Stony Brook University

SGXPecial: Specializing SGX Interfaces Against Code Reuse Attacks

Shachee Mishra
shmishra@cs.stonybrook.edu

Michalis Polychronakis
mikepo@cs.stonybrook.edu

Stony Brook University

SGXPecial: Specializing SGX Interfaces Against Code Reuse Attacks

Shachee Mishra
shmishra@cs.stonybrook.edu

Michalis Polychronakis
mikepo@cs.stonybrook.edu

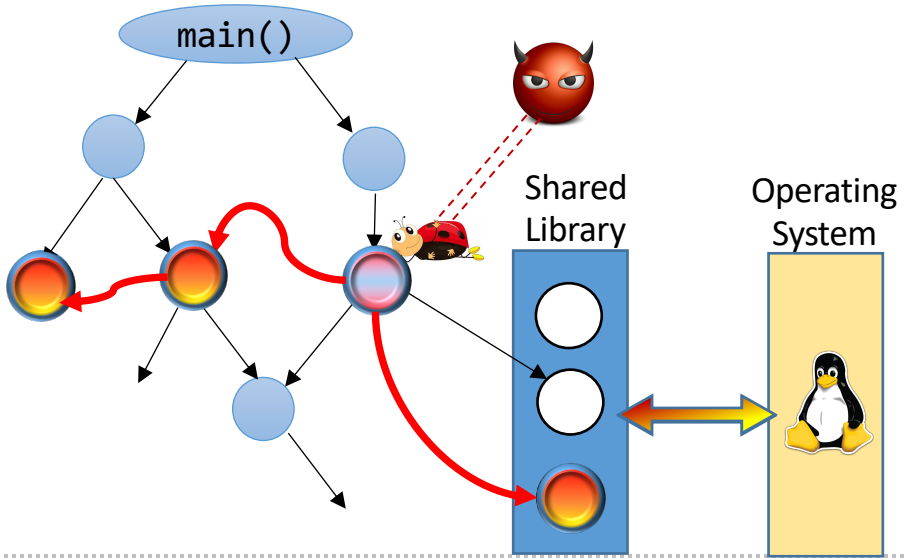
Outline

- Code Reuse Attacks
  - Attack Surface Reduction
- Intel SGX
  - Interfaces
- SGXPecial

Code Reuse Attacks 101

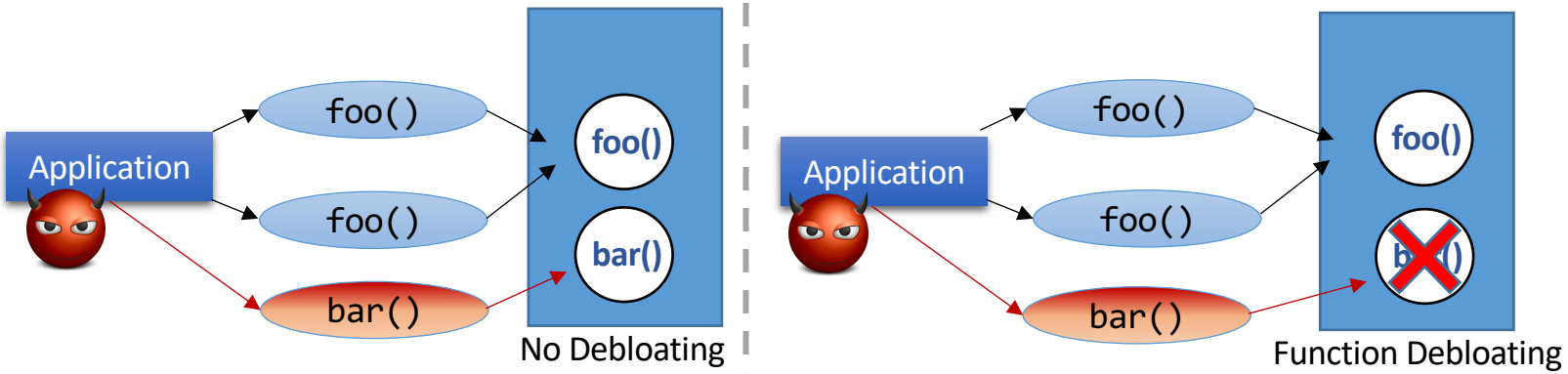
An adversary can **re**use benign code from a process' address space

- Attacker looks for **gadgets** in the address space
- Chain gadgets together to achieve arbitrary functionality



