# PV204 Security technologies

## Team projects

Petr Švenda
Faculty of Informatics, Masaryk University, Brno, CZ

CROCS
Centre for Research on Cryptography and Security

# Project idea: modern secure channel on certified smartcards

1. Study and analyze several Common Criteria security certificates
2. Implement secure channel between smartcard and host PC using ephemeral ECDH keys
3. Attack implementation of other team (obtain PIN, decrypt…)

*Optional: help us automatically process certificates by collecting regexes to match keywords and annotate text*

# Teams

- 3 people per team
  - Formed today (within group), available in IS
- Teams must use GitHub for cooperation
  - Distribute work load evenly between all members
  - Contribution from all team members must be visible in git (git commits from each member)
  - Your evaluation will be partially based on your participation
- The implementation of secure channel on the card side (JavaCard) must be done as new own code, for other parts, existing code can be used (but must make clear attribution to the original author(s))

# Basic hints on successful team work

- Form team from people with similar expectations
  - intended effort, final mark, interactions…
- Plan your work (GitHub milestones + issues)
- Don’t overcommit and fulfil your promises
- Agree on 4 personal session to work on project (at least 1 hour each) and block time in your calendar
  - Mail me the dates
- Every seminar 10 minutes reserved for team sync
  - Update your GitHub project milestones…

# Projects – timeline (details on next slides)

1. Select three target certificates, create repository (9.3.2020)
   - https://www.commoncriteriaportal.org/products/
   - No duplicate libraries allowed, active certificates only, FIFO, mail me!
2. Analyze certificates, report and presentation: 5 points (26.3.2020 ~~19.3.2020~~)
   - Read, understand and extract relevant information from certificates
   - Optionally also annotate the certificates text
   - Report (max. 4 pages A4) + presentation (your seminar)
3. Design and implement secure channel using ECDH: 15 points (23.4.2020 ~~9.4.2020~~)
   - Report (max. 4 pages A4) + presentation (your seminar group)
   - 10 APDU traces obtained from channel establishment and message transfer
4. Audit and attack other implementation: 10 points (11.5.2020)
   - Propose attacks, try to extract PIN
   - Presentation of finding (your seminar)

# PHASE1: ANALYSIS OF SECURITY CERTIFICATES

# Analyze three security certificates

- Must be from three different categories of devices
- ICs, Smart Cards and Smart Card-Related Devices and Systems
  - At least EAL5, at least one maintenance update
  - Download all relevant documents (report, security target…)
- Remaining two devices: at least EAL4, different categories

# Analyze three security certificates (cont.)

- Prepare presentation summarizing:
  - Basics of device certified (ToE), eval. lab …
  - What is the assumed attacker model
  - How was device scrutinized with respect to physical attacks, side channels…
  - List all referenced relevant Security Functional Components (SARs)
    - measures taken during development, discuss details
  - List Security Functional Components (SFRs)
    - security functions provided by product, discuss details
  - What is out of scope of certification
  - Own critical evaluation and conclusions (Would you buy the product? What you were missing? Are you convinced by eval. laboratory testing?)
- Hint: save pdf as text or use pdf2text, use pdf annotation

# PHASE 2

# Requirements

- All documentation and source code will be put into GitHub repository (add me @petrs as collaborator)
- Implementation for card (JavaCard lang) and PC side (any suitable lang, but Java is easiest due to usage of simulator) will be provided
- All members must participate, and the participation be visible from GitHub commits
- The implementation of card side will be executed in jcardsim simulator, and must convert under javacard convertor
- The functionality requirements are listed on next slide(s)

# High-level functionality overview

- User obtains card with pre-personalized PIN and printed PIN (4 digits)

1. User inserts card into reader and type PIN to PC for authentication
2. Both card and PC are mutually authenticated
3. Cryptographic keys for protected channel are established
4. All subsequent exchanged data are sent via protected channel (confidentiality, integrity, freshness…)
5. All temporary secrets are erased when channel is closed
6. Start again from Step 1.

