TrustOTP: Transforming Smartphones into Secure One-Time Password Tokens

He Sun, Kun Sun, Yuewu Wang, and Jiwu Jing

Presented by Fengwei Zhang

Outline

- Introduction
- Motivation
- Architecture
- Implementation
- Evaluation
- Summary

Outline

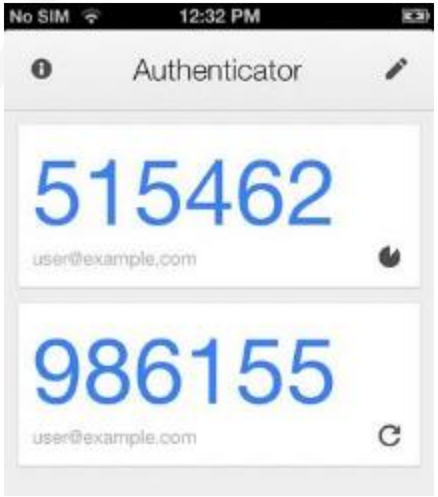
- Introduction
- Motivation
- Architecture
- Implementation
- Evaluation
- Summary

One-time Password (OTP)

- A password that is valid for only one login session or transaction
  - Not vulnerable to reply attacks
  - Widely used in Two-factor Authentication
  - HOTP (Hash-based OTP)
    - Event triggered, key & counter
  - TOTP (Time-based OTP)
    - Time synchronized, key & clock
  - Hardware token & software App

Existing Solutions

- Hardware-based
  - RSA SecurID
  - Yubikey
- Software-based
  - Google authenticator
  - McAfee one-time password



No SIM 12:32 PM

Authenticator

515462

user@example.com

986155

user@example.com

Outline

- Introduction
- Motivation
- Architecture
- Implementation
- Evaluation
- Summary

Limitation

- Hardware-based --- not flexible
  - Unprogrammable
  - Expensive

- Software-based --- not secure
  - Vulnerable to external attacks

Goals

- Confidentiality
  - Malicious mobile OS cannot compromise the keying material (seed) in the OTP generator
  - It cannot read the OTP
- Reliability and Availability
  - Trusted inputs (e.g., clock time) for the OTP genreator
  - Trusted display
  - OTP works even If mobile OS crashes
- Small TCB

