IN SOVIET RUSSIA SMARTCARD HACKS YOU

@DEF CON 26 - 2018

Eric Sesterhenn <eric.sesterhenn@x41-dsec.de>

2018

X41 D-SEC GmbH

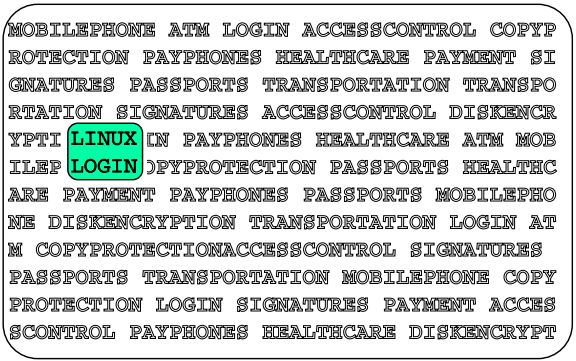
whoami

- Eric Sesterhenn
- Principal Security Consultant
- Pentesting/Code Auditing at X41

Disclaimer

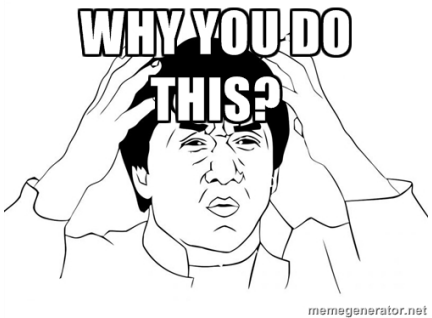
- The issues presented here have been reported and fixed!
- These are open source projects - help them!
- I am not interested in testing / debugging proprietary stuff in my spare time.

Targets

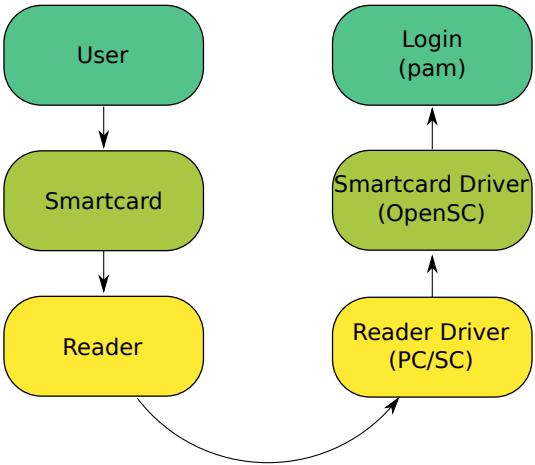

MOBILEPHONE ATM LOGIN ACCESSCONTROL COPYP
ROTECTION PAYPHONES HEALTHCARE PAYMENT SI
GNATURES PASSPORTS TRANSPORTATION TRANSPO
RTATION SIGNATURES ACCESSCONTROL DISKENCR
YPTI LINUX N PAYPHONES HEALTHCARE ATM MOB
ILEP LOGIN PYPROTECTION PASSPORTS HEALTHC
ARE PAYMENT PAYPHONES PASSPORTS MOBILEPHO
NE DISKENCRYPTION TRANSPORTATION LOGIN AT
M COPYPROTECTIONACCESSCONTROL SIGNATURES
PASSPORTS TRANSPORTATION MOBILEPHONE COPY
PROTECTION LOGIN SIGNATURES PAYMENT ACCES
SCONTROL PAYPHONES HEALTHCARE DISKENCRYPT

Why?

- Smartcards control authentication!
- Authentication runs as root!
- Users and programmers subconsciously trust the smartcard!



Smartcards


User
Smartcard
Reader
Login
(pam)
Smartcard Driver
(OpenSC)
Reader Driver
(PC/SC)

What is a Smartcard?

- Physical, tamper-proof device
- Designed to keep information secret
- Contains memory and a processor

https://en.wikipedia.org/wiki/Smart_card#/media/File:SmartCardPinout.svg

Application Protocol Data Unit

- APDUs form the protocol to talk to smartcards
- ISO/IEC 7816-4 Identification cards - Integrated circuit cards
- T=0 is character oriented / T=1 is block-oriented
- Verify: 00 20 00 01 04 31323334

CLA	INS	P1	P2	L_C	Data
1	1	1	1	0-3	N_C

PC/SC API

- PC/SC API can be used on win and *nix
- Other libraries have a similar interface

