Clemmys

Towards Secure Remote Execution in FaaS

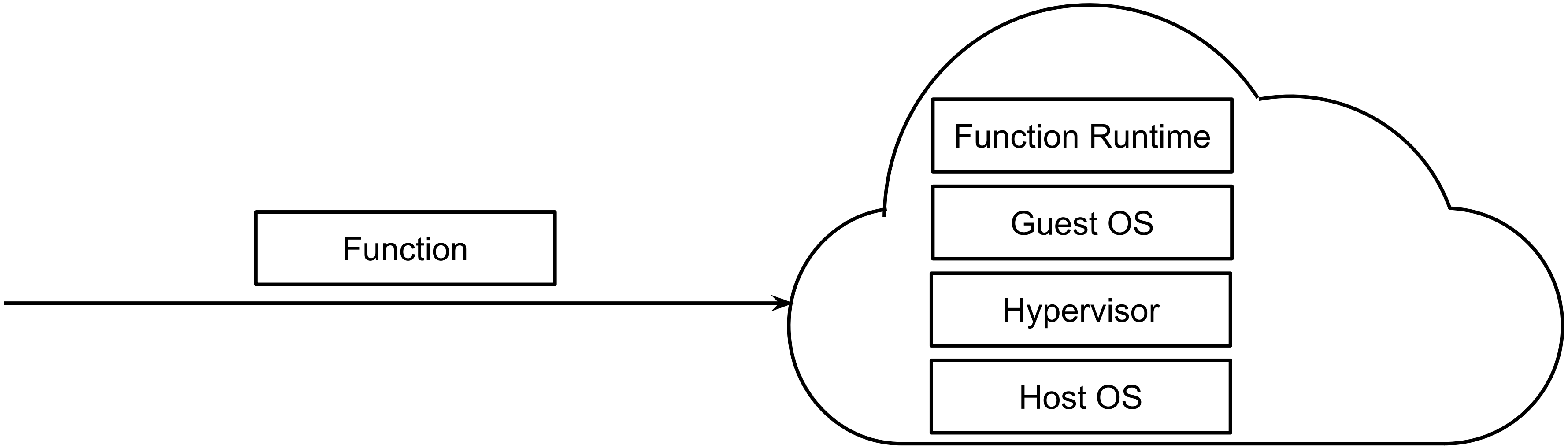
Bohdan Trach, Oleksii Oleksenko, Franz Gregor,
Pramod Bhatotia, Christof Fetzer

TECHNISCHE UNIVERSITÄT DRESDEN

THE UNIVERSITY of EDINBURGH

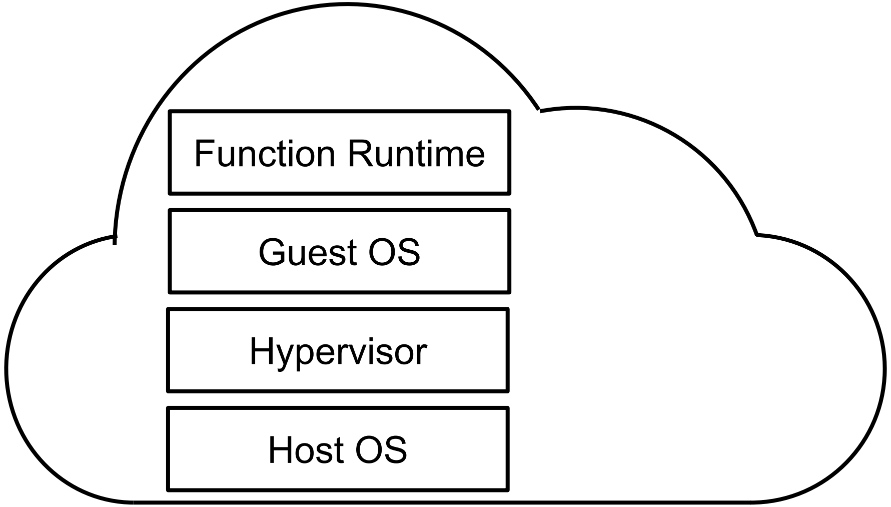
ACM SYSTOR 2019

FaaS Paradigm of Cloud Computing



FaaS Paradigm of Cloud Computing

- Less boilerplate work ☺
- Easy autoscaling ☺



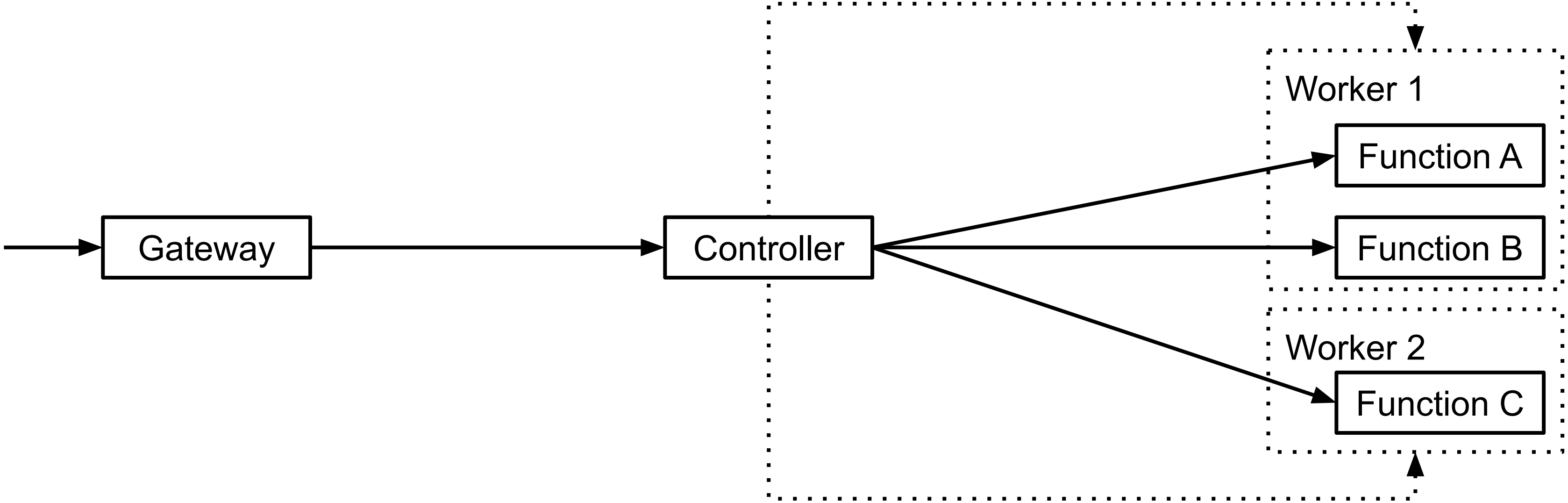
Function Runtime

Guest OS

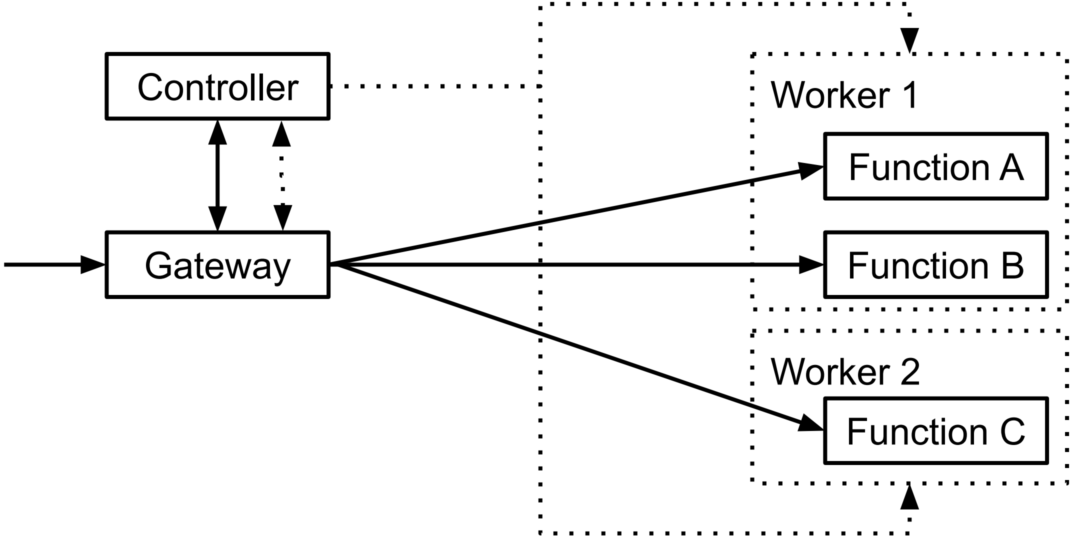
Hypervisor

Host OS

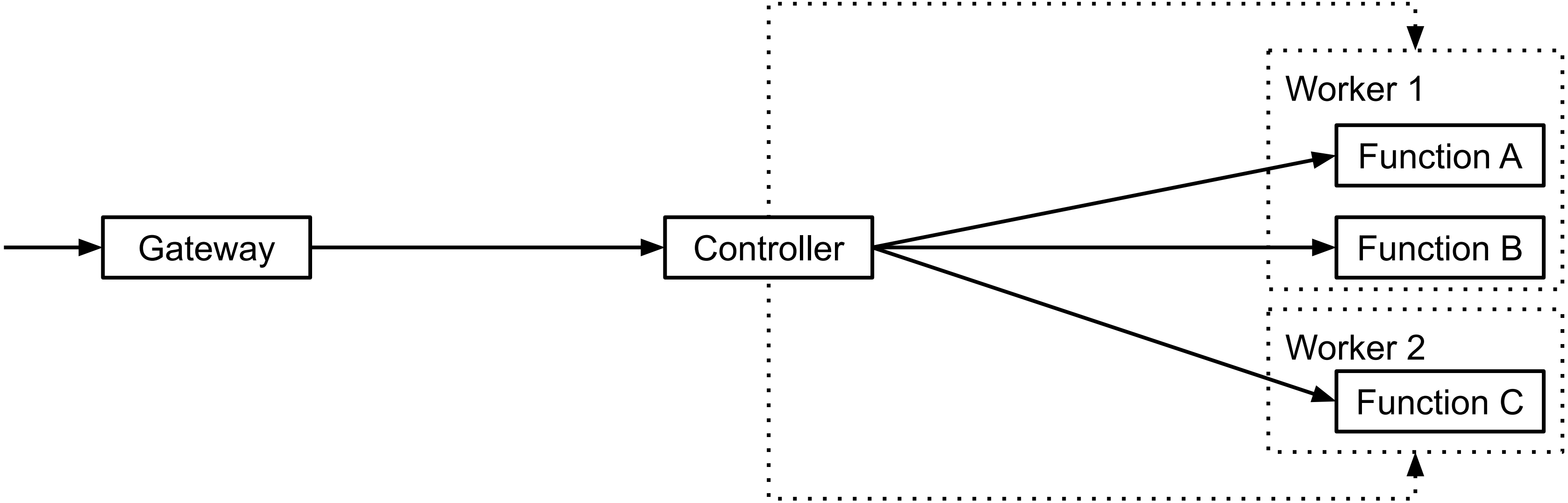
How does FaaS work?



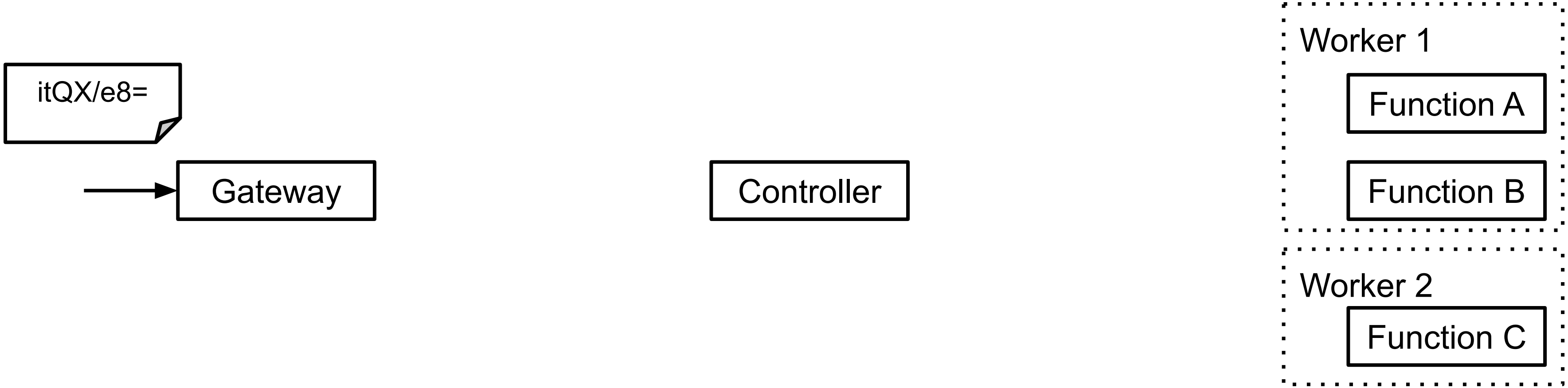
How does FaaS work?



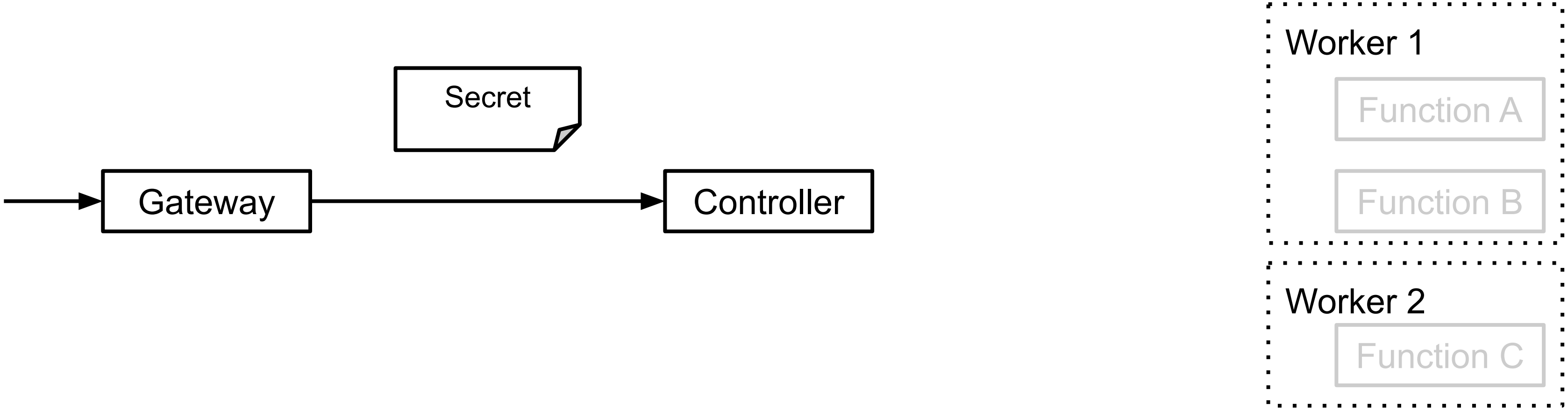
How does FaaS work?



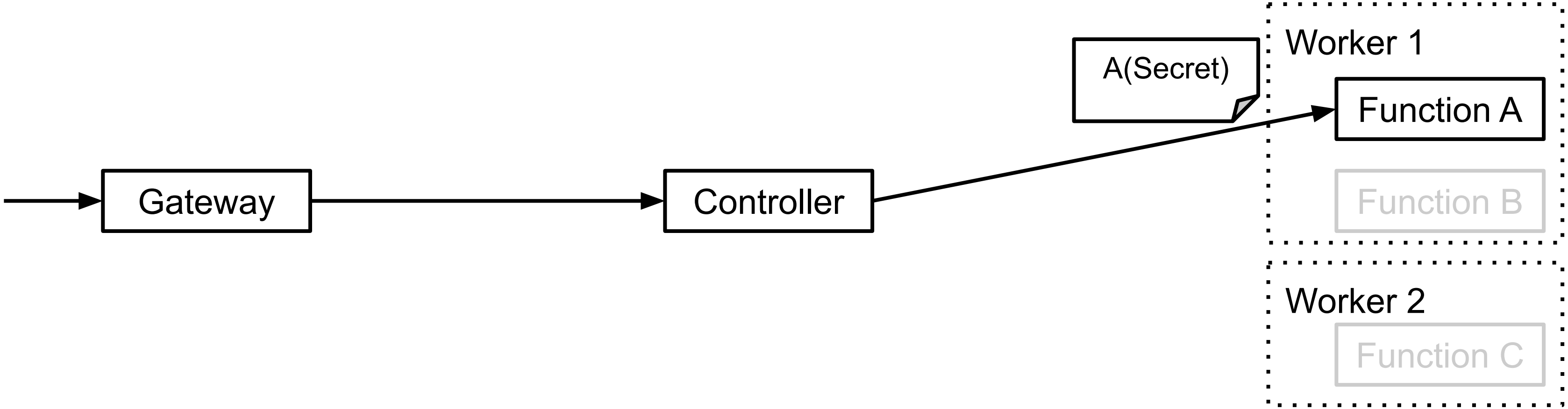
How does FaaS work?



How does FaaS work?

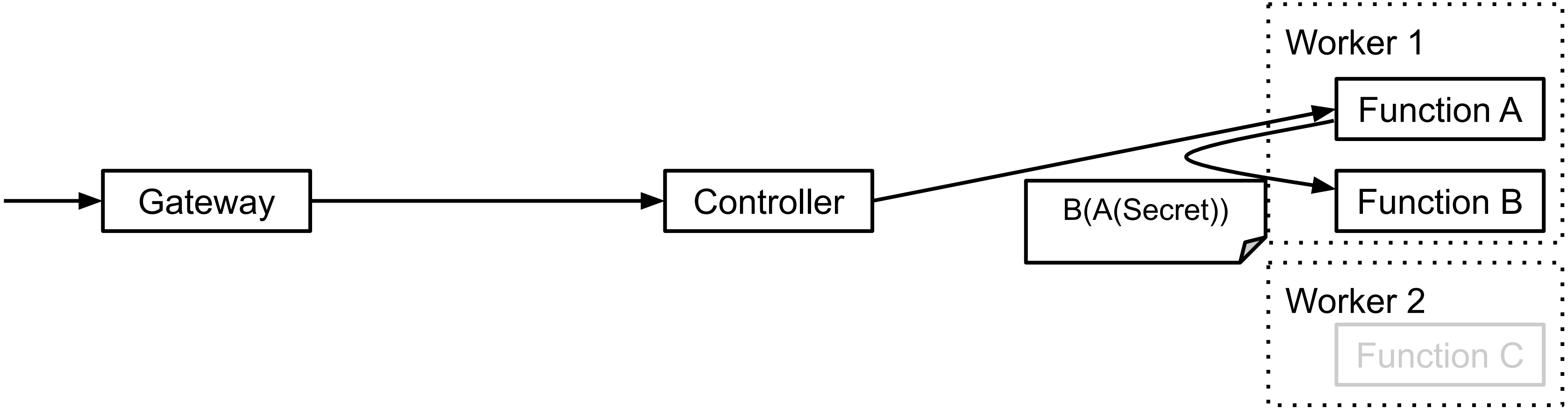


How does FaaS work?



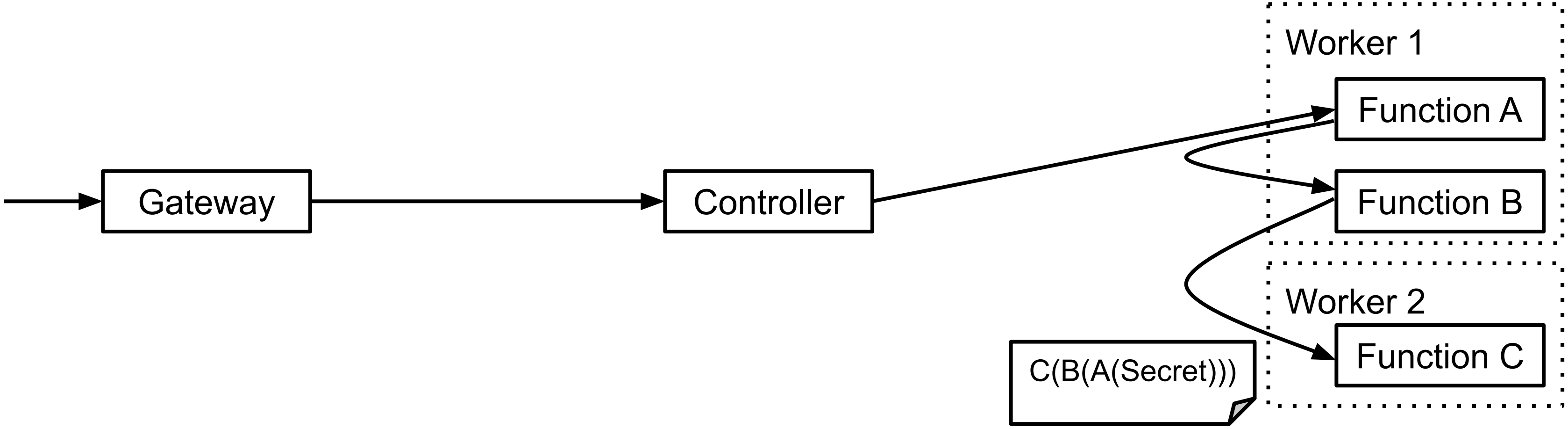
Support for ***function chaining*** is an important requirement for serverless computing

How does FaaS work?