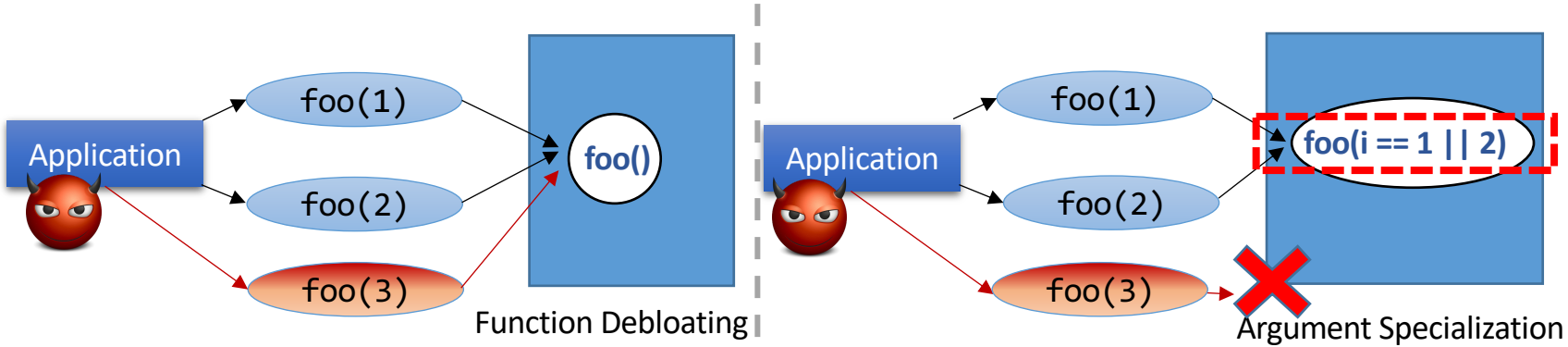
Application Debloating

Remove unneeded code/functions to restrict attacker's capabilities



Argument-level Specialization

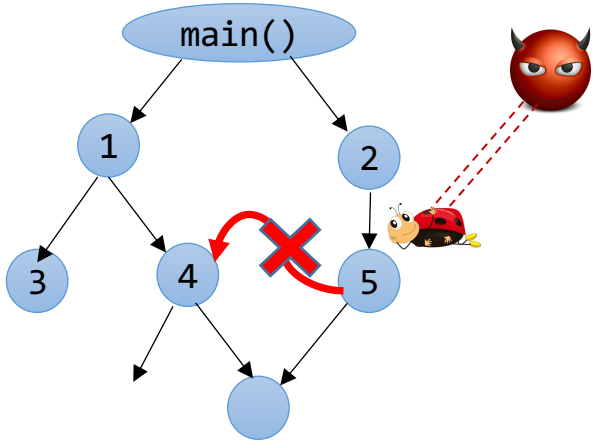
Not all functions can be removed by function debloating



Control Flow Integrity (CFI)

Runtime enforcement technique to prevent control flow hijacking

- Find set of valid targets for every control flow
- Ensure that control flow transfers are only made to these pre-decided sets

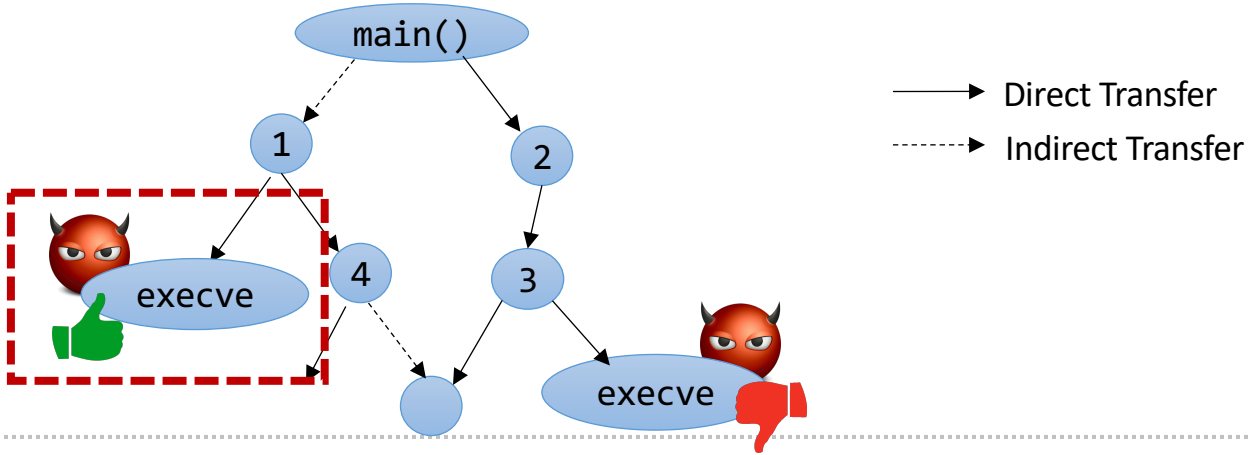



```
[main] -> [1,2]
[1] -> [3,4]
[2] -> [5]
```

So… What happens now?

Not every invocation of a library function is same

- Attackers now are restricted to use specific control flows
- Not all critical function paths are equally protected by CFI rules



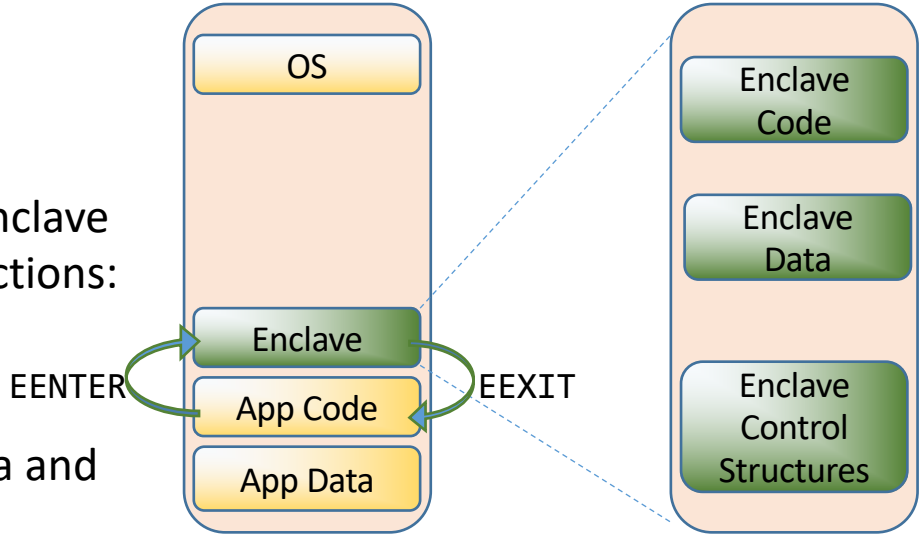
Outline

- Code Reuse Attacks
  - Attack Surface Reduction
- Intel SGX
  - Interfaces
- SGXPecial

Intel Software Guard Extensions (SGX)