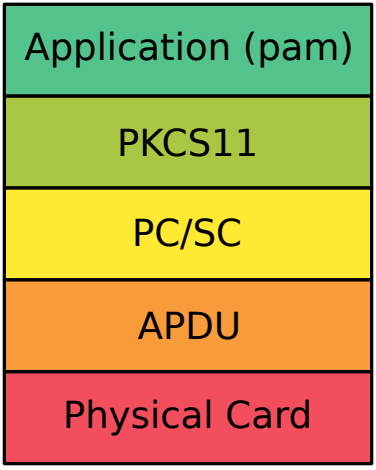
```
LONG WINAPI SCardTransmit(
  SCARDHANDLE         hCard,
  LPCSCARD_IO_REQUEST pioSendPci,
  LPCBYTE             pbSendBuffer,
  DWORD               cbSendLength,
  PSCARD_IO_REQUEST   pioRecvPci,
  LPBYTE              pbRecvBuffer,
  LPDWORD             pcbRecvLength
);
```

PKCS11

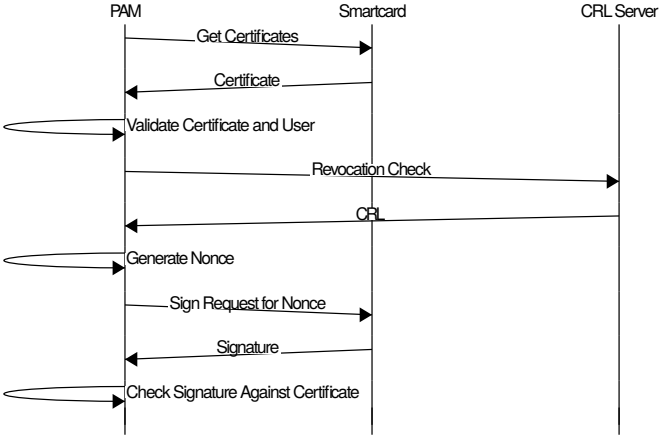
- PKCS11 is a platform independent API for cryptographic token
- Supported by OpenSSL, browsers,... (eg. via libp11)
- Windows uses smartcard Minidriver now
- Driver for each card, uses ATR to match

```
CK_RV C_FindObjectsInit(
  CK_SESSION_HANDLE hSession,
  CK_ATTRIBUTE_PTR pTemplate,
  CK_ULONG ulCount
);
```

Smartcard Stack Summary



Smartcard for Sign-On



Trust the Smartcard

- Driver developers trust the smartcard!
- Let's abuse that
- Mess with the card responses

Bugs

Project	# Bugs
libykneomgr	1
OpenSC	Over 9000 ;-)
pam_pkcs11	1
smartcardservices	2
Yubico-Piv	2

No, I did not fuzz the &$#?@! out of it...
but guess which one I fuzzed the most ;-) Thanks to Frank Morgner for fixing!

Apple Smartcardservices

```
do {
  cacreturn = cacToken.exchangeAPDU(command, sizeof(command), result,
   ↪  resultLength);
  if ((cacreturn & 0xFF00) != 0x6300)
    CACError::check(cacreturn);
...
  memcpy(certificate + certificateLength, result, resultLength - 2);
  certificateLength += resultLength - 2;
  // Number of bytes to fetch next time around is in the last byte
  // returned.
  command[4] = cacreturn & 0xFF;
} while ((cacreturn & 0xFF00) == 0x6300);
```

OpenSC - CryptoFlex

```
u8 buf[2048], *p = buf;
size_t bufsize, keysize;

sc_format_path("I1012", &path);
r = sc_select_file(card, &path, &file);
if (r)
  return 2;
bufsize = file->size;
sc_file_free(file);
r = sc_read_binary(card, 0, buf, bufsize, 0);
```

Popping calcs...

```
snakebyte@smartcard:~$ cryptoflex-tool
Usage: cryptoflex-tool [OPTIONS]
Options:
  -l, --list-keys              Lists all keys in a public key file
  -c, --create-key-files <arg>  Creates new RSA key files for <arg> keys
  -P, --create-pin-file <arg>   Creates a new CHV<arg> file
  -g, --generate-key           Generates a new RSA key pair
  -R, --read-key               Reads a public key from the card
  -V, --verify-pin             Verifies CHV1 before issuing commands
  -k, --key-num <arg>          Selects which key number to operate on [1]
  -a, --app-df <arg>           Selects the DF to operate in
  -p, --prkey-file <arg>       Private key file
  -u, --pubkey-file <arg>      Public key file
  -e, --exponent <arg>         The RSA exponent to use in key generation [3]
  -m, --modulus-length <arg>   Modulus length to use in key generation [1024]
  -r, --reader <arg>           Uses reader <arg>
  -w, --wait                   Wait for card insertion
  -v, --verbose                Verbose operation. Use several times to enable debug output
.
snakebyte@smartcard:~$ cryptoflex-tool -R
Using reader with a card: libfuzzy
Using card driver: Schlumberger Multiflex/Cryptoflex
Unable to read public key file: Card command failed
bc 1.06.95
Copyright 1991-1994, 1997, 1998, 2000, 2004, 2006 Free Software Foundation, Inc.
This is free software with ABSOLUTELY NO WARRANTY.
For details type `warranty'.
3+4
7
:-)
```