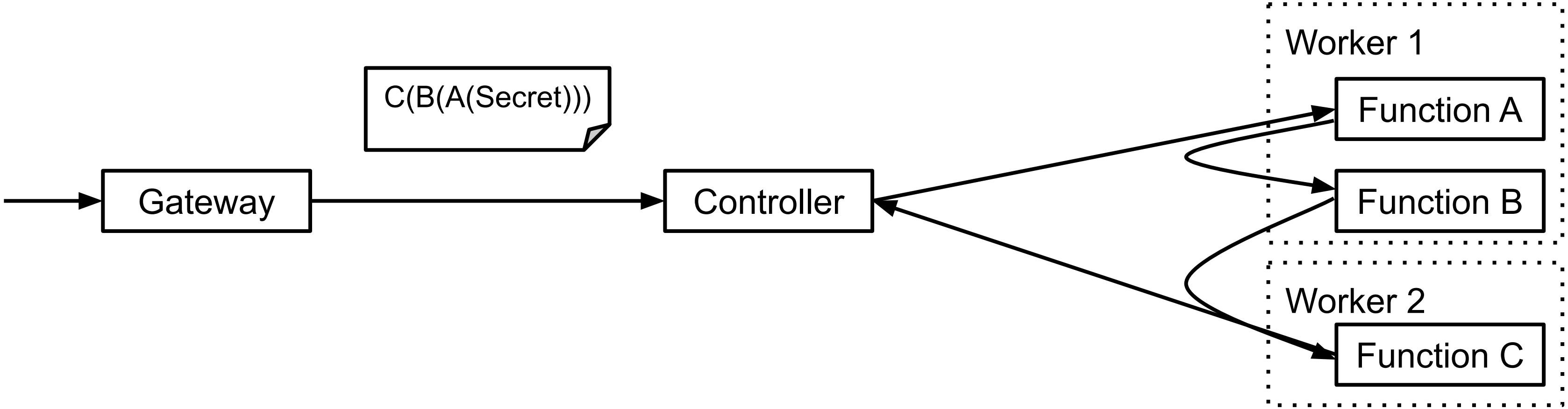
Support for ***function chaining*** is an important requirement for serverless computing

How does FaaS work?

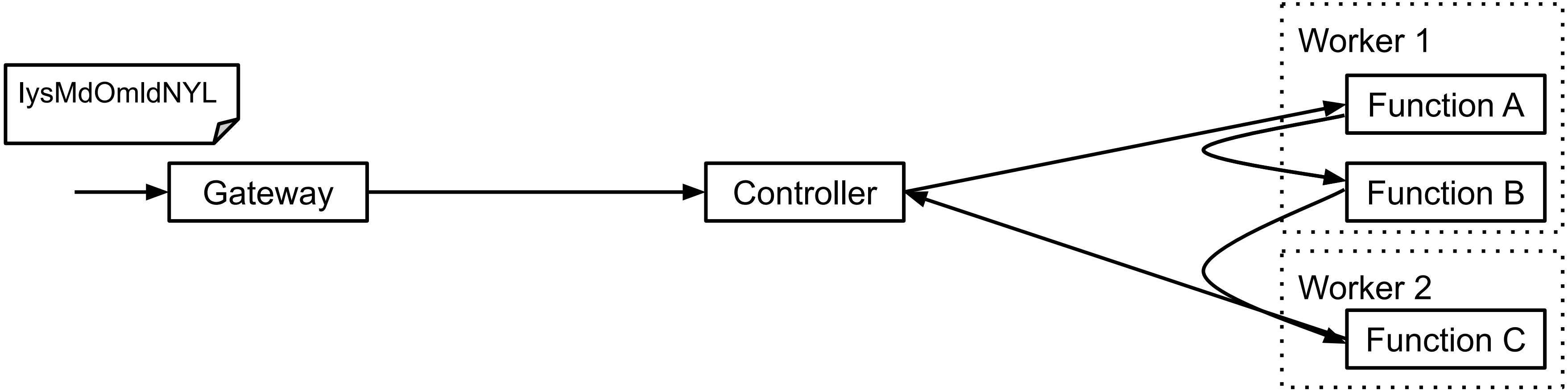


Support for ***function chaining*** is an important requirement for serverless computing

How does FaaS work?



How does FaaS work?



IysMdOmIdNYL

Gateway

Controller

Worker 1

Function A

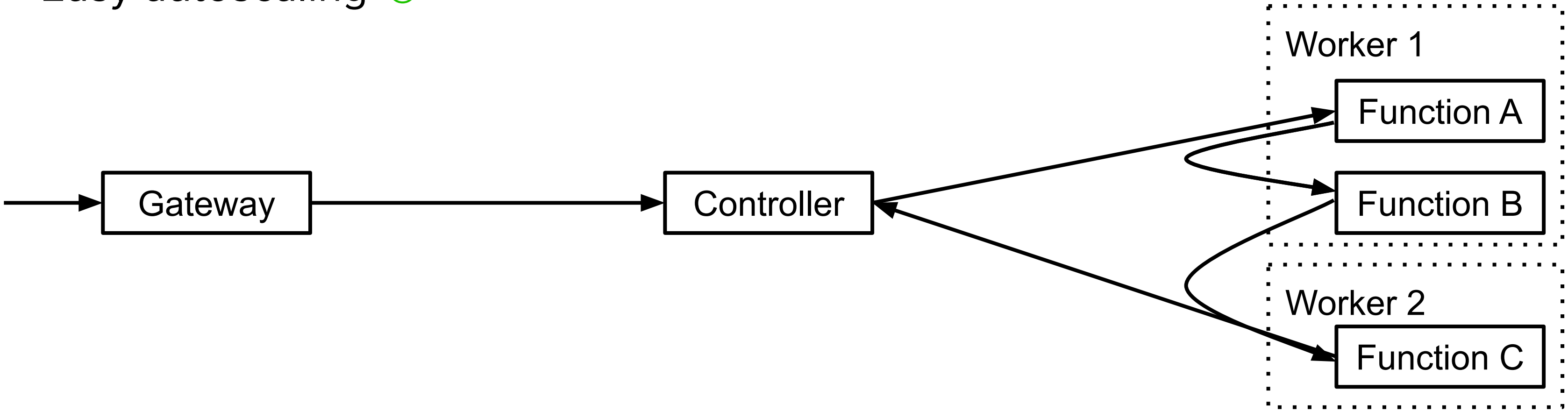
Function B

Worker 2

Function C

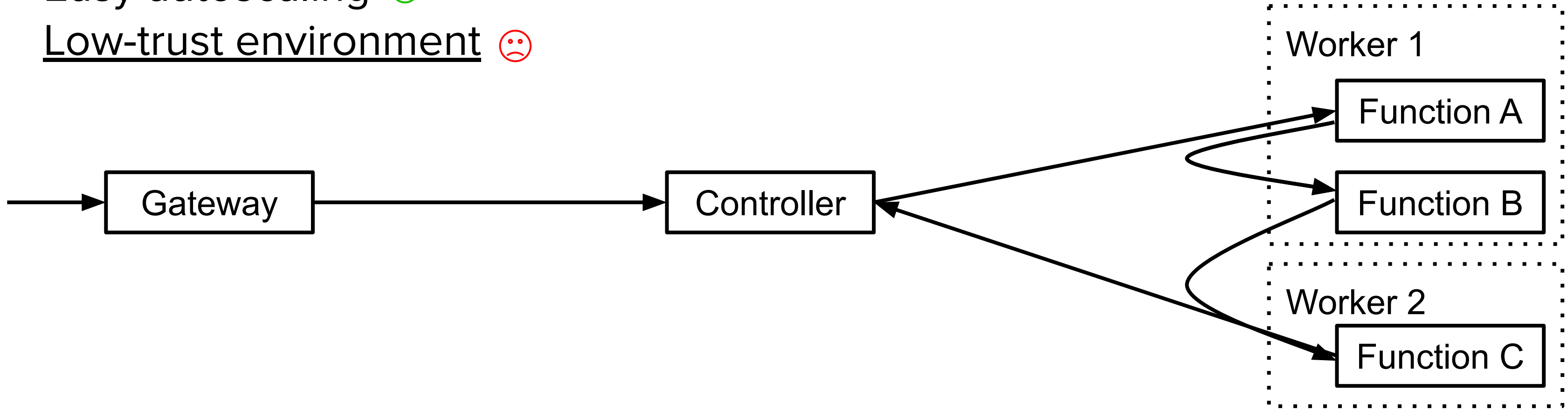
Is Faas secure?

- Less boilerplate work ☺
- Easy autoscaling ☺

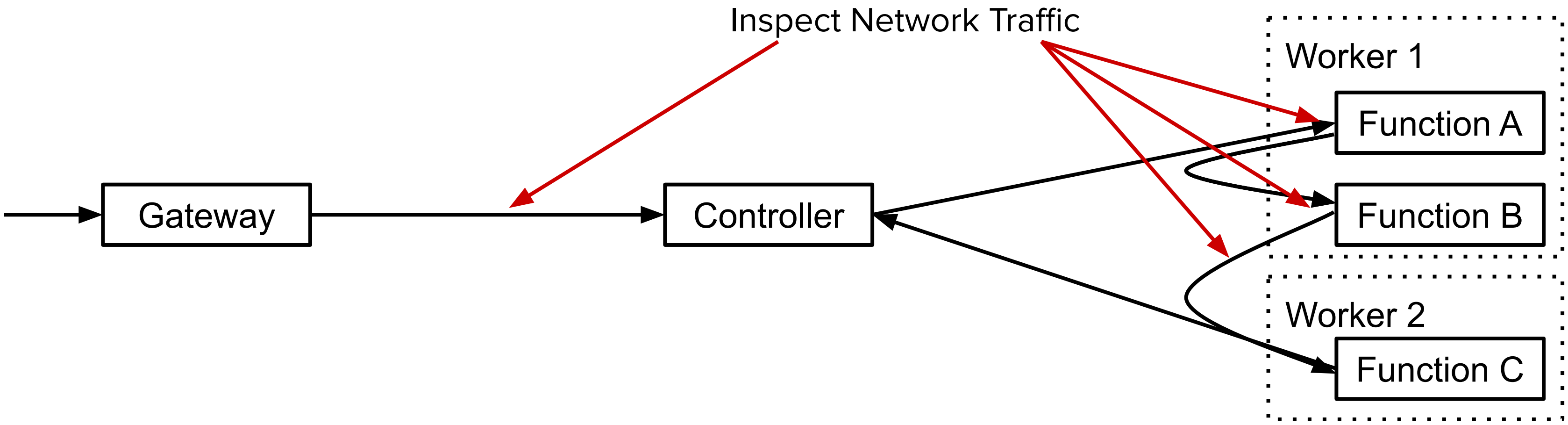


Is Faas secure?

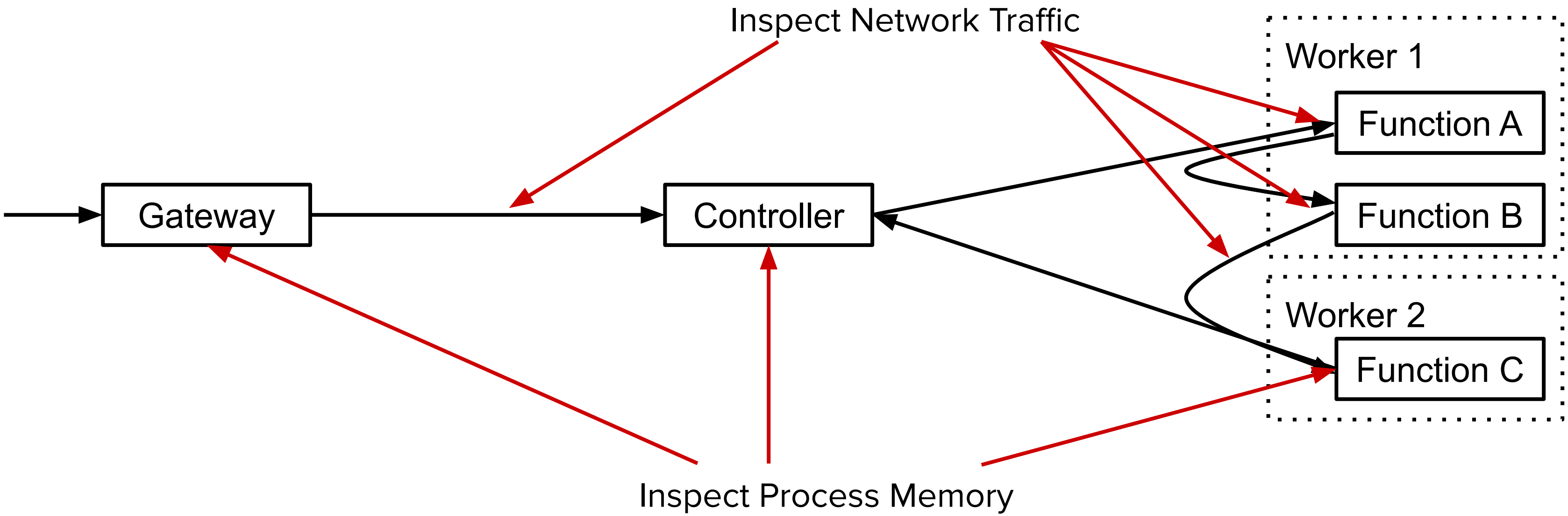
- Less boilerplate work ☺
- Easy autoscaling ☺
- Low-trust environment ☹



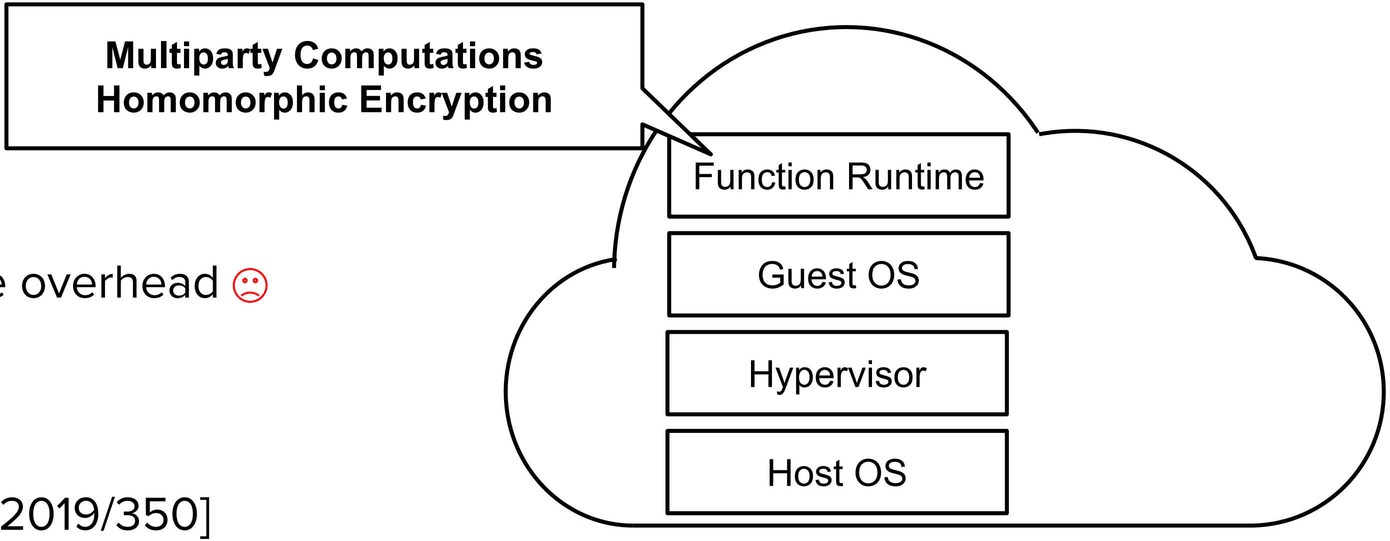
Why is FaaS insecure?



Why is FaaS insecure?



State-of-the-Art: Computing on Untrusted Systems



Multiparty Computations
Homomorphic Encryption

Function Runtime

Guest OS

Hypervisor

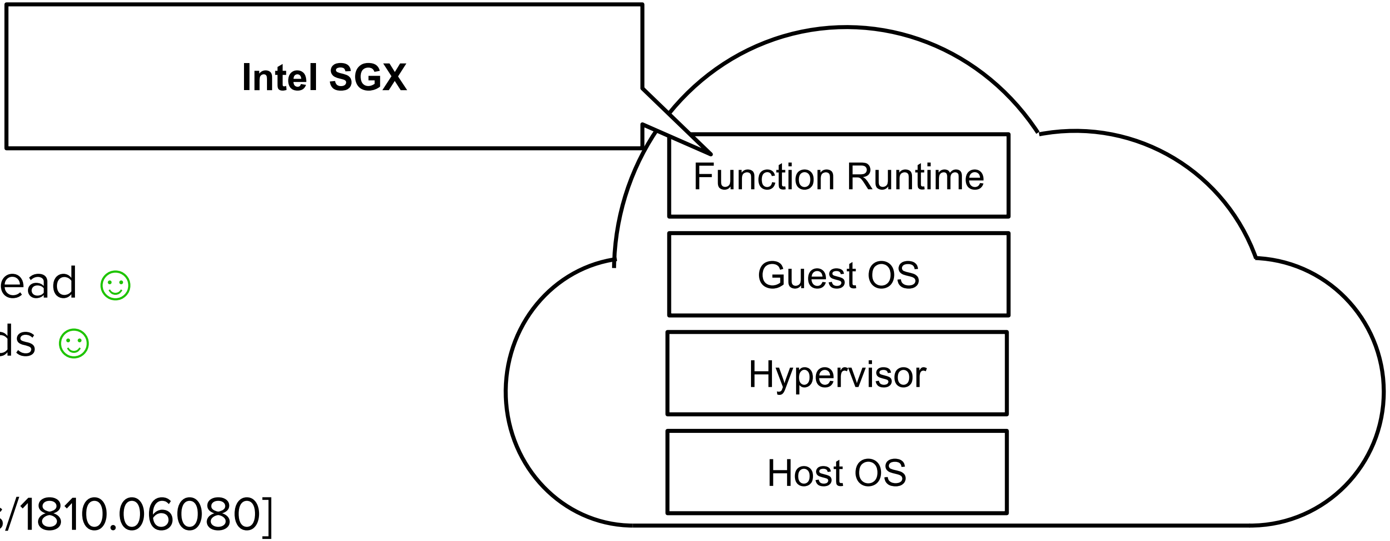
Host OS

- High performance overhead ☹
- Low flexibility ☹

Related Work:

- nGraph-HE [IACR 2019/350]
- PySyft

State-of-the-Art: Computing on Untrusted Systems

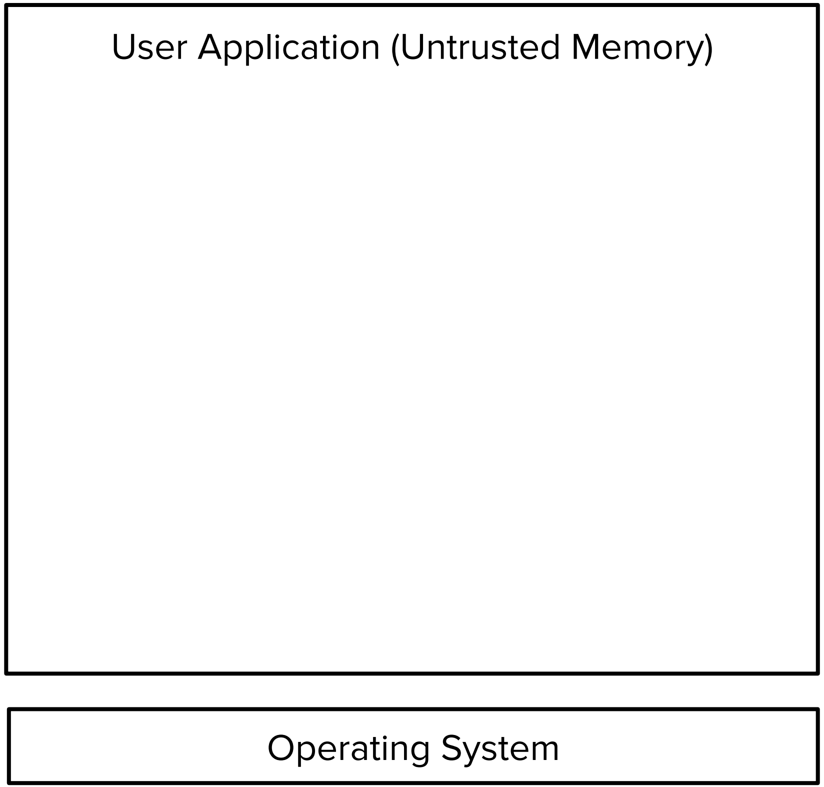


- Acceptable overhead ☺
- Arbitrary workloads ☺

Related Work:

- S-FaaS [CoRR abs/1810.06080]

What is Intel SGX?