# Attacker model

- Can eavesdrop all communication (“wire” between card and reader)
- Can manipulate (inject, remove, delay, modify…) messages on wire
- Can compromise (in future) for limited period of time user PC (=> keys stored in memory)
- Attacker “wins” if:
  - Authenticated instead of card or PC
  - Obtains value of PIN
  - Decrypts communication, modify or drop without detection, send older…

# Functional and security requirements

- Card and user share value of authentication PIN (4 digits)
- PIN is set to card during installation via installation parameters (and distributed to user, e.g., printed)
- User inserts card to reader and provide PIN
- Initial secret for secure channel is established using ECDH with shares authenticated using PIN
  - PIN is used only to authenticate ECDH shares and never transmitted to card or back
- Both card and user (via PC) are authenticated before every session
- Established ECDH share is used to derive session keys
- Subsequent data exchanged between card and PC are protected by secure channel

# Functional and security requirements

- Session ends by explicit command from PC or can be interrupted prematurely by sudden removal of card from the reader
  - All temporary secrets are properly erased
- Create test with demonstration of the functionality
  - (High-level functionality overview workflow)
  - Authenticate, establish channel

# PHASE 3 – DESIGN AND CODE REVIEW

# Project – code review part

- Analyze and try to attack implementation of secure channel of assigned team
- Both design and implementation vulnerabilities are welcomed
- Especially:
  - If PIN value can be retrieved (mitm, offline bruteforcing, failed checks….)
  - If older session can be replayed
  - If authentication of attacker instead of legitimate party can be obtained
  - If multiple instances of parallel protocol runs can be misused
- Be creative, describe well under what circumstances is attack possible and what is impact
- The layers of defense shall be independent. So even if you cannot carry the complete attack, but only breach one of layers, still report it

# Project – code review part (cont.)

- If you need more info, contact target team members
  - Write down log of your interactions with target team
- Open GitHub issues in target repository
  - (repository of team you are reviewing project for)
  - for every separate issue you will find + description
- Write 2-3 pages A4 report from code review
  - What tests did you performed (automated tests, manual review)
  - What did you focus on
  - What did you find out, how serious are the problems
- Prepare presentation for the last lecture May 11

# Present results (Finding summary)

- Location of the vulnerability
- Vulnerability class
- Vulnerability description
- Prerequisites (for exploiting vulnerability)
- Business impact (on assets)
- Risk, Severity, Probability
- Remediation (how to fix)
  - Describe idea how to fix the vulnerabilities identified
- For the issues found, open GitHub issues tickets

# Finding summary - example

**Problem identification: DSA-1571-1 openssl**
**Severity**: critical
**Risk:** high - directly exploitable by external attacker
**Problem description**: crypto/rand/md_rand.c:276 & 473 – The random number generator in Debian's openssl package is predictable. This is caused by an incorrect Debian-specific change to the openssl package. One of the sources of a randomness based on usage of uninitialized buffer *buff* is removed.
**Remediation**: revert back to usage of uninitialized buffer *buff*

# Code review submission

- Presentation will be for all groups at once instead of lecture on 11th May, 16:00
- Presentations: 10 minutes per team + discussion
  - By all team members
  - Please keep the assigned time – we need to fit all groups into 2 hours, will be online via Zoom
- Prepare PPT or PDF slides
- Upload to IS vault ‘Project: Phase3 (review)’
  - List of issues found from design and code review (report)
  - Presentation slides

# Review assignment – team reviews next project (last time reviews the first one)

| | 23.4.2020 |
|---|---|
| **13:00-13:20** | Amal, Jan, Marek |
| **13:20-13:40** | Nagy, Rychlý |
| | |
| **14:00-14:20** | Grabovský, Sharma, Šorf |
| **14:00-14:20** | Ankur, Tomas, Minh |
| | |
| **14:40-15:00** | Florian, Solodkova, Vondráček |
| **15:00-15:20** | Šanta, Pavúk, Gennertová |
| **15:20-15:40** | Obuch, Oravec, Varga |
| **15:40-16:00** | Berka, Jelinek, Galikova |
| **16:00-16:20** | Klunko, Šnajdr, Zat'ovič |