Basic Smartcard Exploitation in 2018

- Basiccard gives you nice control,... yes BASIC!
- Example exploit (Kevin) will be released to the public at beVX
- Other methods would be SIMtrace or certain Javacards

YUBICO PIV

```
if(*out_len + recv_len - 2 > max_out) {
  fprintf(stderr,
    ↪  "Output buffer to small, wanted to write %lu, max was %lu.",
    ↪  *out_len + recv_len - 2, max_out);
}
if(out_data) {
  memcpy(out_data, data, recv_len - 2);
  out_data += recv_len - 2;
  *out_len += recv_len - 2;
}
```

Logging in...


Debian GNU/Linux 9 smartcard tty3
Hint: Num Lock on
smartcard login: _

Challenges in fuzzing a protocol

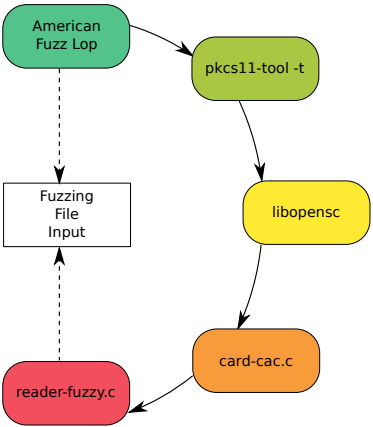
- Most modern fuzzers are file-oriented
- Radamsa: Generates a corpus of files
- Hongfuzz: passes a file (filename different each run)
- libfuzzer: passes a buffer and length
- AFL: passes a file

Challenges in fuzzing a protocol

- SCardTransmit() tells us how much data it expects
- Read this from a file on each call and error out if EOF
- No complicated poll handling like for network sockets required

How to fuzz - OpenSC

- reader-fuzzy.c
- Implements a (virtual) smartcard reader interface
- Responds with malicious data read from file (OPENSC_FUZZ_FILE)
- Have fun with AFL



How to fuzz - Winscard and PC/SC

- Winscard(.dll) on Linux and Unix
- For proprietary code
- Preload the library
- Have fun with non-feedback fuzzers (e.g. radamsa) or AFL in qemu mode

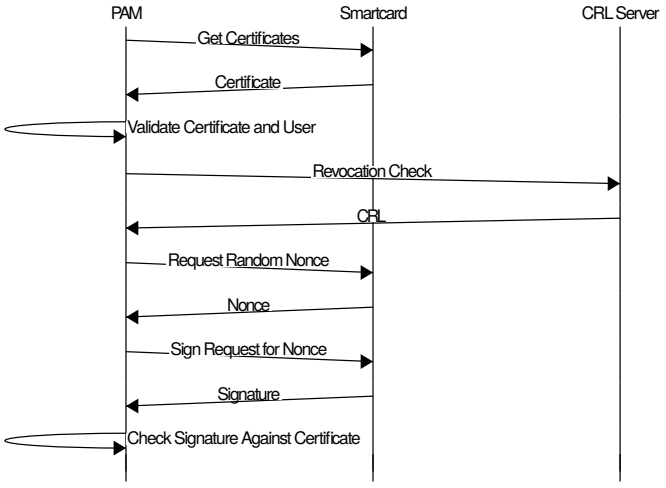
How to fuzz - Winscard 2

- Tavis loadlibrary
- Extended to support Winscard drivers
- Fuzz the windows drivers on linux without all the overhead

Smartcard fuzzing

- Released now!
- https://github.com/x41sec/x41-smartcard-fuzzing

pam_pkcs11: Replay an Authentication



Roadblocks

- Channel back to card is quite limited
- Might need to use revocation list check for information leaks
- Interaction during exploitation not possible with basiccard, get SIMtrace for that
- But: A single bitflip from false to true during login can be enough :)

Takeaways / Conclusions

- Think about trust models!
- Some security measures increase your attack surface big time!
- Fuzz Everything!
- Limit attack surface by disabling certain drivers.
- Do not write drivers in C ;-)

Thanks

- Q & A
- https://github.com/x41sec/x41-smartcard-fuzzing
- eric.sesterhenn@x41-dsec.de
- Sorry no Twitter... stalk me on LinkedIn if you must ;-)