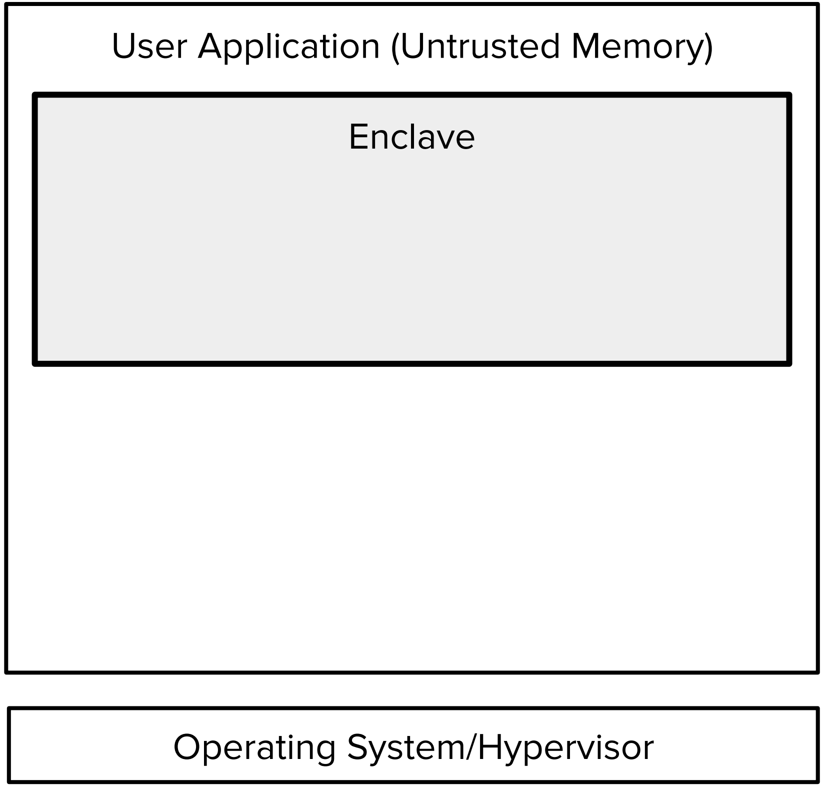


User Application (Untrusted Memory)

Operating System

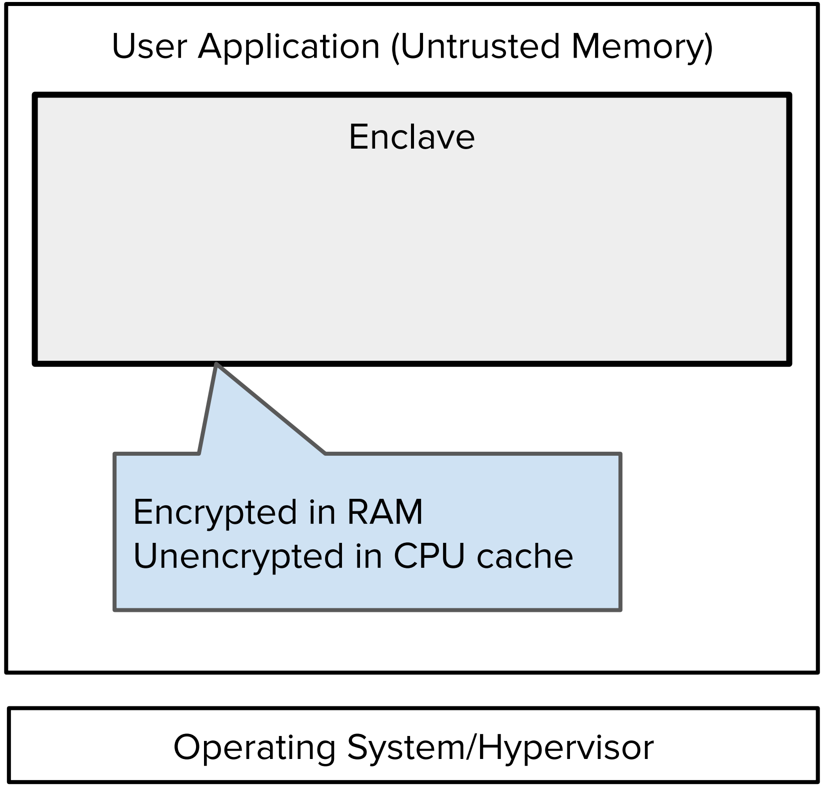
What is Intel SGX?

- Adds *enclave* abstraction


User Application (Untrusted Memory)
Enclave
Operating System/Hypervisor

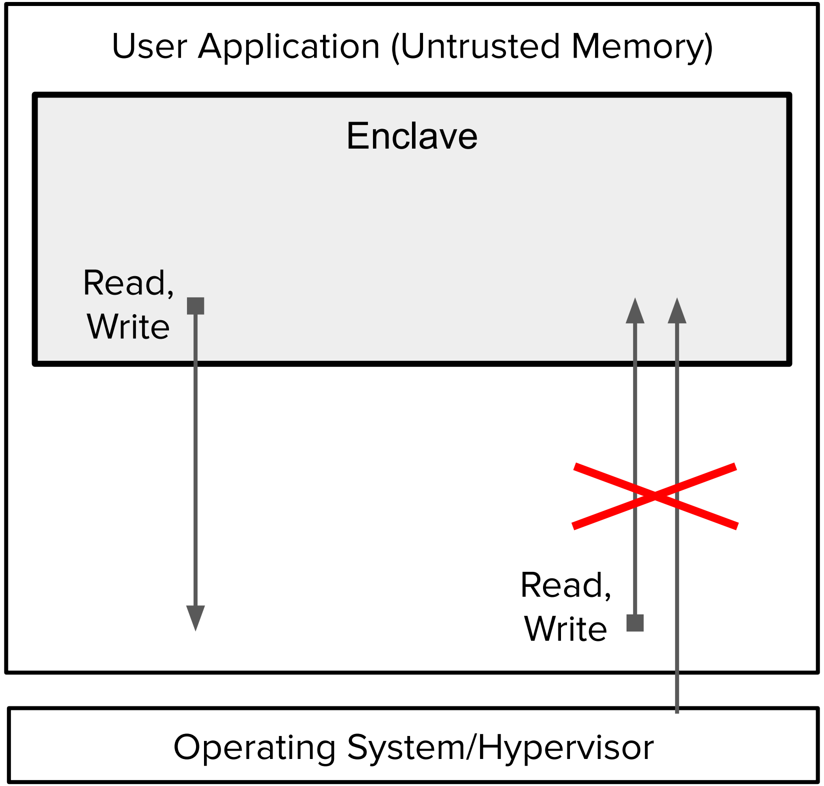
What is Intel SGX?

- Adds *enclave* abstraction
    - Encrypted in RAM only


User Application (Untrusted Memory)
Enclave
Encrypted in RAM
Unencrypted in CPU cache
Operating System/Hypervisor

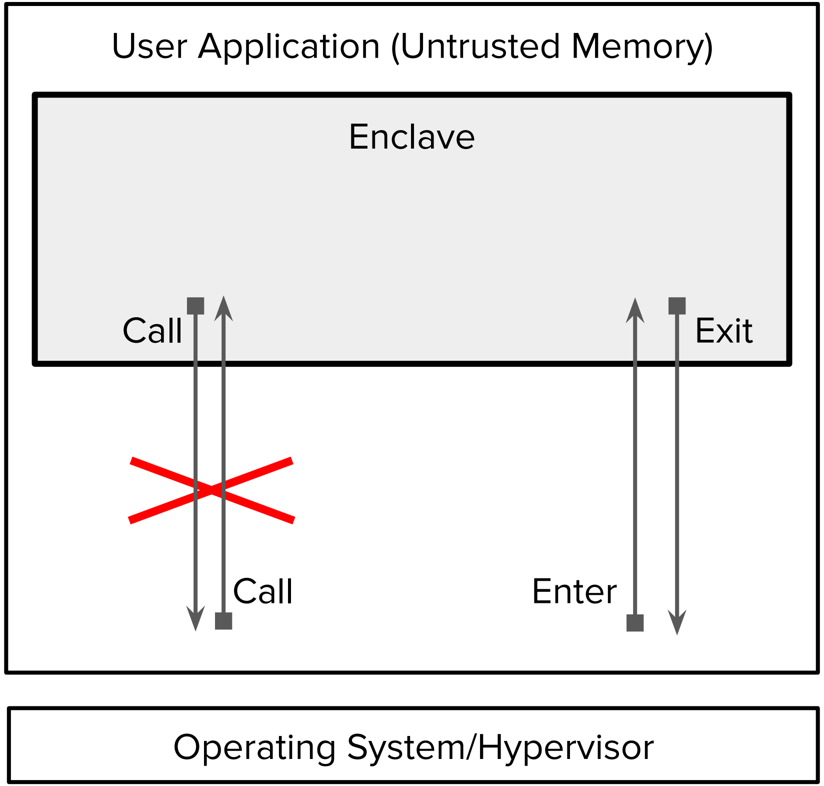
What is Intel SGX?

- Adds *enclave* abstraction
  - Encrypted in RAM only
  - Not accessible from outside


User Application (Untrusted Memory)
Enclave
Read, Write
Read, Write
Operating System/Hypervisor

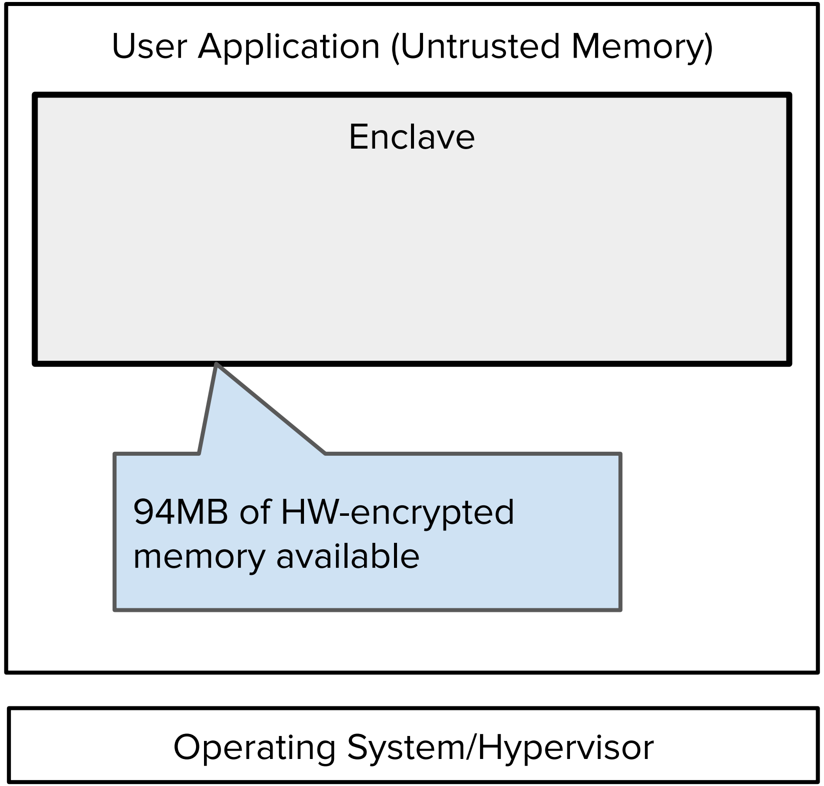
What is Intel SGX?

- Adds *enclave* abstraction
  - Encrypted in RAM only
  - Not accessible from outside
  - Developer-specified entry points


User Application (Untrusted Memory)
Enclave
Call
Exit
Call
Enter
Operating System/Hypervisor

What are the limitations of Intel SGX?

- High overheads for:
  - Secure memory paging
  - Enclave startup with large heap


User Application (Untrusted Memory)
Enclave
94MB of HW-encrypted memory available
Operating System/Hypervisor

Why do Intel SGX limitations matter?

Function startup time as an optimization target:

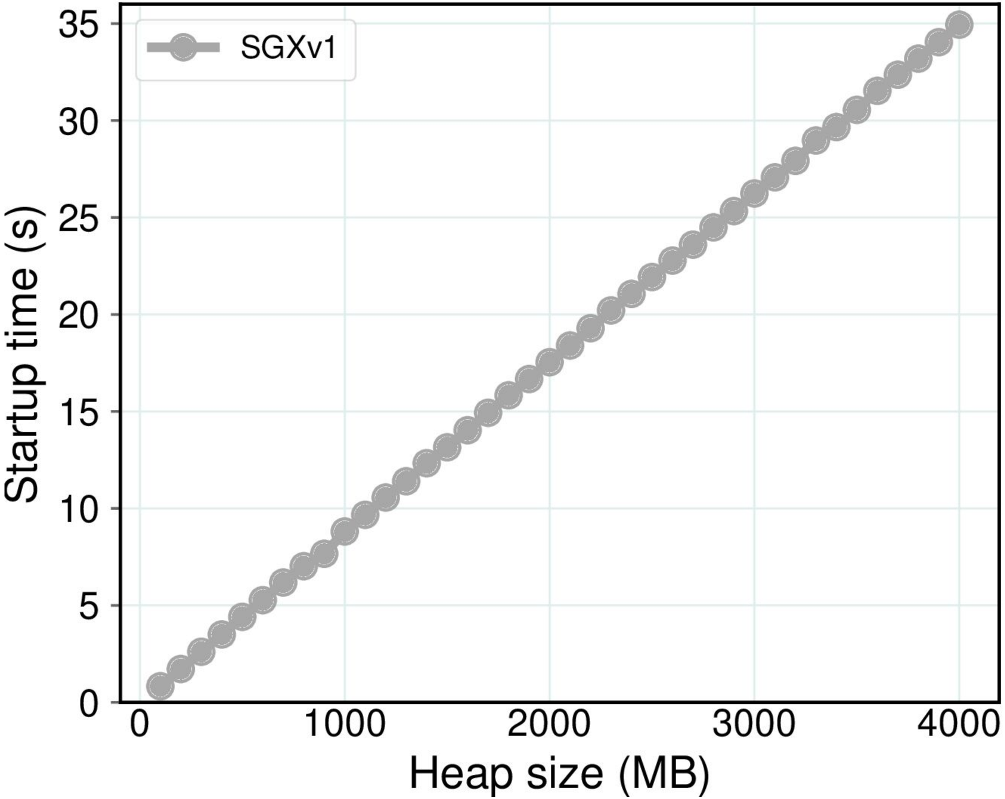
- SAND, SOCK [ATC’18]

Why do Intel SGX limitations matter?

Function startup time as an optimization target:

- SAND, SOCK [ATC'18]

Problem for SGXv1 enclaves



Why do Intel SGX limitations matter?

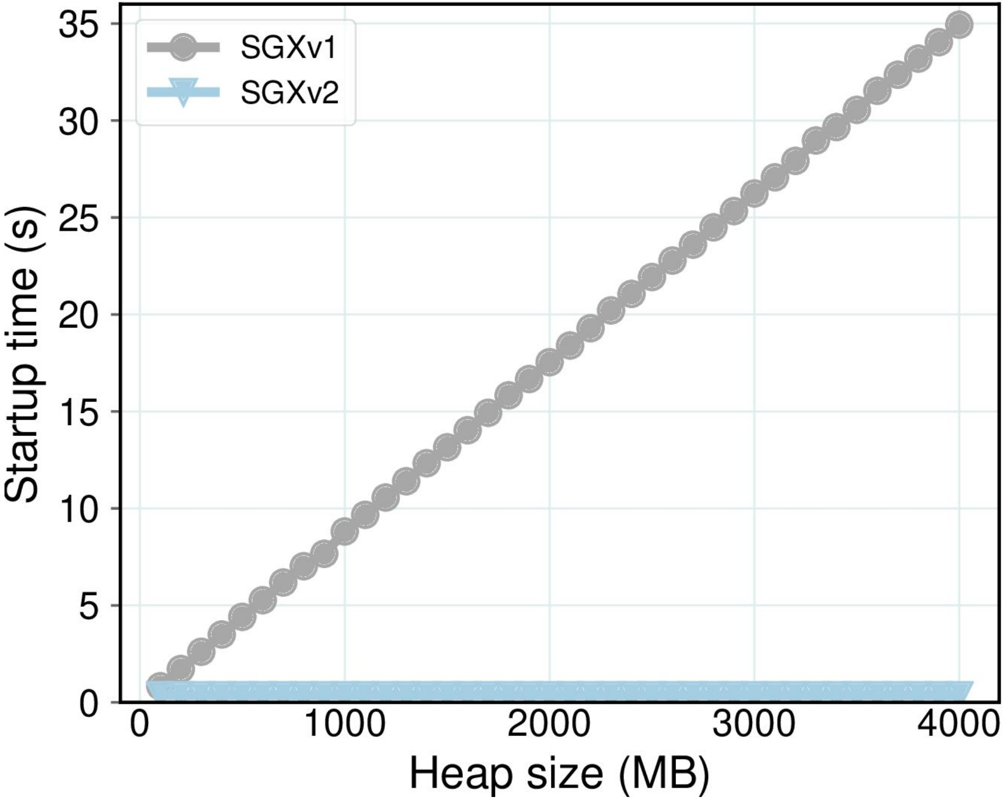
Function startup time as an optimization target:

- SAND, SOCK [ATC'18]

Problem for SGXv1 enclaves

- Can be solved with SGXv2

Additional optimizations are worth investigating.



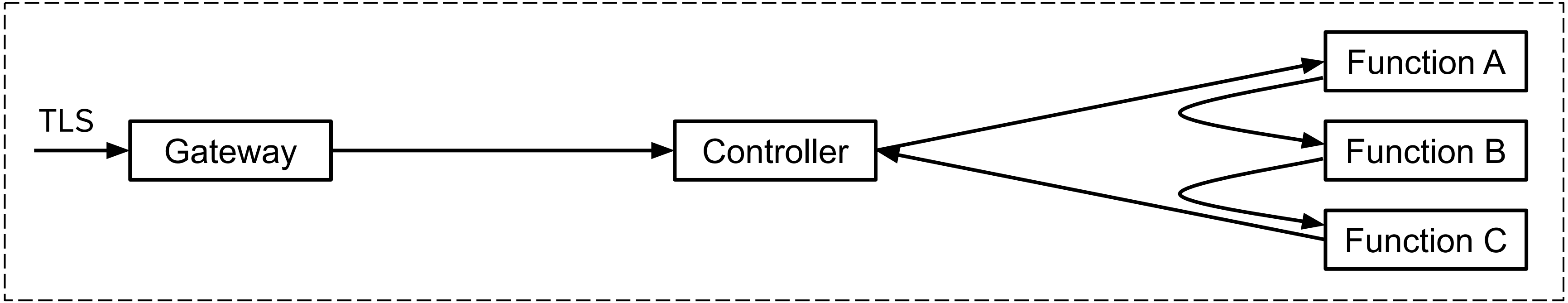
Problem Statement

How to execute a **wide range** of user functions in FaaS in a **trustworthy** and **efficient** manner?

Outline

- ~~Motivation~~
- Design
- Evaluation
- Summary

What is Clemmys?



