SgxElide: Enabling Enclave Code Secrecy via Self-Modification

Erick Bauman[1], Huibo Wang[1],
Mingwei Zhang[2], Zhiqiang Lin[1,3]

[1]University of Texas at Dallas
[2]Intel Labs
[3]The Ohio State University

CGO 2018

Intel SGX

Intel SGX

Intel SGX

- Provides secure *enclaves*

Intel SGX

Intel SGX

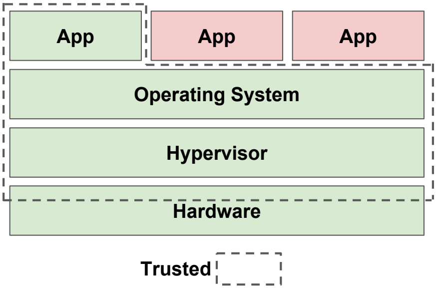
- Provides secure *enclaves*
- Memory regions isolated from all other code

Intel SGX

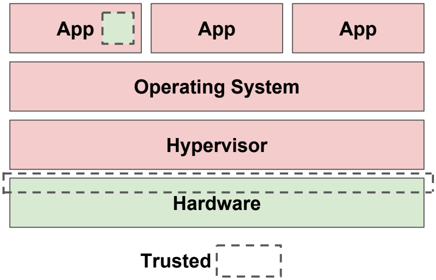
Intel SGX

- Provides secure *enclaves*
- Memory regions isolated from all other code
- Cannot be accessed by OS or hypervisor

