Introduction to InfoSec – Recitation 7

Nir Krakowski (nirkrako at post.tau.ac.il)
Itamar Gilad (itamargi at post.tau.ac.il)

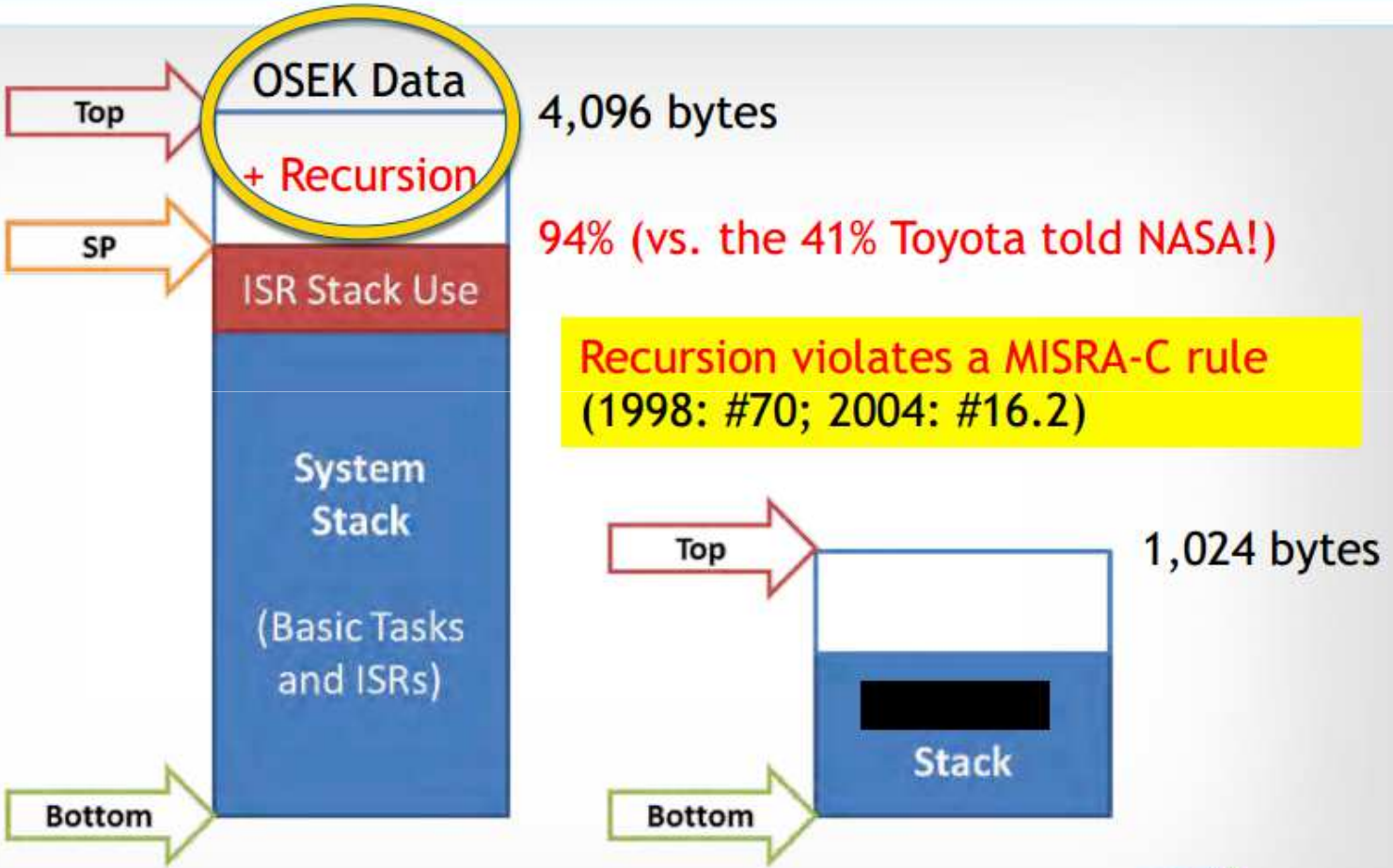
Today

- More vulnerability types!
- More logical
- More illogical

Ariane 5

- A space-launch platform by the ESA / CNES
- Controle software written in Ada, and taken from the Ariane 4
- Redundant hardware (2 identical sets)
- 37 seconds after launch, a cast from 64-bit float to 16-bit signed integer caused a processor trap
- On the Ariane 4, the values were considered to be physically limited
- No one considered the new parameters for the Ariane 5
-