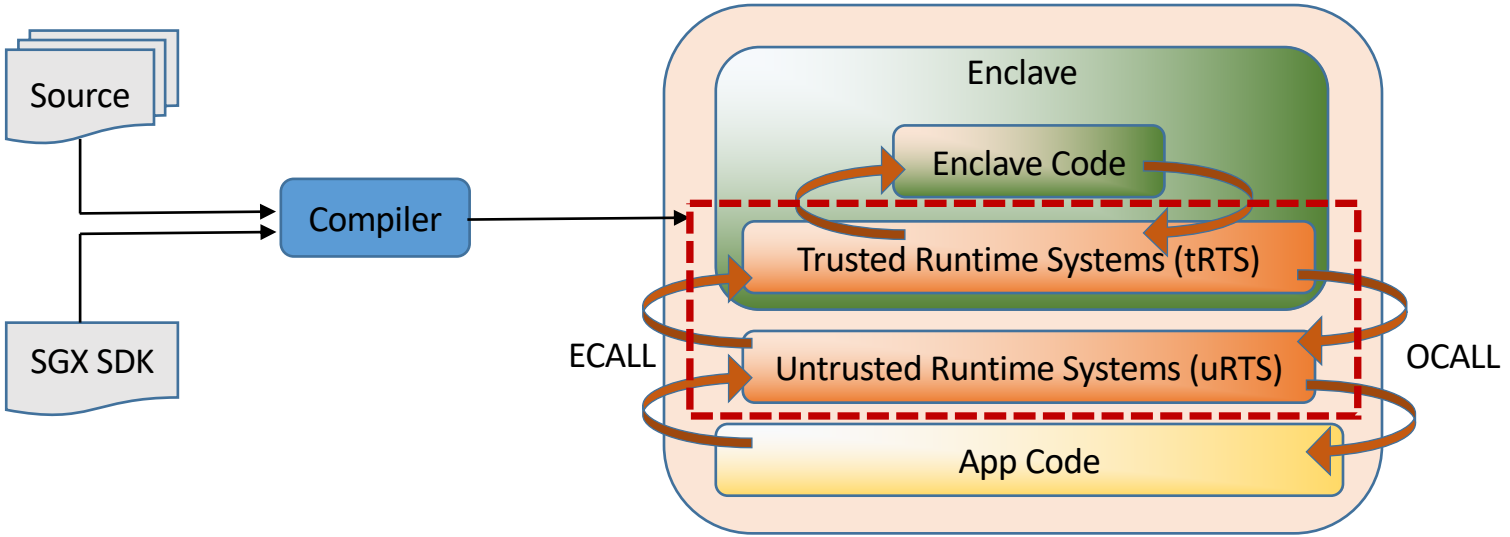
Secret Region “Enclave” protected from malicious actors

- Enclave code and data reside in protected memory
- Entering and Exiting from the enclave are done through special instructions:
  - EENTER
  - EEXIT
- Enclave consists of its code, data and control structures



SGX SDK: Attack Surface

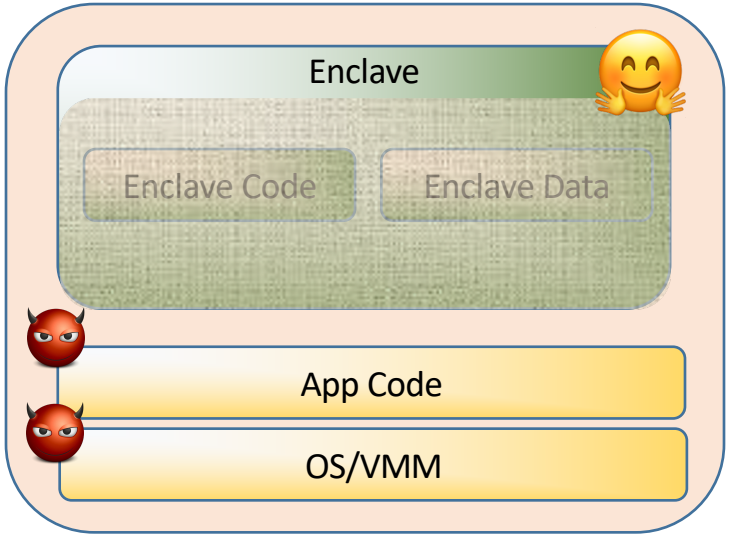
Intel SGX SDK is the most popular tool for building SGX applications



Code Reuse Attacks in SGX: Malicious Host

Standard SGX Threat Model

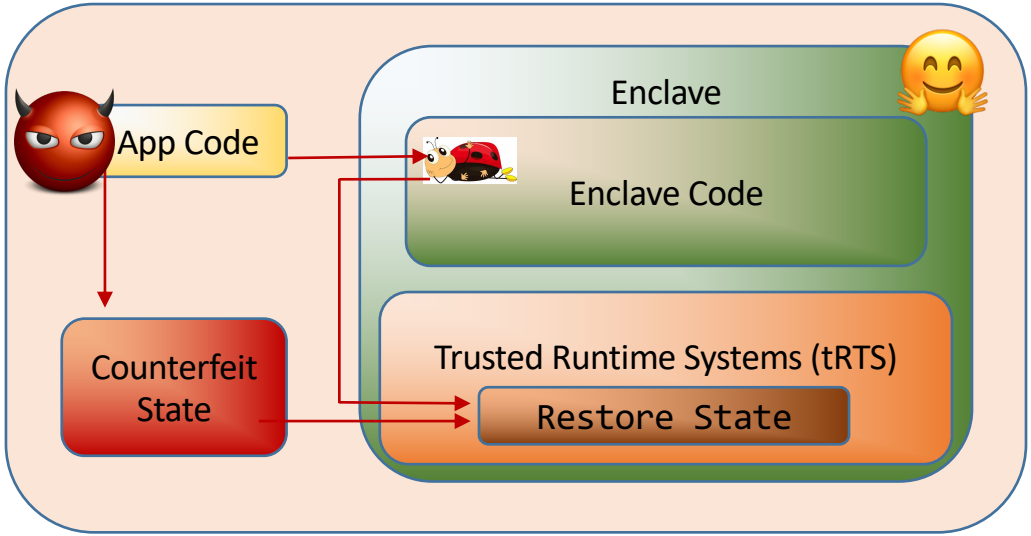
- Everything outside the enclave is untrusted
- OS loads the encrypted enclave binary
- SGX provides Confidentiality and Integrity to the encrypted enclave region



