Intro

Who are we?

⇒ Short introduction of tutors

⇒ Research interests\(^1\)

\(^1\)https://www.sec.in.tum.de/i20/people/momeu-marius
Why should you do this?

⇒ Have you taken *Binary Exploitation* and want to learn advanced methods that mitigate *pwning*?

⇒ Have you learned cool low-level concepts in *Operating Systems* or *Root-Kit Programming* and want to apply them?

⇒ Would you like to find bugs and crash low-level software?

⇒ Are you curious how *Spectre* can be mitigated?

⇒ Are you merely interested in finding out more about *IT Security*?

⇒ Then *Systems Hardening* is the right course for you!
Intro

Objectives

⇒ In this seminar, you are going to **assess state-of-the-art**
   → **hardening mechanisms** (via memory isolation using **VT-x, MPK**, etc.),
   → **kernel and hypervisor fuzzing** agents,
   → **enclaved execution technologies** (**Intel SGX, AMD-SEV, ARM TrustZone**),
   → software mitigation against **microarchitectural flaws** (**Spectre, Forwshadow**, and/or variants).

⇒ **And most importantly**, you are going to
   → **write a paper** about your findings,
   → **give feedback** to (two of) your colleagues’ papers,
   → **give a talk** at the end of the semester.
Technical Content

Overview

⇒ Data & code memory isolation
  → using *Intel VT-x, MPK, HLAT*, etc.
  → to mitigate code-reuse & data-oriented attacks

⇒ Fuzzing low-level software
  → such as the *Linux* kernel or the *Xen* hypervisor via coverage-guided, symbolic execution, and hybrid fuzzers

⇒ Attacks against TEEs
  → such as *AMD-SEV* or *ARM TrustZone*

⇒ Confidential computing
  → using *Intel SGX, MKTME*

⇒ Software mitigations for microarchitectural flaws
  → such as memory management tricks that neutralize *Spectre*

⇒ Heap hardening
⇒ Memory safety via *Rust*
⇒ Live patching
Technical Content

Hands-On

⇒ Operating with command-line *bash* on *Unix* systems

⇒ Kernel & hypervisor development

⇒ Memory management & OS concepts

⇒ *C*, *Assembly* (*x86, ARM, AMD*), *Rust*

⇒ Hardware extensions (*VT-x, MPK*)

⇒ Computer architecture (speculative execution, cache buffers)

⇒ Binary exploitation

⇒ Various executable formats (mostly *ELF*)
Seminar Structure
Phases

Phase I  Choosing your topic  1 week
Phase II  Familiarizing with literature  1 week
Phase III  Writing (first draft with feedback from tutors)  4 weeks
Phase IV  Writing (final draft with feedback from tutors and fellow students)  4 weeks
Phase V  Peer reviewing  1 week
Phase VI  Writing ("camera ready")  1 week
Phase VII  Preparing the final talk  2 weeks
Phase VIII  *Dive deeper with research or thesis*  *optional (TBD)*
Seminar Structure

Sessions

Episode I  Introduction to Scientific Writing
Episode II  More on Scientific Writing
Episode III  Hints on Paper Reviewing
Episode IV  Hints on Public Speaking
Episode V  Final Talks
## Seminar Structure

### Grading

<table>
<thead>
<tr>
<th>Percentage</th>
<th>Component</th>
</tr>
</thead>
<tbody>
<tr>
<td>50%</td>
<td>Final Paper (Content, Style, Language, Scope, ...)</td>
</tr>
<tr>
<td>20%</td>
<td>Presentation (Content, Speaking, Style, Timeliness, ...)</td>
</tr>
<tr>
<td>15%</td>
<td>Prototype / Design / Experiments</td>
</tr>
<tr>
<td>10%</td>
<td>Peer Review</td>
</tr>
<tr>
<td>5%</td>
<td>Discussion</td>
</tr>
</tbody>
</table>

\[ \Sigma 100\% \text{ Total} \]
Seminar Structure

⇒ When?
   → with presentations from tutors and optionally from you (updates on your findings)
   → online or hybrid (depending on the regulations)
   → exact weekday and time TBA
   → final talks at the end of the semester

⇒ Capacity
   → 14-16 students: individual work or in groups of two
   → no qualification challenge
   → but see Hands-On slide to get an idea on what you will be working with during this seminar
   → don’t forget to register in the matching system!

⇒ Master’s and Bachelor’s students are welcome

⇒ Language of instruction: English

⇒ Moodle for accessing seminar material
Resources

Infrastructure

⇒ access to our (resourceful) x86 servers running Xen
  → we can assist you in extending Xen's hypercall interface on demand

⇒ dedicated server for fuzzing the kernel

⇒ AMD and ARM servers TBA

⇒ anything else you need, we’re here for you and we’ll find a solution
  → don’t be shy, ask us at any time :)}
Resources

Reading Material

⇒ Literature access
   → https://scholar.google.com/
   → https://semanticscholar.org/
   → https://dblp.uni-trier.de/
   → https://arxiv.org/

⇒ Get around paywalls using: https://www-ub-tum-de.eaccess.ub.tum.de/datenbanken

⇒ Researchers’ homepages can be valuable!
   → the paper, source code, raw data, instructions, technical information
Outro

Questions?

Thank you!

momeu@sec.in.tum.de

@MariusMomeu

proskurin@sec.in.tum.de