STACK ANALYSIS FOR 2005 CAMRY L4



Slide taken from a technical report by Michael Barr prepared for the BOOKOUT V. TOYOTA court case

BARR group

Let's get back on topic

What's wrong here?

```
inp_str = user supplied data
char * tmp = strstr(inp_str, "%n")
*tmp = '\0';
printf(inp_str);
```

Brief analysis

- Attacker omits the token completely
- String will be far longer than expected
- Attacker can read stack contents – Information Leakage Vulnerability

- Since attacker can also control printf's format string, they could just as well read throught the stack in another way – "%x%x%x%x…"

Information Leaks

- Anything that the attacker can learn despite not having the right to
- Often serves as a way to make the attack feasible / more efficient
- Example – ASLR can defeat simple ROP exploits, but the ability to read arbitrary memory and re-write the ROP chain can create a "perfect" ROP exploit

What's wrong here?

```
hashOut.data = hashes + SSL_MD5_DIGEST_LEN;
hashOut.length = SSL_SHA1_DIGEST_LEN;
if ((err = SSLFreeBuffer(&hashCtx)) != 0)
    goto fail;
if ((err = ReadyHash(&SSLHashSHA1, &hashCtx)) != 0)
    goto fail;
if ((err = SSLHashSHA1.update(&hashCtx, &clientRandom)) != 0)
    goto fail;
if ((err = SSLHashSHA1.update(&hashCtx, &serverRandom)) != 0)
    goto fail;
if ((err = SSLHashSHA1.update(&hashCtx, &signedParams)) != 0)
    goto fail;
    goto fail;
if ((err = SSLHashSHA1.final(&hashCtx, &hashOut)) != 0)
    goto fail;

err = sslRawVerify(...);
```

What's wrong here?

```
…
if ((err = SSLHashSHA1.update(&hashCtx, &serverRandom)) != 0)
    goto fail;
if ((err = SSLHashSHA1.update(&hashCtx, &signedParams)) != 0)
    goto fail;
    goto fail;  /* THIS LINE SHOULD NOT BE HERE */
if ((err = SSLHashSHA1.final(&hashCtx, &hashOut)) != 0)
    goto fail;
…
```

Brief analysis

- The first ‘goto’ is “in” the if-statement
- **The second one isn’t, so it will always be run!**

- In this case, this led to a compromise of the SSL/TLS security for many versions of iOS up to iOS 7.0.6

- Goto’s aren’t bad! Bad programmers are bad!
- **In this recitation – we’re interested in bad programmers!**

Directory Traversal

- Assume an otherwise secure FTP server
- Supports requests for
  - GET [file]
  - PUT [file]
  - CD [path]
  - etc.
- CD requests are well filtered to remain within the exposed ('public') directory
- But what about GET and PUT requests?
- Try using escaping sequences – "../", "/////", etc.

Command Injection

```
def perform_calculation(expression):
    exec("ret = %s")
    return ret
```