Guard's Dilemma[1]

ROP attack that uses gadgets from SDK libraries

- Malicious host exploits an in-enclave bug
- Uses gadgets from tRTS to restore to a fake state introduced by the host
- No backward CFI in SGX

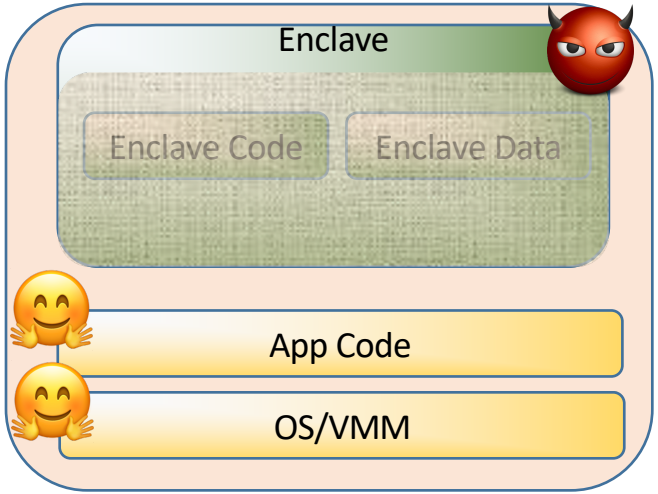


[1] The Guard's Dilemma: Efficient Code-Reuse Attacks Against Intel {SGX}. In 27th USENIX Security Symposium (USENIX Security 18)

Code Reuse Attacks in SGX: Malicious Enclave

Intel SGX does not guarantee that the code executed in the enclave is from a trusted source - Intel

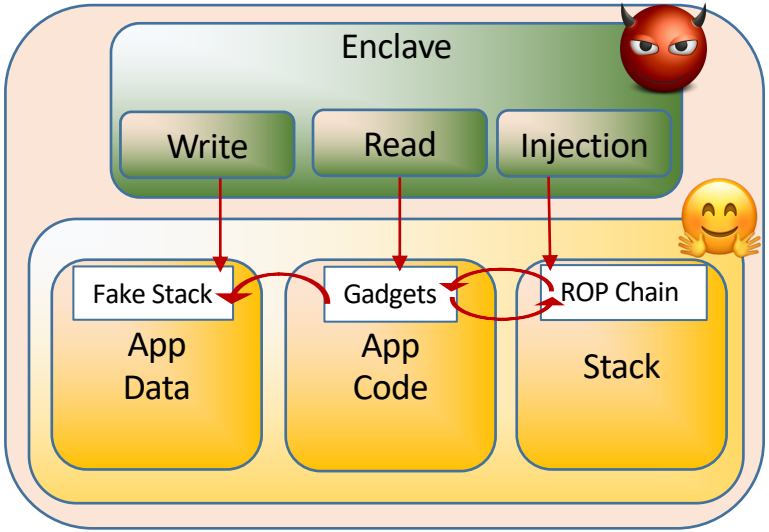
- Enclaves are black boxes
- Application and OS can not look inside the enclaves
- What if they come from untrusted sources? What if they contain malicious code?



SGX-ROP[1]

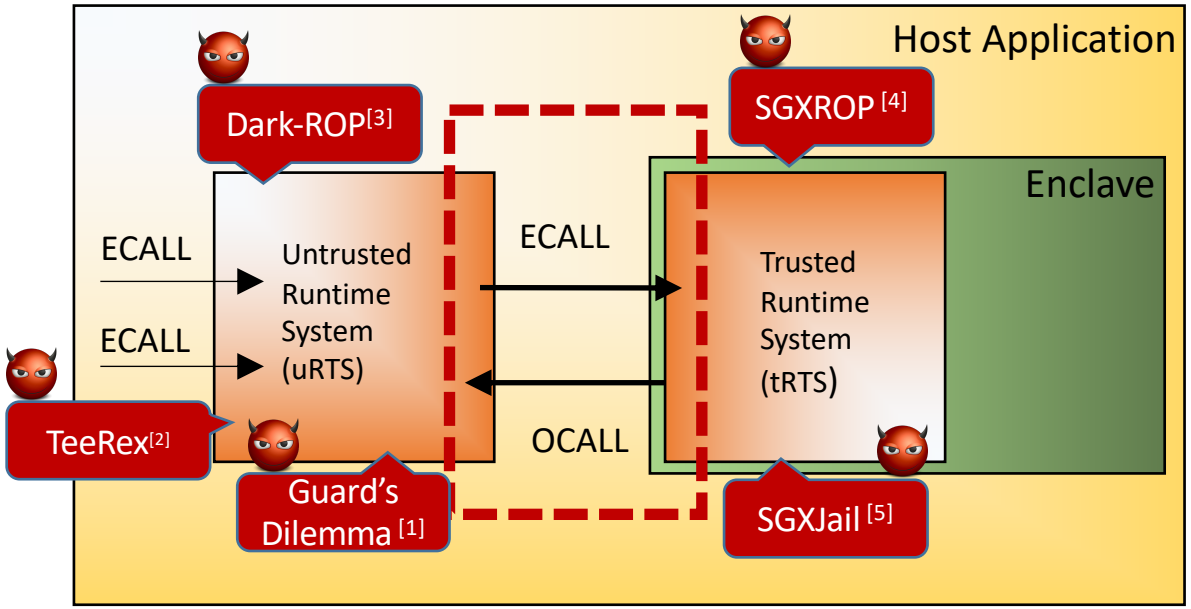
Uses Intel TSX to look for gadgets in host application

- Reads and Writes are atomic
- Error -> Abort and Rollback
- Uses Read and Write abilities of Enclave to read gadgets and inject ROP chains



[1] Practical enclave malware with Intel SGX. In International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA 2019)

