Small bugs and big security problems

(Professional Master in Information Security)

PRO.M.IS
security built in

Dragos Ilie (dragos.ilie@bth.se)

agenda

- Bugs: the danger of buffer overflows

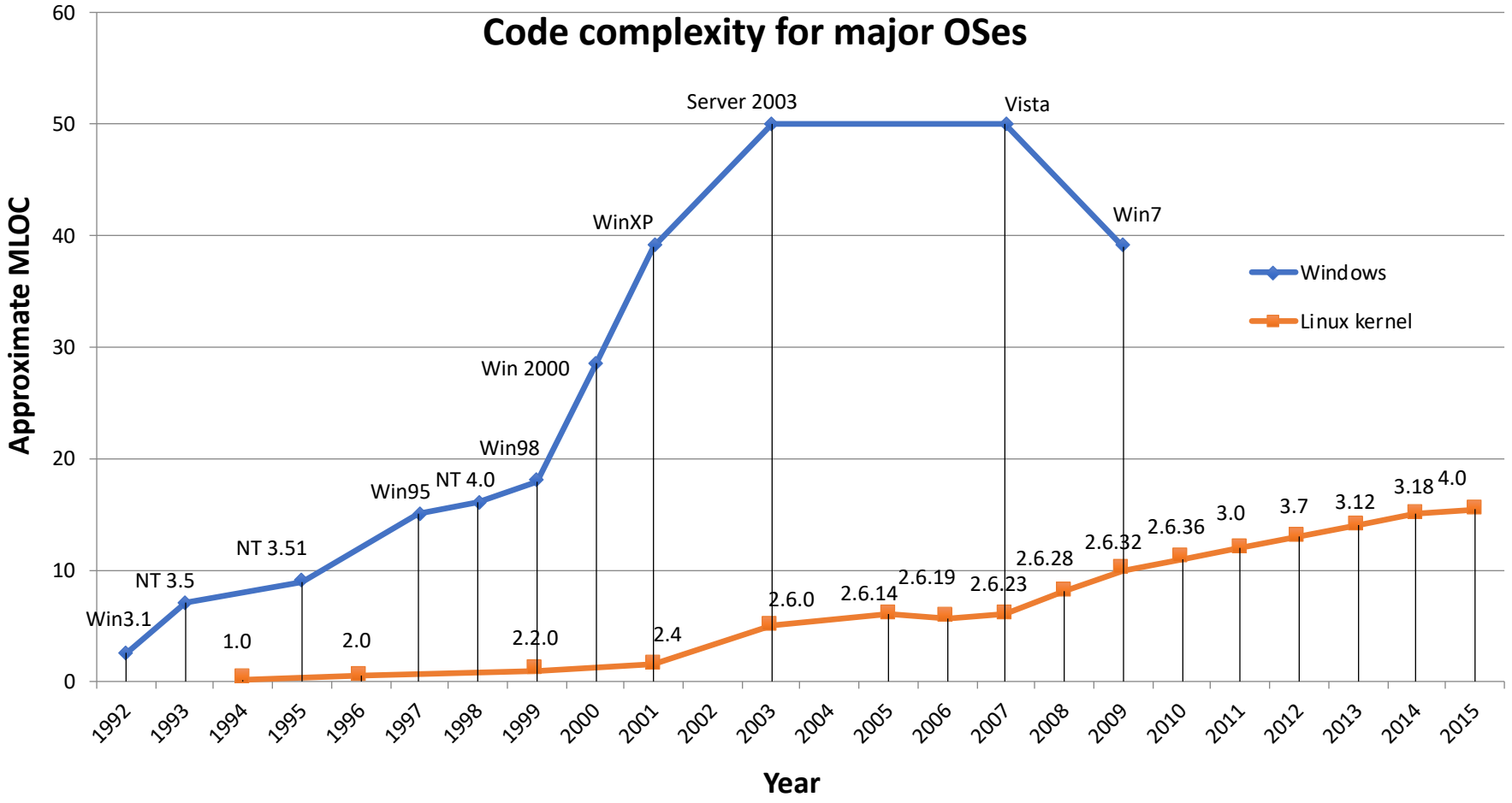
- PROMIS general information
- Courses
- How to apply