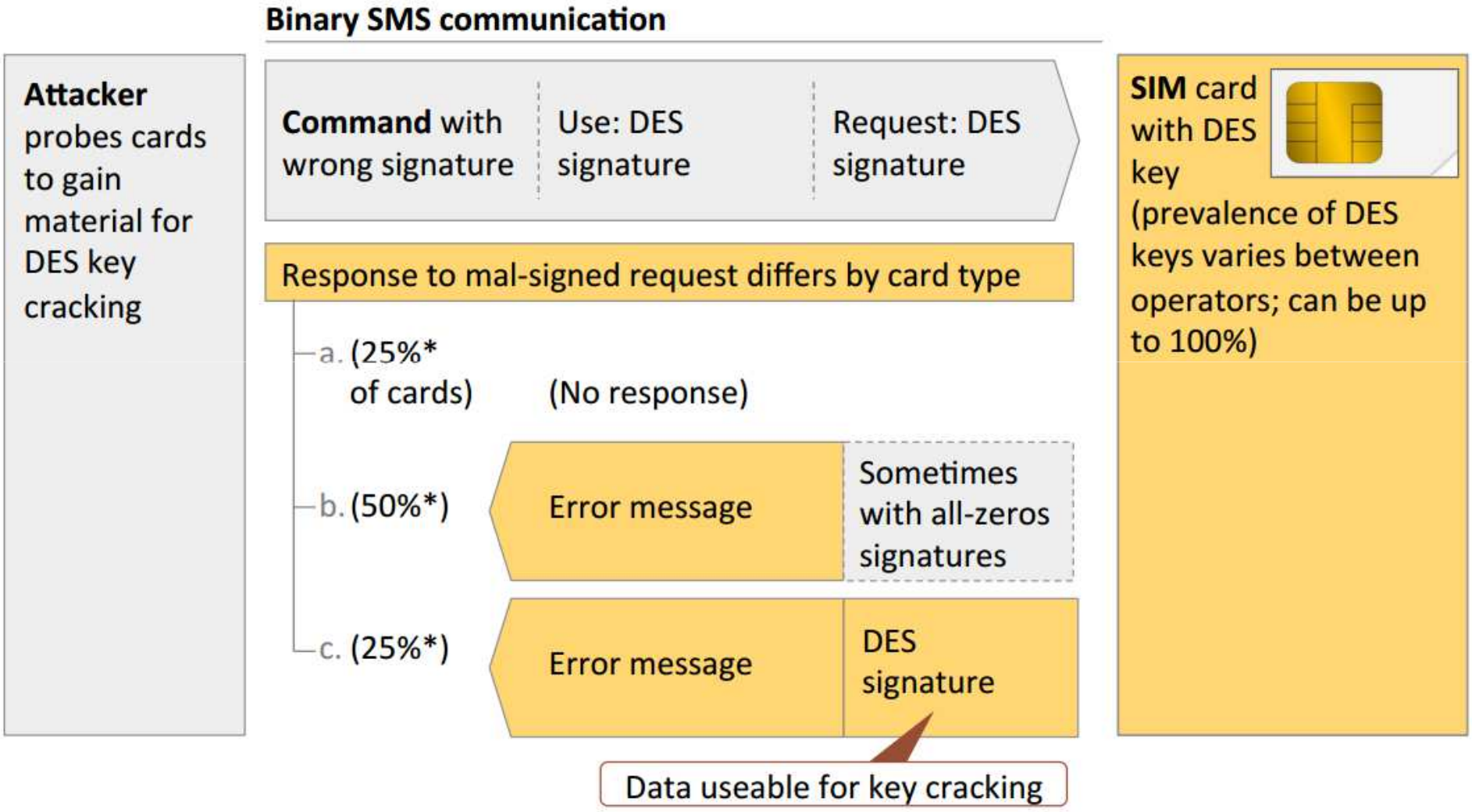
- The user can control 'expression'
- And can thus run arbitrary python code!
- See also: eval()

CVE-2010-2568

- Windows shell wants to display an LNK file
- LNK's icon is stored in a CPL file
- CPL file is actually a standard PE (exe) file, which will be loaded and initialized in the windows shell process