Two-sided Attack Surface



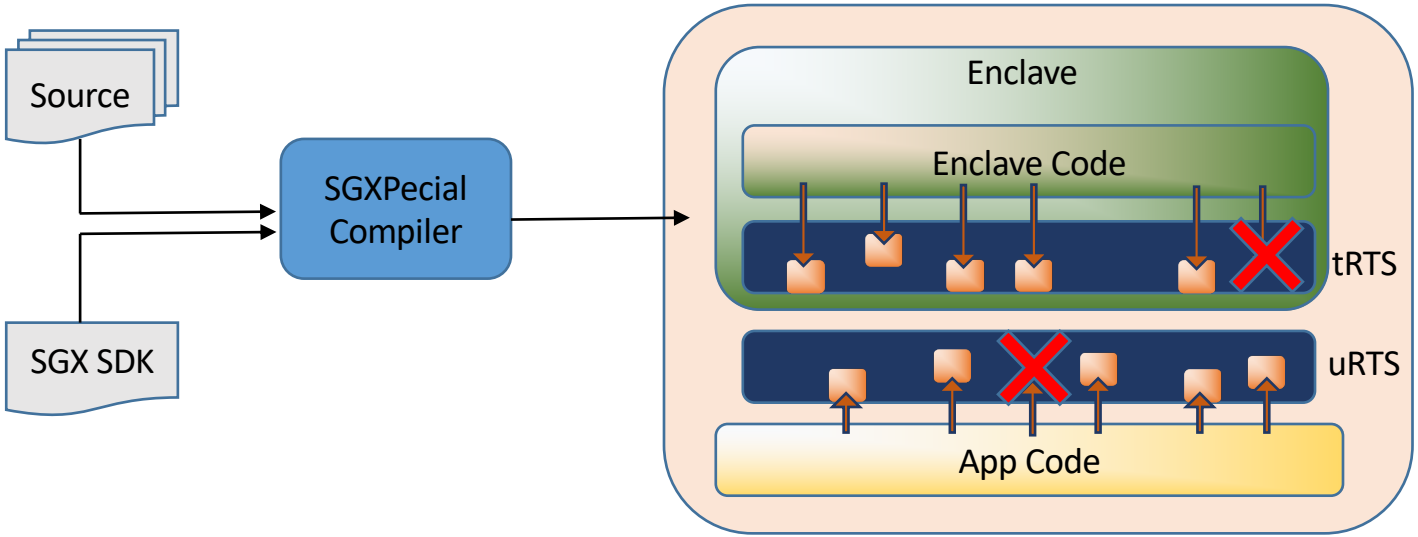
[1] The Guard's Dilemma: Efficient Code-Reuse Attacks Against Intel (SGX). In 27th USENIX Security Symposium (USENIX Security 18)
[2] TeeRex: Discovery and Exploitation of Memory Corruption Vulnerabilities in SGX Enclaves. In 29th USENIX Security Symposium (USENIX Security 20)
[3] Hacking in darkness: Return oriented programming against secure enclaves. In 26th USENIX Security Symposium (USENIX Security 17)
[4] Practical enclave malware with Intel SGX. In International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA 2019)
[5] SGXJail: Defeating Enclave Malware via Confinement. In 22nd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2019)

Outline

- Code Reuse Attacks
  - Attack Surface Reduction
- Intel SGX
  - Interfaces
- SGXPecial

SGXPecial

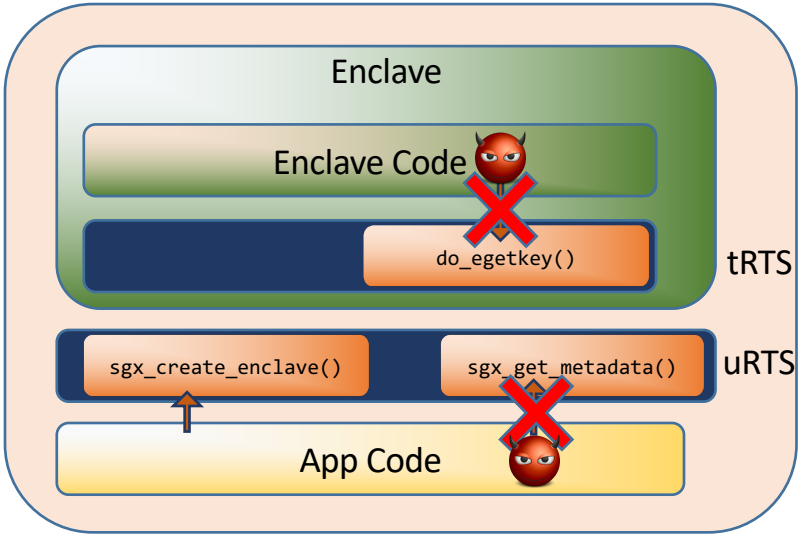
Specialize SGX Interfaces according to individual applications