Annual cost of software bugs: trends

- 2002, US: 60 billion USD (NIST)
- 2013, global: 312 billion USD (Cambridge University)
- 2017, global: 1.7 trillion USD (Tricentis)
  World population: 7.4 billion, affected: 3.7 billion (50%)

- Swedish government expense budget 2020: ~110 billion USD
- Many root causes for bugs:
  - Complexity
  - Ubiquitous connectivity
  - Extensibility
  - …

Complexity: example



Dormant bugs: a guessing game

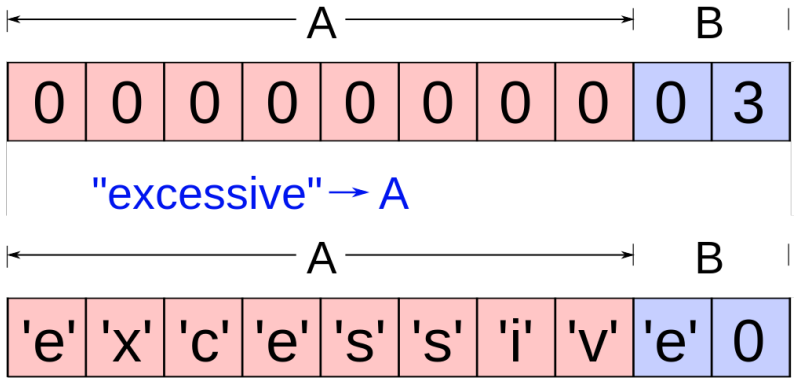
- Very old statistic: approx. 0.5 errors per 1000 LOCs in released code (Microsoft 1990)
- Assume 10-fold improvement over 25 years (*not likely*!): 0.05 bugs per 1000 LOCs
  - Linux kernel 4.0 (2015): 750 dormant bugs
  - Windows 7 (2015): 1950 dormant bugs
- Hard fact:
  - 105 bug fixes in Linux stable kernel 5.6.13 (2020-05-14)

Input parsing bugs

- “There is a programming issue (input parsing) that is the single cause of most exploits… (Fixing this) would do more to improve security of our computers, cars, smartphones, and devices than would any other change.”

  - *Rik Farrow, Editorial of USENIX ;login, Aug 2015, vol. 40, no. 4*

Input parsing: absent check of buffer size



From: Wikimedia Commons

- Buffer overflows occur when one writes more data then a buffer can hold
- The overflow data spills over into the contents of neighbor variables and buffers
- C pseudo-code:
  char B[2], A[8];
  B = {0,3};
  scanf("%s", A);
- User enters: "excessive" from the command-line when prompted.

Demo time!

Input parsing: lenient syntax control

iOS sandbox escape (fixed in iOS 13.5)

https://siguza.github.io/psychicpaper/

```
<key>application-identifier</key>
<string>...</string>
<!---><!-->
<key>platform-application</key>
<true/>
<key>com.apple.private.security.no-container</key>
<true/>
<key>task_for_pid-allow</key>
<true/>
<!-- -->
```

Correct XML comment syntax

```
<!--this is a comment-->
```

iOS uses 4 different XML parsers

- OSUnserializeXML in the kernel
- IOCFUnserialize in IOKitUser
- CFPropertyListCreateWithData in CoreFoundation
- xpc_create_from_plist in libxpc (closed-source)

- "Implementing 4 different parsers is just asking for trouble, and the 'fix' is of the crappiest sort, bolting on more crap to check they're doing the right thing in this single case. None of this is encouraging."