TLS

Gateway

Controller

Function A

Function B

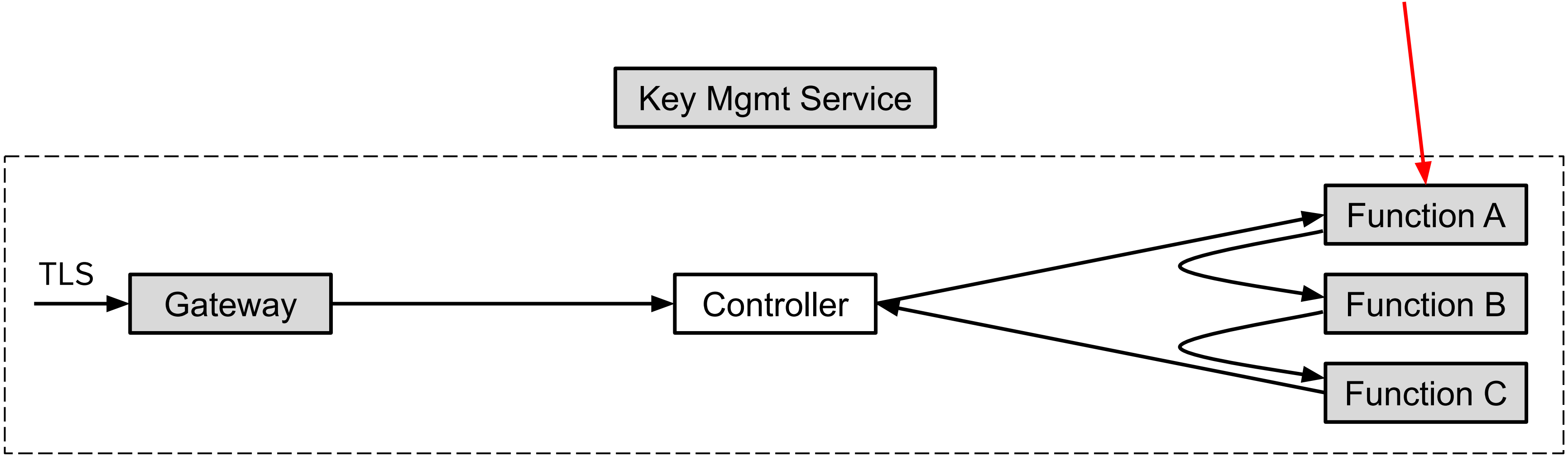
Function C

Based on Apache OpenWhisk

▨ SGX Enclave □ Native Application

What is Clemmys?

1. Trustworthy environment for function execution



Key Mgmt Service

TLS

Gateway

Controller

Function A

Function B

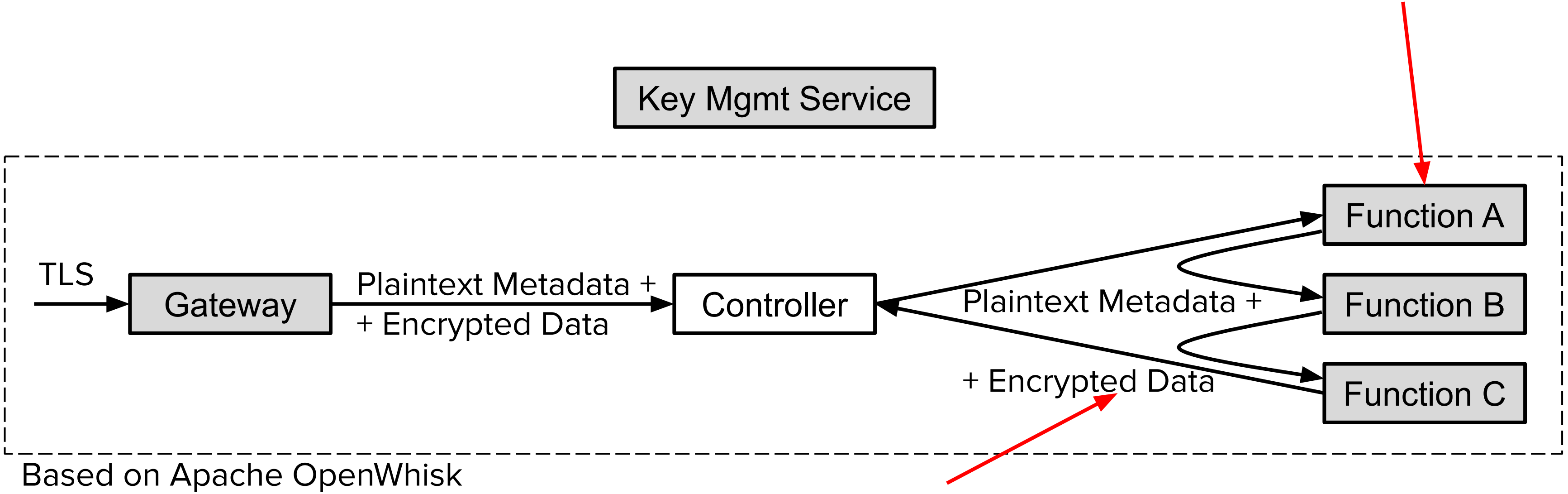
Function C

Based on Apache OpenWhisk

▨ SGX Enclave □ Native Application

What is Clemmys?

1. Trustworthy environment for function execution

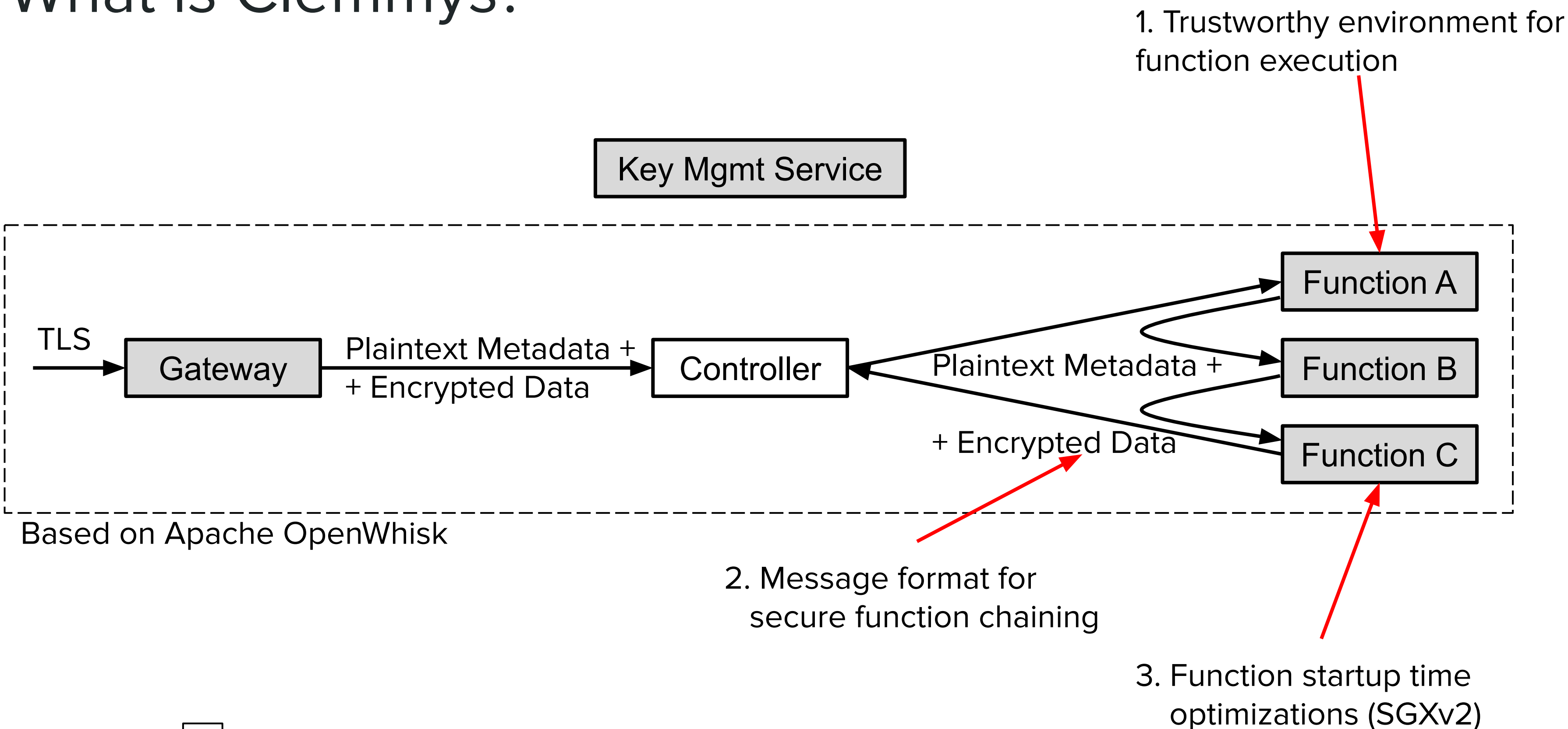


Based on Apache OpenWhisk

2. Message format for secure function chaining

▨ SGX Enclave □ Native Application

What is Clemmys?



1. Trustworthy environment for function execution

2. Message format for secure function chaining

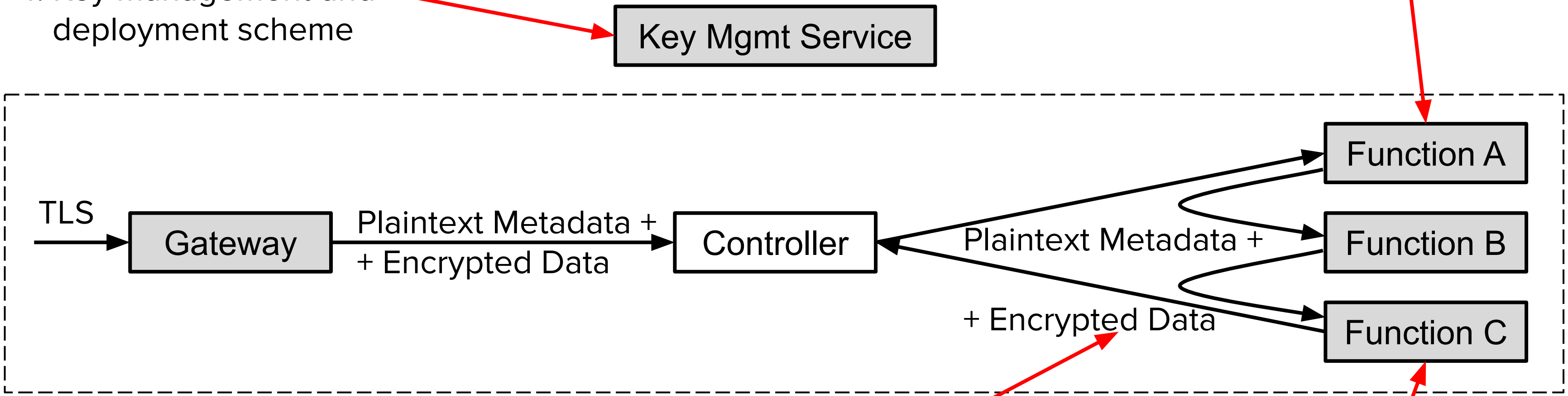
3. Function startup time optimizations (SGXv2)

▨ SGX Enclave □ Native Application

What is Clemmys?

1. Trustworthy environment for function execution

4. Key management and deployment scheme



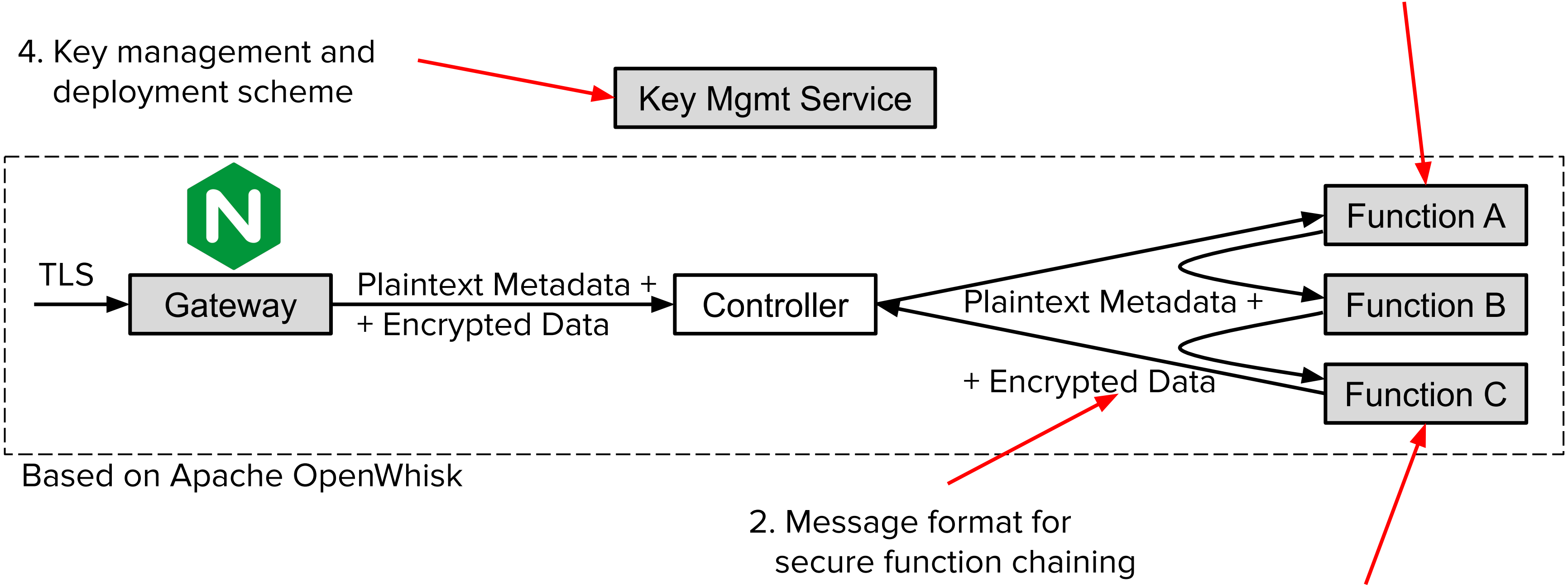
Based on Apache OpenWhisk

2. Message format for secure function chaining

3. Function startup time optimizations (SGXv2)

▨ SGX Enclave □ Native Application

What is Clemmys?



1. Trustworthy environment for function execution

2. Message format for secure function chaining

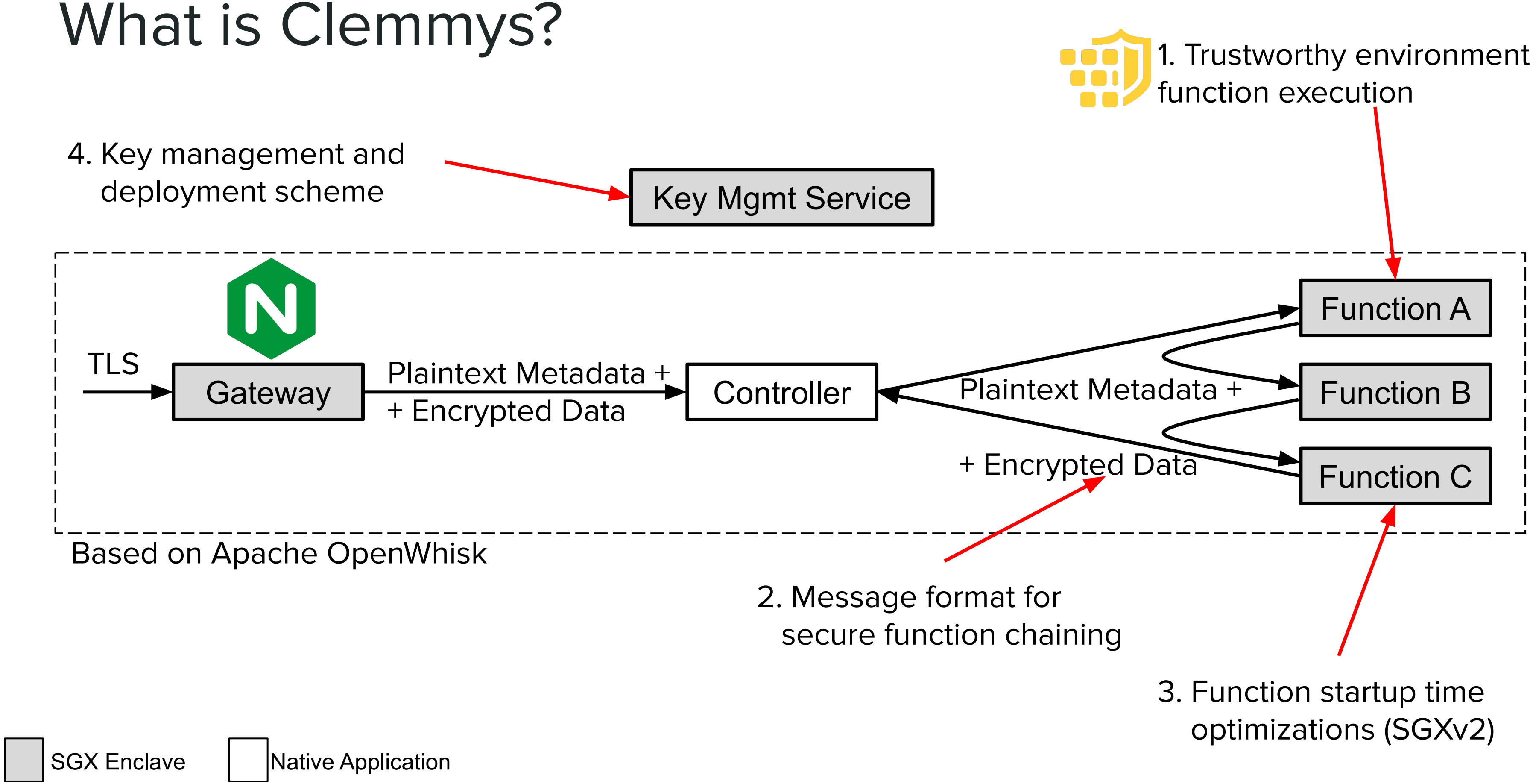
3. Function startup time optimizations (SGXv2)

4. Key management and deployment scheme

Based on Apache OpenWhisk

▨ SGX Enclave □ Native Application

What is Clemmys?



1. Trustworthy environment for function execution

4. Key management and deployment scheme

Key Mgmt Service

TLS

Gateway

Plaintext Metadata + + Encrypted Data

Controller

Plaintext Metadata + + Encrypted Data

Function A

Function B

Function C

Based on Apache OpenWhisk

2. Message format for secure function chaining

3. Function startup time optimizations (SGXv2)

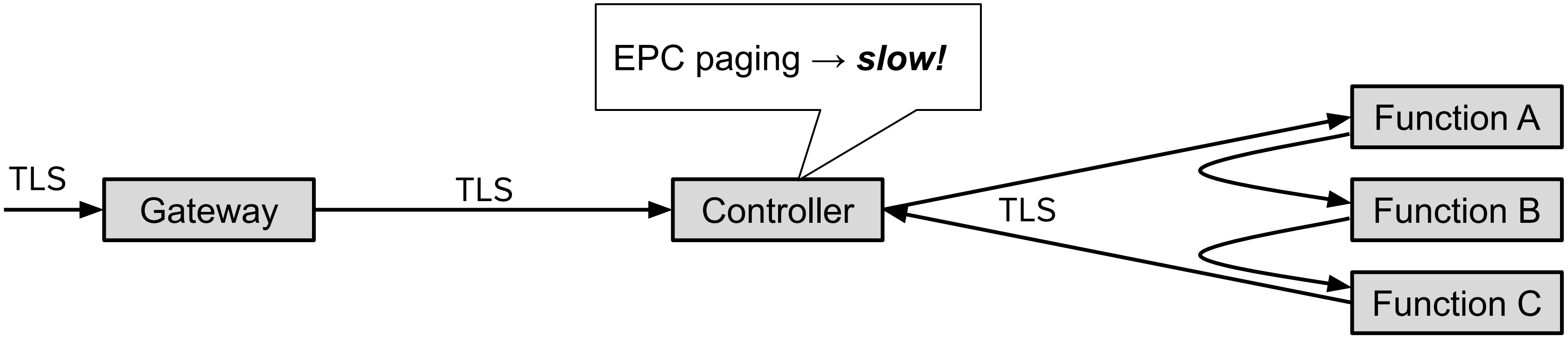
SGX Enclave

Native Application

Components of Clemmys

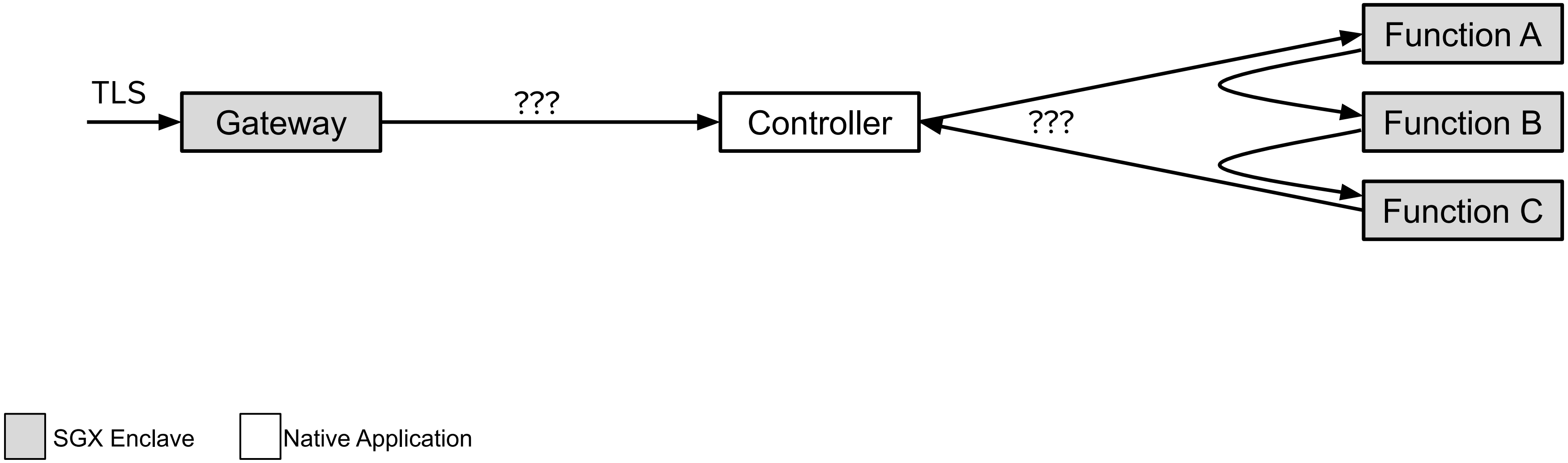
- **Internal encryption**
- Function chain verification
- Function startup optimizations
- Function deployment and key management

How does Clemmys secure communication?



