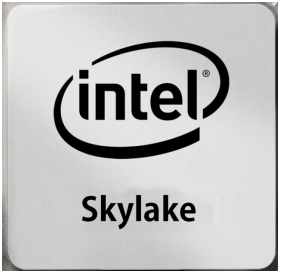
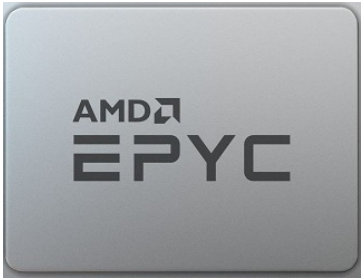
Towards Distributed and Virtualized Trusted Execution Environments

Zhiqiang Lin
Distinguished Professor of Engineering
zlin@cse.ohio-state.edu

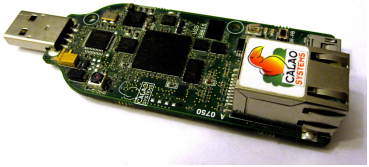
12/06/2022

Trusted Execution Environments (TEEs)





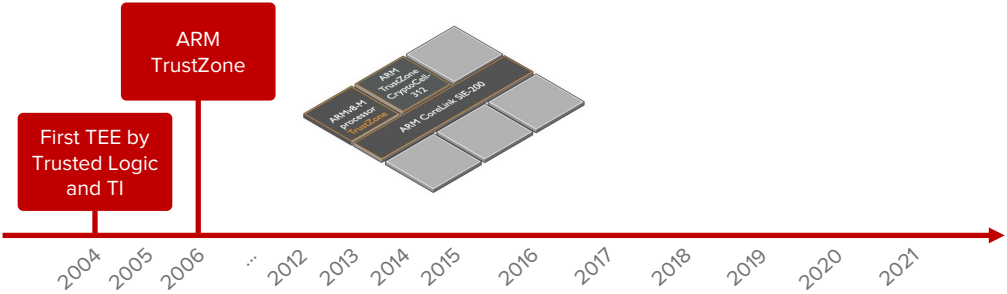
Trusted Execution Environments (TEEs)



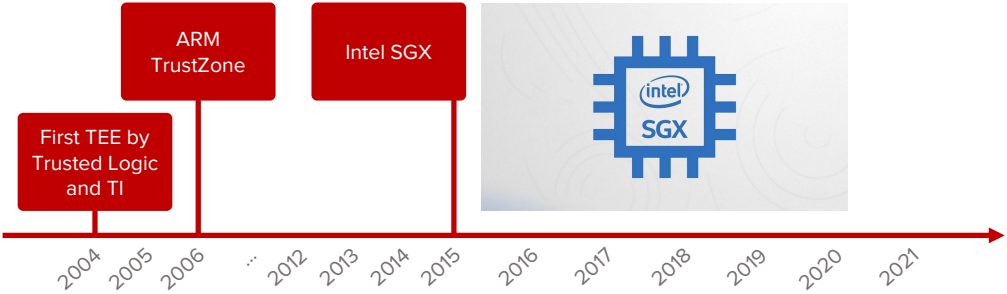
First TEE by Trusted Logic and TI

2004 2005 2006 ... 2012 2013 2014 2015 2016 2017 2018 2019 2020 2021

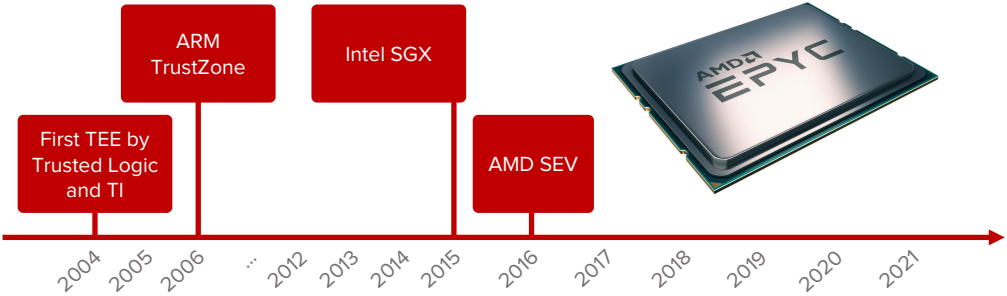
Trusted Execution Environments (TEEs)



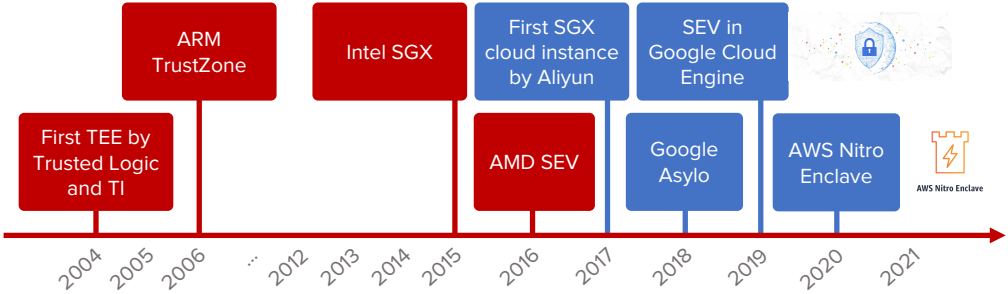
Trusted Execution Environments (TEEs)



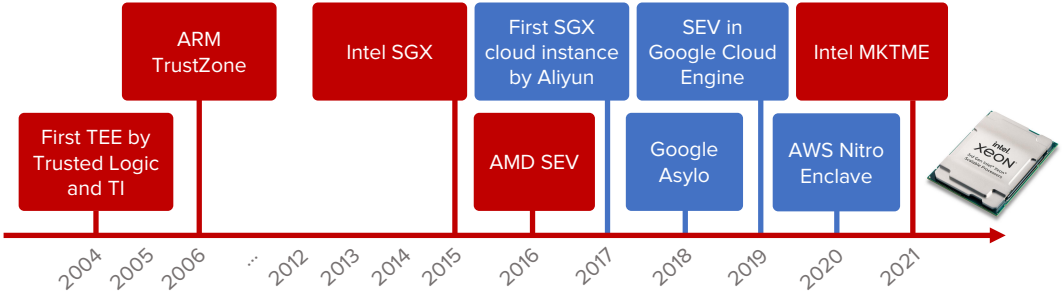
Trusted Execution Environments (TEEs)



Trusted Execution Environments (TEEs)



Trusted Execution Environments (TEEs)



Numerous Applications w/ TEEs

1. SGX-based password manager [KKP+18]
2. SGX-based anonymity network [KHH+17]
3. Privacy-preserving data analytics (e.g., [SCF+15]) and machine learning (e.g., [KPM+16, OSF+16]),
4. SGX-based game protection [PAL20]
5. Privacy-preserving contact-tracing (e.g., SafeTrace [eni]) and blockchains [CXZW21]

Our Efforts in TEE Research

1. "A Case for Protecting **Computer Games** With SGX". SysTex 2016
2. "SGX-Log: Securing **System Logs** With SGX". ASIACCS 2017
3. "Securing **Data Analytics** on SGX With Randomization". ESORICS 2017
4. "SGX-BigMatrix: A Practical Encrypted **Data Analytic** Framework With Trusted Processors". CCS 2017
5. "SGX-Elide: Enabling Enclave **Code Secrecy** via Self-Modification". CGO 2018
6. "Exploiting **Unprotected I/O Operations** in AMD's SEV", USENIX Security 2019
7. "Automatically Identifying **Side-Channel Vulnerabilities in Enclave Binaries**". RAID 2019
8. "Towards **Memory Safe** Enclave Programming with Rust-SGX". CCS 2019
9. "CROSSLINE: Breaking 'Security-by-Crash' based Memory Isolation in AMD SEV". CCS 2021
10. "Towards **Formal Verification** of State Continuity for **Enclave Programs**". USENIX Security 2021
11. "vSGX: **Virtualizing SGX on AMD SEV**". Oakland 2022
12. "Towards A **TEE-based V2V Protocol** For Connected And Autonomous Vehicles". AutoSec 2022
13. Controlled Data Races in Enclaves: Attacks and Detection". USENIX Security 2023

Our Efforts in TEE Research

1. ~~"A Case for Protecting **Computer Games** With SGX". SysTex 2016~~
2. ~~"SGX-Log: Securing **System Logs** With SGX". ASIACCS 2017~~
3. ~~"Securing **Data Analytics** on SGX With Randomization". ESORICS 2017~~
4. ~~"SGX-BigMatrix: A Practical Encrypted **Data Analytic** Framework With Trusted Processors". CCS 2017~~
5. ~~"SGX-Elide: Enabling Enclave **Code Secrecy** via Self-Modification". CGO 2018~~
6. ~~"Exploiting **Unprotected I/O Operations** in AMD's SEV", USENIX Security 2019~~
7. ~~"Automatically Identifying **Side-Channel Vulnerabilities in Enclave Binaries**". RAID 2019~~
8. ~~"Towards **Memory Safe** Enclave Programming with Rust-SGX". CCS 2019~~
9. ~~"CROSSLINE: Breaking 'Security-by-Crash' based Memory Isolation in AMD SEV". CCS 2021~~
10. "Towards **Formal Verification** of State Continuity for **Enclave Programs**". USENIX Security 2021
11. "vSGX: **Virtualizing SGX on AMD SEV**". Oakland 2022
12. "Towards A **TEE-based V2V Protocol** For Connected And Autonomous Vehicles". AutoSec 2022
13. ~~Controlled Data Races in Enclaves: Attacks and Detection". USENIX Security 2023~~

Problems in Current TEEs: (1) Vendor lock-in and (2) Slow evolution

1. SGX is an ISA extension → apps have to be written specifically for SGX and can't run else where (e.g., you can't run it on AMD chips)
2. Enumerous side channel vulnerabilities and attacks (e.g., [XCP15, SCNS16, SWG+17, BMD+17, HCP17, GESM17, LSG+17, VBMW+18, vSMÖ+19, SLM+19, CGG+19, CCX+19]) → The defense requires faster patching

The Need to Decouple TEEs from the Hardware (Virtualized TEE)

Recent Trend

1. Unifying TEE SDKs
   - Asoly by Google [Goo], by Vmware [VMw]
2. Software defined TEE
   - Komodo [FBHP17], but not binary compatibile

The Need to Decouple TEEs from the Hardware (Virtualized TEE)

Recent Trend

1. Unifying TEE SDKs
   - Asoly by Google [Goo], by Vmware [VMw]
2. Software defined TEE
   - Komodo [FBHP17], but not binary compatibile

Benefits of Software Defined TEE

1. No vendor lock-in
2. Flexibility on deployment
3. Faster feature evolution
4. Faster bug fixes
5. Migration

The Need to Decouple TEEs from the Hardware (Virtualized TEE)

Recent Trend

1. Unifying TEE SDKs
   - Asoly by Google [Goo], by Vmware [VMw]
2. Software defined TEE
   - Komodo [FBHP17], but not binary compatibile

Benefits of Software Defined TEE

1. No vendor lock-in
2. Flexibility on deployment
3. Faster feature evolution
4. Faster bug fixes
5. Migration

Our goal: Can we design a binary comptible, software defined TEE? If so, how?