TrustZone-related Work

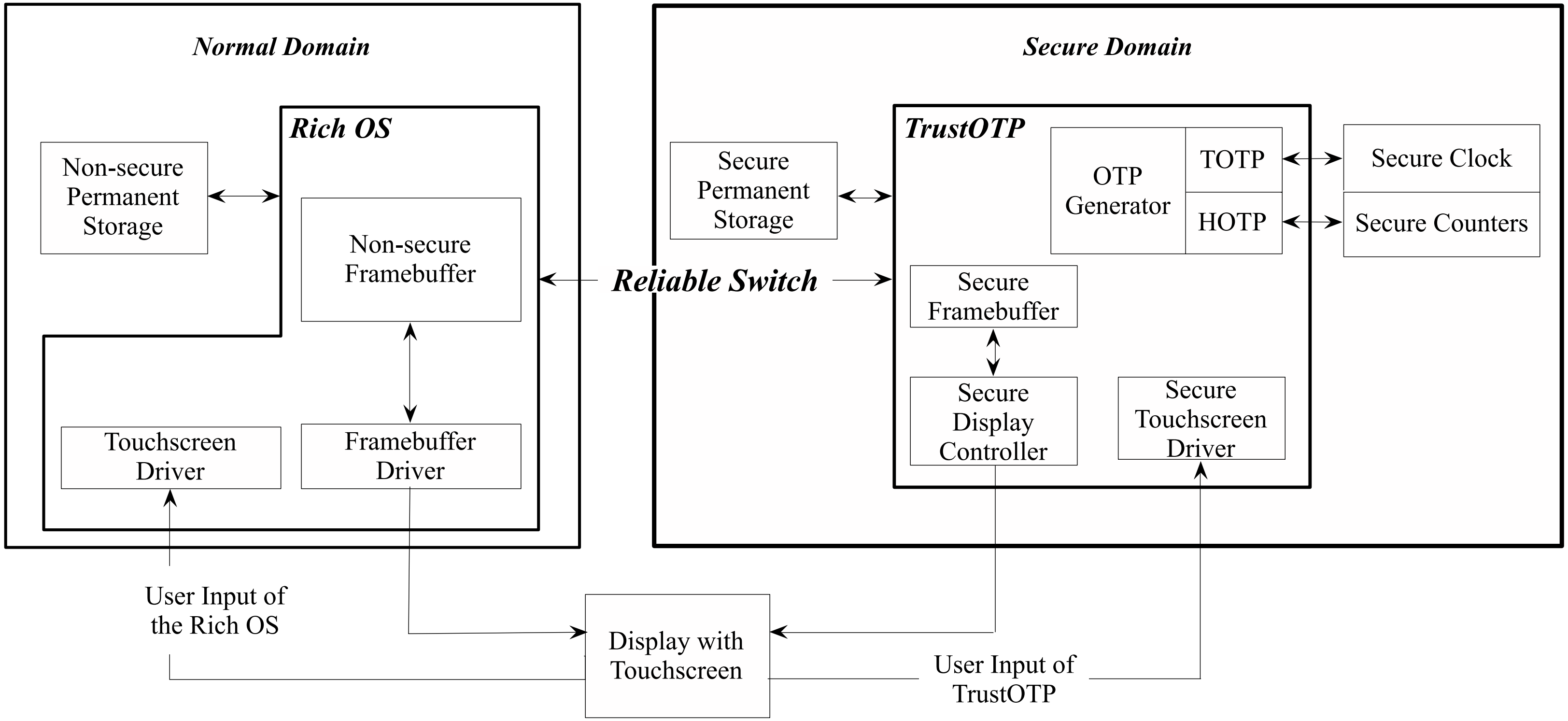
- TrustICE (Sun et al.[1])
  - Isolated Computing Environment in the normal domain
- SeCReT (Jang et al.[2])
  - Secure channel between secure domain and normal application
- Hypervision (Azab et al.[3])
  - Real-time kernel protection in the normal domain
- TrustDump (Sun et al.[4])
  - Reliable Memory Acquisition of the mobile OS
- Smartphone as location verification token for payments (Marforio et al.[5])
- Trusted Language Runtime for trusted applications in the secure domain (Santos et al.[6])

Outline

- Introduction
- Motivation
- Architecture
- Implementation
- Evaluation
- Summary

TrustOTP Architecture

- In the secure domain
- Shared I/O device with the rich OS
- Reliable switch between domains



Challenges

- Secure input and display though shared touchscreen
- Reliable switch
- Generator protection
  - Static code
  - Execution environment
- Availability