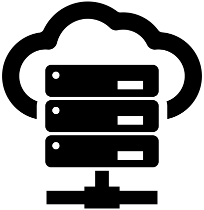
Intel SGX

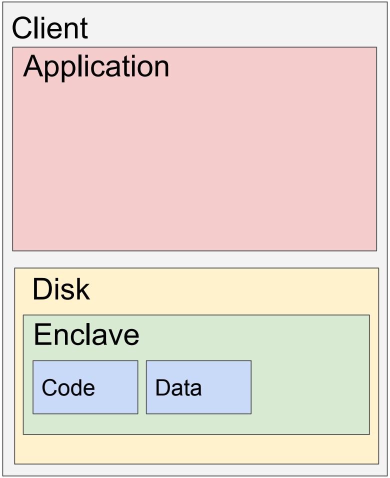


Intel SGX

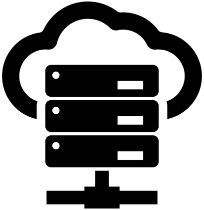


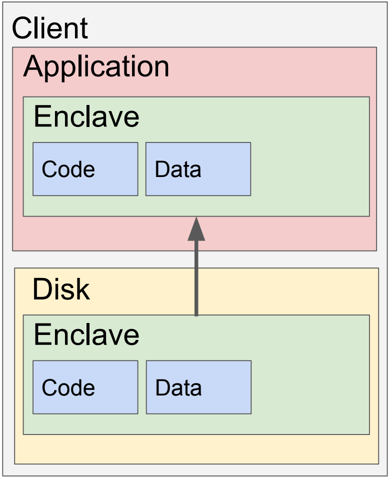
Intel SGX



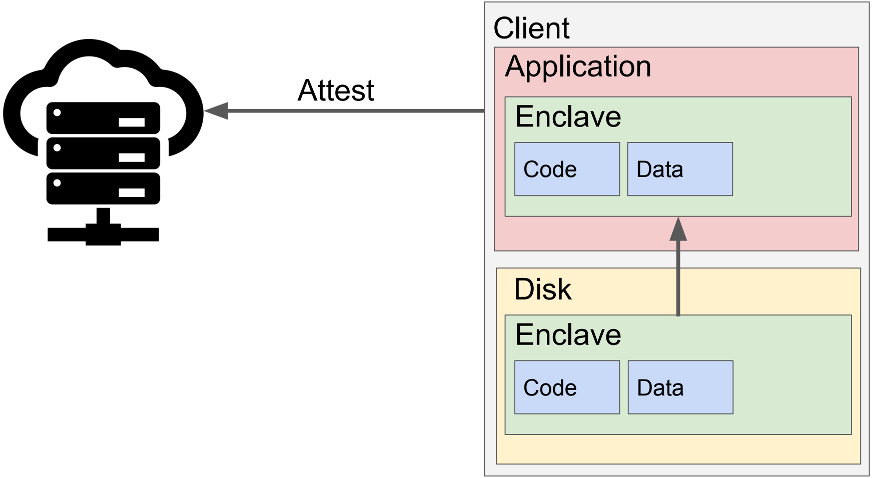


Intel SGX

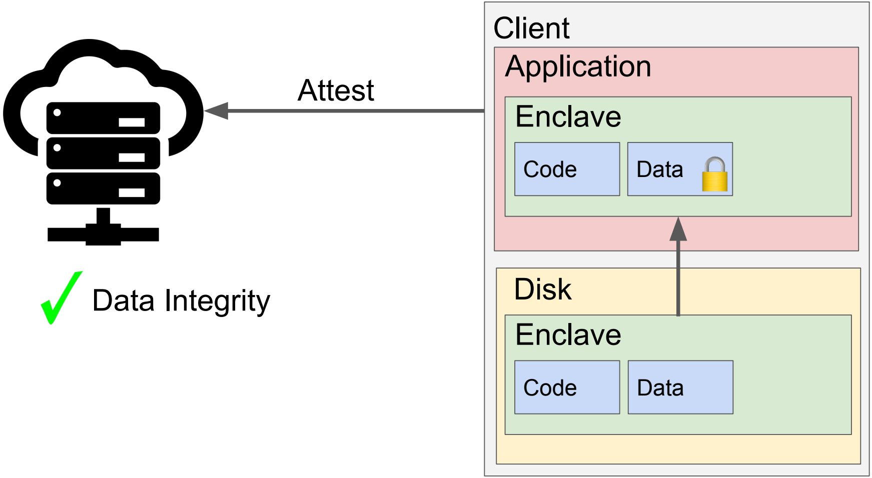



Client
Application
Enclave
Code
Data
Disk
Enclave
Code
Data

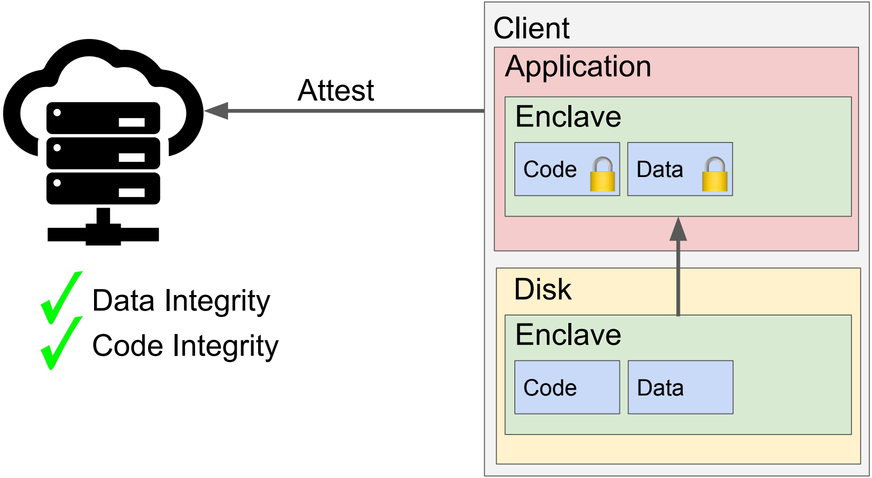
Intel SGX


Attest
Client
Application
Enclave
Code
Data
Disk
Enclave
Code
Data

Intel SGX

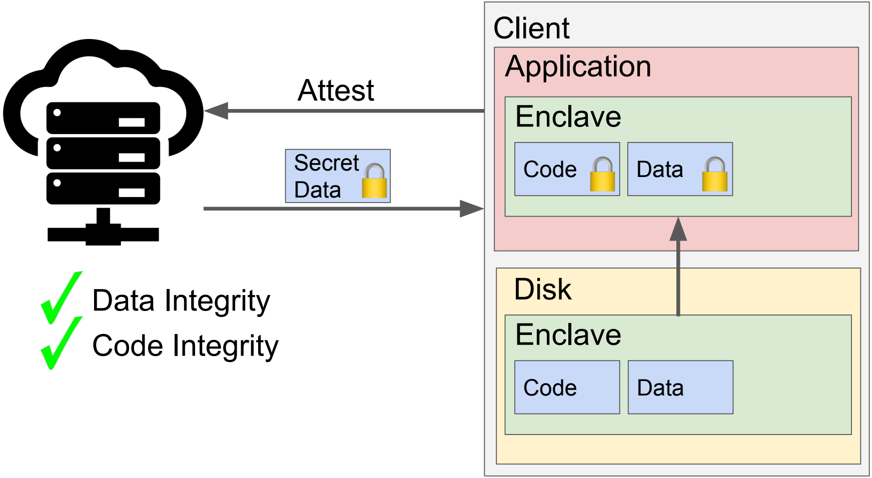

Attest
Data Integrity
Client
Application
Enclave
Code
Data
Disk
Enclave
Code
Data

Intel SGX

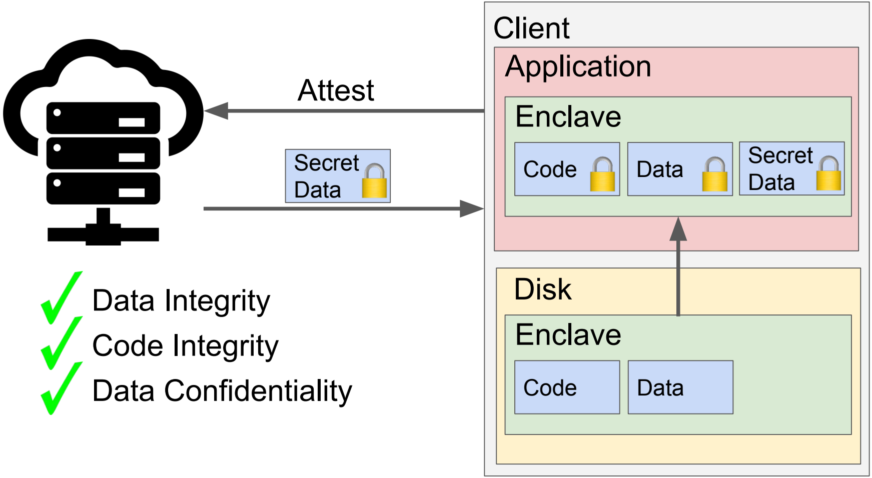


Attest
Data Integrity
Code Integrity
Client
Application
Enclave
Code
Data
Disk
Enclave
Code
Data