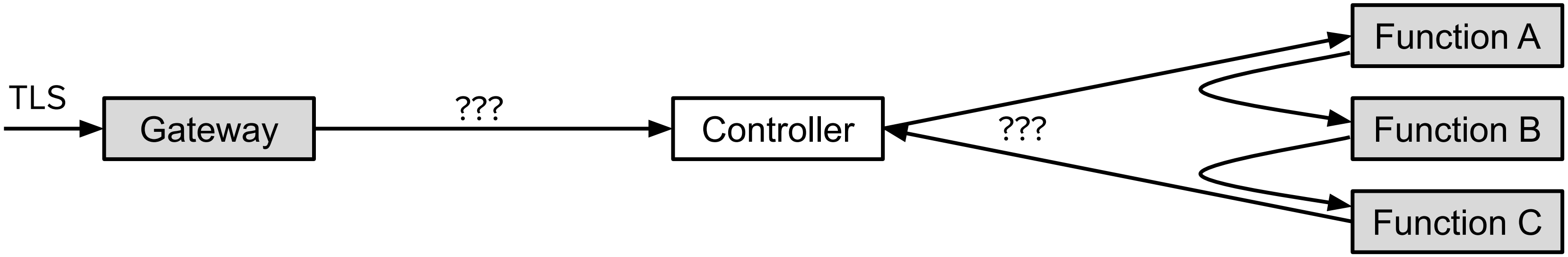
▨ SGX Enclave □ Native Application

How does Clemmys secure communication?



How does Clemmys secure communication?

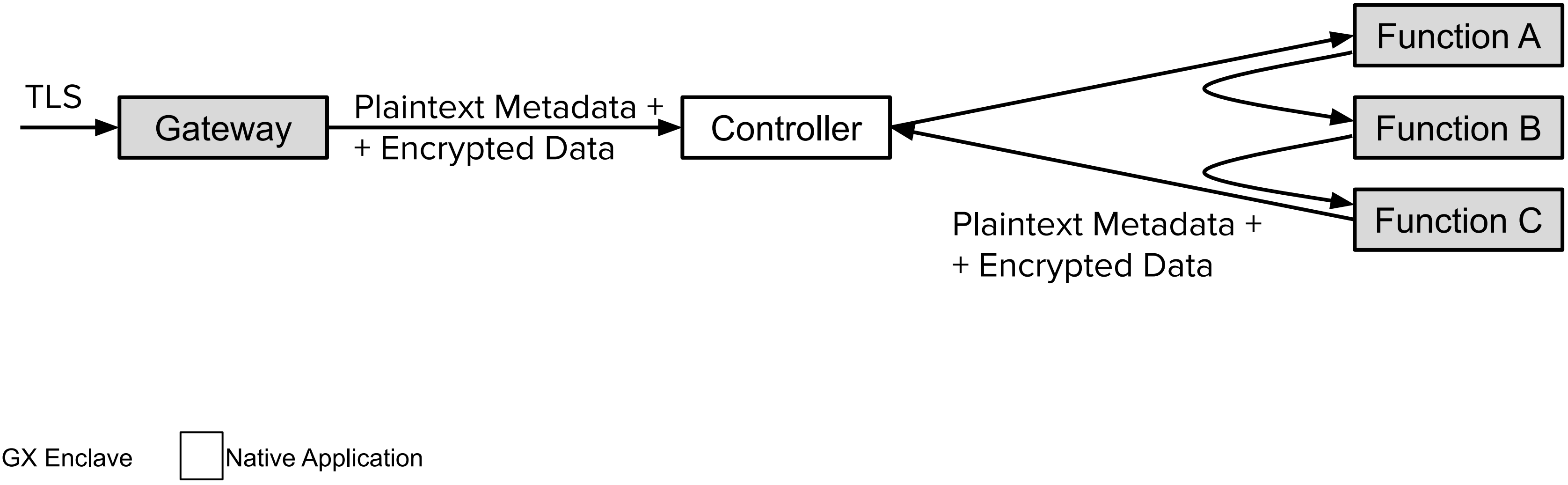
Idea: separate controller metadata (plaintext) from function arguments (encrypted)



▨ SGX Enclave □ Native Application

How does Clemmys secure communication?

Idea: separate controller metadata (plaintext) from function arguments (encrypted)

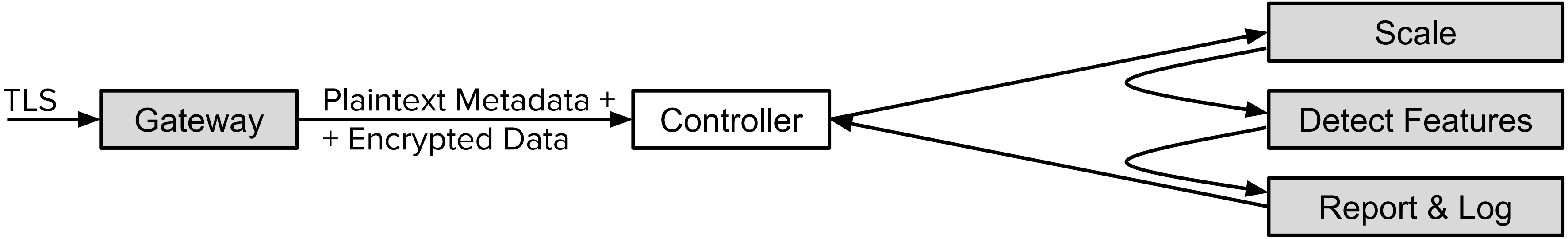


Components of Clemmys

- Internal encryption
- **Function chain verification**
- Function startup optimizations
- Function deployment and key management

Why should function chain order be enforced?

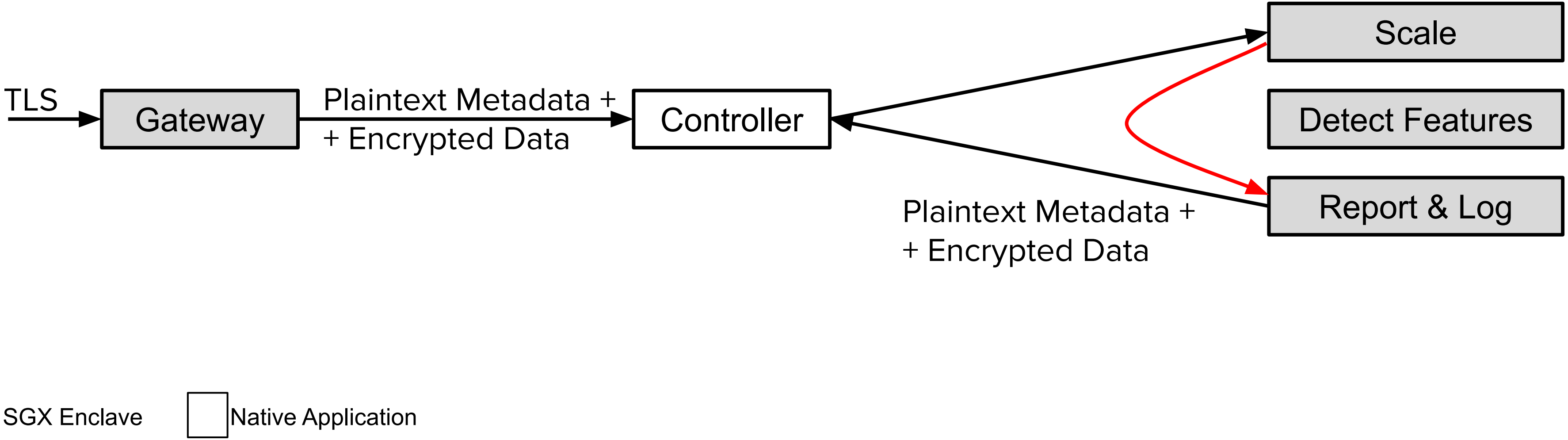
- Naive encryption does not preserve function order.



▨ SGX Enclave □ Native Application

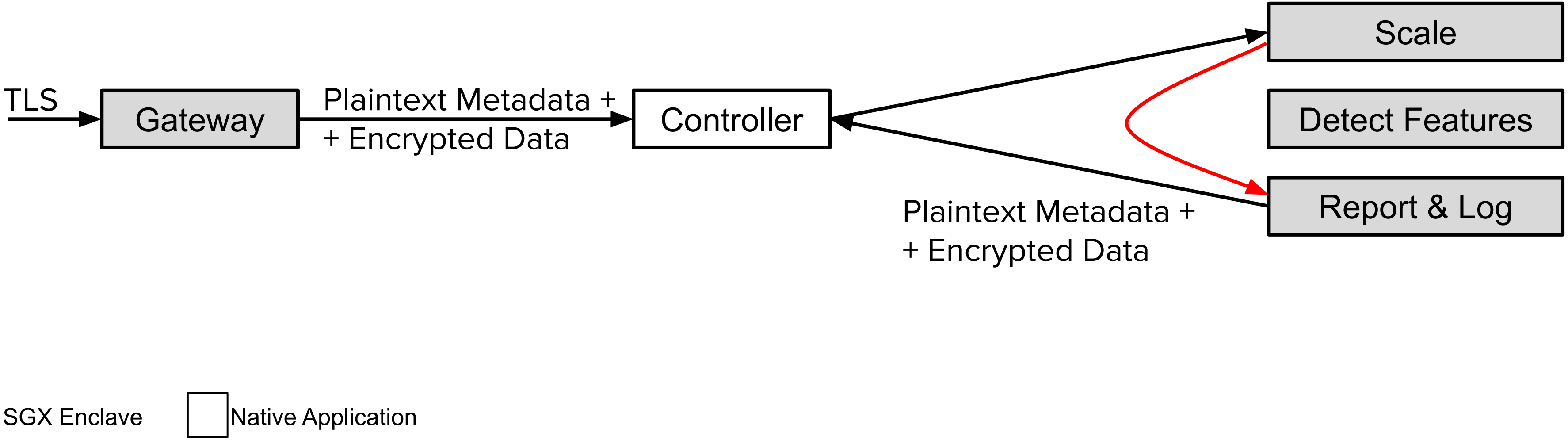
Why should function chain order be enforced?

- Naive encryption does not preserve function order.



Why should function chain order be enforced?

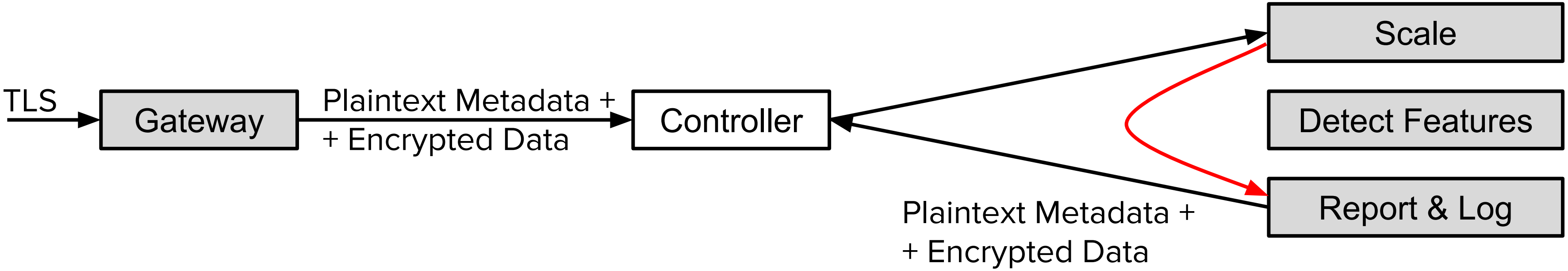
- Naive encryption does not preserve function order.
- Message format should preclude these attack vector.



Why should function chain order be enforced?

- Naive encryption does not preserve function order.
- Message format should preclude these attack vector.

See paper for technical details



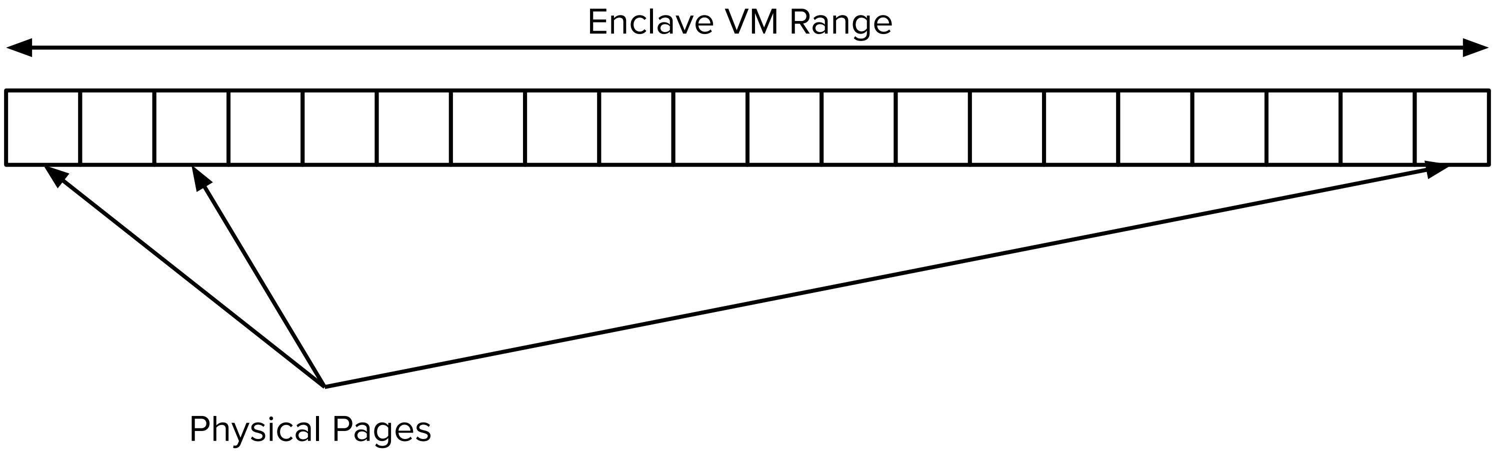
▨ SGX Enclave □ Native Application

Components of Clemmys

- Internal encryption
- Function chain verification
- **Function startup optimizations**
- Function deployment and key management

