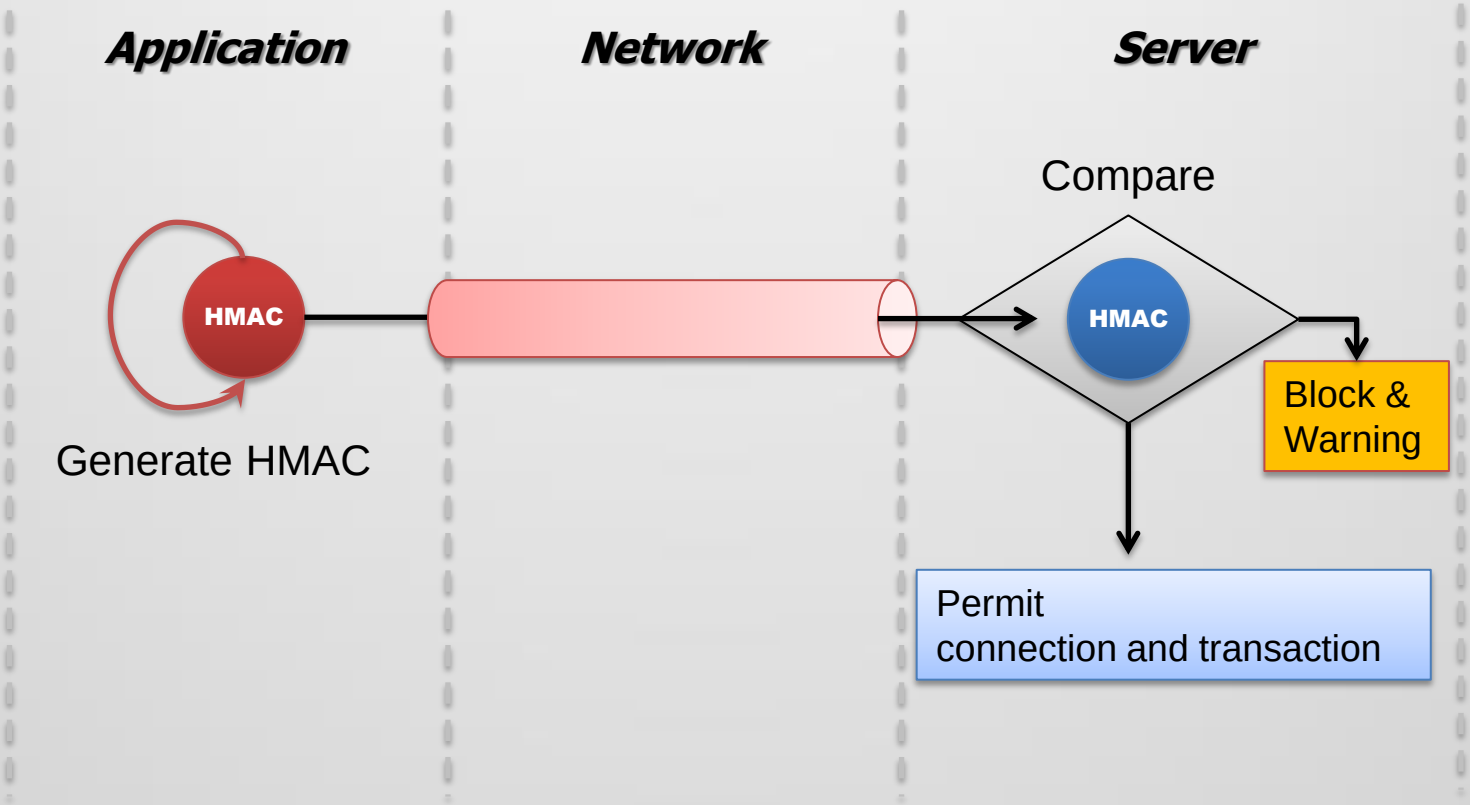




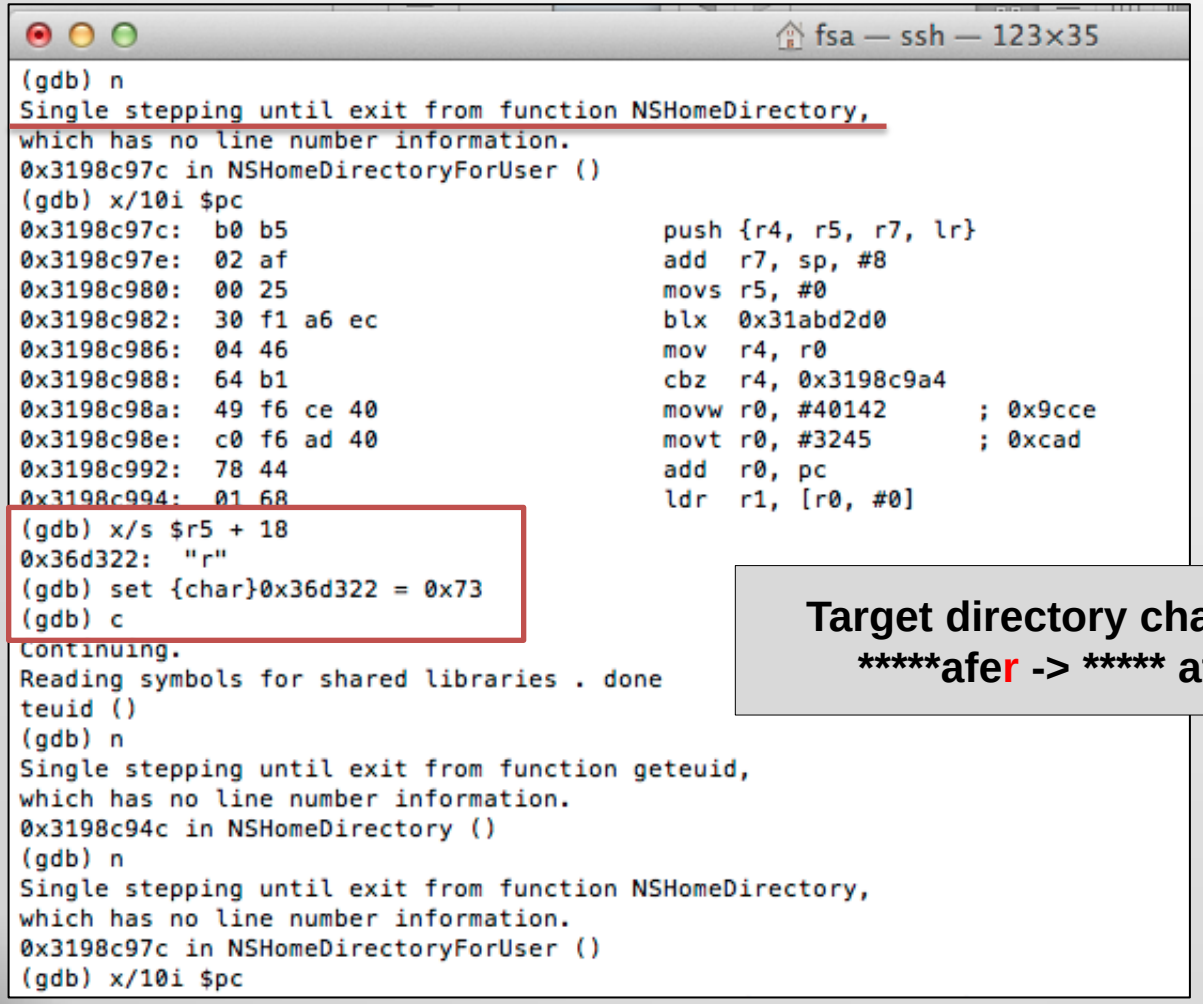


Binary Protection

- **Compare client HMAC with HMAC on server**



Binary Protection



Target directory changed
*******afer -> ***** afes**

Countermeasure

- **Ways to protect mobile payment**

  - Virtual Keypad
  - MDM(Mobile Device Management)
  - Authenticator(Internal, External)
  - Secure Element
  - Fingerprint
  - Anti virus
  - FDS(Fraud Detection System)

Approaches to test mobile payment

- **Security test and application design to make a secure mobile payment**
  - 1st goal is to protect a transaction from illegal access
  - But, various services are coming up, we need more effective approaches to test it
- **Set the direction for test based on**
  - Critical data
  - Process of the service
  - Data flow on architecture(Internal / External)
- **Threat modeling**
  - Figure out vulnerability from threats
- **Enough?**

Thank you

Question?

E-mail : ric3box@gmail.com
Facebook : ricebox0