Intel SGX

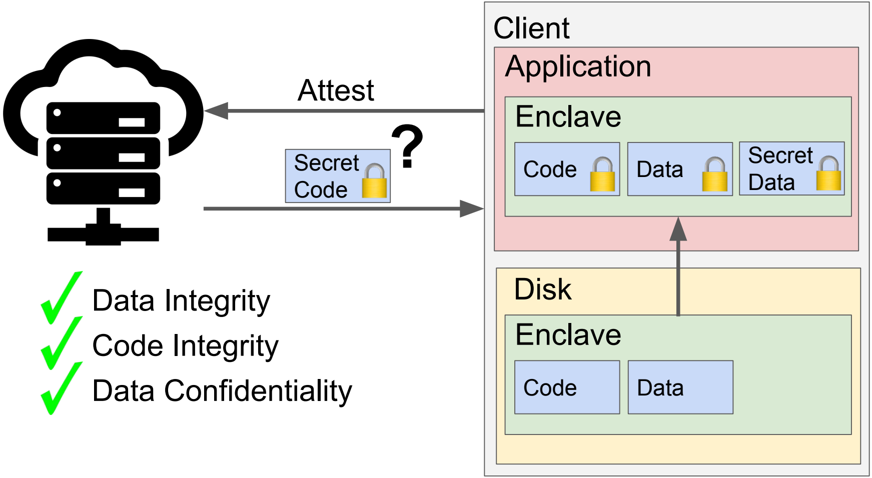

Attest
Secret Data
Client
Application
Enclave
Code
Data
Disk
Enclave
Code
Data
Data Integrity
Code Integrity

Intel SGX



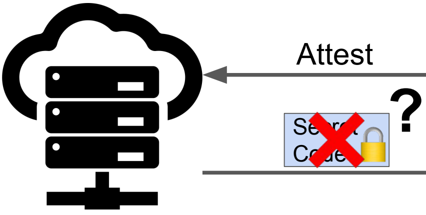
- ✓ Data Integrity
- ✓ Code Integrity
- ✓ Data Confidentiality

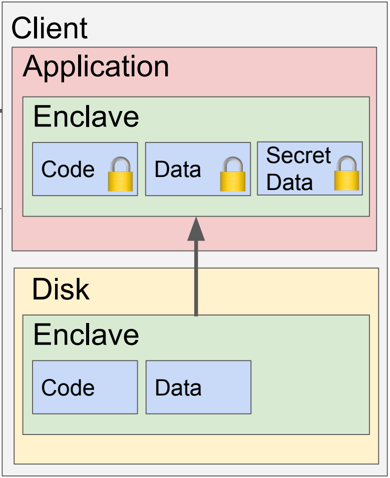
Intel SGX



- Data Integrity
- Code Integrity
- Data Confidentiality

Intel SGX





- ✓ Data Integrity
- ✓ Code Integrity
- ✓ Data Confidentiality
- ✗ Code Confidentiality

Intel SGX

"The enclave file can be disassembled, so the algorithms used by the enclave developer will not remain secret."
–SGX SDK Manual

SgxElide

Definition

Elide: To leave out or omit

Challenges

- Enclaves must be signed and unmodified until initialization

Challenges

- Enclaves must be signed and unmodified until initialization
- The entire enclave cannot be encrypted

Challenges

- Enclaves must be signed and unmodified until initialization
- The entire enclave cannot be encrypted
- Any secrets cannot be stored in the enclave

Challenges

- Enclaves must be signed and unmodified until initialization
- The entire enclave cannot be encrypted
- Any secrets cannot be stored in the enclave
- There should be minimal toolchain changes

Main Idea

Redact (or *sanitize*) secrets and restore at runtime

Blacklist vs. Whitelist

Blacklist

Blacklist vs. Whitelist

Blacklist

- User specifies secrets (e.g. annotations)

Blacklist vs. Whitelist

Blacklist

- User specifies secrets (e.g. annotations)
- Minimizes code that must be encrypted

Blacklist vs. Whitelist

Blacklist

- User specifies secrets (e.g. annotations)
- Minimizes code that must be encrypted
- Burden of annotating secrets on developer

Blacklist vs. Whitelist

Blacklist

- User specifies secrets (e.g. annotations)
- Minimizes code that must be encrypted
- Burden of annotating secrets on developer
- Risk of mistakes

Blacklist vs. Whitelist

Blacklist

- User specifies secrets (e.g. annotations)
- Minimizes code that must be encrypted
- Burden of annotating secrets on developer
- Risk of mistakes

Whitelist

Blacklist vs. Whitelist

Blacklist

- User specifies secrets (e.g. annotations)
- Minimizes code that must be encrypted
- Burden of annotating secrets on developer
- Risk of mistakes

Whitelist

- Only specify code that must not be redacted

Blacklist vs. Whitelist

Blacklist

- User specifies secrets (e.g. annotations)
- Minimizes code that must be encrypted
- Burden of annotating secrets on developer
- Risk of mistakes

Whitelist

- Only specify code that must not be redacted
- Applicable to any enclave

Blacklist vs. Whitelist

Blacklist

- User specifies secrets (e.g. annotations)
- Minimizes code that must be encrypted
- Burden of annotating secrets on developer
- Risk of mistakes

Whitelist

- Only specify code that must not be redacted
- Applicable to any enclave
- No need for developer to mark secrets

Blacklist vs. Whitelist

Blacklist

- User specifies secrets (e.g. annotations)
- Minimizes code that must be encrypted
- Burden of annotating secrets on developer
- Risk of mistakes

Whitelist

- Only specify code that must not be redacted
- Applicable to any enclave
- No need for developer to mark secrets
- More code must be encrypted

Our Solution

- Sign sanitized enclave and restore secrets after initializing

Our Solution

- Sign sanitized enclave and restore secrets after initializing
- Encrypt all nonessential functions