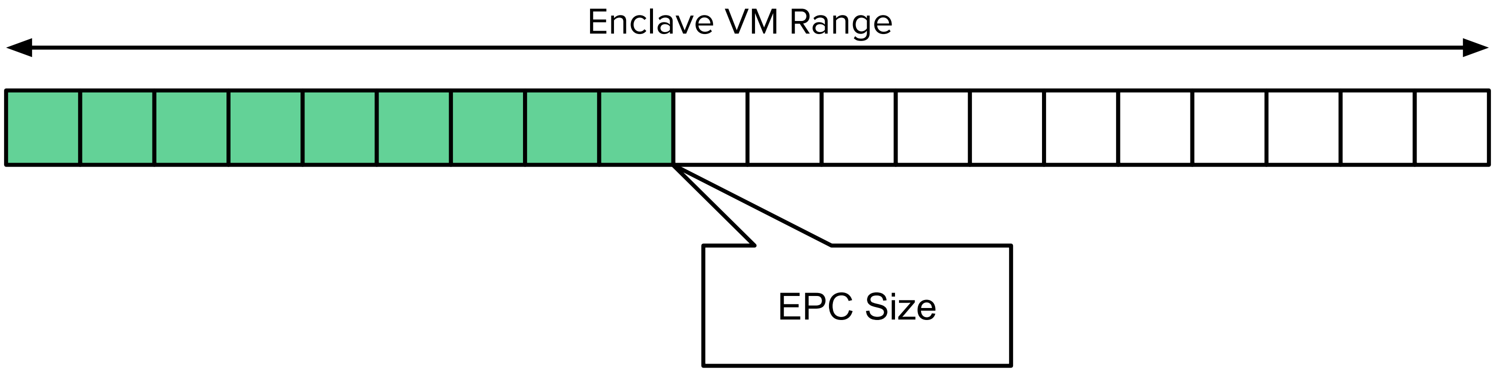
Startup Optimizations

1. SGXv1 Enclave Creation



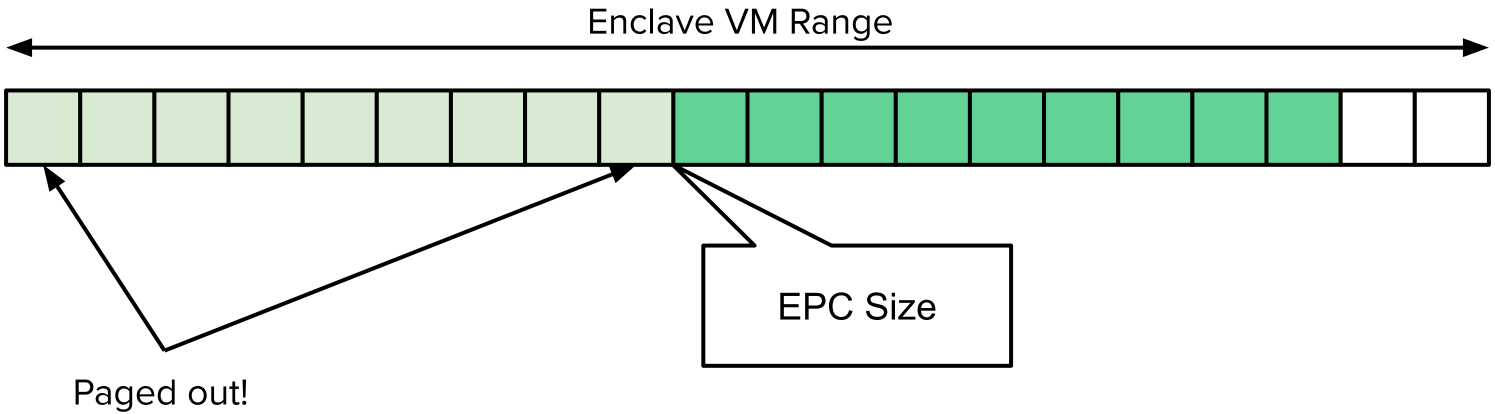
Startup Optimizations

1. SGXv1 Enclave Creation



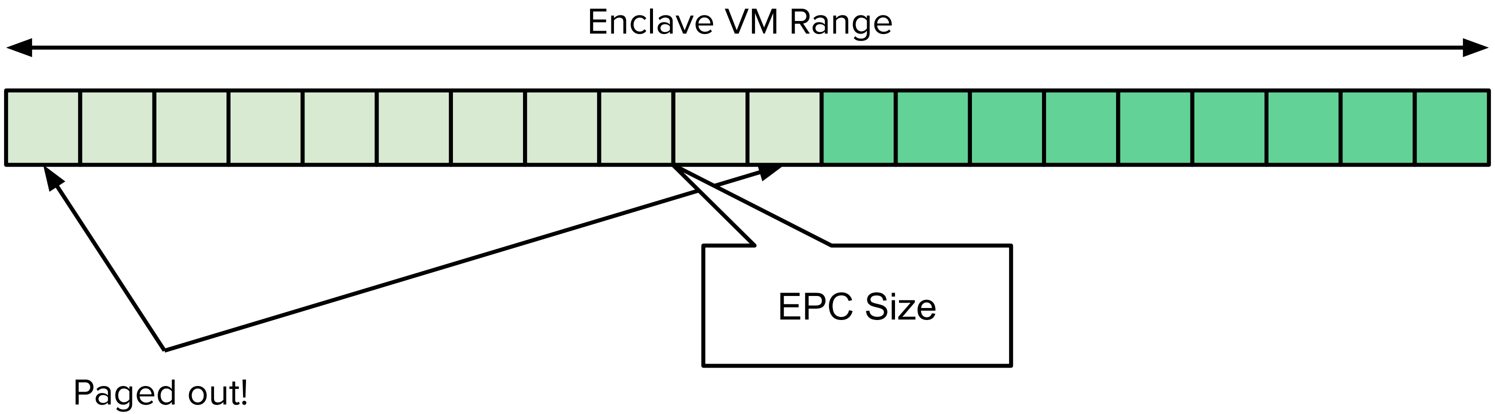
Startup Optimizations

1. SGXv1 Enclave Creation



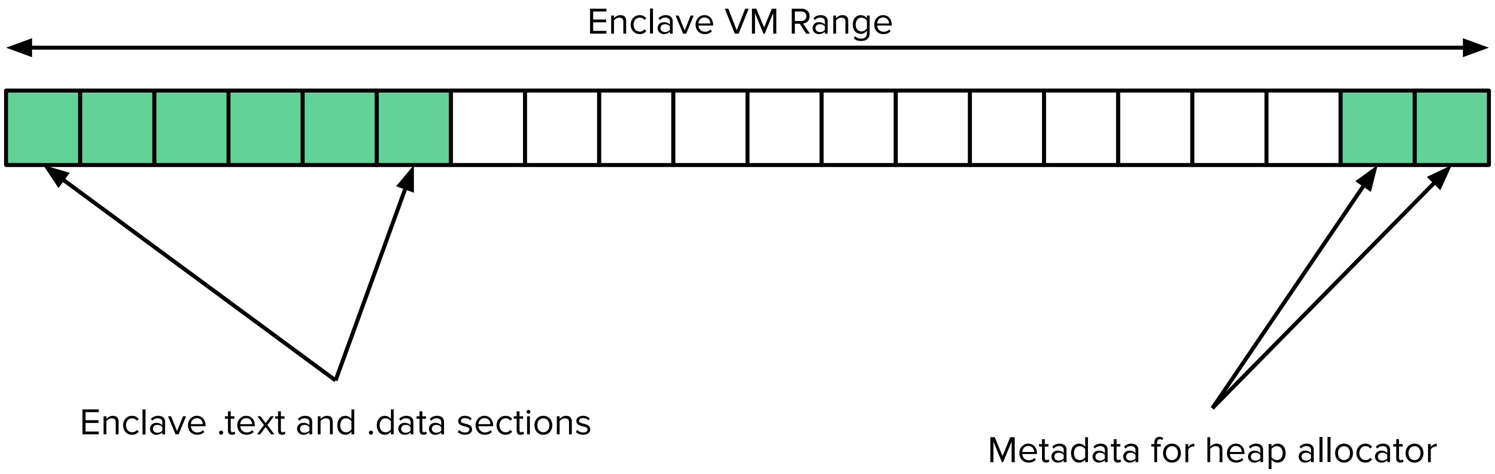
Startup Optimizations

1. SGXv1 Enclave Creation



Startup Optimizations

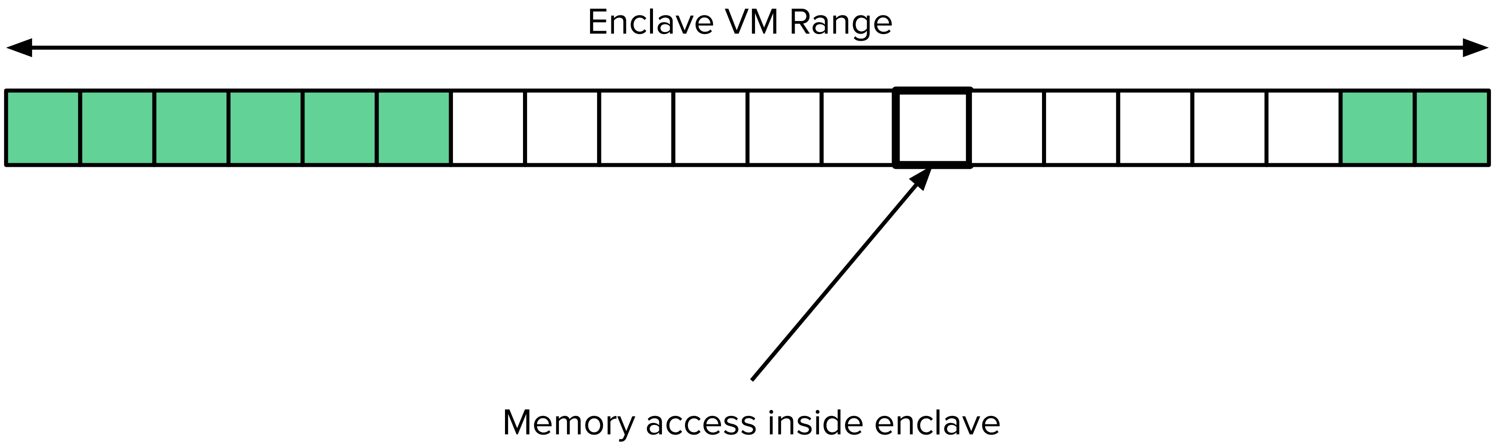
1. **SGXv2 Enclave Creation**



SGXv2 allows adding pages at runtime

Startup Optimizations

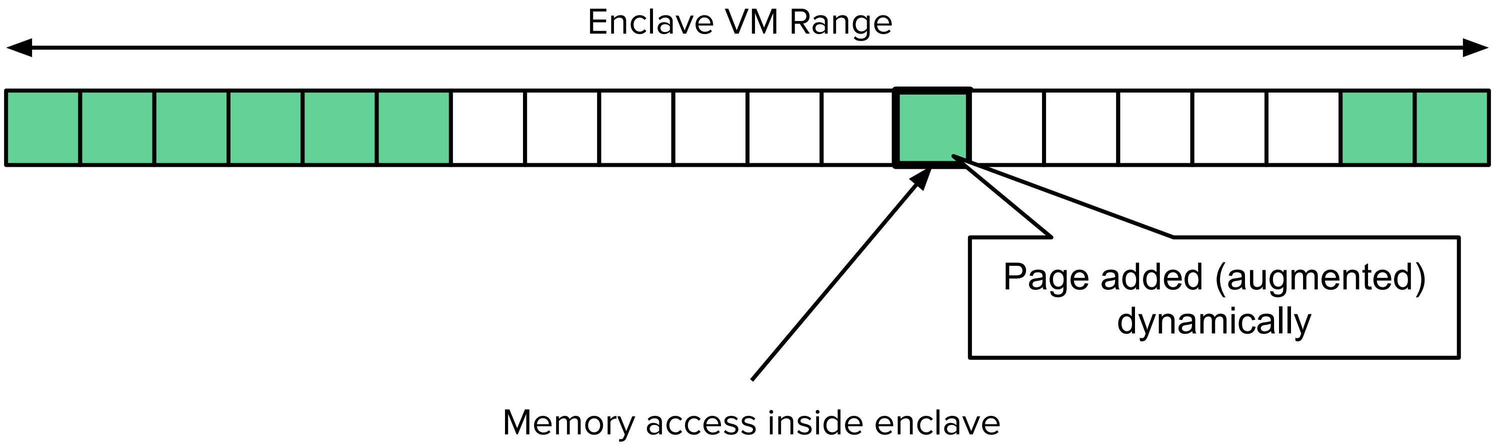
1. SGXv2 Enclave Creation



SGXv2 allows adding pages at runtime

Startup Optimizations

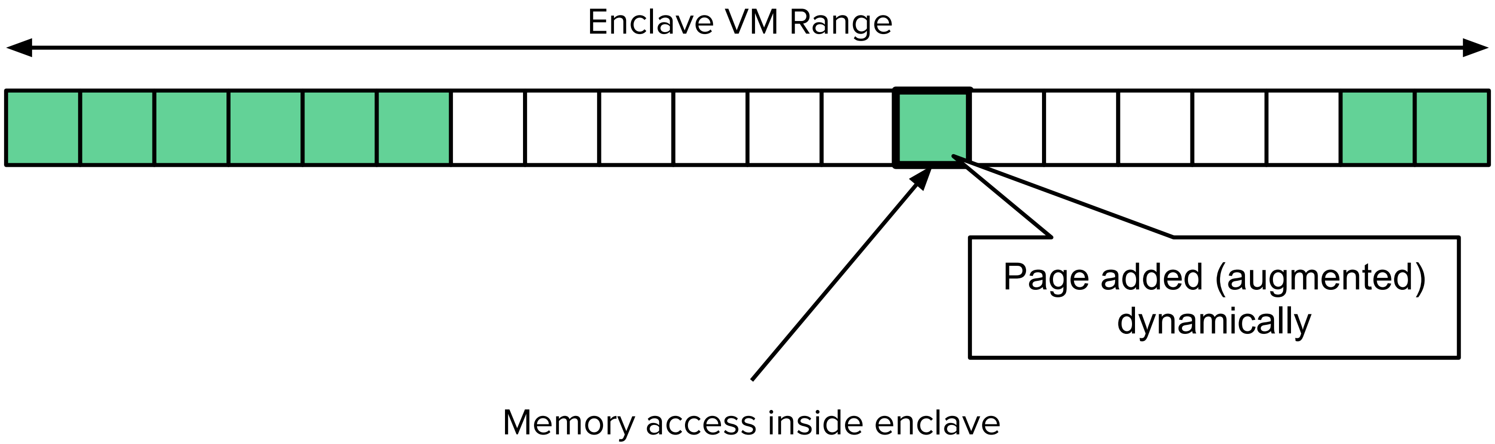
1. SGXv2 Enclave Creation


Enclave VM Range
Page added (augmented) dynamically
Memory access inside enclave

SGXv2 allows adding pages at runtime

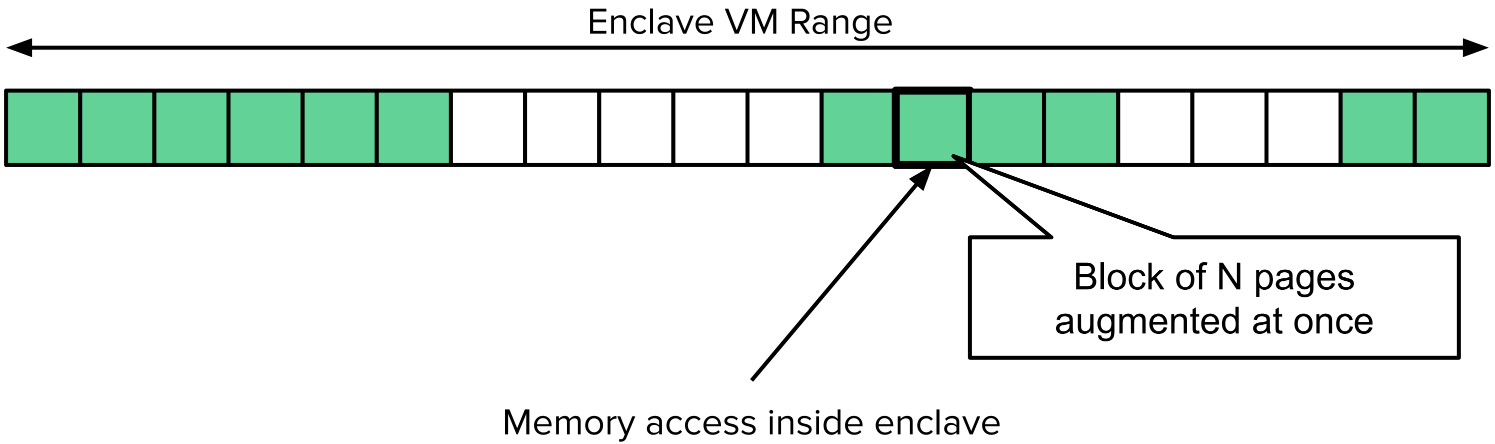
Startup Optimizations

1. SGXv2 Enclave Creation
2. **EPC Batch Augmentation**


Enclave VM Range
Page added (augmented) dynamically
Memory access inside enclave

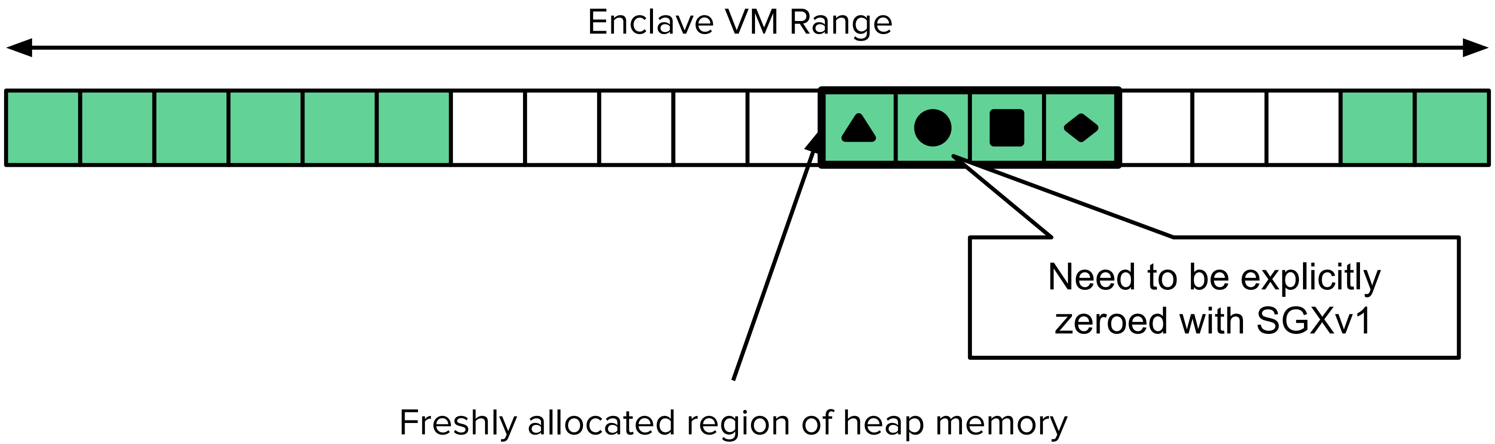
Startup Optimizations

1. SGXv2 Enclave Creation
2. EPC Batch Augmentation


Enclave VM Range
Block of N pages augmented at once
Memory access inside enclave

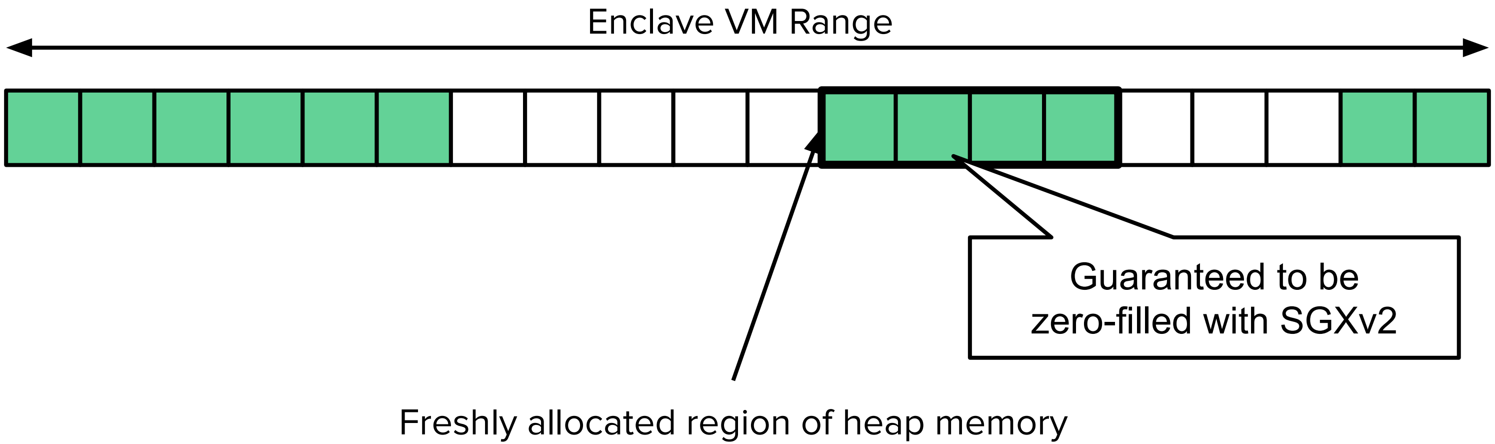
Startup Optimizations

1. SGXv2 Enclave Creation
2. EPC Batch Augmentation
3. Memory zeroing on deallocation


Enclave VM Range
Need to be explicitly zeroed with SGXv1
Freshly allocated region of heap memory

Startup Optimizations

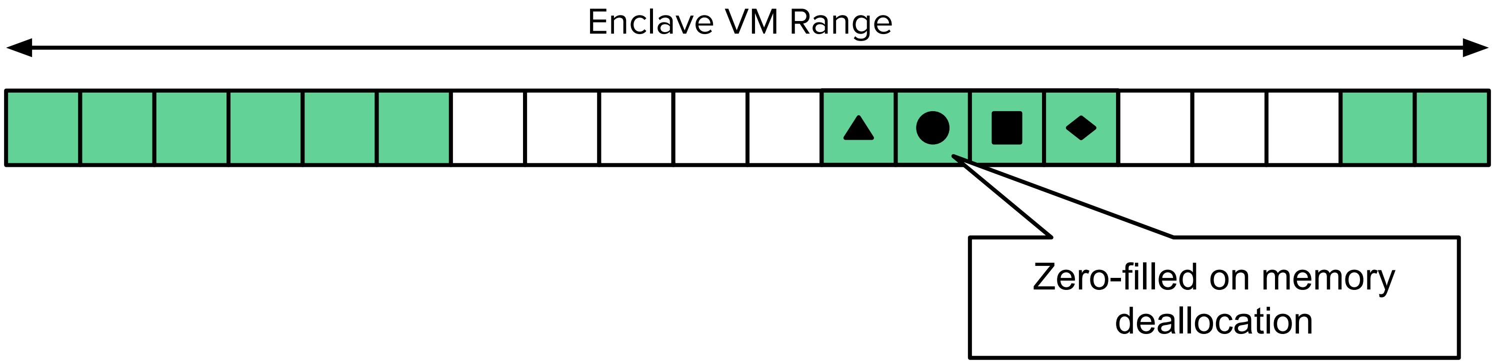
1. SGXv2 Enclave Creation
2. EPC Batch Augmentation
3. Memory zeroing on deallocation



Enclave VM Range
Guaranteed to be zero-filled with SGXv2
Freshly allocated region of heap memory

Startup Optimizations

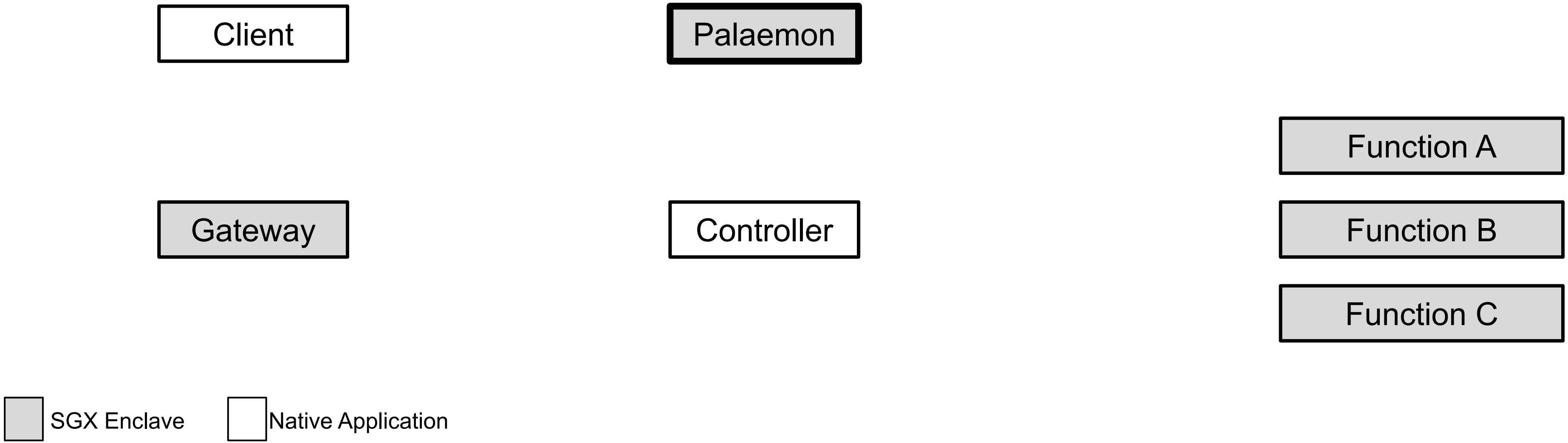
1. SGXv2 Enclave Creation
2. EPC Batch Augmentation
3. Memory zeroing on deallocation



Components of Clemmys

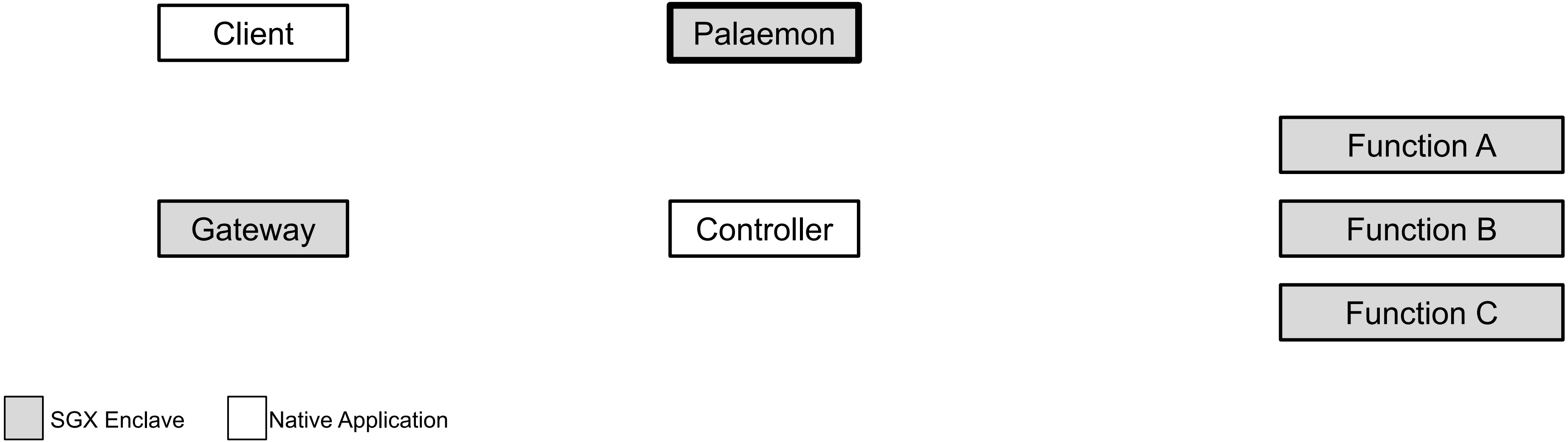
- Internal encryption
- Function chain verification
- Function startup optimizations
- **Function deployment and key management**

How is Clemmys function deployed?