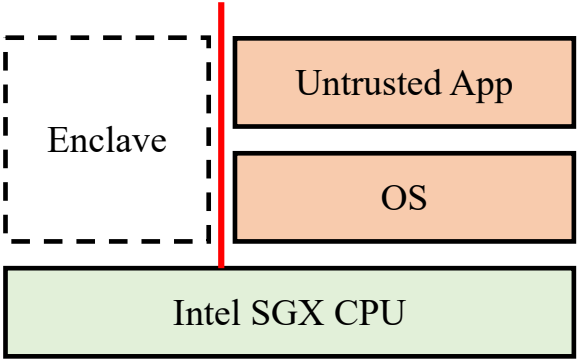
SGX 101: SGX Instructions

SGX Version	User Space `enclu`	Kernel Space `encls`	Total
SGX-v1	5	13	18
SGX-v2	5+3	13+3	18+6

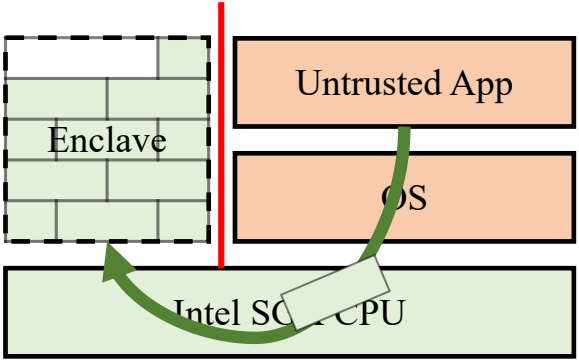
SGX 101: SGX Instructions

Privilege	Type	Instruction	Description	Version
P	MEM	EADD	Add a page	v1
P	MEM	EBLOCK	Block an EPC page	v1
P	EXE	ECREATE	Create an enclave	v1
P	DBG	EDBGRD	Read data by debugger	v1
P	DBG	EDBGWR	Write data by debugger	v1
P	MEM	EEXTEND	Extend EPC page measurement	v1
P	EXE	EINIT	Initialize an enclave	v1
P	MEM	ELDB	Load an EPC page as blocked	v1
P	MEM	ELDU	Load an EPC page as unblocked	v1
P	SEC	EPA	Add version array	v1
P	MEM	EREMOVE	Remove a page from EPC	v1
P	MEM	ETRACK	Activate EBLOCK checks	v1
P	MEM	EWB	Write back/invalidate an EPC page	v1
P	MEM	EAUG	Allocate page to an existing enclave	v2
P	SEC	EMODPR	Restrict page permissions	v2
P	EXE	EMODT	Change the type of an EPC page	v2
U	EXE	EENTER	Enter an enclave	v1
U	EXE	EEXIT	Exit an enclave	v1
U	SEC	EGETKEY	Create a cryptographic key	v1
U	SEC	EREPORT	Create a cryptographic report	v1
U	EXE	ERESUME	Re-enter an enclave	v1
U	MEM	EACCEPT	Accept changes to a page	v2
U	SEC	EMODPE	Enhance access rights	v2
U	MEM	EACCEPTCOPY	Copy page to a new location	v2

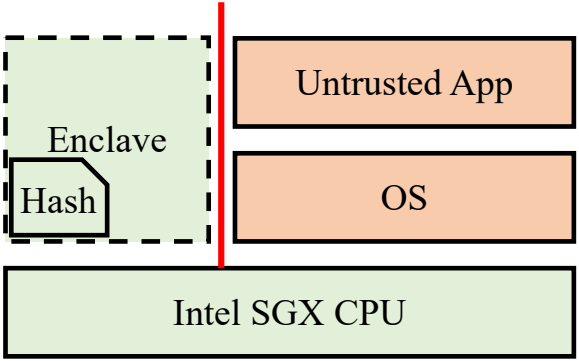
SGX 101: ECREATE, EADD; EEXTEND, EINIT



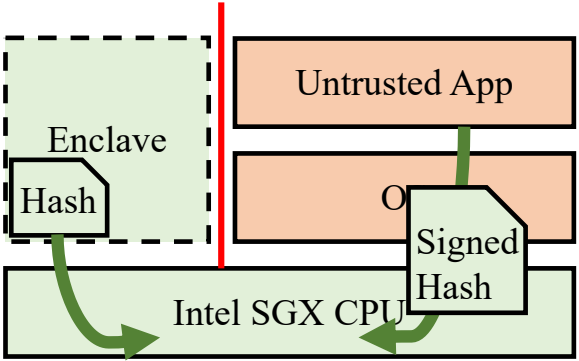
SGX 101: ECREATE, EADD; EEXTEND, EINIT



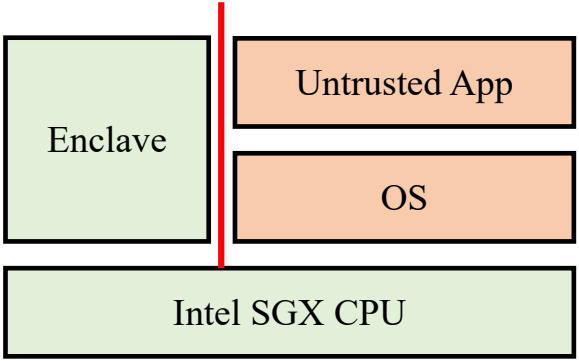
SGX 101: ECREATE, EADD; EEXTEND, EINIT



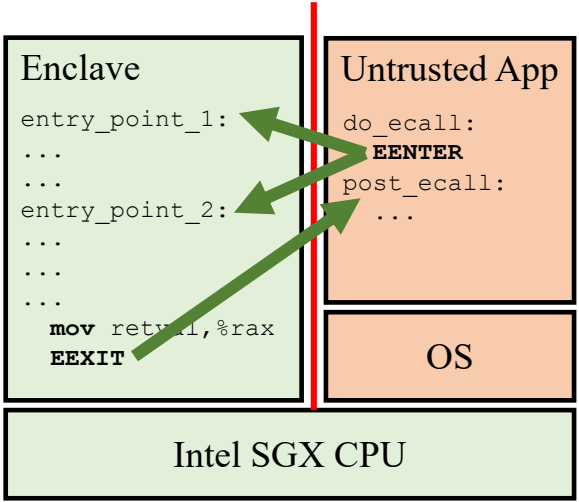
SGX 101: ECREATE, EADD; EEXTEND, EINIT



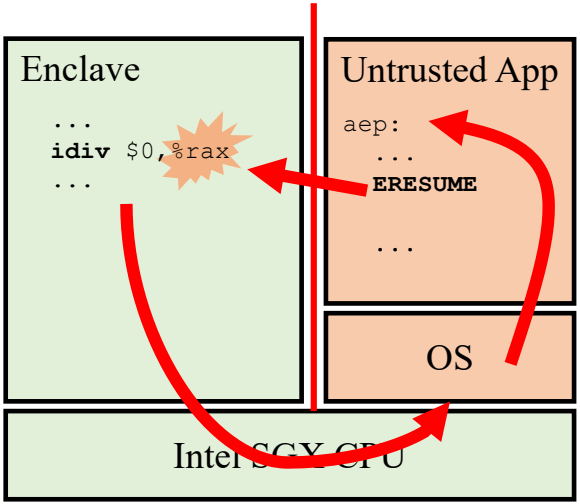
SGX 101: ECREATE, EADD; EEXTEND, EINIT



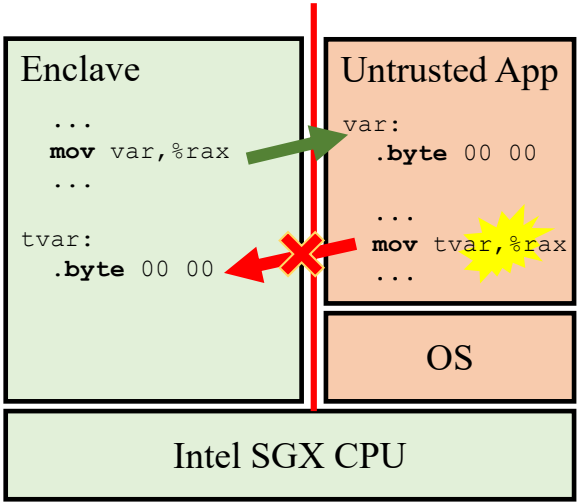
SGX 101: EENTER, EEXIT; AEX, ERESUME



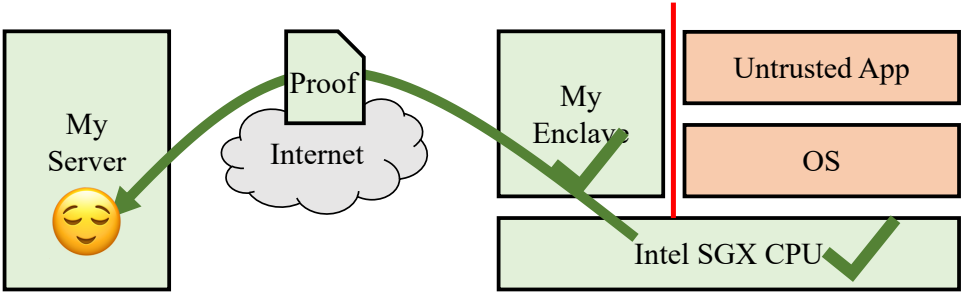
SGX 101: EENTER, EEXIT; AEX, ERESUME



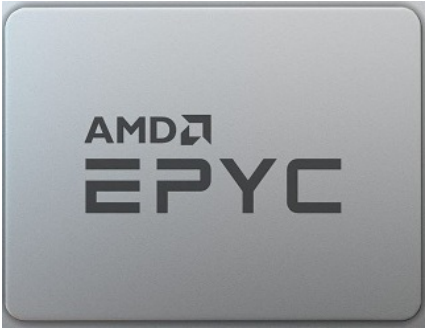
SGX 101: Enclave Memory in SGX

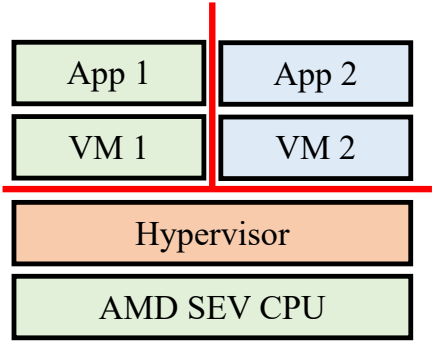


SGX 101: Attestation in SGX

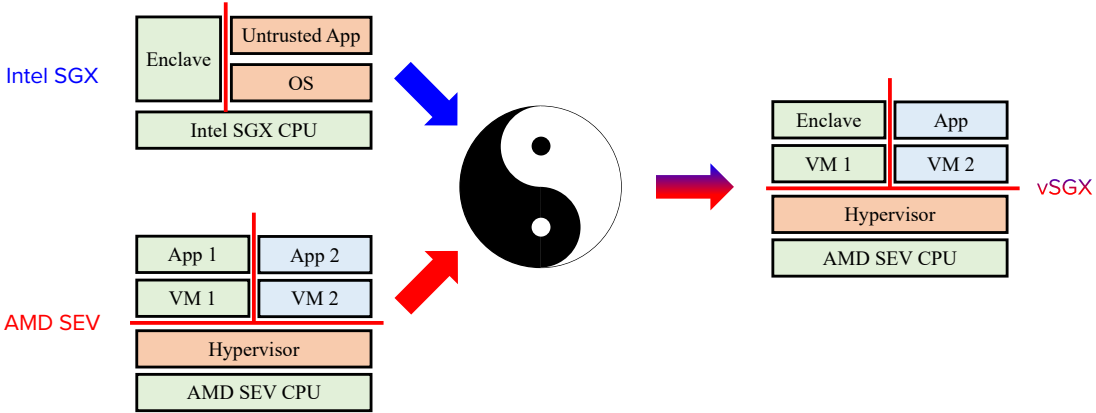


AMD Secure Encrypted Virtualization (SEV)