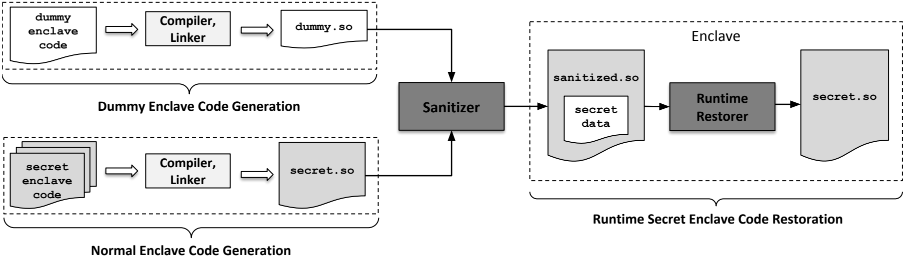
Our Solution

- Sign sanitized enclave and restore secrets after initializing
- Encrypt all nonessential functions
- Use remote attestation

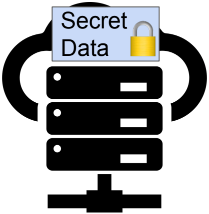
Our Solution

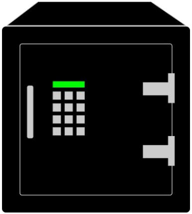
- Sign sanitized enclave and restore secrets after initializing
- Encrypt all nonessential functions
- Use remote attestation
- Use both local and remote storage