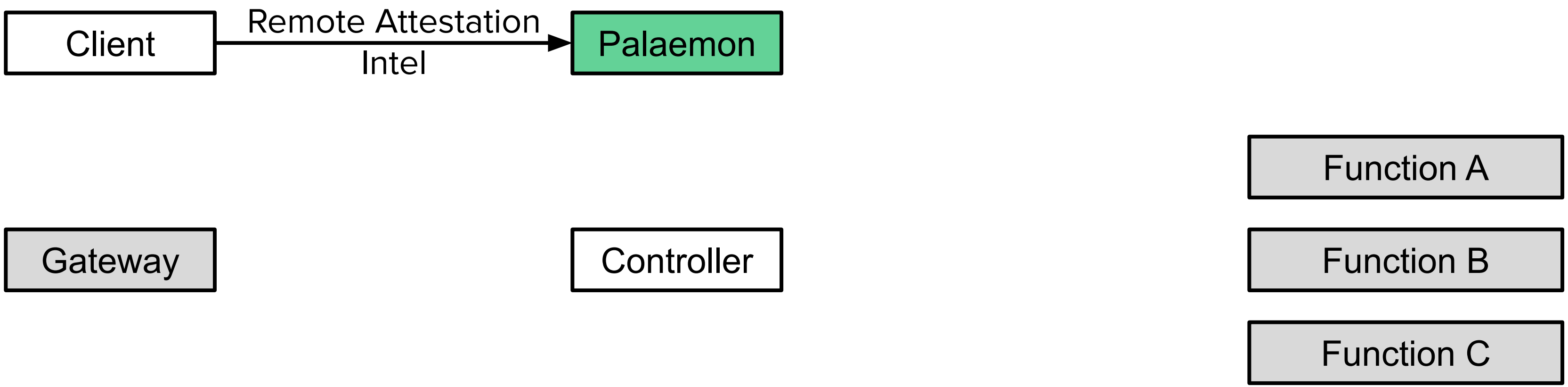


How is Clemmys function deployed?

- Palaemon - remote attestation and configuration service
- Transparent configuration management:
  - Environment variables and command-line arguments

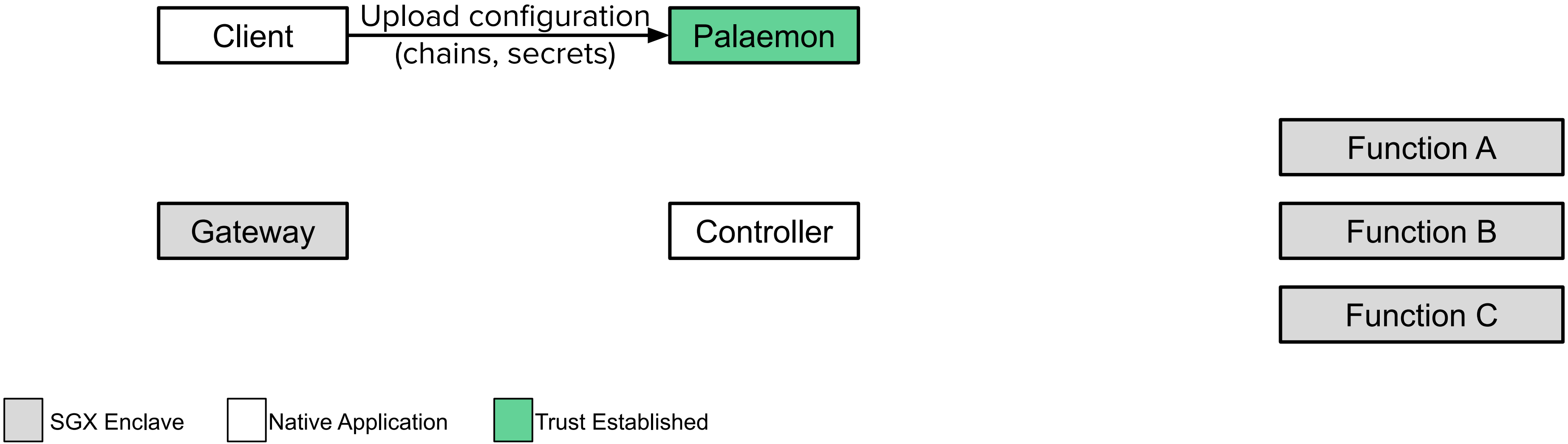


How is Clemmys function deployed?

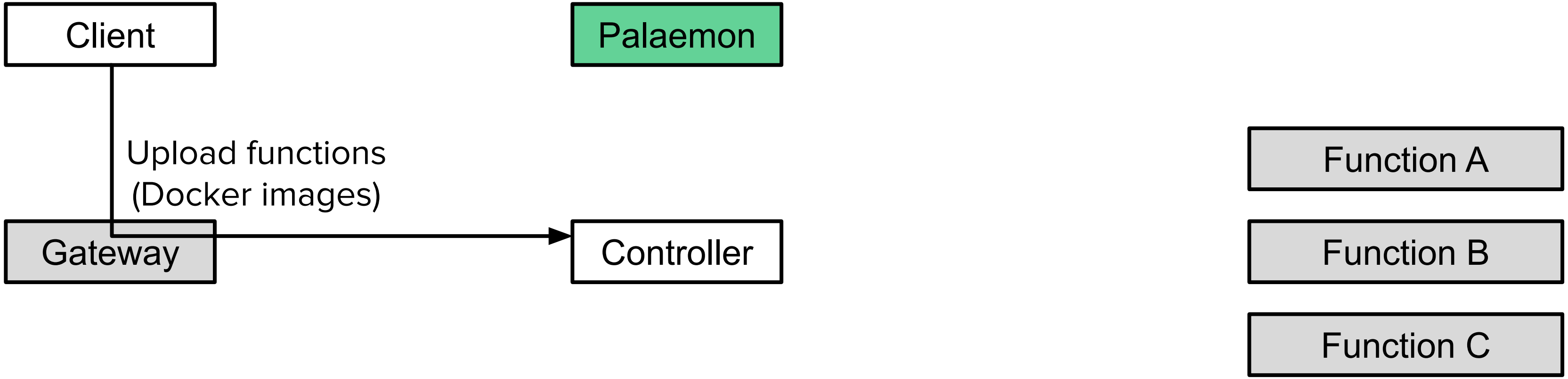


▨ SGX Enclave □ Native Application 🟩 Trust Established

How is Clemmys function deployed?



How is Clemmys function deployed?



Client

Palaemon

Upload functions (Docker images)

Gateway

Controller

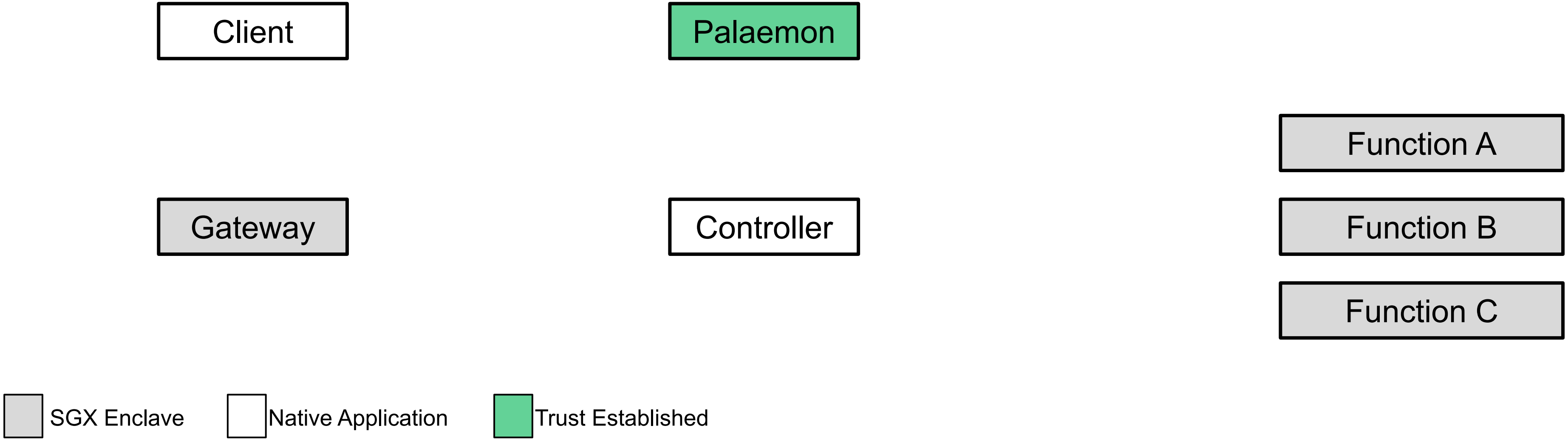
Function A

Function B

Function C

⬜ SGX Enclave ⬜ Native Application 🟩 Trust Established

How is Clemmys function invoked?



Client

Palaemon

Function A

Gateway

Controller

Function B

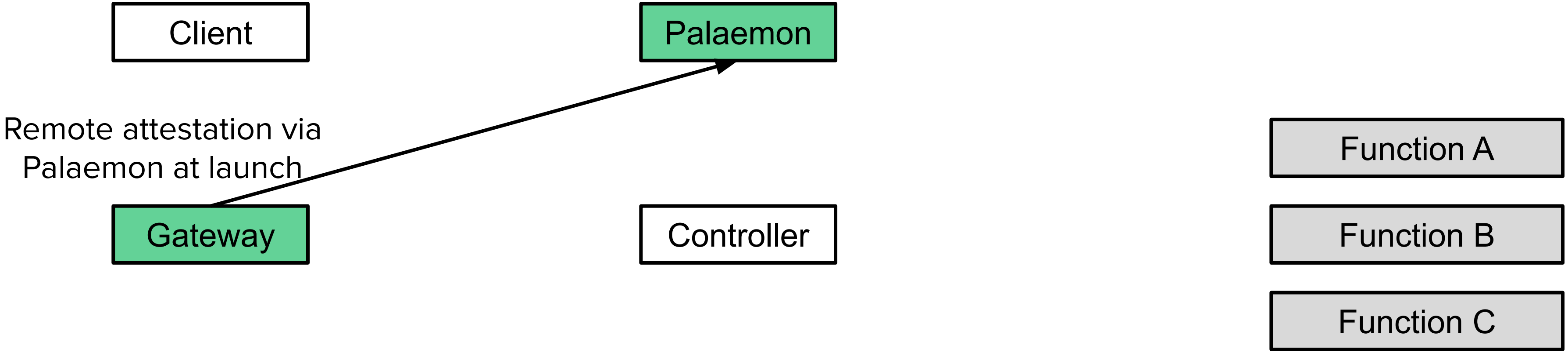
Function C

▨ SGX Enclave

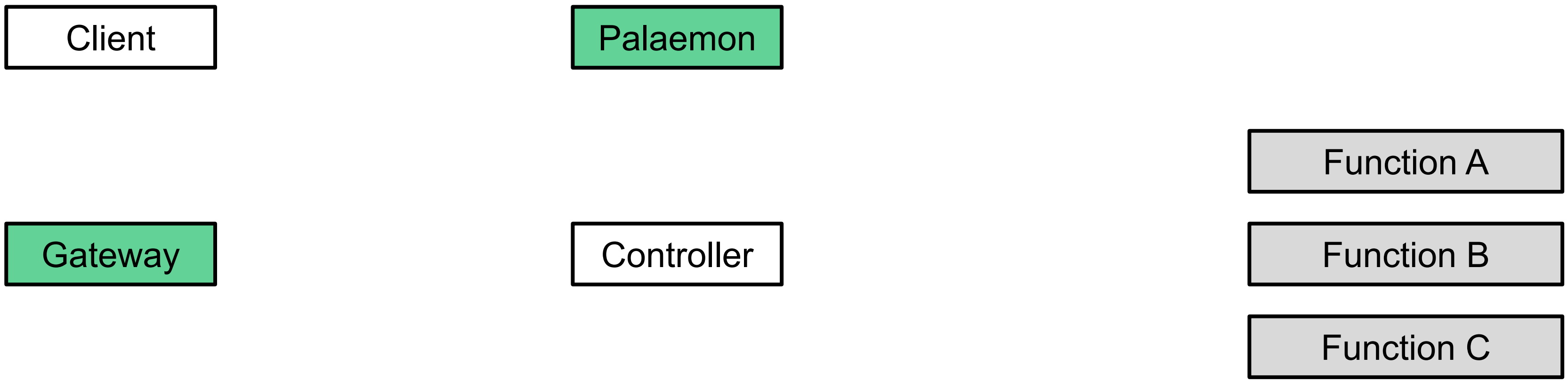
□ Native Application

🟩 Trust Established

How is Clemmys function invoked?

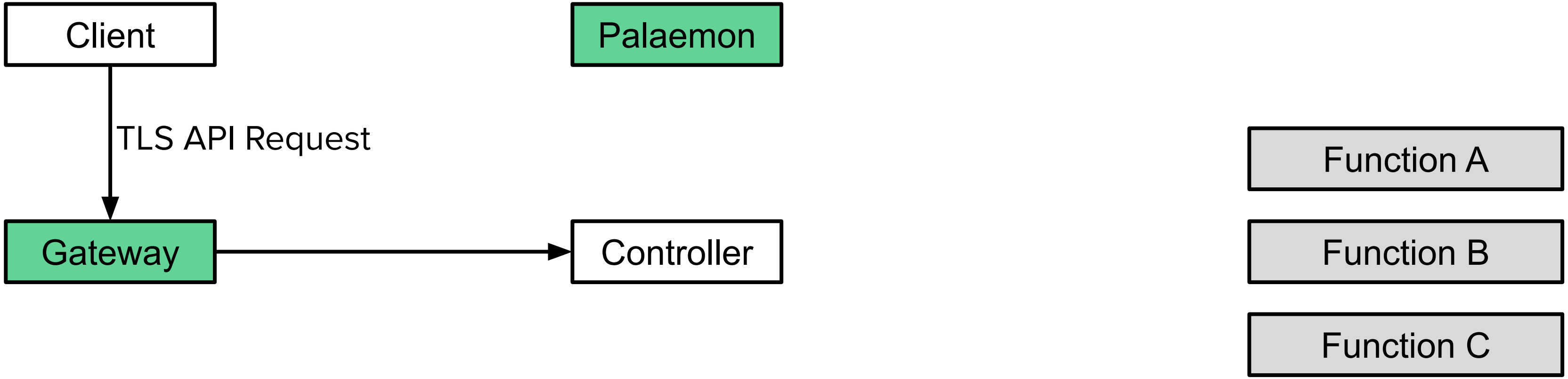


How is Clemmys function invoked?

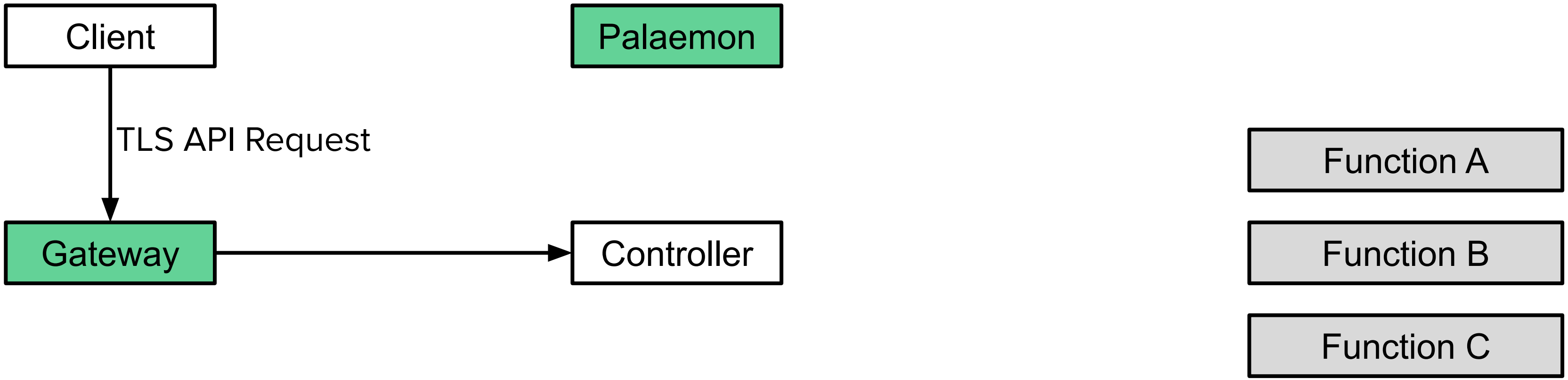


▨ SGX Enclave □ Native Application 🟩 Trust Established

How is Clemmys function invoked?

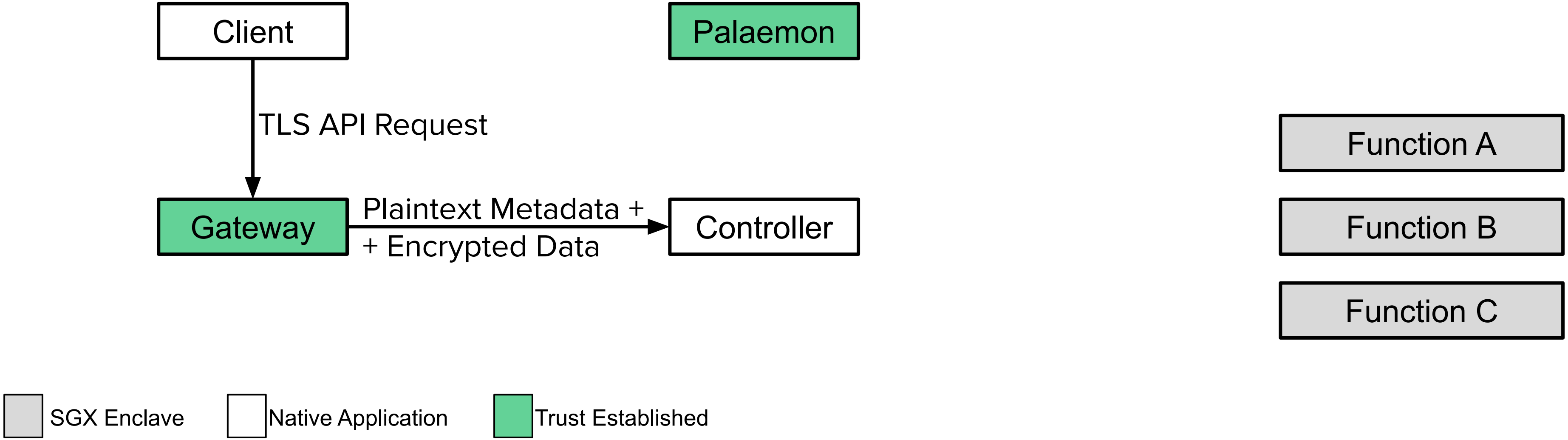


How is Clemmys function invoked?