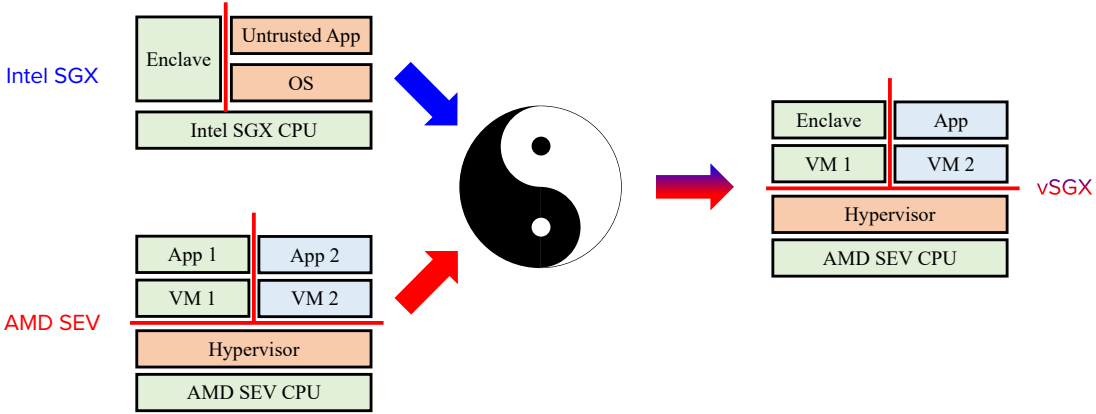




vSGX: Virtualizing SGX Enclaves on AMD SEV

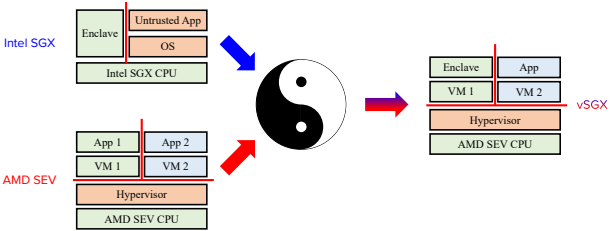


vSGX: Virtualizing SGX Enclaves on AMD SEV



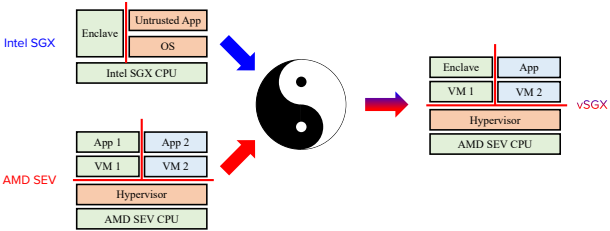
"*vSGX: Virtualizing SGX Enclaves on AMD SEV*". S. Zhao, M. Li, Y. Zhang, and Z. Lin. S&P'22.

Design Goals



1. Binary compatibility
2. Comparable security guarantee with both SGX and SEV
3. Reasonable performance

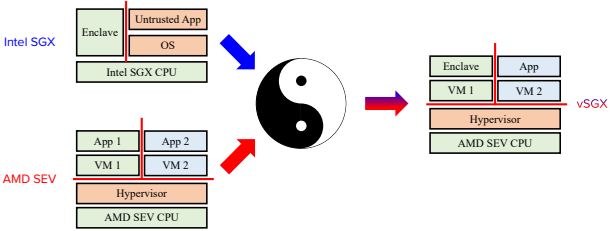
Design Goals



1. Binary compatibility
2. Comparable security guarantee with both SGX and SEV
3. Reasonable performance

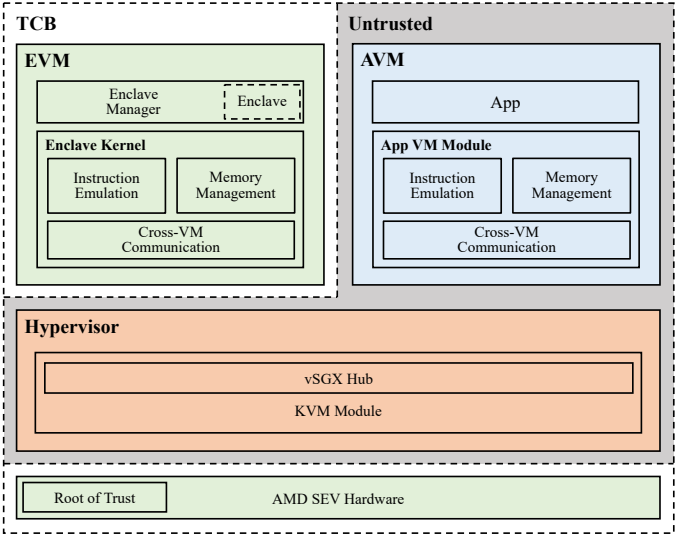
vSGX should work like an SGX module plugged onto an SEV machine

Challenges



1. How to isolate the enclave from others components?
2. How to execute SGX instructions on SEV?
3. How to handle memory access inside an enclave?
4. How to connect between an enclave and other components?

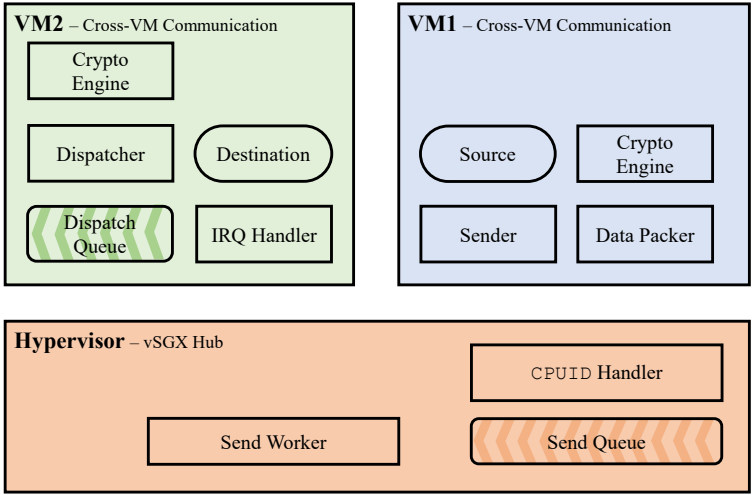
vSGX Architecture



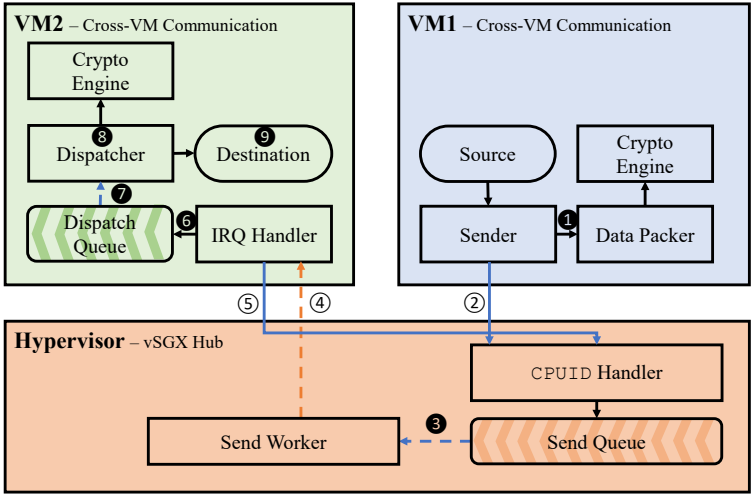
SGX Instructions Emulation in vSGX

	Leaf Instruction	Instruction Descriptions	Support	New Enclave	EPCM Modification	Add EPC Page	Remove EPC Page	VM Transmission EVM - AVM	Req. Pkg. Size	Rsp. Pkg. Size
ENCLS	EADD	Add an page to an uninitialized enclave	✓	✗	✓	✓	✗	● ← ◯	4185	19
	EAUG	Add an page to an initialized enclave	✓	✗	✓	✓	✗	● ← ◯	25	19
	EBLOCK	Block an EPC page	✓	✗	✓	✗	✗	● ← ◯	9	19
	ECREATE	Create a SECS page in EPC	✓	✓	✓	✓	✗	● ← ◯	4105	19
	EDBGRD	Read from a debug enclave	✗	-	-	-	-	-	-	-
	EDBGWR	Write to a debug enclave	✗	-	-	-	-	-	-	-
	EEXTEND	Extend uninitialized enclave's measurement	✓	✗	✗	✗	✗	● ← ◯	9	19
	EINIT	Initialize an enclave	✓	✗	✗	✗	✗	● ← ◯	2137	19
	ELDB/ELDU	Load a page to enclave	✓	P**	✓	✓	✗	● ← ◯	8370	4131
	EMODPR	Restrict an EPC page's permission	✓	✗	✓	✗	✗	● ← ◯	12	19
	EMODT	Change an EPC page's type	✓	✗	✓	✗	✗	● ← ◯	12	19
	EPA	Add version array	✓	✗	✓	✓	✗	● ← ◯	9	4131
	EREMOVE	Remove a page from EPC	✓	✗	✓	✗	✓	● ← ◯	9	19
	ETRACK	Block until EBLOCK is done	✓	-	-	-	-	-	-	-
	EWB	Write an EPC page to main memory	✓	✗	✓	✗	✓	● ← ◯	4137	8355
ENCLU	EACCEPT	Accept changes to an EPC page	✓	-	✓	✗	✗	◎ -	-	-
	EACCEPTCOPY	Copy a page to a new EPC page	✓	-	✓	✗	✗	◎ -	-	-
	EENTER	Enter an enclave	✓	-	✗	✗	✗	● ← ◯	177	19
	EEXIT	Exit an enclave	✓	-	✗	✗	✗	◎ → ●	153	-
	EGETKEY	Derive a key	✓	-	✗	✗	✗	◎ -	-	-
	EMODPE	Extend permission of an EPC page	✓	-	✓	✗	✗	◎ -	-	-
	EREPORT	Create a cryptographic report	✓	-	✗	✗	✗	◎ -	-	-
	ERESUME	Resume an enclave	✓	-	✗	✗	✗	● ← ◯	33	19
Behavior										
	AEX	Exit an enclave due to interrupt or fault	✓*	-	✗	✗	✗	◎ → ●	166	-

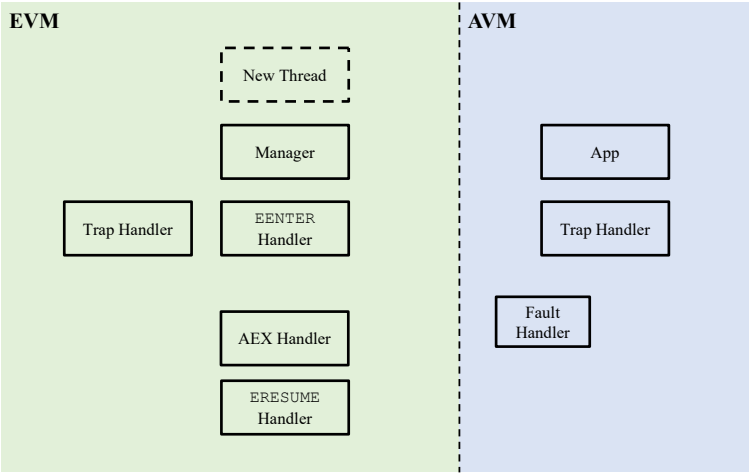
Cross VM Communication



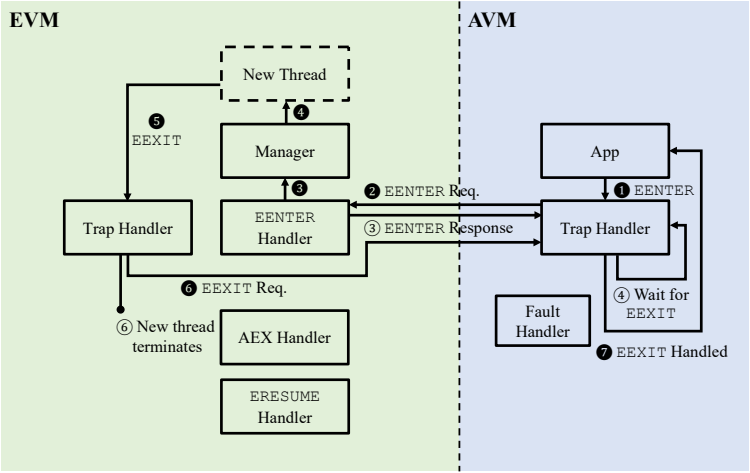
Cross VM Communication



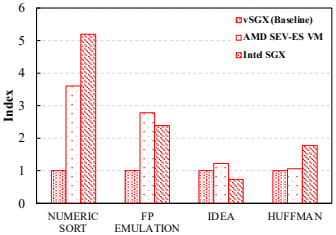
Control Flow Transferring: EENTER, EEXIT