SgxElide Overview

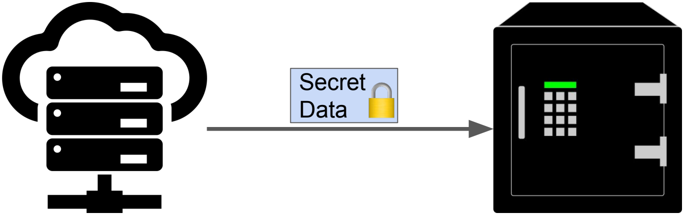


Remote vs. Local Data

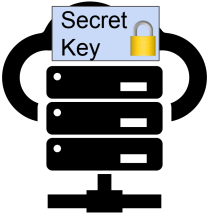


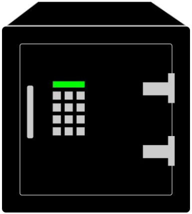


Remote vs. Local Data

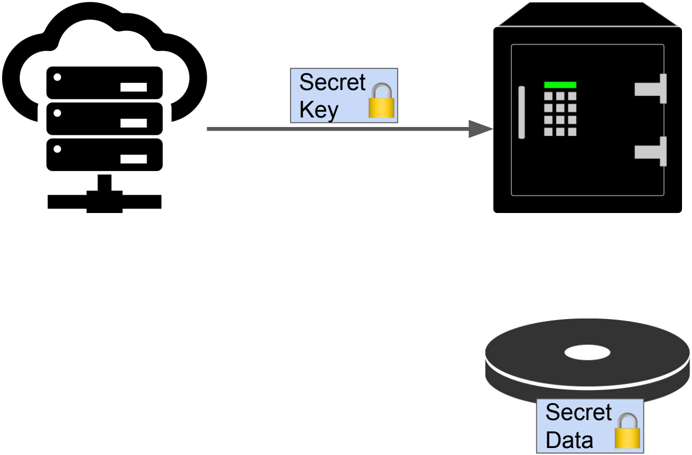


Remote vs. Local Data

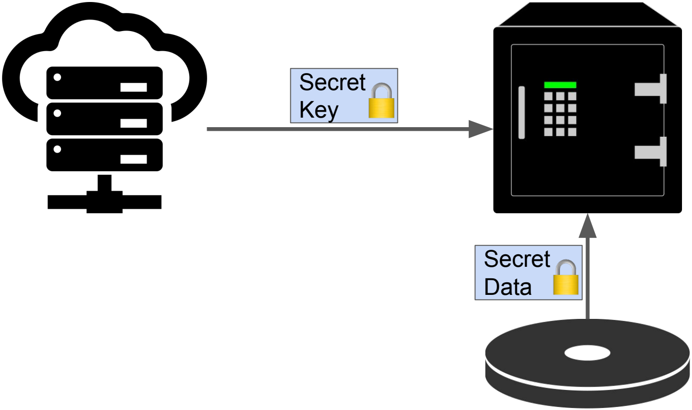




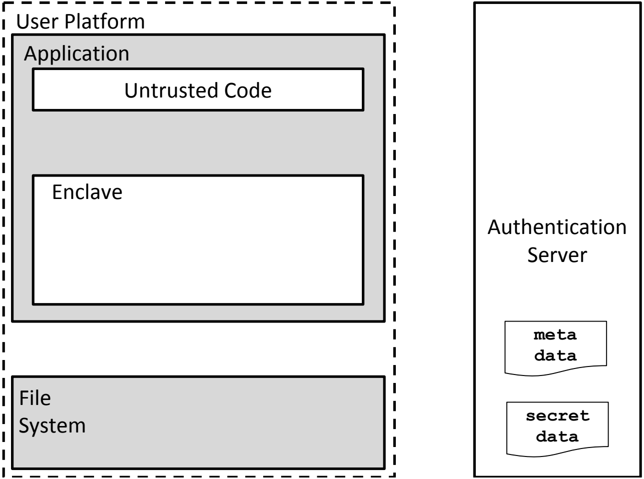
Remote vs. Local Data



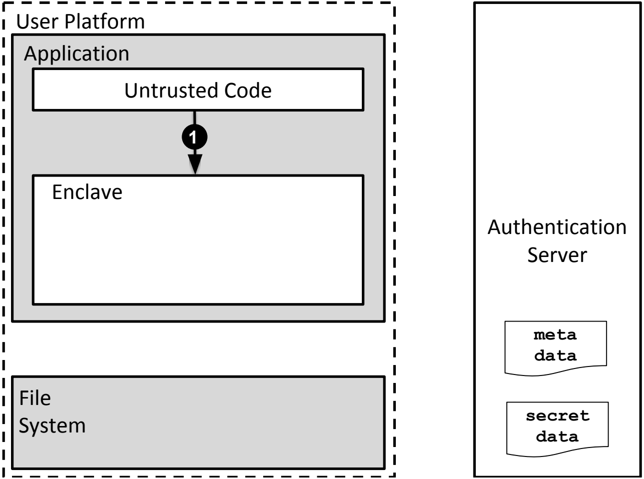
Remote vs. Local Data



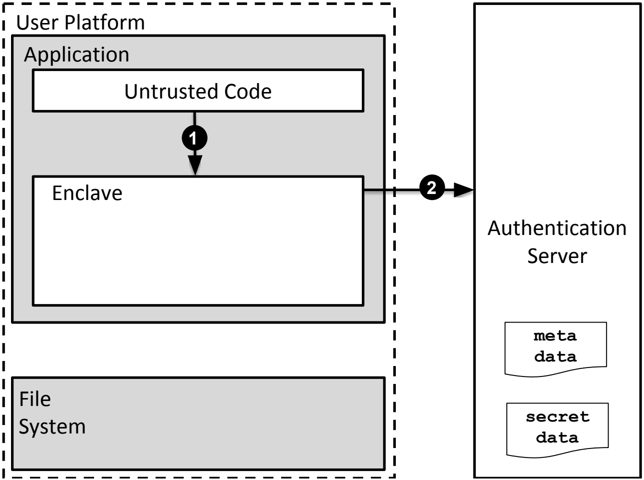
SgxElide Design - Remote Data



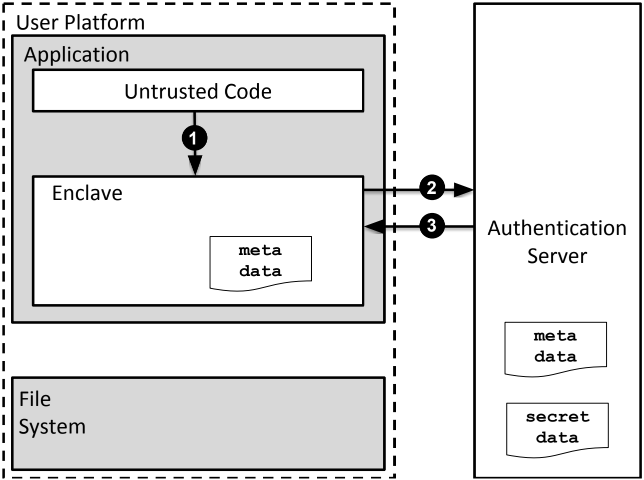
SgxElide Design - Remote Data



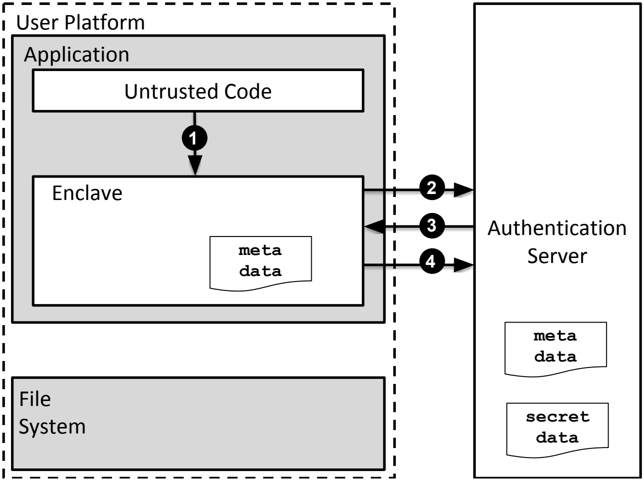
SgxElide Design - Remote Data