Outline

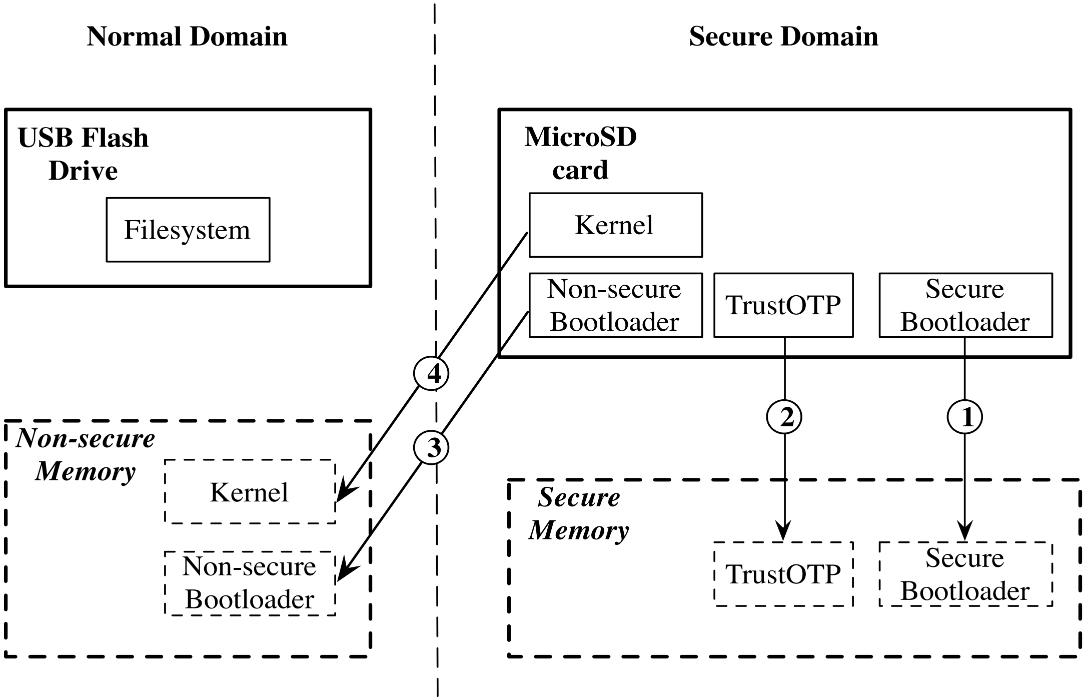
- Introduction
- Motivation
- Architecture
- Implementation
- Evaluation
- Summary

Security Analysis

- Information leakage
  - Generated OTPs
  - Shared keys
- Control flow tampering
  - Code integrity
  - Execution integrity (e.g., Interrupt)
- Denial-of-service
  - Switch between domains
  - Static & dynamic code
  - Display

Boot Sequence

- Secure storage
  - MicroSD card
- Memory Isolation
  - TZASC (TrustZone Address Space Controller)
  - Watermark mechanism
  - Secure boot
- Secure bootloader
  - Non-secure bootloader
  - Rich OS

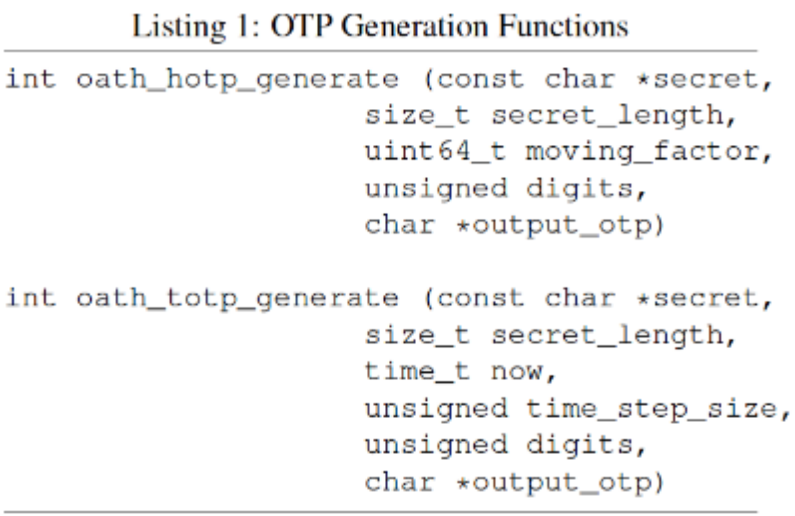


TrustOTP Triggering

- Reliable switch
  - Non-maskable interrupt (NMI)
    - The rich OS cannot block or intercept
  - Secure Interrupt (FIQ)
    - The rich OS cannot manipulate
  - Interrupt source (configurable)
    - Physical button
    - Timer