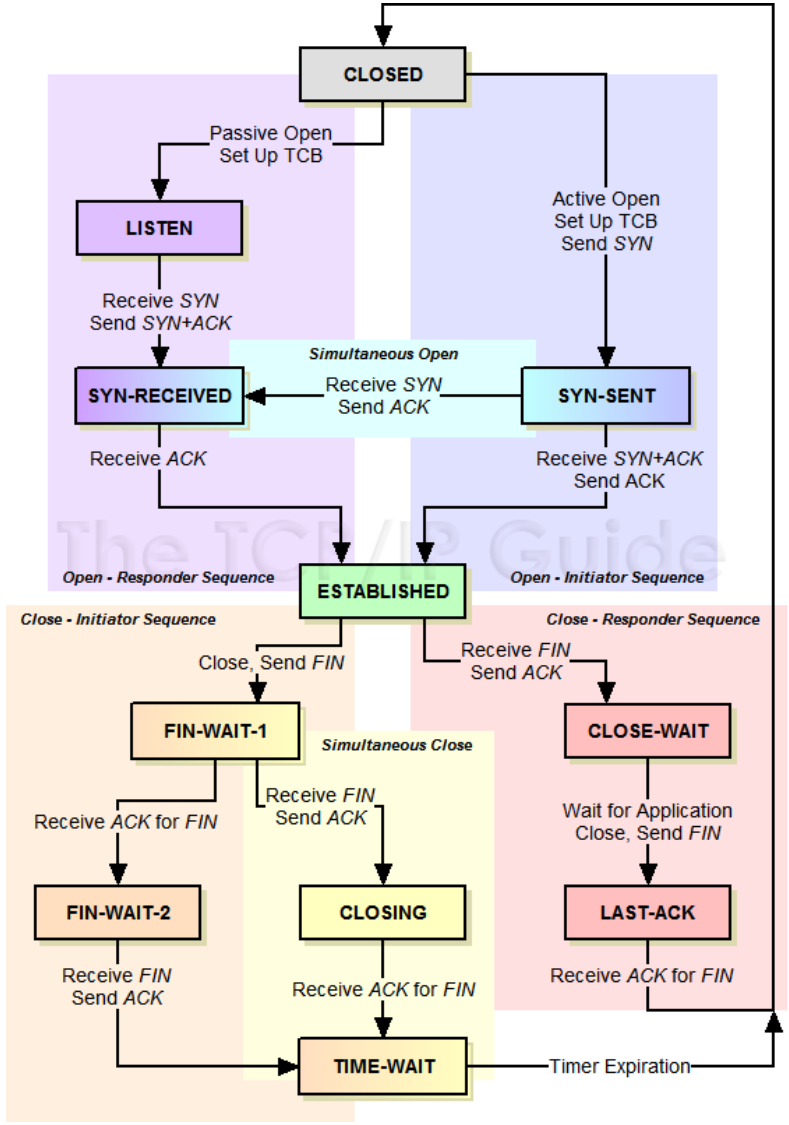
  Rob Hiller, tweet on May 3, 2020

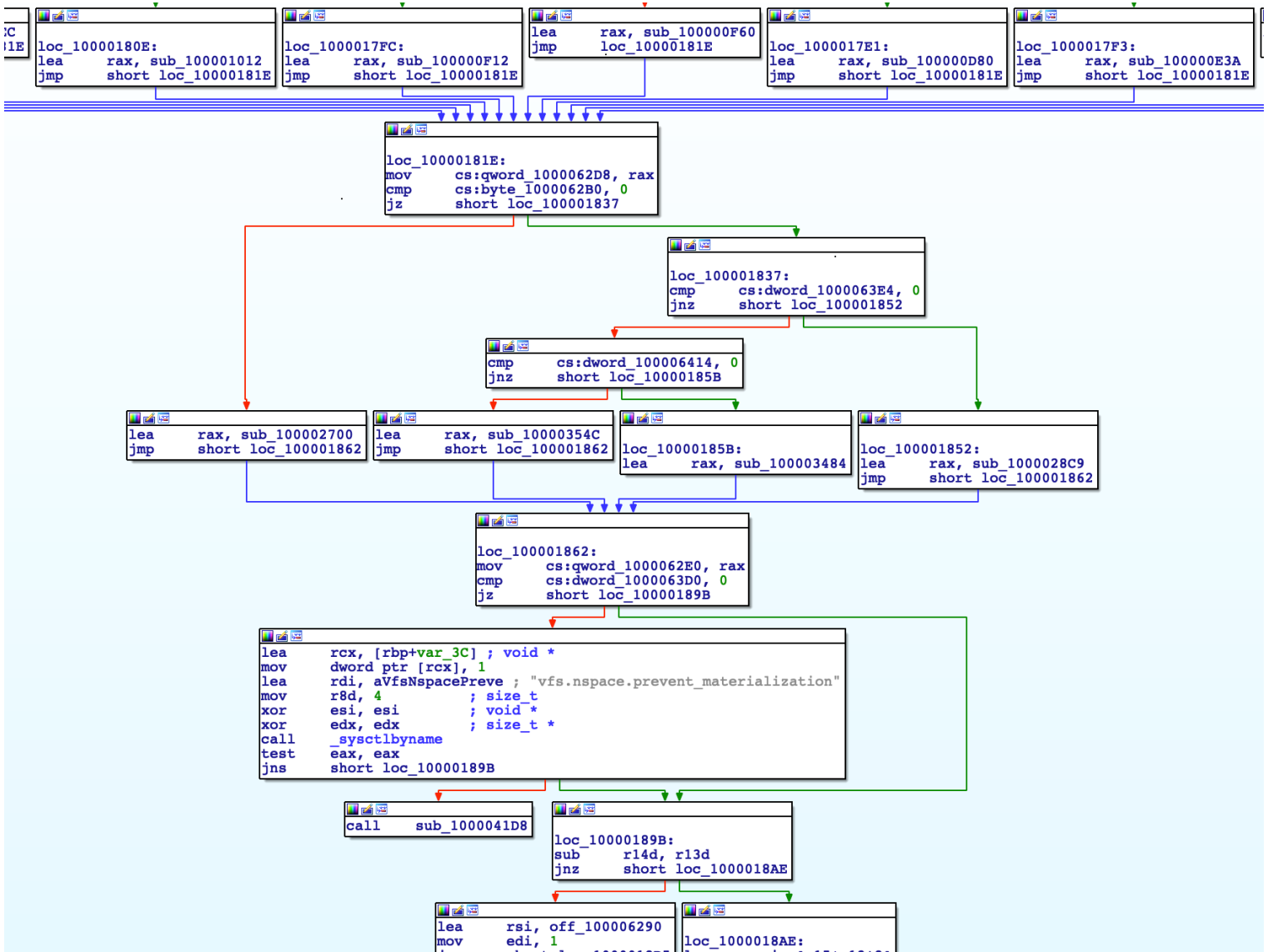
Why is input parsing difficult?

"As a species we are pretty bad at writing code,
we are even worse at writing code that deals with data,
and we are astonshingly bad at writing code that deals with
arbitrary user provided data."

– Matthew Garrett, "Beyond Anti Evil Maid", 2015

Complexity: state diagram





Complexity: System model

- The programmer builds a mental representation of the state machine
  - To understand the (mis)behavior of the program
  - To identify error states and unwanted transitions
  - To find best place to make modifications/add features
  - To anticipate any side effects that changes may bring
- Most practical state machines too large to fit into our mind: ➔ incomplete!
- Incomplete models or unexpected stimuli can cause the process to enter undefined states
  - The process can do things beyond the intention of the programmer

State transitions are controlled by data!

“The illusion that your program is manipulating its data is powerful. But it is an illusion: The data is controlling your program.”

– Taylor Hornby, @DefuseSec 2014

Demo time!