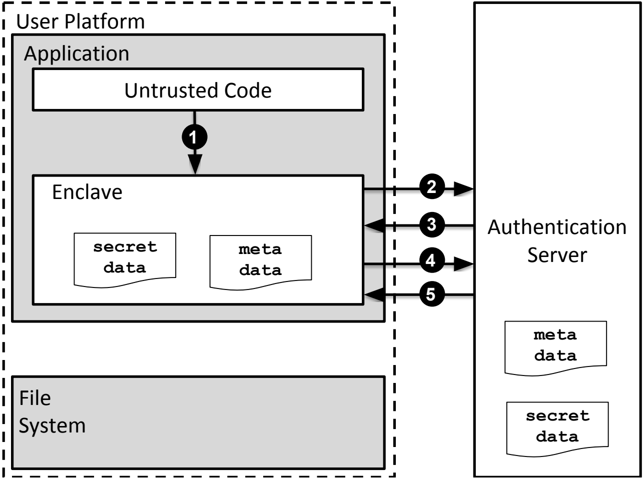
SgxElide Design - Remote Data



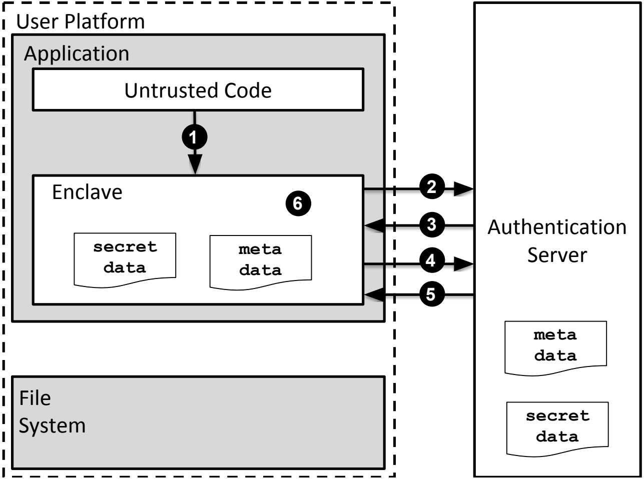
SgxElide Design - Remote Data



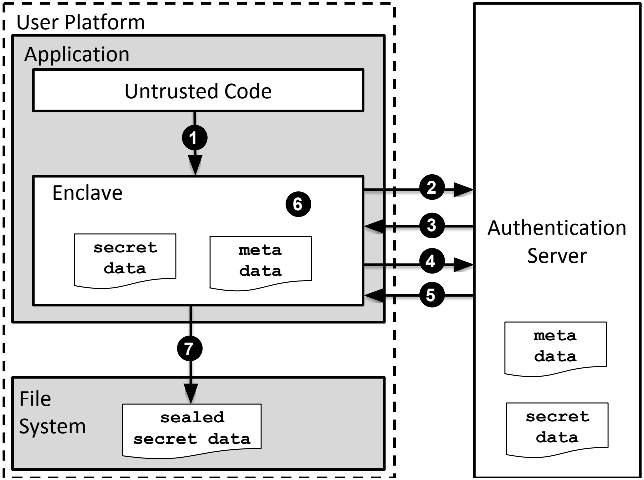
SgxElide Design - Remote Data



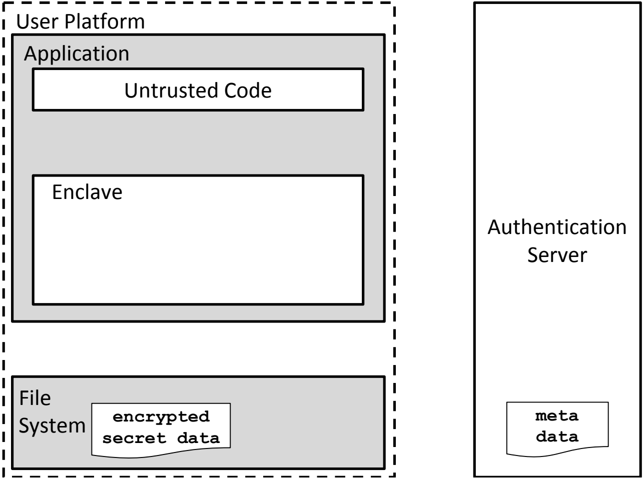
SgxElide Design - Remote Data



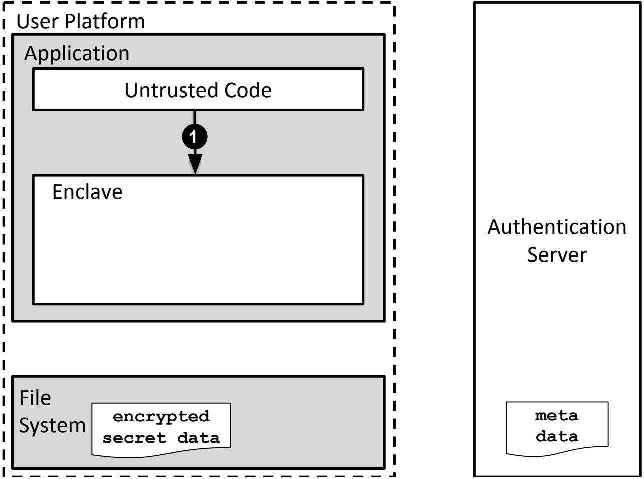
SgxElide Design - Remote Data



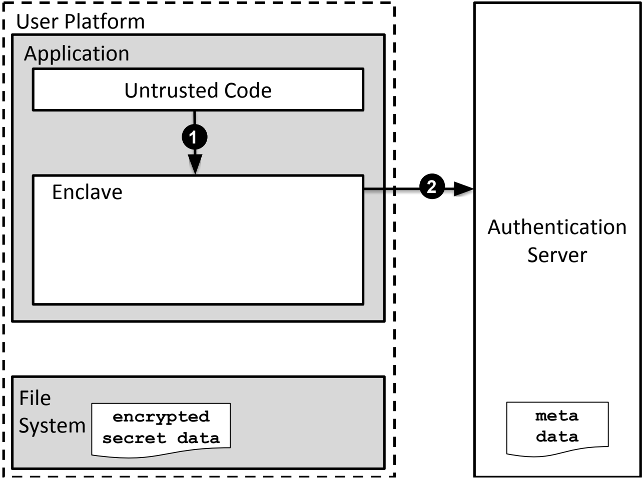
SgxElide Design - Local Data



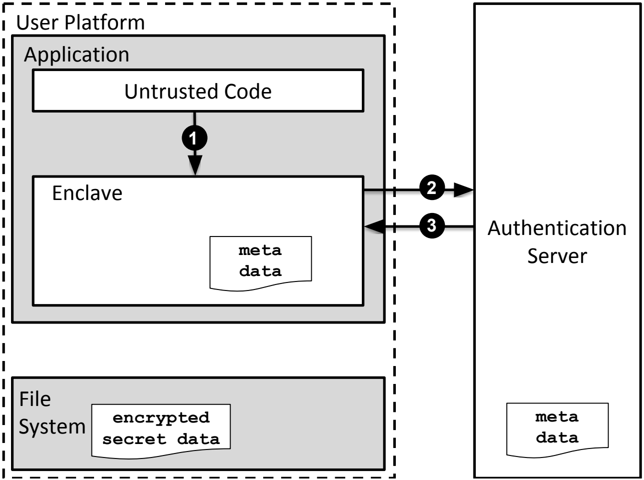
SgxElide Design - Local Data



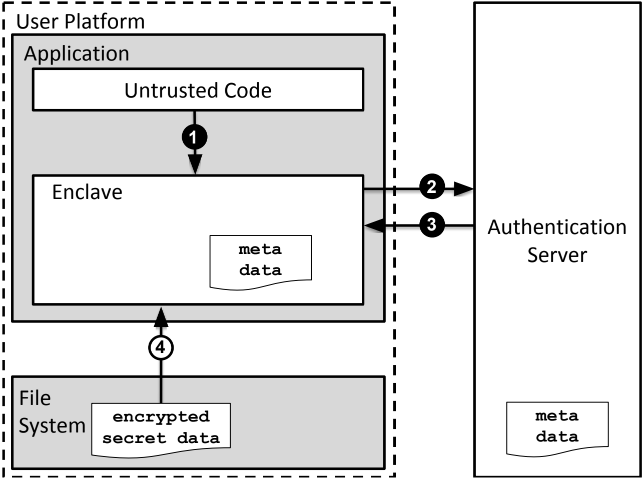
SgxElide Design - Local Data



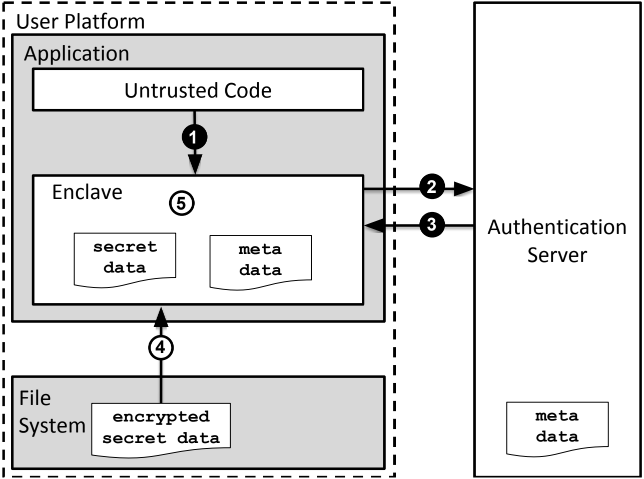
SgxElide Design - Local Data



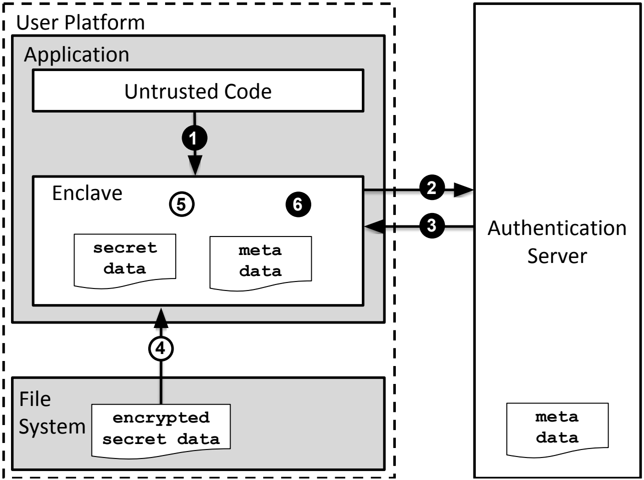