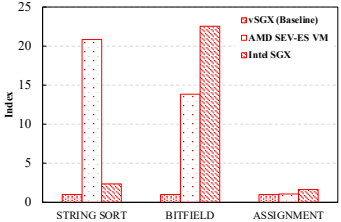
Control Flow Transferring: EENTER, EEXIT



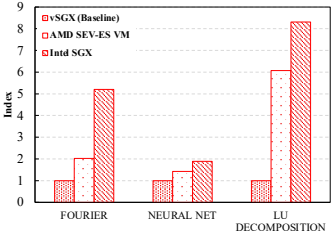
Performance - BYTEmark



(a) **CPU Intensive Test**

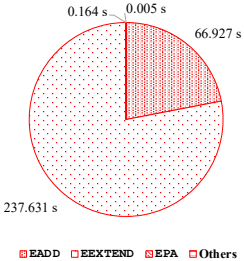


(b) **Memory Intensive Test**

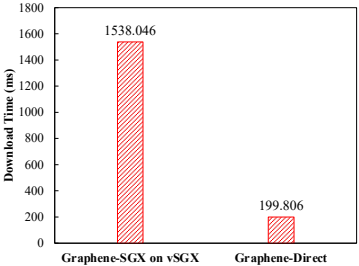


(c) **FP Intensive Test**

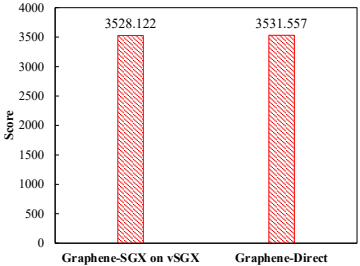
Performance - Graphene



(a) Time Consumption Launching Graphene SGX on vSGX



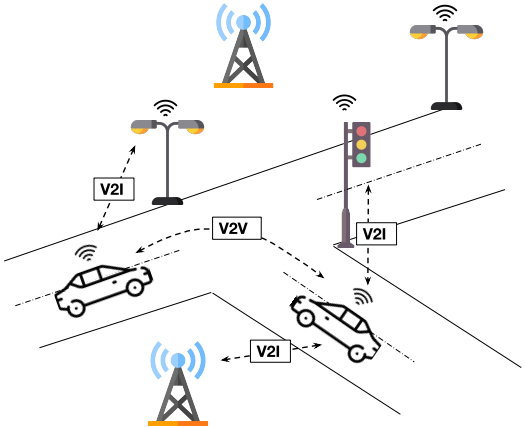
(b) cURL Execution Time



(c) GMPbench 0.2 Score

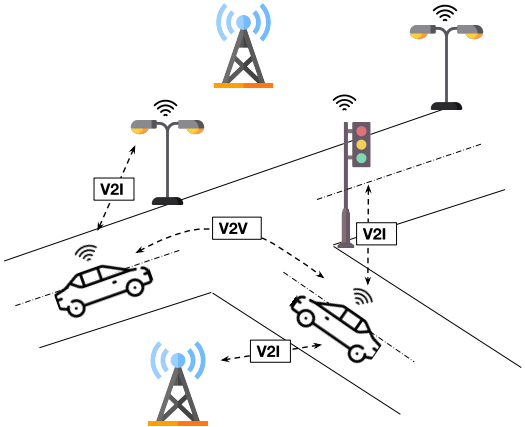
Connected and Automotive Vehicles (CAVs)

- Vehicle To Infrastructure (V2I)
- Vehicle To Vehicle (V2V)
- Benefits
  1. Traffic Safety
  2. Traffic Efficiency
  3. Energy Savings

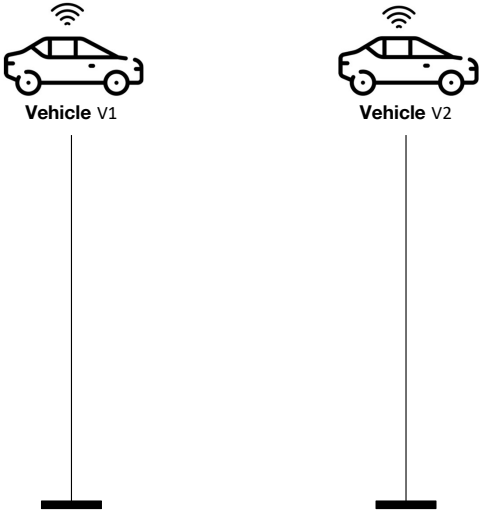


V2V Protocol Requirements

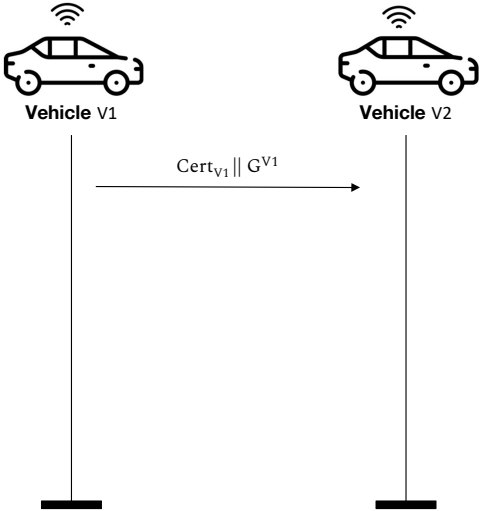
1. **(Instant) Authentication**
2. Message Integrity (Tamper Resiliency)
3. Message Confidenality (Privacy)
4. **Anti-tracking** (Privacy)
5. Revocation and **Accountability**
6. Scalability



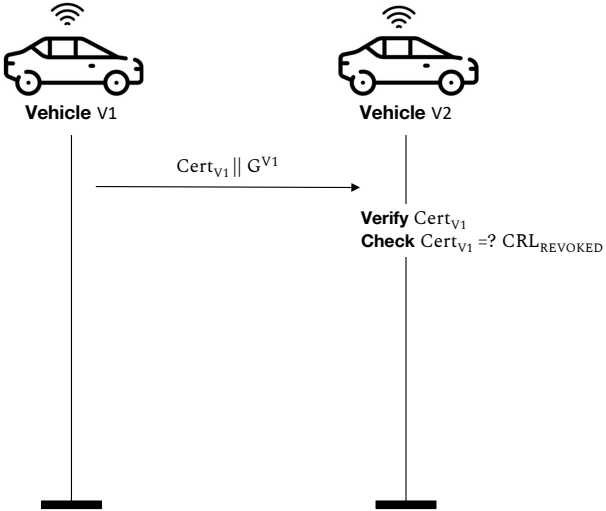
PKI-based V2V Encrypted Broadcast



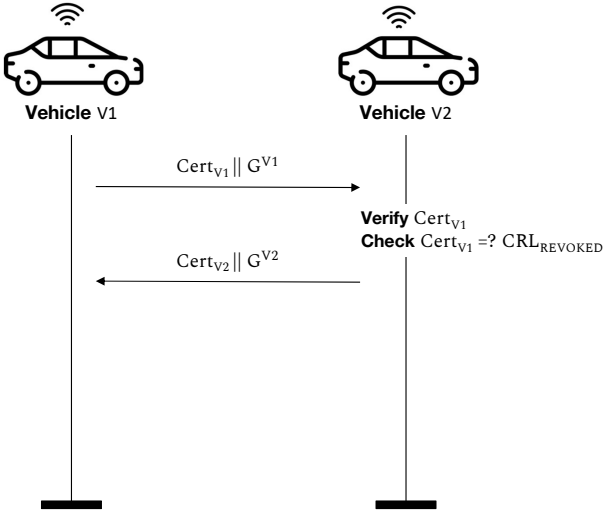
PKI-based V2V Encrypted Broadcast



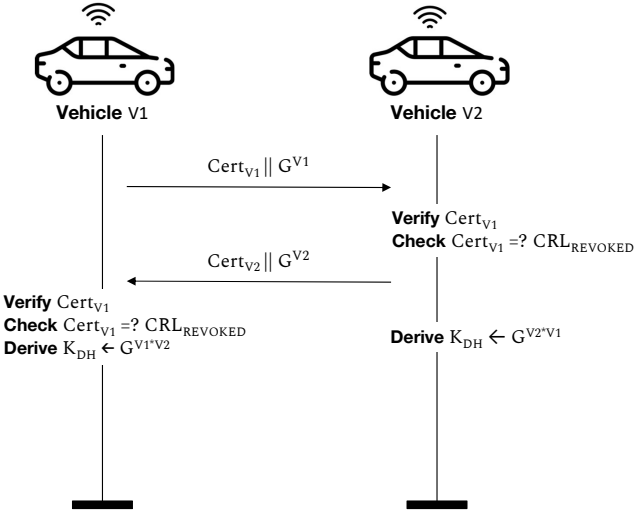
PKI-based V2V Encrypted Broadcast



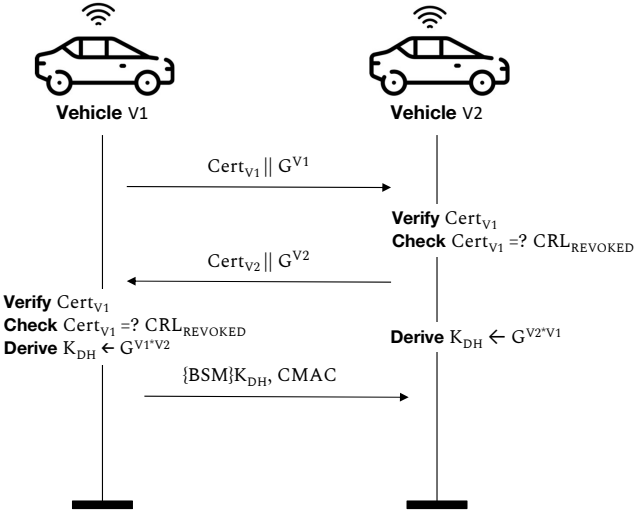
PKI-based V2V Encrypted Broadcast



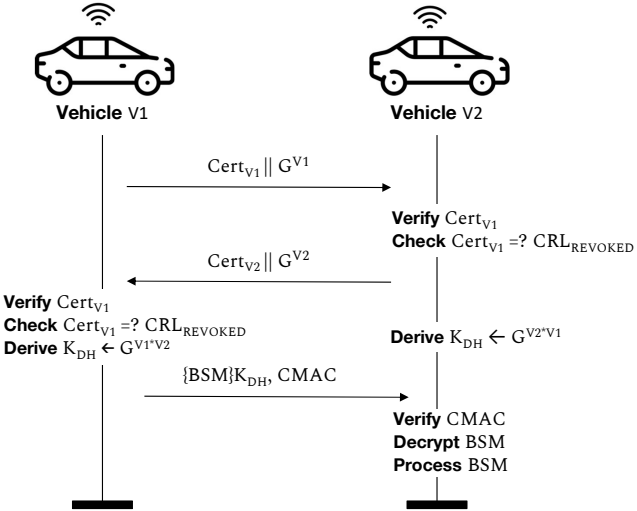
PKI-based V2V Encrypted Broadcast



PKI-based V2V Encrypted Broadcast



PKI-based V2V Encrypted Broadcast



Our Approach

Key Objective

1. **(Instant) Authentication**
2. Message Integrity (Tamper Resiliency)
3. Message Confidenality (Privacy)
4. **Anti-tracking** (Privacy)
5. Revocation and **Accountability**
6. Scalability