# Project groups

- GROUP1: Amal Chukkinin, Marek Hrašna and Jan Kubeša
  - Dencrypt Talk for iPhone version 4.2.794
  - SMARTY IQ-GPRS/LTE, Version 1.0
  - NXP JCOP 5.1 on SN100.C48 Secure Element
- GROUP2: Jen Jelínek, Šimon Berka, Miriam Gáliková
  - NXP Crypto Library V3.1.x on P6022y VB
  - Trusted Plattform Module SLB9670_2.0 v7.83.3358.00, v7.83.3360.00
  - Huawei 3900 Series LTE eNodeB Access Control Software version V100R008C01SPC820
- GROUP3: Mykhailo Klunko, Vojtěch Šnajdr, Daniel Zaťovič
  - https://github.com/danzatt/pv204_project
  - NXP eDoc Suite v3.0 - cryptovision ePasslet Suite
  - NPCT7xx TPM2.0 rev1.38 Hardware version LAG019 Firmware version 7.2.1.0
  - Thales Trusted Security Filter TSF201

# Project groups

- GROUP4: Ankur Lochab, Tomáš Madeja, Tran Anh Minh
  - https://github.com/TomasMadeja/pv204_Team_Project_Supercalifragilisticexpialidocious
  - IC&SC: FM1280 V05
  - Network Devices: genuscreen 7.0
  - Other Devices: Thinklogical TLX1280
- GROUP5: Martin Vondracek, Elena Solodkova, Oldrich Tristan Florian
  - https://github.com/mvondracek/PV204_smartcards_Emerald
  - NXP Secure Smart Card Controller P60x080/052/040yVC(Y/Z/A)/yVG with IC Dedicated Software
  - MIFARE DESFire EV2
  - WatchGuard Firebox Security Appliances with Fireware v11.11 and WatchGuard Dimension 2.1
- GROUP6: Philippe Bize, Luc Tatu
  - S3D350A / S3D300A / S3D264A / S3D232A / S3D200A / S3K350A / S3K300A 32-bit RISC Microcontroller for Smart Card with optional AT1 Secure Libraries
  - IDeal Citiz v2.1 Open platform
  - Motion Sensor for Digital (smart) Tachographs Lesikar TACH3 HW version 02, SW version 03 r43

# Project groups

- GROUP7: Matěj Grabovský, Milan Šorf, Nomit Sharma
  - NXP JCOP 4 P71 (smart cards category),
  - vinCERTcore, versión 4.0.5.5733 (products for digital signatures)
  - genuscreen 7.0 (network-related category)
- GROUP8: Marek Šanta, Štěpánka Gennertová, Michal Pavúk
  - Fox Fort Hardware Data Diode FFHDD3_1/10 z Boundary protection
  - ST31G480 C01 including optional cryptographic library NESLIB and optional technologies MIFARE DESFire EV1 and MIFARE Plus X,
  - Digital Tachograph DTCO 1381 Release 3.0

# Project groups

- GROUP9: Denis Varga, Samuel Obuch, Roman Oravec
  - https://github.com/St4lkerino/PV204_PROJEKT
  - SLS 32TLC00xS(M) CIPURSE™4move v1.00.00
  - Nexor Sentinel 3E Filtering System
  - Infineon Technologies AG Trusted Platform Module SLB9665_2.0
- GROUP10: Daniel Rychlý, Imrich Nagy
  - ST33TPHF2E mode TPM 2.0, TPM Firmware 0x49.0x40 & 0x49.0x41
  - Red Hat Enterprise LinuxVersion 7.1
  - NXP JCOP 5.2 on SN100.C58 Secure Element