SgxElide Design - Local Data



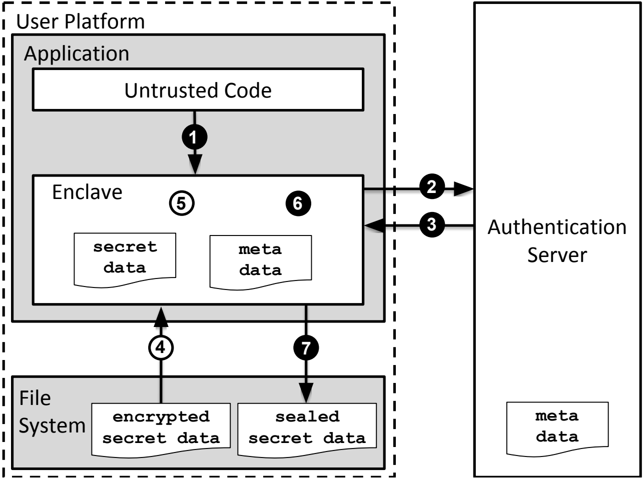
SgxElide Design - Local Data



SgxElide Design - Local Data



SgxElide Design - Local Data



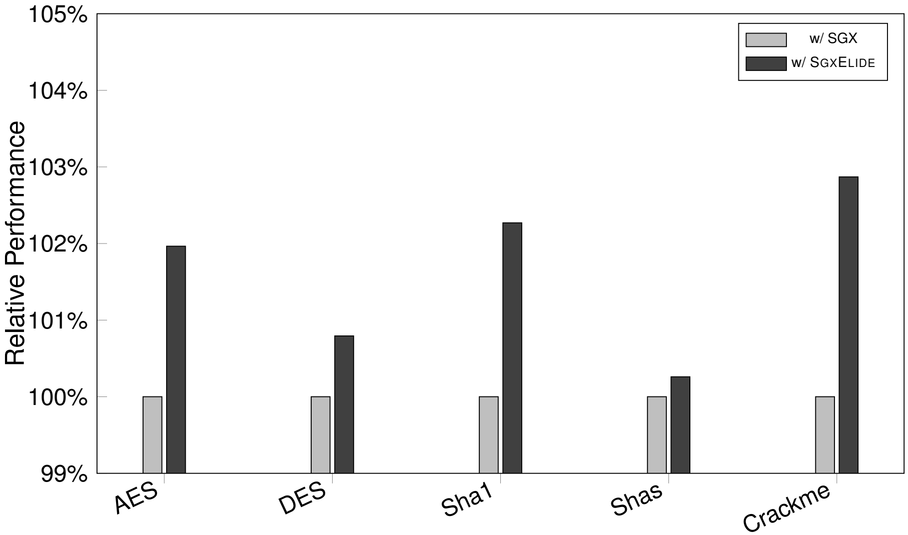
Benchmarks

Benchmarks	Original LOC	LOC w/ SGX		LOC w/ SGXELIDE		TC Functions	TC Bytes	Sanitized Functions	Sanitized Bytes
		UC	TC	UC	TC				
AES	802	472	427	522	540	185	75999	15	3840
DES	473	463	372	513	485	179	75455	9	3296
Sha1	315	423	251	473	364	179	73791	9	1632
Shas	2417	1529	1240	1579	1353	224	80127	54	7968
2048	413	551	192	601	305	208	76351	38	4448
Biniax	3523	3582	193	3632	306	208	76351	38	4448
Crackme	48	316	93	366	206	182	73711	12	1536