Our Approach

Key Objective

1. **(Instant) Authentication**
2. Message Integrity (Tamper Resiliency)
3. Message Confidenality (Privacy)
4. **Anti-tracking** (Privacy)
5. Revocation and **Accountability**
6. Scalability

Our Approach

- Leveraging TEEs to protect secrets
- Using a shared Daily Symmetric (DS) Key among vehicles
- Storing DS key in the enclave
- Using Remote Attestation to ensure the right environment
- Generating temporal IDs for privacy

Our Approach

Key Objective

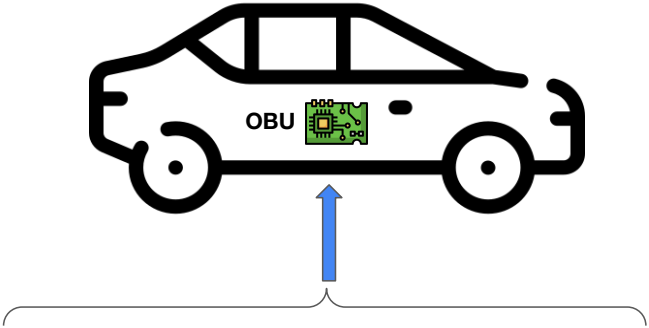
1. **(Instant) Authentication**
2. Message Integrity (Tamper Resiliency)
3. Message Confidenality (Privacy)
4. **Anti-tracking** (Privacy)
5. Revocation and **Accountability**
6. Scalability

Our Approach

- Leveraging TEEs to protect secrets
- Using a shared Daily Symmetric (DS) Key among vehicles
- Storing DS key in the enclave
- Using Remote Attestation to ensure the right environment
- Generating temporal IDs for privacy

"*Towards A TEE-based V2V Protocol For Connected And Autonomous Vehicles*". Mohit K. Jangid and Zhiqiang Lin. AutoSec 2022.

The Protocol: Manufacture Phase (One Time)

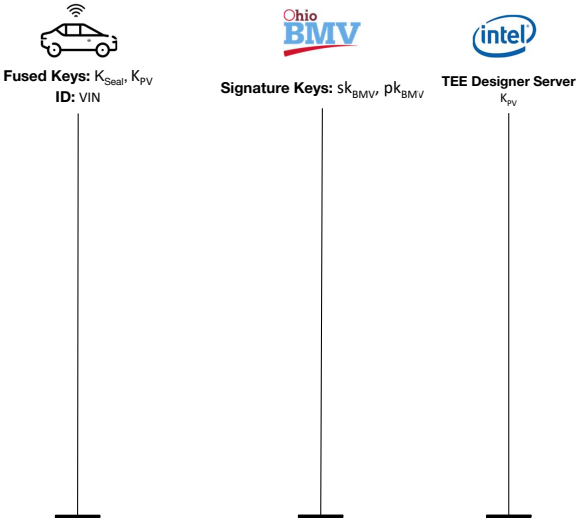


Vehicle Manufacturer

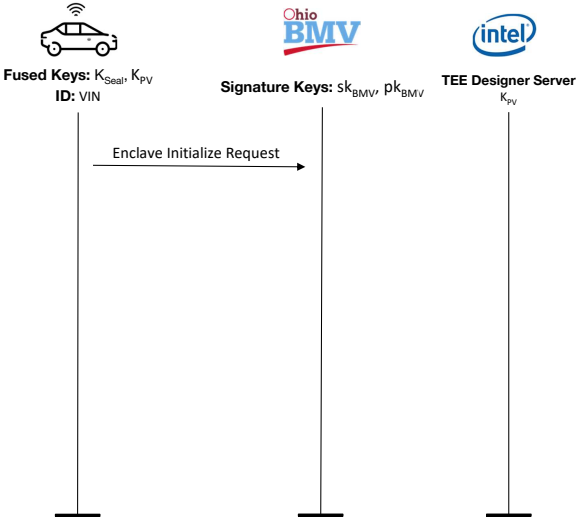
TEE Designer

Collaborate and Integrate into Vehicle On-Board unit

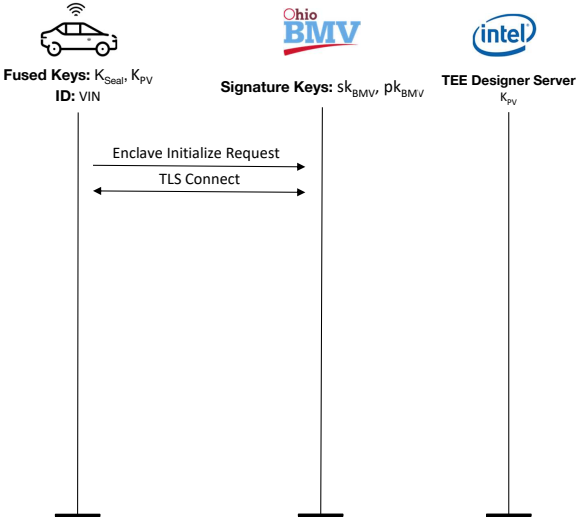
The Protocol: Registration Phase (One Time)



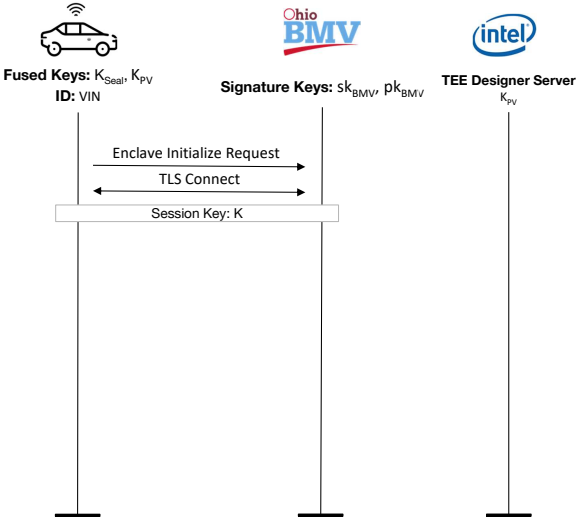
The Protocol: Registration Phase (One Time)



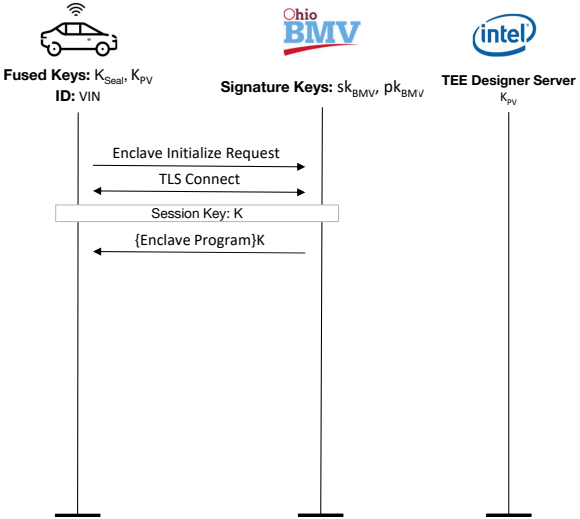
The Protocol: Registration Phase (One Time)



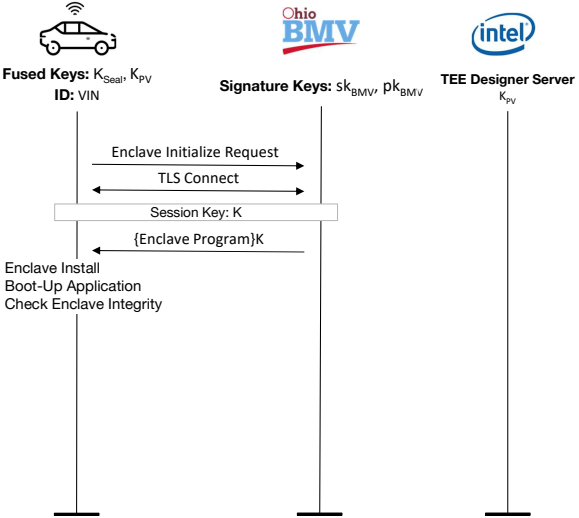
The Protocol: Registration Phase (One Time)



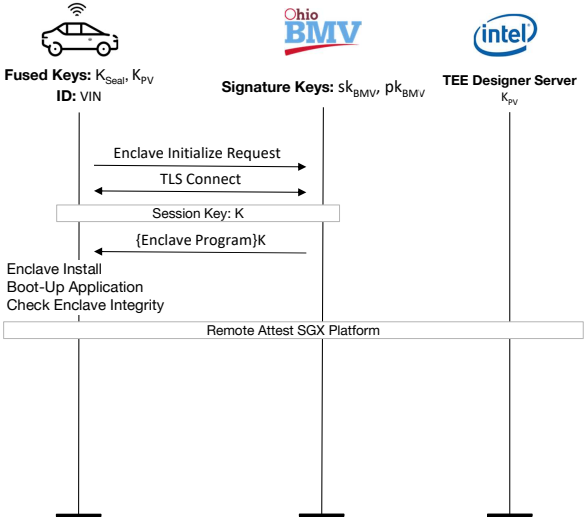
The Protocol: Registration Phase (One Time)



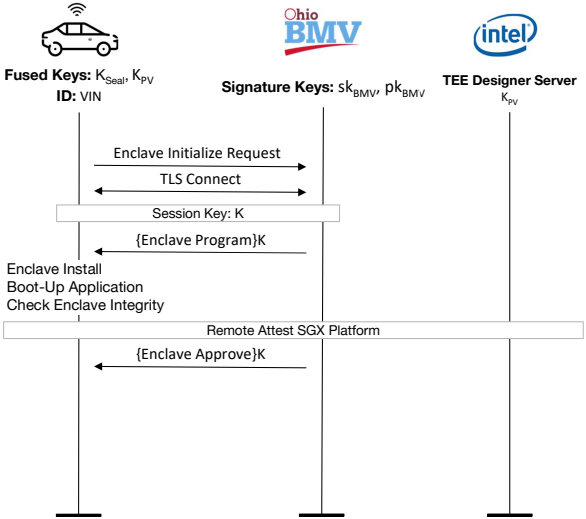
The Protocol: Registration Phase (One Time)



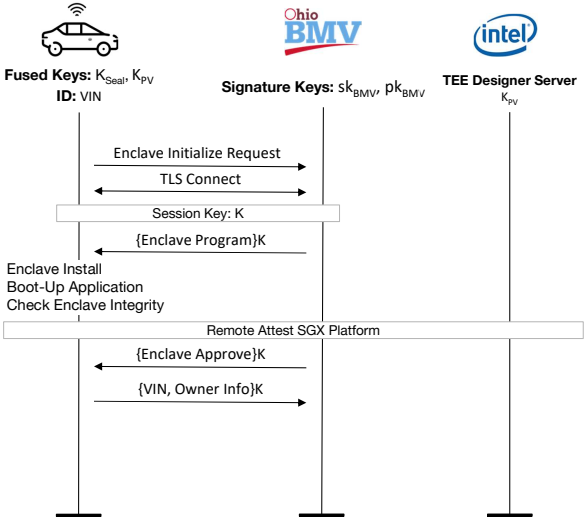
The Protocol: Registration Phase (One Time)