OTP Generation

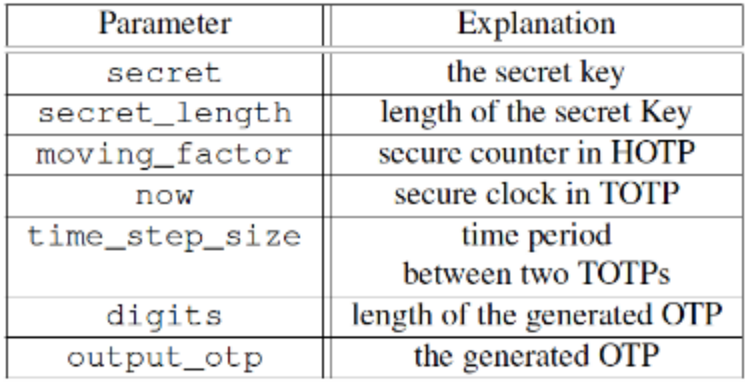
- Hash-based one-time password (HOTP)
  - Key, counter
- Time-based one-time password (TOTP)
  - Key, Clock



Listing 1: OTP Generation Functions

```
int oath_hotp_generate (const char *secret,
                        size_t secret_length,
                        uint64_t moving_factor,
                        unsigned digits,
                        char *output_otp)

int oath_totp_generate (const char *secret,
                        size_t secret_length,
                        time_t now,
                        unsigned time_step_size,
                        unsigned digits,
                        char *output_otp)
```



Parameter	Explanation
secret	the secret key
secret_length	length of the secret Key
moving_factor	secure counter in HOTP
now	secure clock in TOTP
time_step_size	time period between two TOTPs
digits	length of the generated OTP
output_otp	the generated OTP

OTP Display

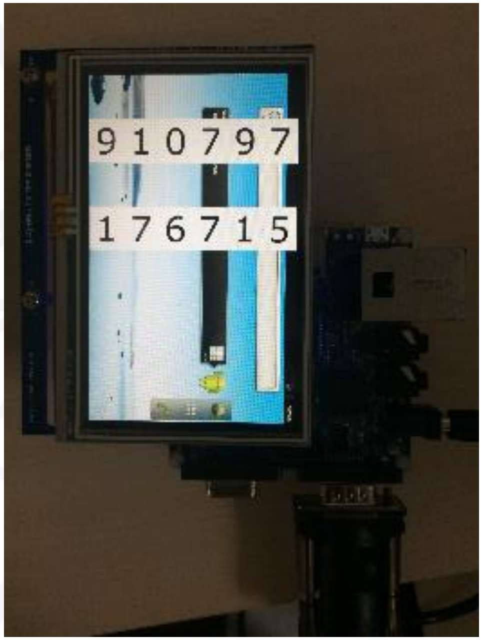
- Secure I/O
  - – Display: IPU (Image Processing Unit) + LCD
  - – Input: 4-wire resistive touchscreen
- User-friendly manner
  - – Rich OS and TrustOTP run concurrently
  - – Watchdog timer
  - – 1.5 seconds / cycle
    - 0.5 second for display
    - 1 second for input 2~3 numbers

Outline

- Introduction
- Motivation
- Architecture
- Implementation
- Evaluation
- Summary

Evaluation

- Freescale i.MX53 QSB
  - A Cortex-A8 1GHz processor
  - 1GB DD3 RAM
  - 4GB microSD card
- Monsoon power monitor
  - Power measurement
  - Power logging



TrustOTP Performance

- Before OTP display (60.48 ms)