Function Level Specialization

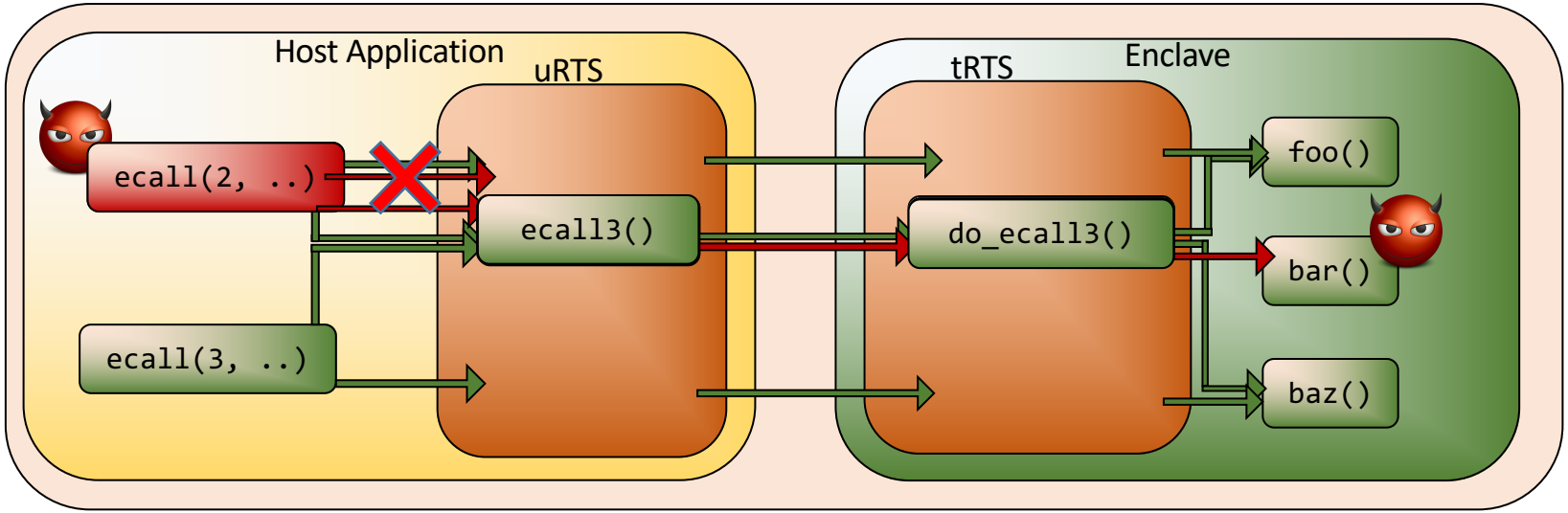
Identify functions that the application and enclave use.

- Analyze application and create specialized interfaces
- Functions might be used by the enclave for bookkeeping
- `do_egetkey()` – used to get SGX keys; but, not used by application



Context-Sensitive Argument Specialization

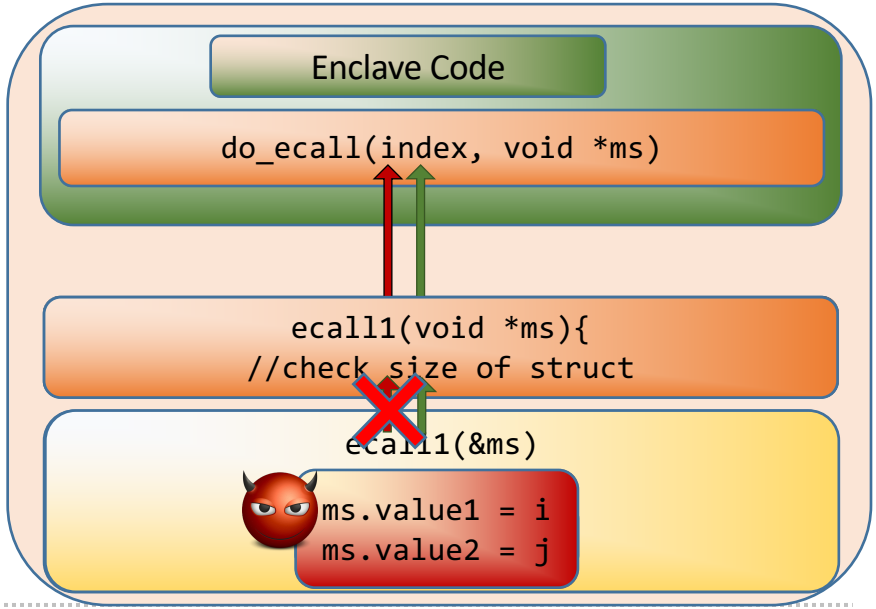
Extract and Neutralize Static Arguments to function calls



Type-based Filtering

SGXPecial: Verifies types of arguments marshalled for ECALLs/OCALLs

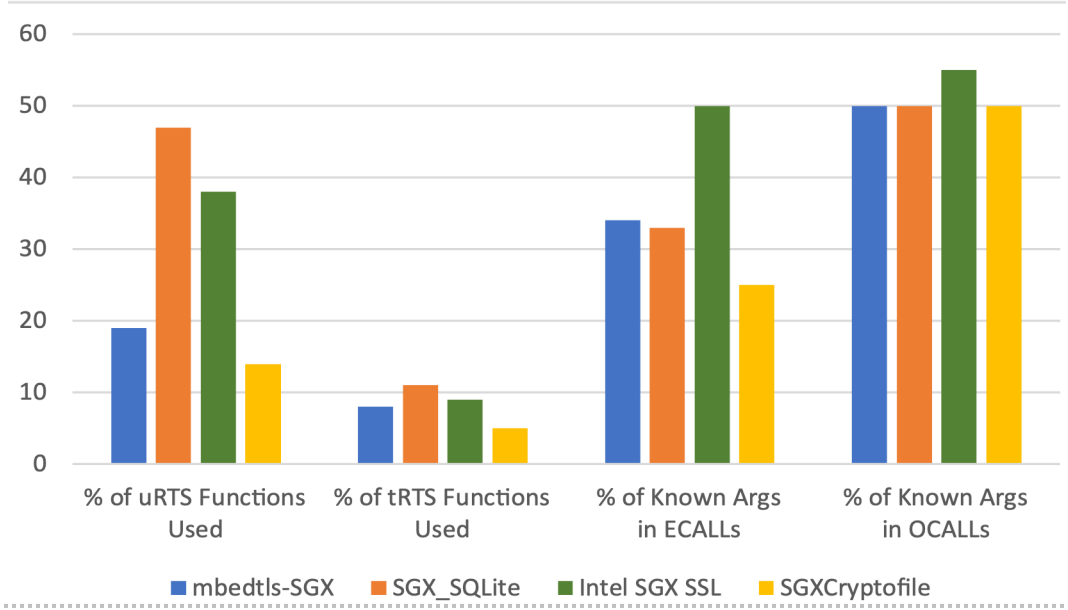
- Host can not read/write enclave memory
- Parameters are marshalled from the host application to the enclave, and return values are un-marshalled
- Generic marshalling functions used for all ECALLs/OCALLs