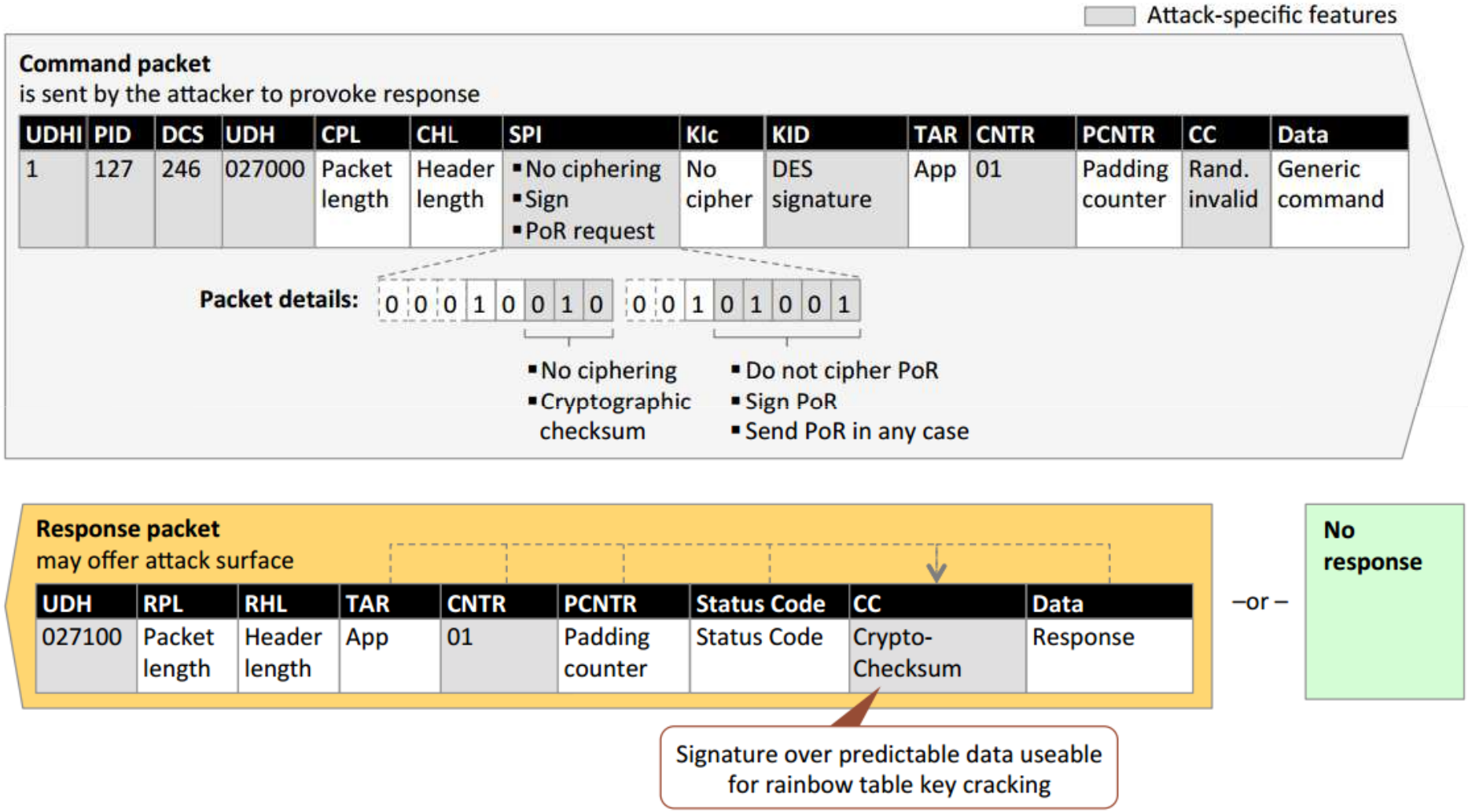
- When installed at a location the user will see (like the root of a removable drive) – the attacker's code will be executed!

OTA error handling is underspecified, possibly opening attack surface



Slides taken from Karsten Nohl's talk at BlackHat 2013

Attacker SMS asks for DES-signed SMS response with fully predictable content



Slides taken from Karsten Nohl's talk at BlackHat 2013

Logical Vulnerabilities

- Often a higher-level mistake (i.e.: not a buffer size check, but a whole concept)
- More common when there are quick-and-dirty solutions, or when someone takes a shortcut
- Can subvert most protection mechanisms with one small (large) mistake
- Exploitation is usually easier and much more reliable
- Much harder to find automatically, since there are fewer clear patterns

Side Channel Attacks

- You cannot get what you want directly
- So you'll get it indirectly!

- By measuring the time something takes (Timing analysis)
- By measuring the power usage of a processor/device (Power analysis)
- By generating faults (Differential Fault Analysis)
- By measuring acoustic noise (Acoustic analysis)
- By measuring RF emissions (TEMPEST)
- By reading uninitialized data (Data remanence)

Questions?

- חג פסח שמח!