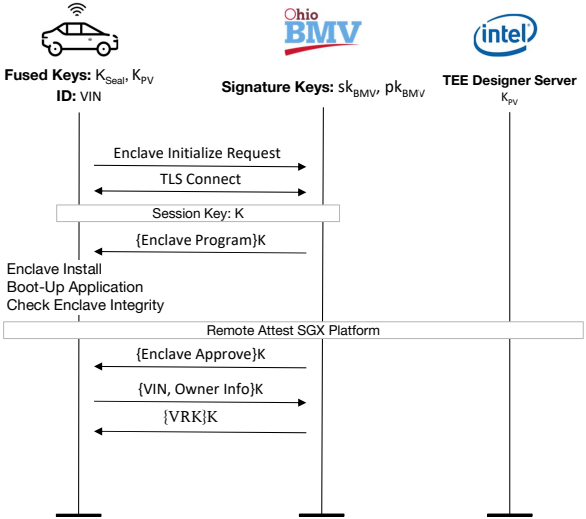
The Protocol: Registration Phase (One Time)



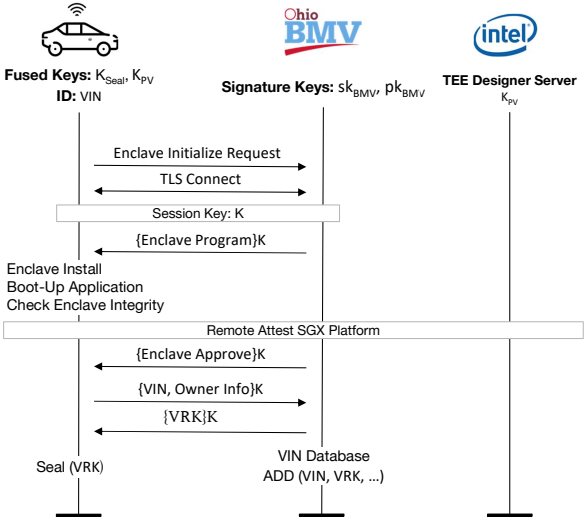
The Protocol: Registration Phase (One Time)



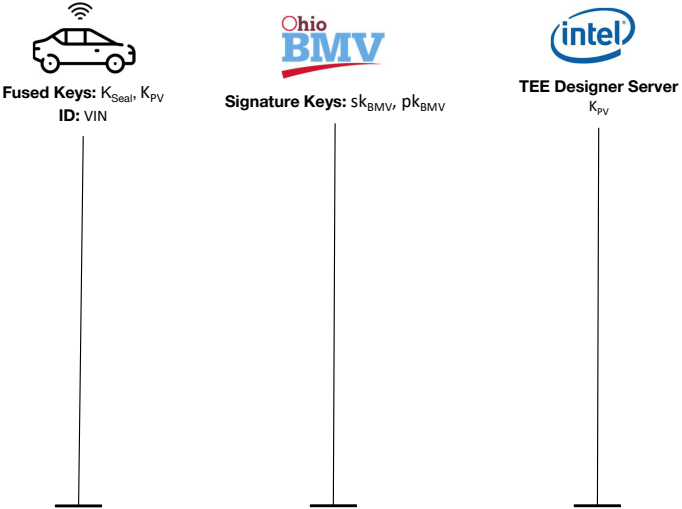
The Protocol: Registration Phase (One Time)



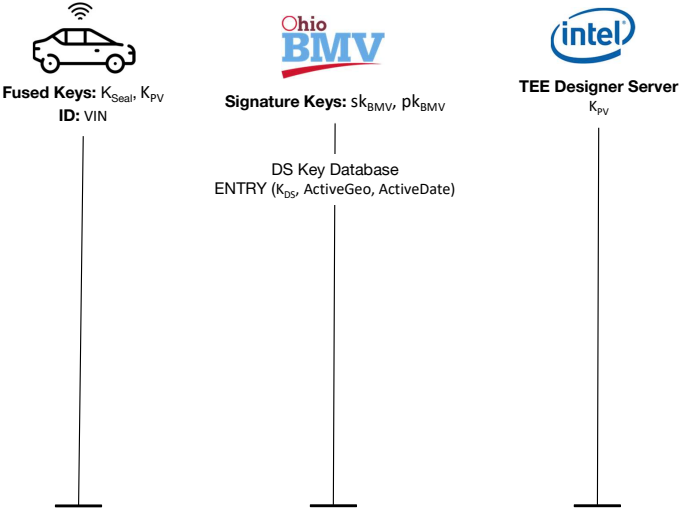
The Protocol: Registration Phase (One Time)



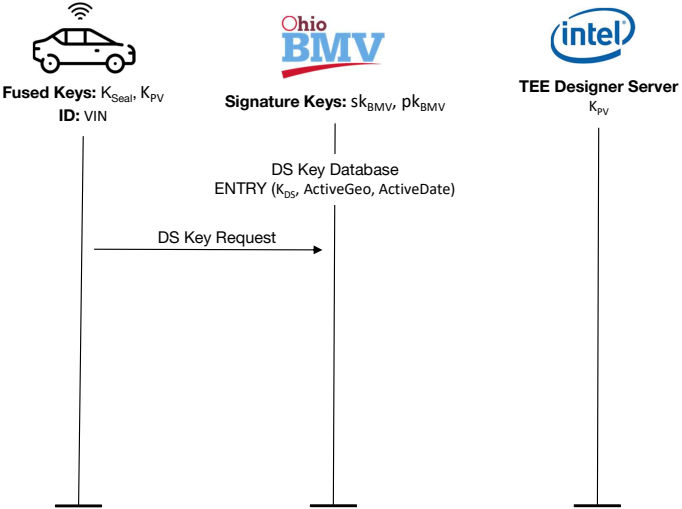
The Protocol: DS Key Renewal (On Demand)



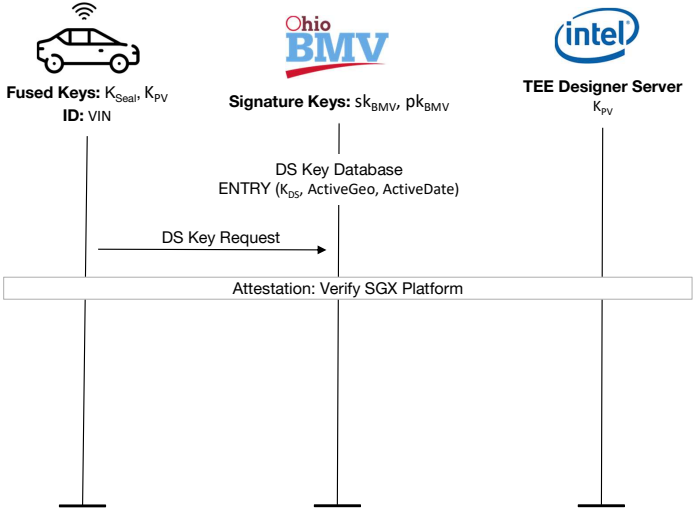
The Protocol: DS Key Renewal (On Demand)



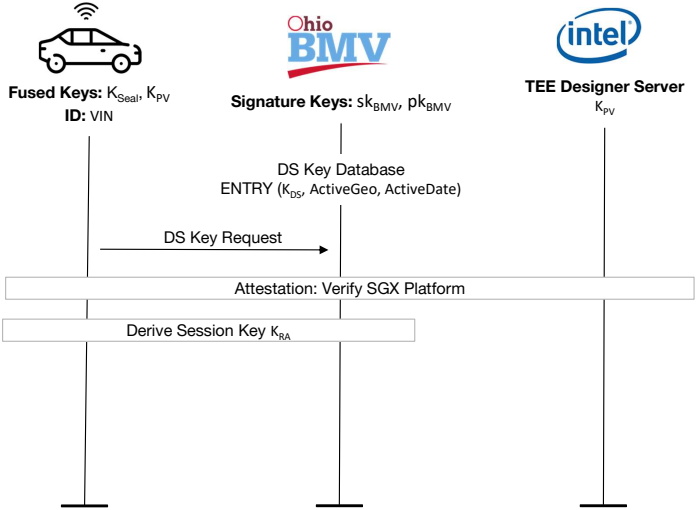
The Protocol: DS Key Renewal (On Demand)



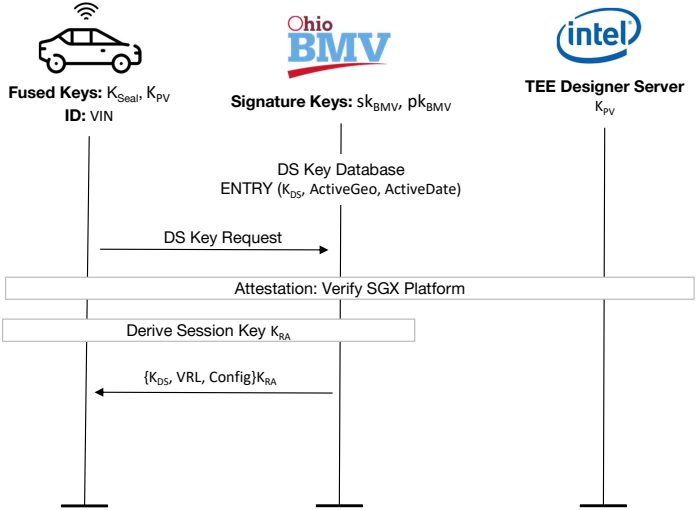
The Protocol: DS Key Renewal (On Demand)



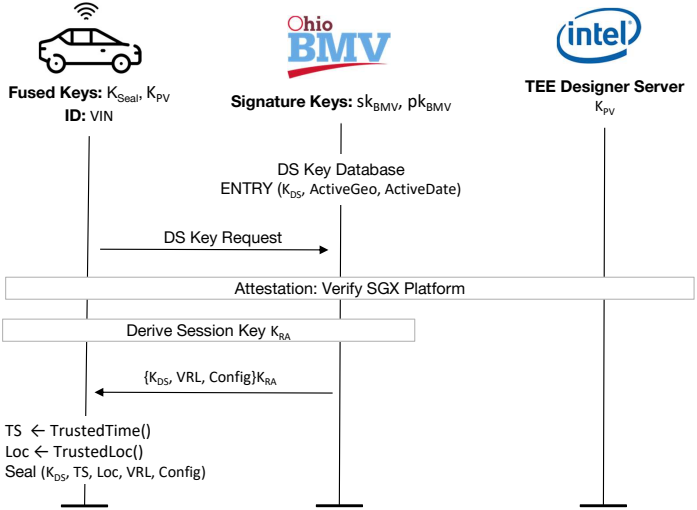
The Protocol: DS Key Renewal (On Demand)