Evaluation: Applications

Tested with Sample and Real-world Applications

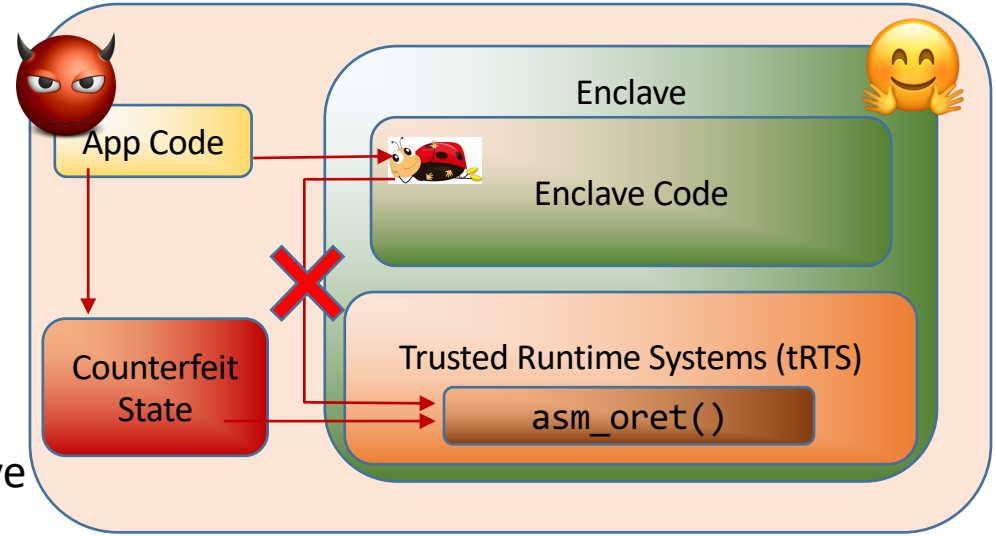
- 7 Sample Applications:
  - SampleEnclave
  - SealUnseal etc.
- Real-world Applications:
  - Mbedtls-SGX
  - Sqlite
  - SGX SSL
  - SGXCryptofile



Security Case Study: Guard's Dilemma[1]

ROP attack that uses gadgets from tRTS

- Vulnerable enclave code calls `asm_oret()`
- `asm_oret()` → `xregs_restore()`
- SGXPecial identifies `asm_oret()` not used by enclave

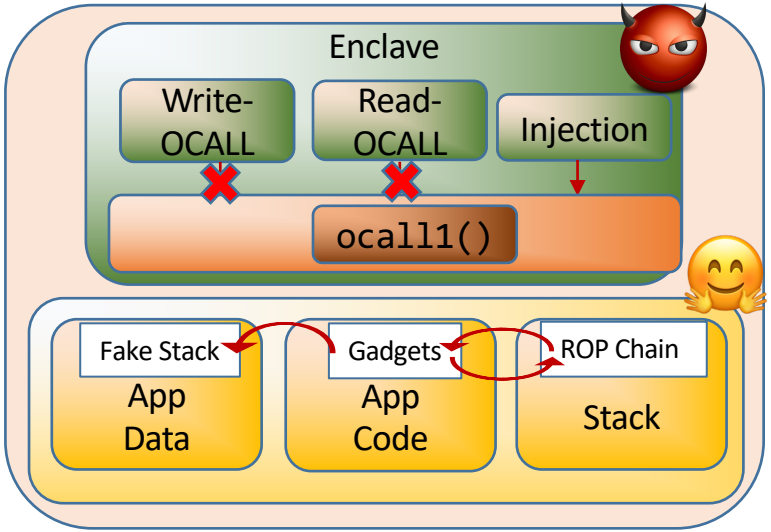


[1] The Guard's Dilemma: Efficient Code-Reuse Attacks Against Intel {SGX}. In 27th USENIX Security Symposium (USENIX Security 18)

Security Case Study: SGX-ROP[1]

Uses Intel TSX to look for gadgets in host application

- write()/read() are OCALLs
- OCALL functions are identified
- If the enclave does not use the functions, SGXPecial blocks the functions from executing



[1] Practical enclave malware with Intel SGX. In International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA 2019)

Conclusion

- We presented SGXPecial: Modifies SGX SDK to generate interfaces specialized to the current application and enclave.
- SGXPecial performs: Function-level, Argument-level and Type-based specialization.
- Tested SGXPecial with sample and real-world applications and show that real code reuse exploits can be prevented.

For questions/ comments/ concerns:  shmishra@cs.stonybrook.edu

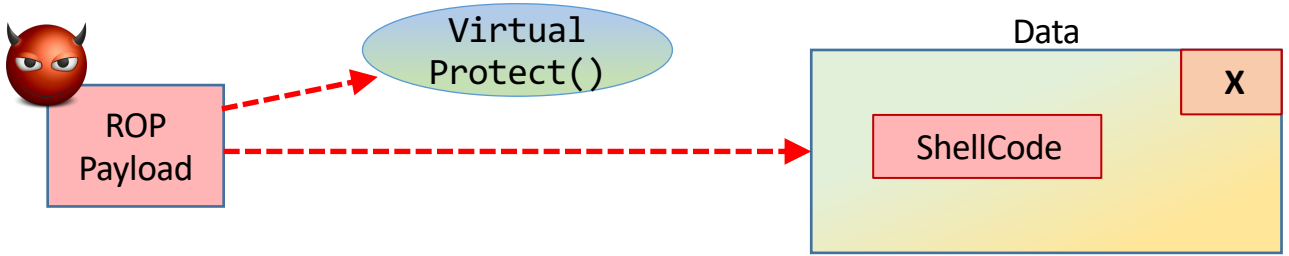
Thank You!