DISCLAIMER

- OS and compiler defenses were turned off to keep demo simple
  - Address Space Layout Randomization (ASLR)
  - StackGuard
  - Non-executable stack (NX)
- However, they can be circumvented with advanced techniques

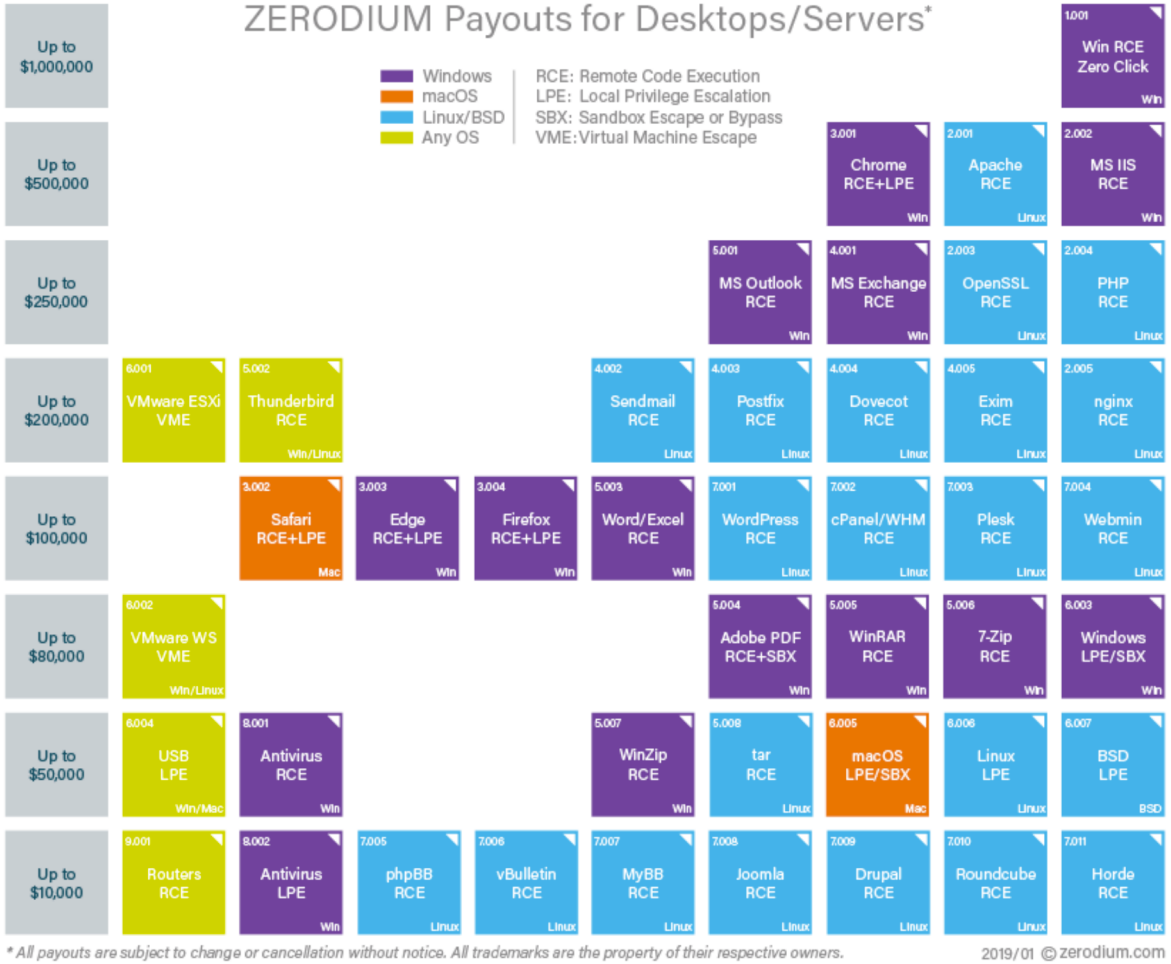
Exploit capabilities

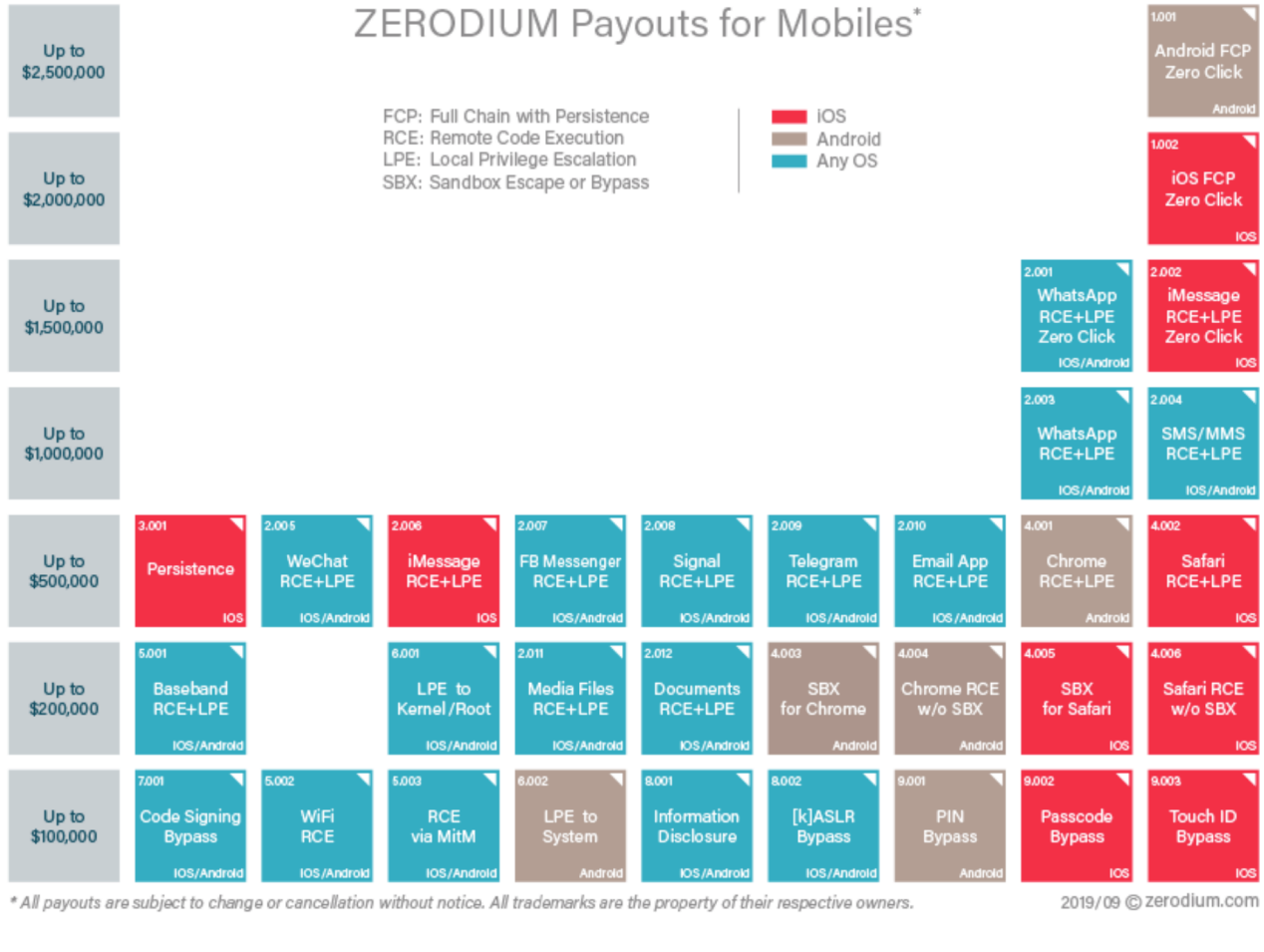
“Any sufficiently complex input format is indistinguishable from bytecode; the code receiving it is indistinguishable from a virtual machine.”