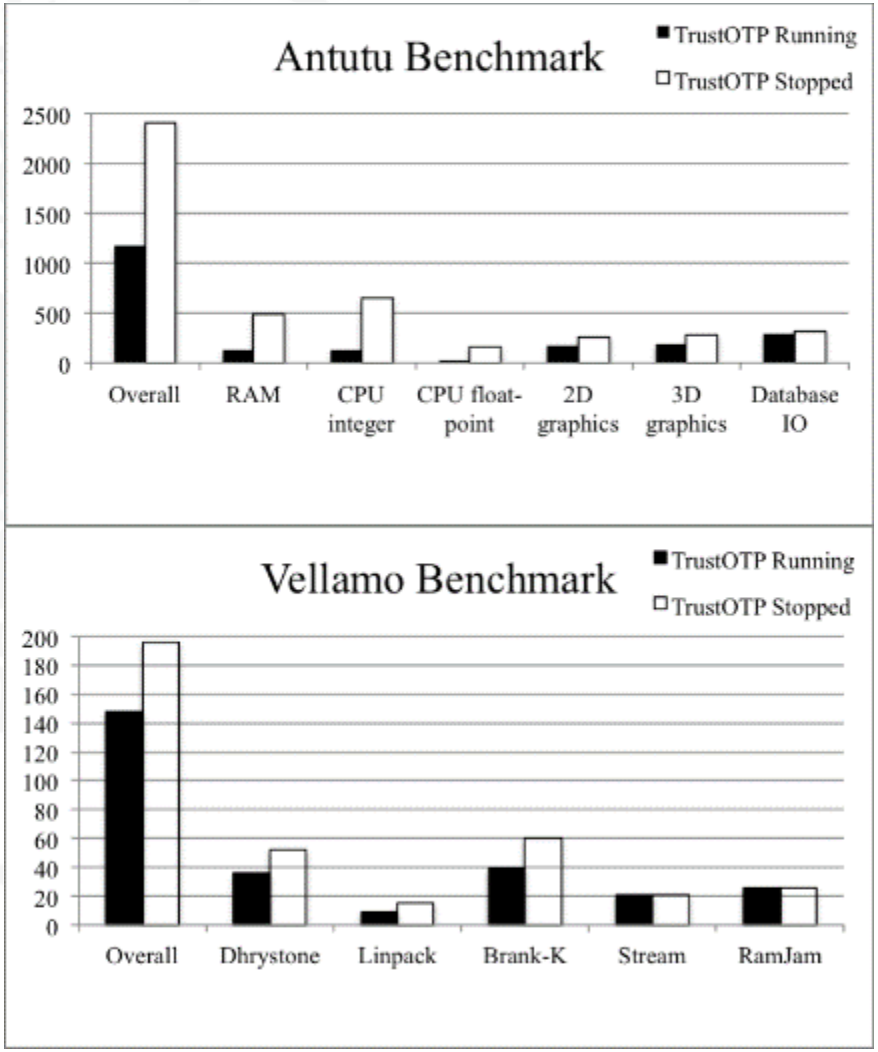
Step	Operation	Time (ms)
1	Domain Switching	0.002
2	Context Saving	0.0006
3	TOTP/HOTP Generation	0.048/0.044
4	Background Matching	49.85
5	OTP Drawing	8.029
6	IPU Check	2.22
7	Framebuffer Replacement	0.28

- After OTP display (7.52 ms)

Step	Operation	Time (ms)
1	Flushing IPU & Rich OS Recovery	7.47
2	Domain Switching	0.05