Backup Slides

Hybrid Attack

Code Reuse is generally used as first step in a hybrid attack

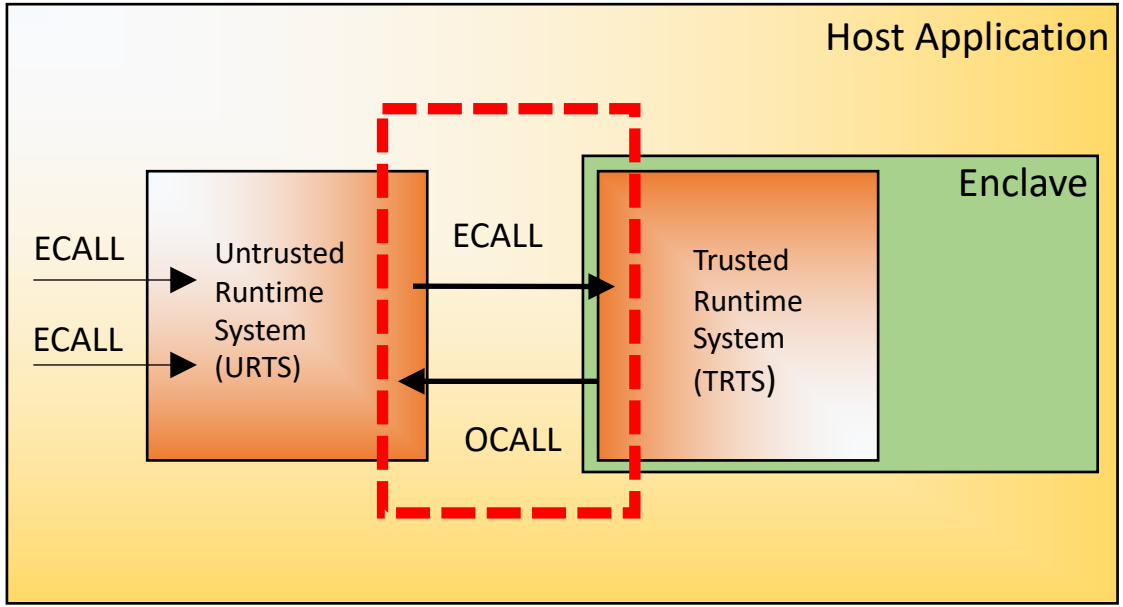
- ROP gadgets are Turing complete, but are complex to implement
- ROP is used as first step to attain an “Executable Memory”



- Writing a “shellcode” to the memory and executing ➔ Code Injection Attack

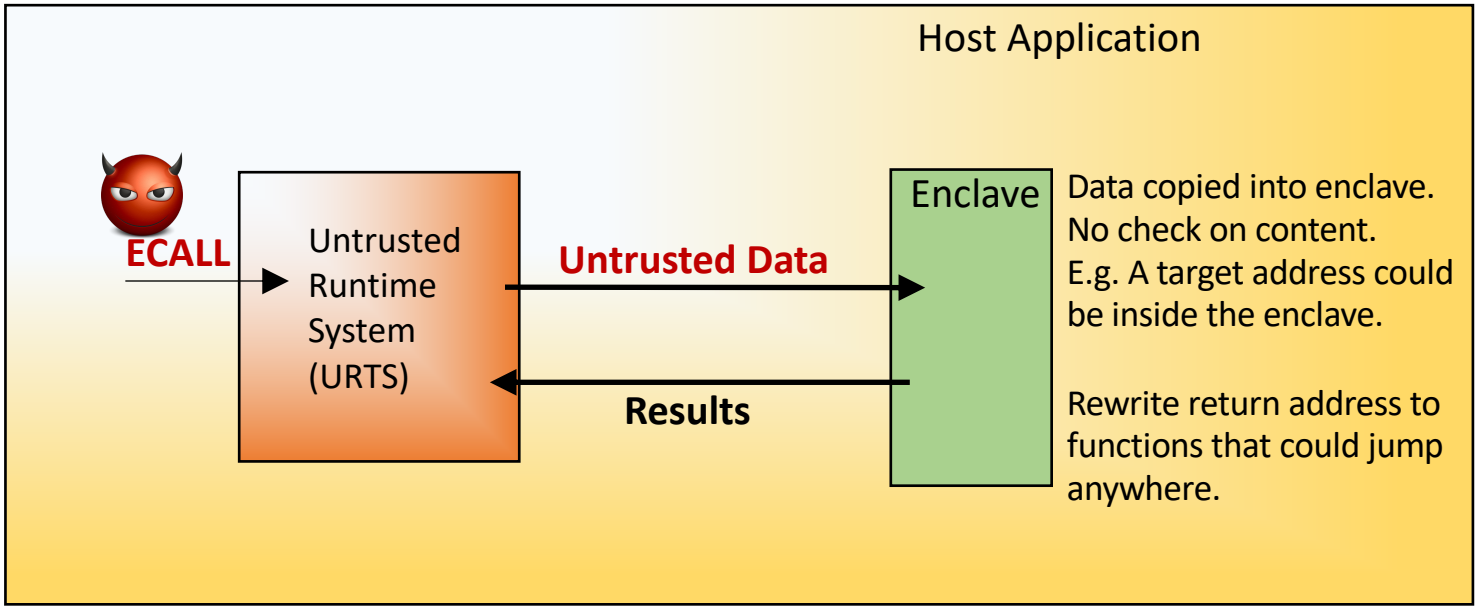
SGX Runtime Environment

Trusted and Untrusted Runtime Systems communicate over SGX's Interface



TeeRex[1]

Public SGX Enclaves could be exploited by host applications



[1] Cloosters, T., Rodler, M., & Davi, L. (2020). TeeRex: Discovery and Exploitation of Memory Corruption Vulnerabilities in SGX Enclaves. In *29th USENIX Security Symposium (USENIX Security 20)*

SgxPecial

Specialize interface between enclave and host application