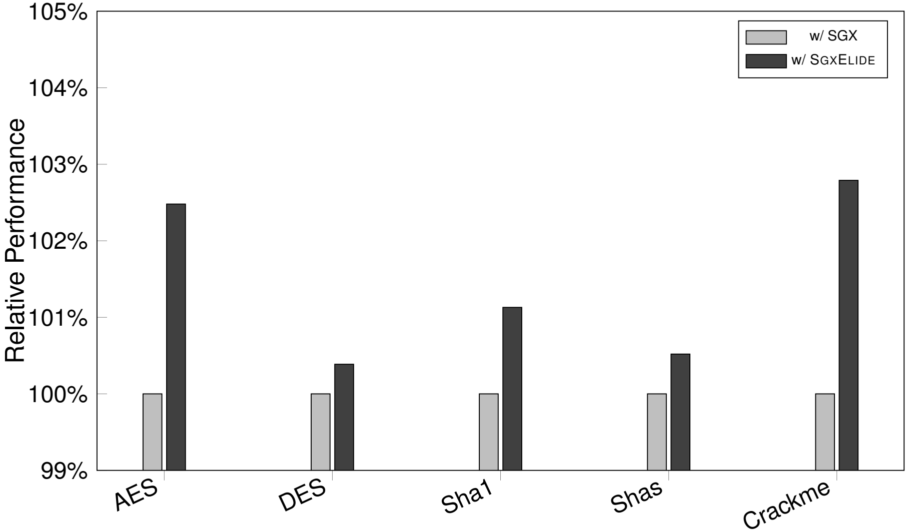
Sanitization/Restoration Time

Benchmarks	Remote Data				Local Data			
	Sanitize Time	Stand. Dev.	Restore Time	Stand. Dev.	Sanitize Time	Stand. Dev.	Restore Time	Stand. Dev.
AES	0.09	0.01	4.06	0.54	0.15	0.01	3.76	0.20
DES	0.09	0.01	3.99	0.52	0.14	0.01	3.97	0.75
Sha1	0.09	0.01	3.67	0.35	0.14	0.01	3.97	0.98
Shas	0.09	0.00	4.06	0.53	0.15	0.01	4.26	0.97
2048	0.09	0.01	3.78	0.52	0.15	0.01	3.73	0.28
Biniax	0.09	0.00	4.44	0.61	0.15	0.01	4.32	0.92
Crackme	0.09	0.01	3.53	0.28	0.15	0.00	3.54	0.78

SGXELIDE Overhead - Remote Data



SgxElide Overhead - Local Data



Discussions

SgxElide enclaves are self-modifying!

Discussions

SGXELIDE enclaves are self-modifying!

- How do we defend against malicious enclaves?

Discussions

SgxElide enclaves are self-modifying!

- How do we defend against malicious enclaves?
- How do we protect vulnerable enclaves?

Discussions

SgxElide enclaves are self-modifying!

- How do we defend against malicious enclaves?
- How do we protect vulnerable enclaves?
- How does this influence side-channel attacks?

Discussions

SGXELIDE enclaves are self-modifying!

- How do we defend against malicious enclaves?
- How do we protect vulnerable enclaves?
- How does this influence side-channel attacks?

Limitations and future work

Discussions

SgxElide enclaves are self-modifying!

- How do we defend against malicious enclaves?
- How do we protect vulnerable enclaves?
- How does this influence side-channel attacks?

Limitations and future work

- Framework not completely transparent

Discussions

SgxElide enclaves are self-modifying!

- How do we defend against malicious enclaves?
- How do we protect vulnerable enclaves?
- How does this influence side-channel attacks?

Limitations and future work

- Framework not completely transparent
- Would be useful to test SgxElide with large-scale software

Discussions

SGXELIDE enclaves are self-modifying!

- How do we defend against malicious enclaves?
- How do we protect vulnerable enclaves?
- How does this influence side-channel attacks?

Limitations and future work

- Framework not completely transparent
- Would be useful to test SGXELIDE with large-scale software
- Framework is proof-of-concept and not production ready

Conclusion

SgxElide

- Presented framework for SGX that ensures code confidentiality
- Sanitize enclave and dynamically restore code at runtime
- Evaluated SgxElide's performance with SGX benchmarks we developed
- Showed SgxElide has very little overhead with no performance penalty after restoration

Conclusion

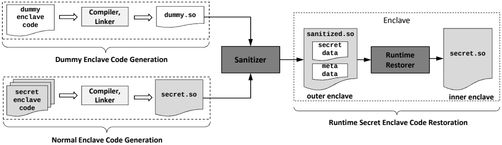
SgxElide

- Presented framework for SGX that ensures code confidentiality
- Sanitize enclave and dynamically restore code at runtime
- Evaluated SgxElide's performance with SGX benchmarks we developed
- Showed SgxElide has very little overhead with no performance penalty after restoration

SgxElide Source

github.com/utds3lab/sgxelide

Thank You



Q&A

erick.bauman@utdallas.edu

github.com/utds3lab/sgxelide