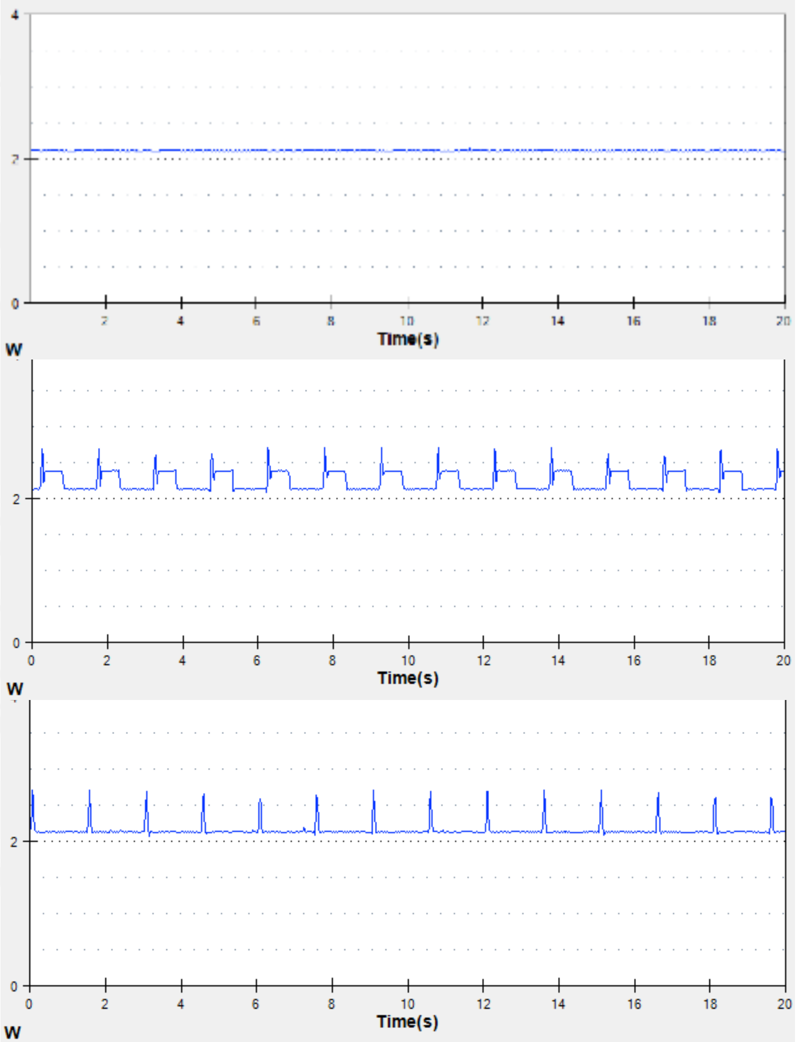
Impact on the Rich OS

- Rich OS vs. TrustOTP
- Anutu
  - CPU & RAM
  - I/O devices
- Vellamo



Power Consumption

- Rich OS
  - Average = 2,128 mW
- TrustOTP running
  - Average =2,230 mW
- TrustOTP without display



Outline

- Introduction
- Motivation
- Architecture
- Implementation
- Evaluation
- Summary

Summary

- TrustOTP: Hardware-assisted OTP Token on smartphones
  - Security (confidentiality, integrity, availability)
  - Flexibility (various and multiple OTPs)
- Low performance overhead on the Rich OS
  - No need to modify the Rich OS
  - Low power consumption

References

1. H. Sun, K. Sun, Y. Wang, J. Jing, and H. Wang, "TrustICE: Hardware-assisted Isolated Computing Environments on Mobile Devices," in Proceedings of the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'15), June 22-25, 2015.
2. J. Jang, S. Kong, M. Kim, D. Kim, and B. B. Kang, "Secret: Secure channel between rich execution environment and trusted execution environment," in 21st Annual Network and Distributed System Security Symposium, NDSS 2015, February 8-11, 2015.
3. A. M. Azab, P. Ning, J. Shah, Q. Chen, R. Bhutkar, G. Ganesh, J. Ma, and W. Shen, "Hypervision across worlds: Real-time kernel protection from the ARM trustzone secure world," in Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, November 3-7, 2014.
4. H. Sun, K. Sun, Y. Wang, J. Jing, and S. Jajodia, "Trustdump: Reliable memory acquisition on smartphones," in Proceedings of 19th European Symposium on Research in Computer Security (ESORICS'14), September 7-11, 2014.
5. C. Marforio, N. Karapanos, C. Soriente, K. Kostiainen, and S. Capkun, "Smartphones as practical and secure location verification tokens for payments," in 21st Annual Network and Distributed System Security Symposium, NDSS 2014, February 23-26, 2014.
6. N. Santos, H. Raj, S. Saroiu, and A. Wolman, "Using ARM trustzone to build a trusted language runtime for mobile applications," in Architectural Support for Programming Languages and Operating Systems, ASPLOS '14, March 1-5, 2014