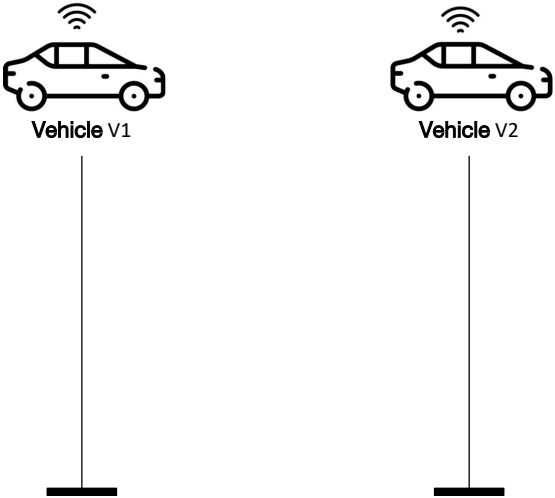
The Protocol: DS Key Renewal (On Demand)



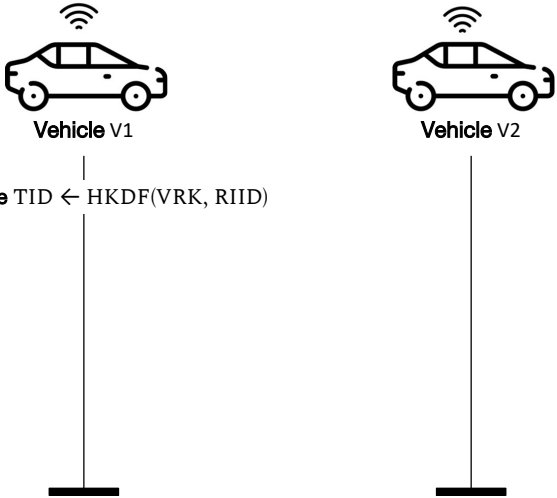
The Protocol: DS Key Renewal (On Demand)



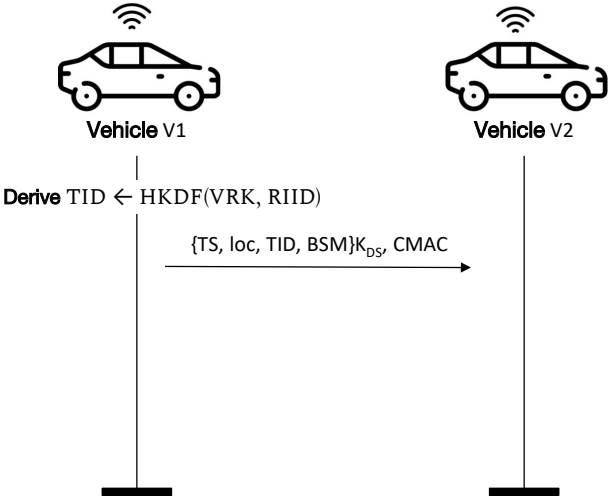
The Protocol: V2V (On Demand)



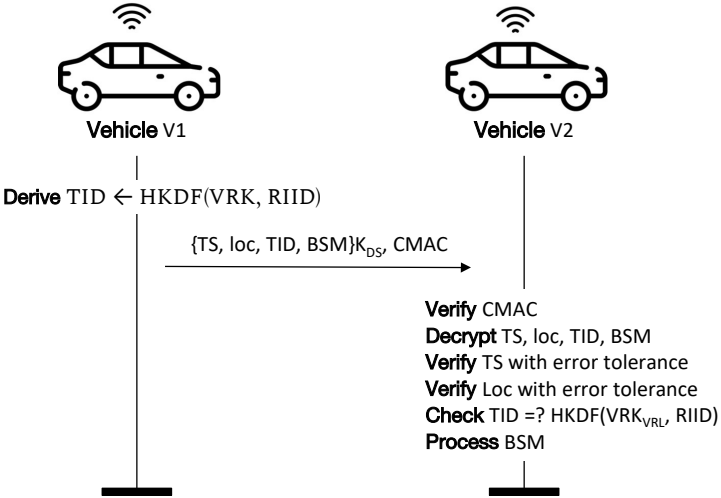
The Protocol: V2V (On Demand)



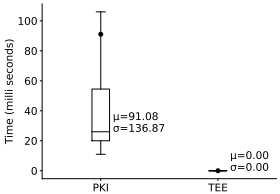
The Protocol: V2V (On Demand)



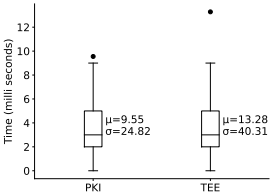
The Protocol: V2V (On Demand)



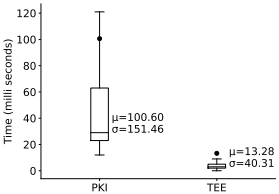
Preliminary Results: TEE vs. PKI



(a) Key Exchange Latency



(b) BSM Broadcast Latency



(c) End To End Latency

Next Step: Formal Verification of the TEE-based V2V Protocol

Modeling New Threats from TEEs

- New TCB modules in TEEs:
  1. Enclave memory (local/global variables)
  2. Non-volatile memory (monotonic counters)
  3. Persistent storage (sealed data)
- New threat model in TEEs:
  1. Controls the privileged code (e.g., OS, or hypervisors)
  2. Arbitrary thread and process instantiation
  3. Permute, reorder enclave calls
  4. Access to ecall/ocall arguments and returns
  5. Replay, modify of data in untrusted code

Next Step: Formal Verification of the TEE-based V2V Protocol

Modeling New Threats from TEEs

- New TCB modules in TEEs:
  1. Enclave memory (local/global variables)
  2. Non-volatile memory (monotonic counters)
  3. Persistent storage (sealed data)
- New threat model in TEEs:
  1. Controls the privileged code (e.g., OS, or hypervisors)
  2. Arbitrary thread and process instantiation
  3. Permute, reorder enclave calls
  4. Access to ecall/ocall arguments and returns
  5. Replay, modify of data in untrusted code

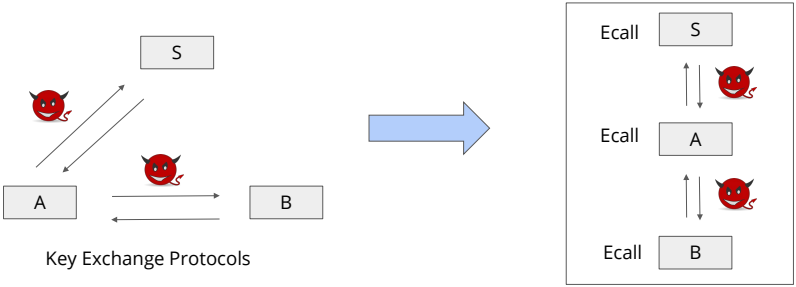
"Towards **Formal Verification** of State Continuity for **Enclave Programs**". USENIX Security 2021

Next Step: Formal Verification of the TEE-based V2V Protocol

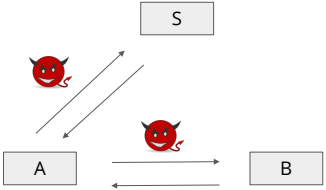
Verifying SGX-V2V Protocol (Tamarin)

1. Absence of state continuity or reentrancy vulnerability
2. Absence of non-repudiation attacks (identity of a vehicle cannot be forged)
3. Absence of replay attacks against key distribution
4. Absence of replay attacks against V2V messages
5. Consensus on the same secrete key for all the members

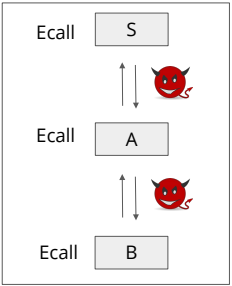
Key Intuition: Cryptographic Protocols and SGX Environment Share Similarities



Key Intuition: Cryptographic Protocols and SGX Environment Share Similarities



Key Exchange Protocols

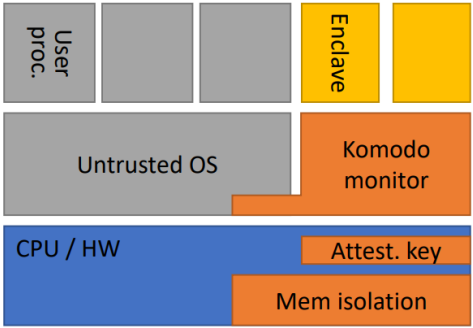


SGX Environment

Tamarin MSR and query language

State Cont. Properties
SGX Thread Model

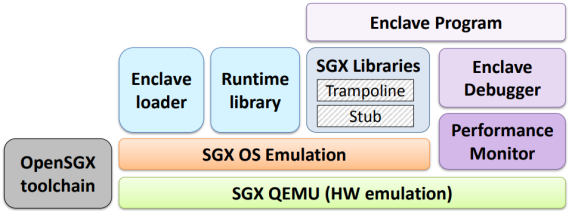
Komodo



Source: https://www.sigops.org/s/conferences/sosp/2017/slides/komodo-sosp17-slides.pdf

- **Komodo [FBHP17]:** An enclave implementation using ARM's TrustZone. It is implemented using software with formal verifications to provide support for feature evolving.
- **Comparing to ours:** Komodo has its own spec, and it does not have the binary compatibility issue with legacy SGX programs. There are aslo additional challenges due to the difference between ARM-TrustZone (already enclave-like) and SEV.

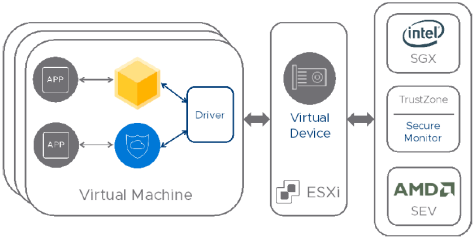
OpenSGX



Source: https://gts3.org/assets/papers/2016/jain:opensgx-slides.pdf

- OpenSGX [JDKH16] is a QEMU implementation of SGX spec without any security promise. It has no security features and not strictly compatible with Intel SGX
- It is a project when Intel has not released its SDKs, and useful for SGX programs understanding, debugging

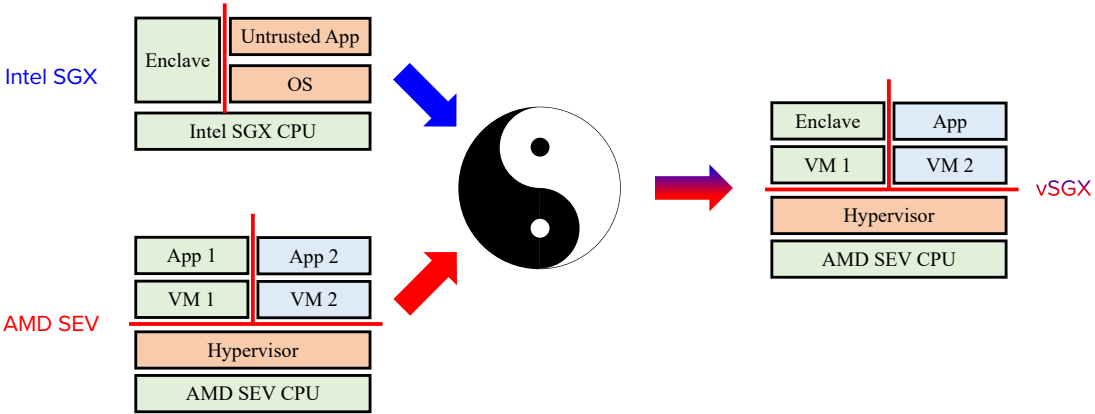
The Vmware's Unified TEE Framework for Virtualized Environments



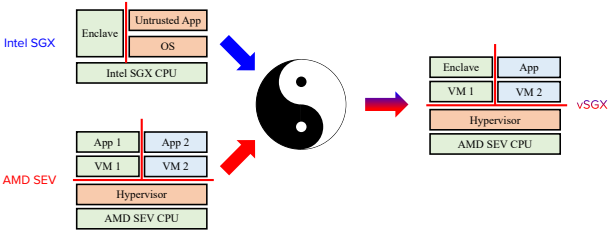
Source: https://research.vmware.com/projects/unified-tee-framework-for-virtualized-environments