- Bratus, “*The Bugs We Have to Kill*”, ;login, Aug 2015

- Dangerous **zero-day** vulnerabilities are in high demand

Zero-day exploit pricing





Does this sound interesting?

PROMIS (Professional Master in Information Security)

GENERAL FORMAT

Active industrials studying and working at the same time

- *University grade **COURSES for professionals**!*
- *Extend current competence in **an area ("security")***
- Case-based pedagogy (bring your own problems!)
- On-line collaborative didactics
- Distance capability overall incl. lab and tools

Courses under development with input from companies

- Keep relevant and right level (companies advise us)
- DO YOU want to be part of the companies advising on courses?
  - CONTACT: anna.eriksson@bth.se

more to come

Courses (3 thus far)

PROMIS (Professional Master in Information Security)

https://promisedu.se/

Software Security (DV2595)
https://www.bth.se/eng/courses/D5816/20202/
Course responsible: dragos.ilie@bth.se

- the ability to understand how attackers exploit risky programming practices
- the ability to detect risky programming practices
- the ability to understand and reason about efficiency and limitations in existing software security mechanisms
- the ability to to compare and weight the benefits and costs associated with binary analysis and instrumentation techniques

more to come

Courses (3 thus far)

PROMIS (Professional Master in Information Security)

https://promisedu.se/

Web System Security (DV2596)
https://www.bth.se/eng/courses/D5816/20202/
Course anders.carlsson@bth.se

- be able to explain web protocols based on known vulnerabilities and weaknesses
- be able to describe the Common Vulnerability Scoring System (CVSS)
- be able to explain web protocols based on known vulnerabilities and weaknesses
- be able to explain the security aspects when using languages and framework, eg. PHP, JavaScript, and SQL
- be able to explain authentication mechanisms and counter techniques to bypass authentication
- understand Cross-site scripting (XSS) attacks and SQL injections
- be able to explain impacts of one or more combined vulnerabilities that limit or extend the damage given
- be able to install and configure the web server for high security independently
- be able to use and search open vulnerability databases (Common Vulnerability databases CV -DB)
  to prevent and find security problems
- be able to use best practice of known design patterns for secure web applications
- be able to utilize OWASP where applicable
- be able to conduct internal and external penetration testing of web applications and related infrastructure

more to come

Courses (3 thus far)

PROMIS (Professional Master in Information Security)

https://promisedu.se/

Security in Software-intensive products and service development (PA2582)
https://www.bth.se/eng/courses/D5818/20202/
Course responsible: tony.gorschek@bth.se

- the ability to understand the technology, operational aspects, and engineering aspects of security - albeit the focus on the course is on "engineering security”
- the ability to plan for "pre-emptive" security in the planning and development of products and services
- the ability to do a risk assessment and take ROI into account
- the ability to develop and use secure architectures that allows for a more stable base for products and services
- the ability to compare and weigh the benefits and costs of non-functional aspects in combination to security
- the ability to estimate how security aspects impact, and are impacted on quality-/non-functional aspects such as usability, performance and maintainability of a product

more to come

PROMIS

HOW TO

https://promisedu.se/

Spread information about courses @ your company

Entry Requirements

PROMIS courses requires at least 120 credits, of which at least 90 credits are in a technical area, and a minimum of 2 years professional experience within an area related to software-intensive product and/or service development (shown by, for example, a work certificate from an employer).

Even if you don't have the formal academic merits, you might be qualified for the course through validation (reell kompetens)!

Apply for course:

1. **Create a user account at antagning.se / universityadmission.se**
2. **Search for PROMIS courses by the name Fill in and send in your application**
3. **Upload your required documents (employer's certificate)**
4. **Reply to any offers of admission**

Questions about the course: contact course responsible
Questions about applying and validation (reell kompetens): :
anna.eriksson@bth.se
Visit promisedu.se for more info about courses, application and template for employer's certificate

more to come