- Exploring a framework for **unifying TEEs in virtual environments** [VMw]
- "The scheme is designed with **virtualization** in mind and offers capabilities that simplify the use of TEEs for guest VM environments."
- Our project is **inspired** and closely aligns with this vision, but with an emphasis on **binary compatibility**

vSGX: Virtualizing SGX Enclaves on AMD SEV [S&P22]



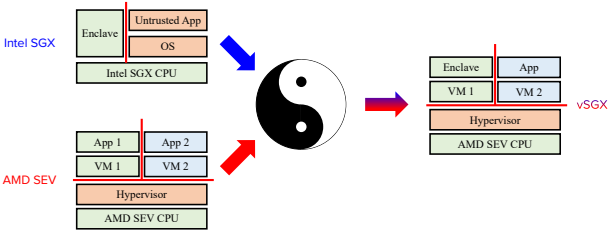
vSGX: Virtualizing SGX Enclaves on AMD SEV [S&P22]



1. Binary compatibility
2. Comparable security guarantee with both SGX and SEV
3. Reasonable performance

vSGX works like an SGX module plugged onto an SEV machine

vSGX: Virtualizing SGX Enclaves on AMD SEV [S&P22]



1. Binary compatibility
2. Comparable security guarantee with both SGX and SEV
3. Reasonable performance

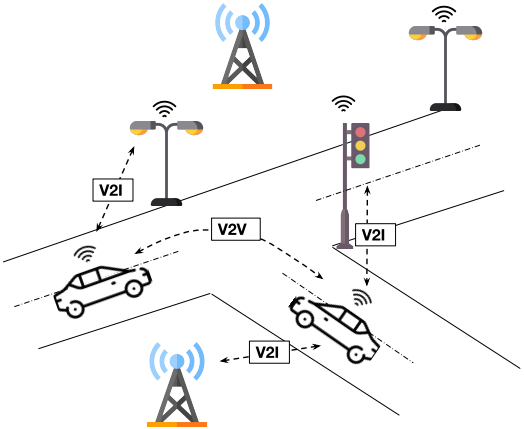
vSGX works like an SGX module plugged onto an SEV machine

The source code has been made available at https://github.com/OSUSecLab/vSGX

A TEE-based V2V Protocol [AutoSec'22]

Objectives

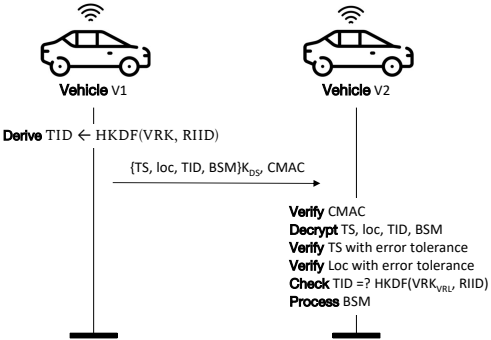
1. **(Instant) Authentication**
2. Message Integrity (Tamper Resiliency)
3. Message Confidenality (Privacy)
4. **Anti-tracking** (Privacy)
5. Revocation and **Accountability**
6. Scalability



A TEE-based V2V Protocol [AutoSec'22]

Objectives

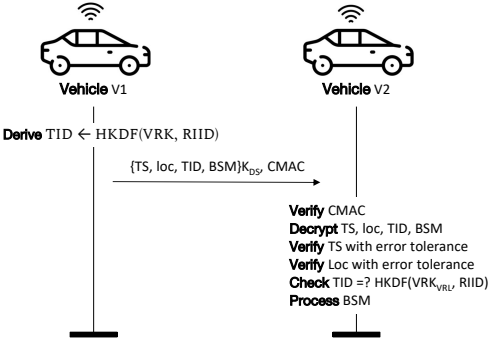
1. **(Instant) Authentication**
2. Message Integrity (Tamper Resiliency)
3. Message Confidenality (Privacy)
4. **Anti-tracking** (Privacy)
5. Revocation and **Accountability**
6. Scalability



A TEE-based V2V Protocol [AutoSec'22]

Our Approach

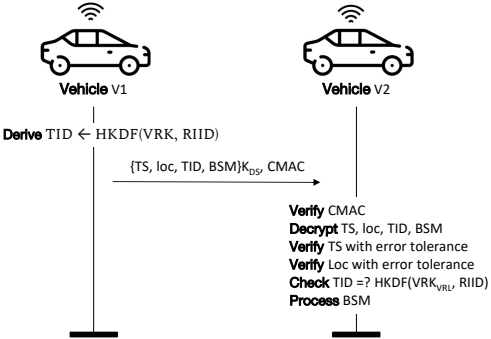
- Leveraging TEEs to protect secrets
- Using a shared Daily Symmetric (DS) Key among vehicles
- Storing DS key in the enclave
- Using Remote Attestation to ensure the right environment
- Generating temporal IDs for privacy



A TEE-based V2V Protocol [AutoSec'22]

Our Approach

- Leveraging TEEs to protect secrets
- Using a shared Daily Symmetric (DS) Key among vehicles
- Storing DS key in the enclave
- Using Remote Attestation to ensure the right environment
- Generating temporal IDs for privacy



https://github.com/OSUSecLab/tee-v2v

Thank You

Towards Distributed and Virtualized Trusted Execution Environments

Zhiqiang Lin
Distinguished Professor of Engineering
zlin@cse.ohio-state.edu

12/06/2022

References I

Ferdinand Brasser, Urs Müller, Alexandra Dmitrienko, Kari Kostiainen, Srdjan Capkun, and Ahmad-Reza Sadeghi, *Software grand exposure: SGX cache attacks are practical*, 11th USENIX Workshop on Offensive Technologies (Vancouver, BC), USENIX Association, August 2017.

Guoxing Chen, Sanchuan Chen, Yuan Xiao, Yinqian Zhang, Zhiqiang Lin, and Ten H. Lai, *SGXPECTRE: Stealing Intel Secrets from SGX Enclaves via Speculative Execution*, 4th IEEE European Symposium on Security and Privacy, IEEE Computer Society, 2019.

Claudio Canella, Daniel Genkin, Lukas Giner, Daniel Gruss, Moritz Lipp, Marina Minkin, Daniel Moghimi, Frank Piessens, Michael Schwarz, Berk Sunar, et al., *Fallout: Leaking data on meltdown-resistant cpus*, ACM SIGSAC Conference on Computer and Communications Security, 2019, pp. 769–784.

Chengjun Cai, Lei Xu, Anxin Zhou, and Cong Wang, *Toward a secure, rich, and fair query service for light clients on public blockchains*, IEEE Transactions on Dependable and Secure Computing (2021), 1–1.

enigmampc/SafeTrace: Privacy preserving voluntary COVID-19 self-reporting platform., `https://github.com/enigmampc/SafeTrace`, (Accessed on 08/05/2020).

Andrew Ferraiuolo, Andrew Baumann, Chris Hawblitzel, and Bryan Parno, *Komodo: Using verification to disentangle secure-enclave hardware from software*, Proceedings of the ACM Symposium on Operating Systems Principles, October 2017.

Johannes Götzfried, Moritz Eckert, Sebastian Schinzel, and Tilo Müller, *Cache attacks on Intel SGX*, Proceedings of the 10th European Workshop on Systems Security, 2017.

Google, *Github - google/asylo: An open and flexible framework for developing enclave applications*, `https://github.com/google/asylo`, (Accessed on 11/17/2021).

References II

Marcus Hähnel, Weidong Cui, and Marcus Peinado, *High-resolution side channels for untrusted operating systems*, USENIX Annual Technical Conference, 2017.

Prerit Jain, Soham Jayesh Desai, Brent Byunghoon Kang, and Dongsu Han, *OpenSGX: An Open Platform for SGX Research.*, Proceedings of the Network and Distributed System Security Symposium, 2016.

Seongmin Kim, Juhyeng Han, Jaehyeong Ha, Taesoo Kim, and Dongsu Han, *Enhancing security and privacy of tor's ecosystem by using trusted execution environments*, 14th USENIX Symposium on Networked Systems Design and Implementation, 2017, pp. 145–161.

Klaudia Krawiecka, Arseny Kurnikov, Andrew Paverd, Mohammad Mannan, and N Asokan, *Safekeeper: Protecting web passwords using trusted execution environments*, Proceedings of the 2018 World Wide Web Conference, 2018, pp. 349–358.

Kubilay Ahmet Küçük, Andrew Paverd, Andrew Martin, N Asokan, Andrew Simpson, and Robin Ankele, *Exploring the use of intel sgx for secure many-party applications*, Proceedings of the 1st Workshop on System Software for Trusted Execution, ACM, 2016, p. 5.

Sangho Lee, Ming-Wei Shih, Prasun Gera, Taesoo Kim, Hyesoon Kim, and Marcus Peinado, *Inferring fine-grained control flow inside SGX enclaves with branch shadowing*, 26th USENIX Security Symposium, 2017.

Olga Ohrimenko, Felix Schuster, Cédric Fournet, Aastha Mehta, Sebastian Nowozin, Kapil Vaswani, and Manuel Costa, *Oblivious multi-party machine learning on trusted processors*, 25th USENIX Security Symposium (USENIX Security 16) (Austin, TX), USENIX Association, 2016, pp. 619–636.

Seonghyun Park, Adil Ahmad, and Byoungyoung Lee, *Blackmirror: Preventing wallhacks in 3d online fps games*, Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security, 2020, pp. 987–1000.

References III

Felix Schuster, Manuel Costa, Cedric Fournet, Christos Gkantsidis, Marcus Peinado, Gloria Mainar-Ruiz, and Mark Russinovich, *VC3: Trustworthy Data Analytics in the Cloud using SGX*, 2015 IEEE Symposium on Security and Privacy, IEEE, 2015, pp. 38–54.

Shweta Shinde, Zheng Leong Chua, Viswesh Narayanan, and Prateek Saxena, *Preventing page faults from telling your secrets*, 11th ACM on Asia Conference on Computer and Communications Security, 2016.

Michael Schwarz, Moritz Lipp, Daniel Moghimi, Jo Van Bulck, Julian Stecklina, Thomas Prescher, and Daniel Gruss, *Zombieload: Cross-privilege-boundary data sampling*, arXiv preprint arXiv:1905.05726 (2019).

Michael Schwarz, Samuel Weiser, Daniel Gruss, Clémentine Maurice, and Stefan Mangard, *Malware guard extension: Using sgx to conceal cache attacks*, Springer International Publishing, 2017.

Jo Van Bulck, Marina Minkin, Ofir Weisse, Daniel Genkin, Baris Kasikci, Frank Piessens, Mark Silberstein, Thomas F Wenisch, Yuval Yarom, and Raoul Strackx, *Foreshadow: Extracting the keys to the intel SGX kingdom with transient out-of-order execution*, 27th USENIX Security Symposium, 2018, pp. 991–1008.

Vmware research — unified tee framework for virtualized environments, `https://research.vmware.com/projects/unified-tee-framework-for-virtualized-environments`, (Accessed on 11/17/2021).

Stephan van Schaik, Alyssa Milburn, Sebastian Österlund, Pietro Frigo, Giorgi Maisuradze, Kaveh Razavi, Herbert Bos, and Cristiano Giuffrida, *RIDL: Rogue in-flight data load*, IEEE Symposium on Security and Privacy (2019).

Yuanzhong Xu, Weidong Cui, and Marcus Peinado, *Controlled-channel attacks: Deterministic side channels for untrusted operating systems*, IEEE Symposium on Security and Privacy, IEEE, 2015.