▨ SGX Enclave □ Native Application 🟩 Trust Established

How is Clemmys function invoked?



Client

TLS API Request

Gateway

Plaintext Metadata +
+ Encrypted Data

Controller

Palaemon

Function A

Function B

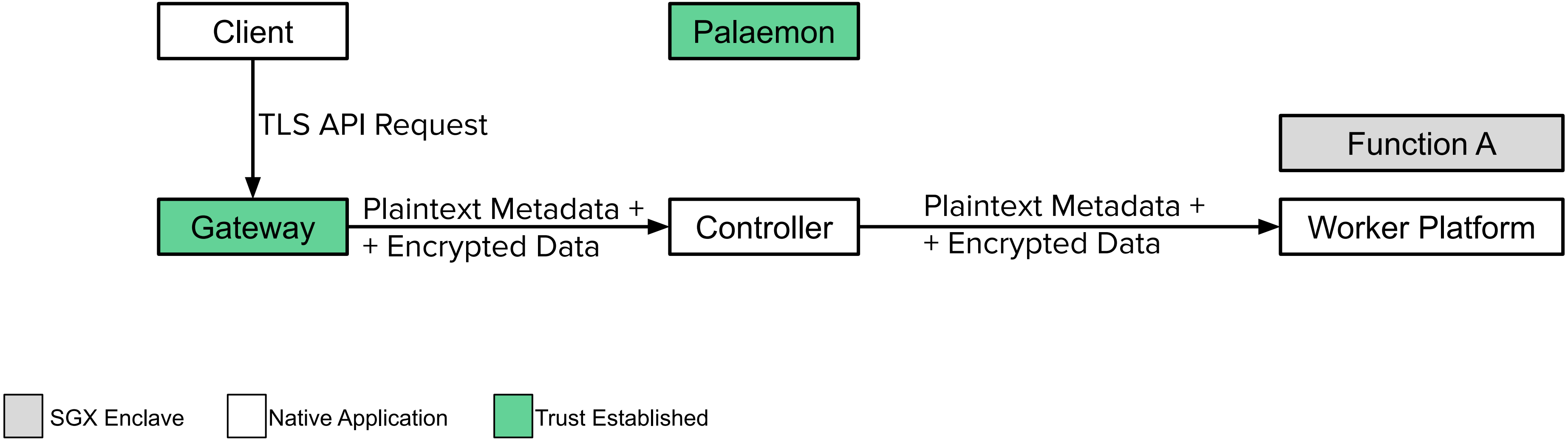
Function C

▨ SGX Enclave

□ Native Application

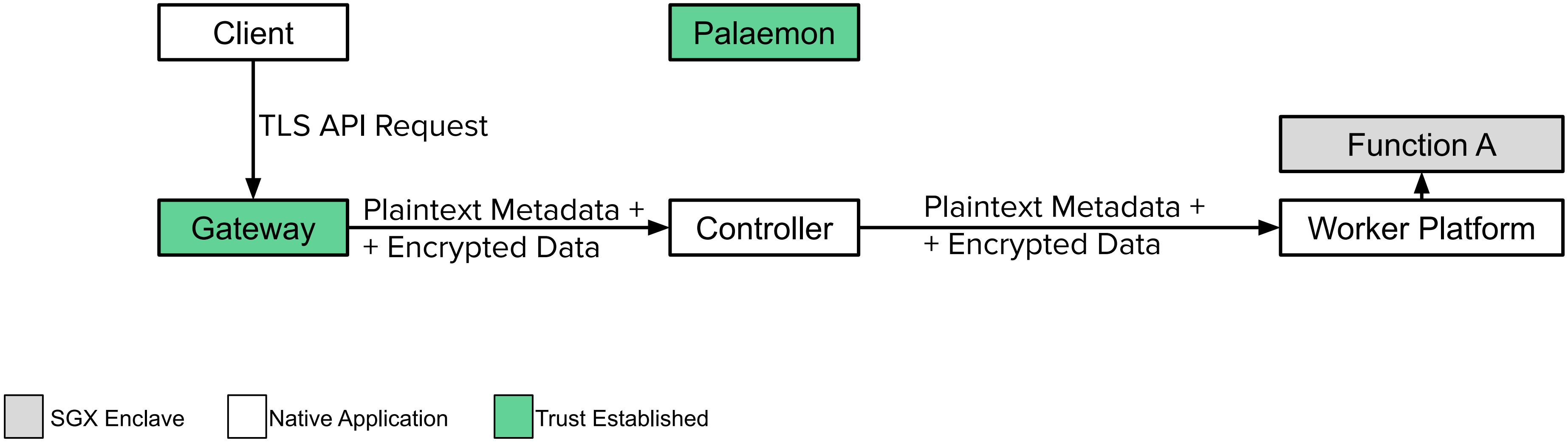
🟩 Trust Established

How is Clemmys function invoked?



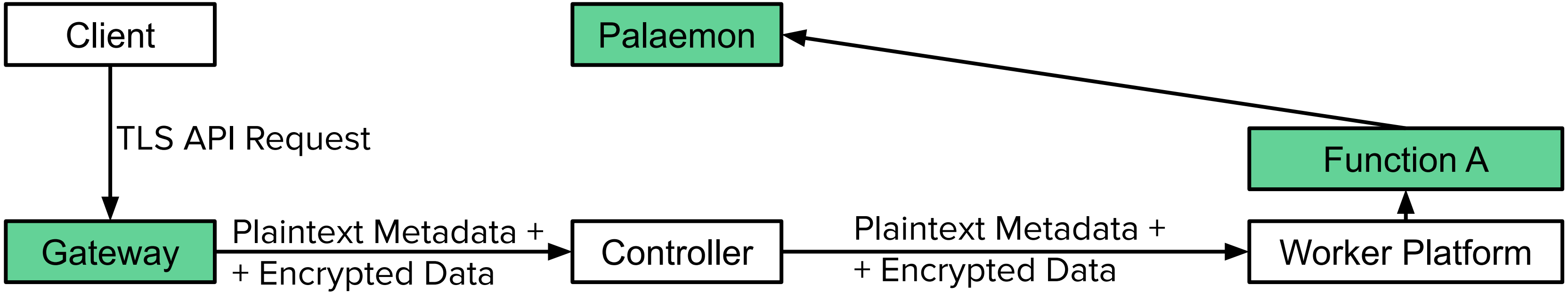
How is Clemmys function invoked?

1. Platform launches the enclave using the plaintext metadata



How is Clemmys function invoked?

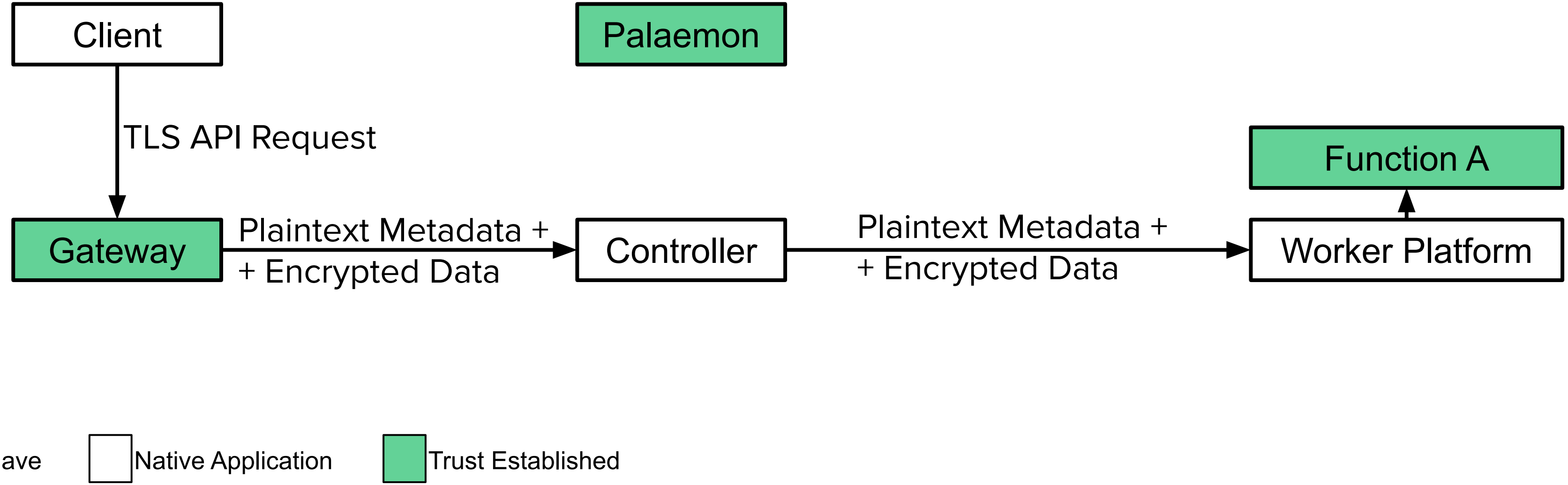
1. Platform launches the enclave using the plaintext metadata
2. Enclave performs remote attestation and configuration with Palaemon



⬜ SGX Enclave □ Native Application 🟩 Trust Established

How is Clemmys function invoked?

1. Platform launches the enclave using the plaintext metadata
2. Enclave performs remote attestation and configuration with Palaemon
3. Enclave decrypts and processes the request

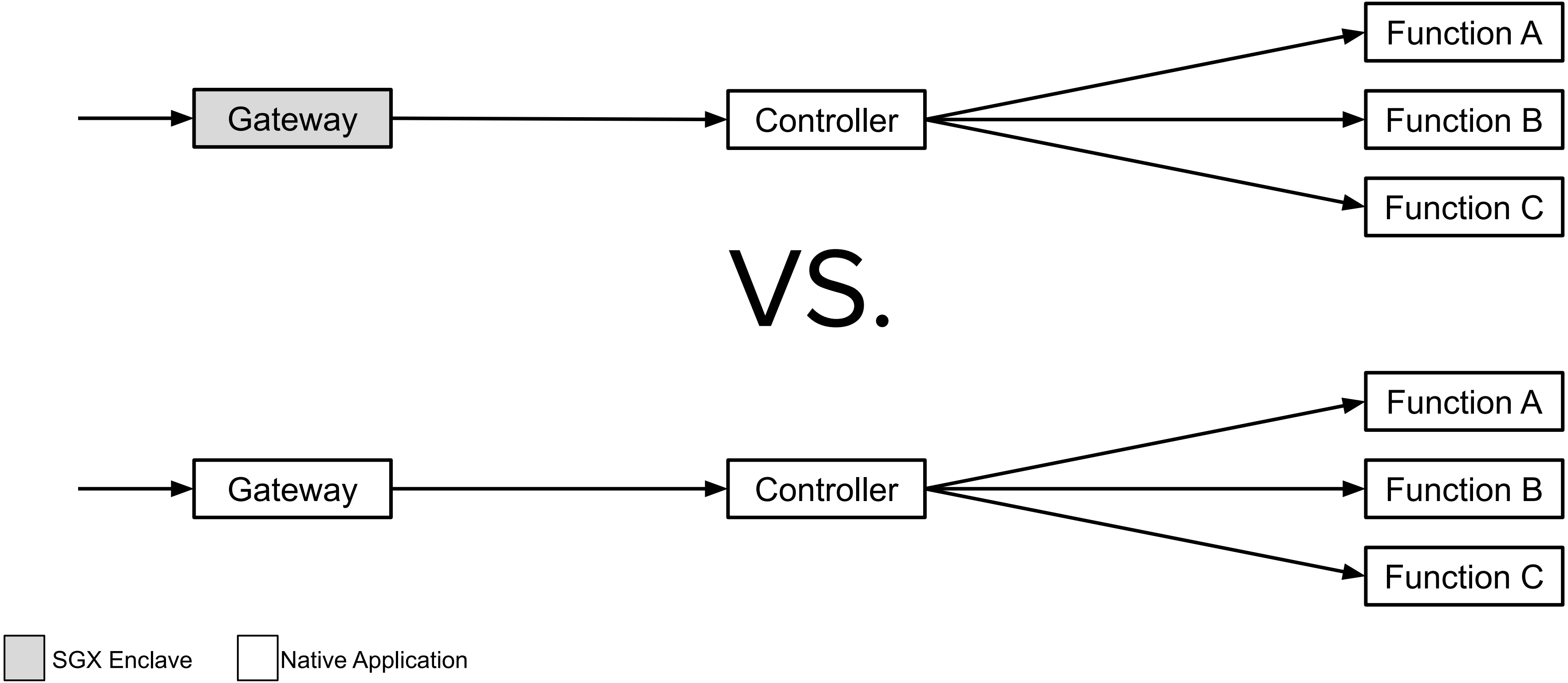


⬜ SGX Enclave □ Native Application 🟩 Trust Established

Outline

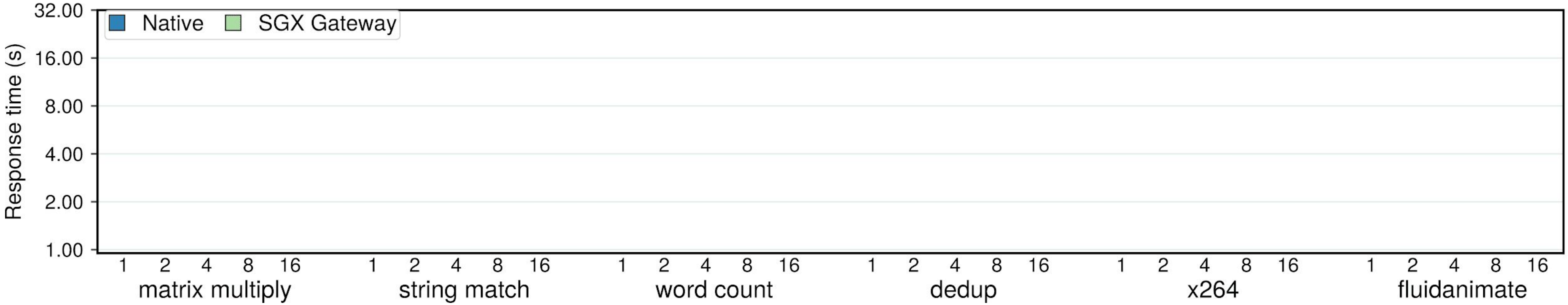
- ~~Motivation~~
- ~~Design~~
- Evaluation
- Summary

Gateway Overhead

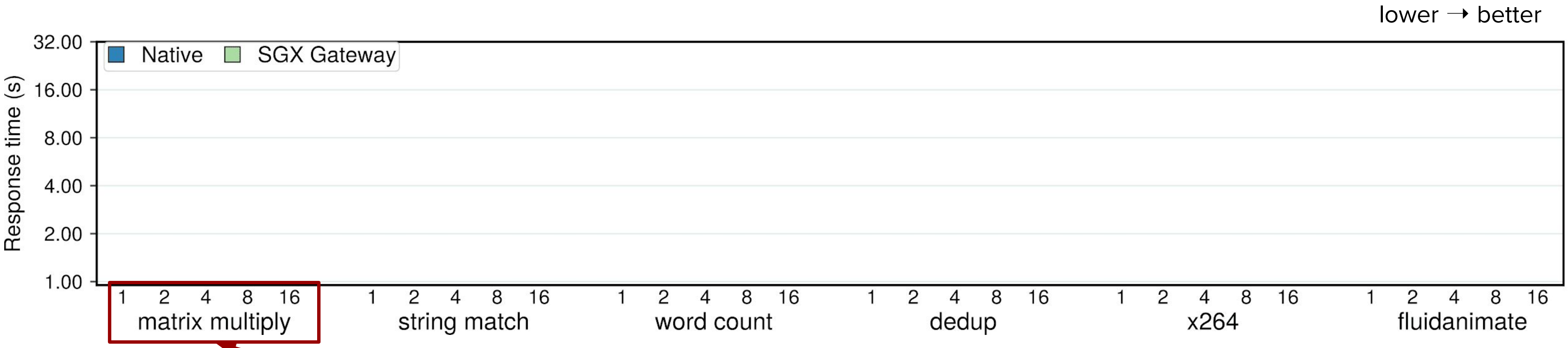


Gateway Overhead

lower → better



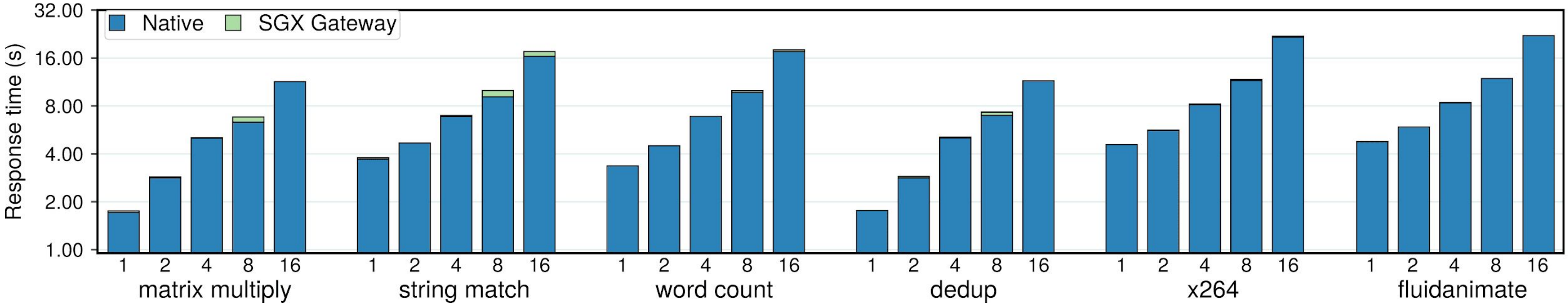
Gateway Overhead



Number of functions running on the worker node

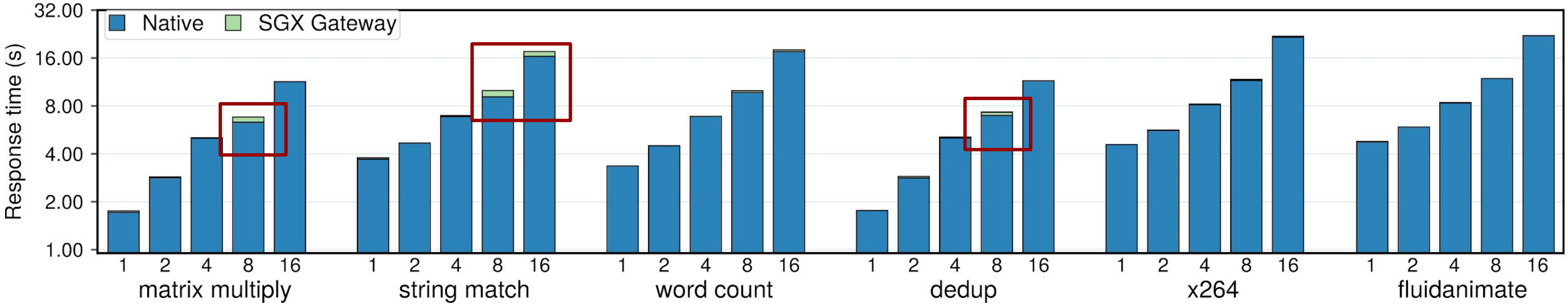
Gateway Overhead

lower → better



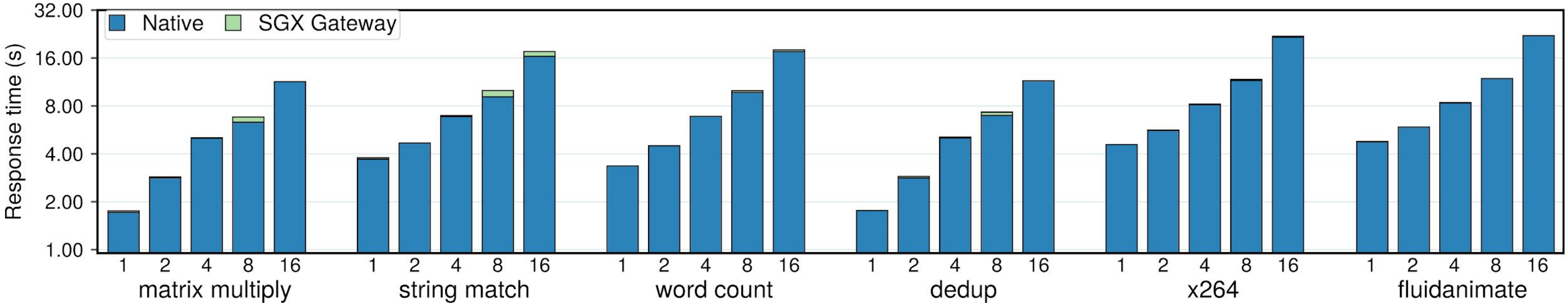
Gateway Overhead

lower → better



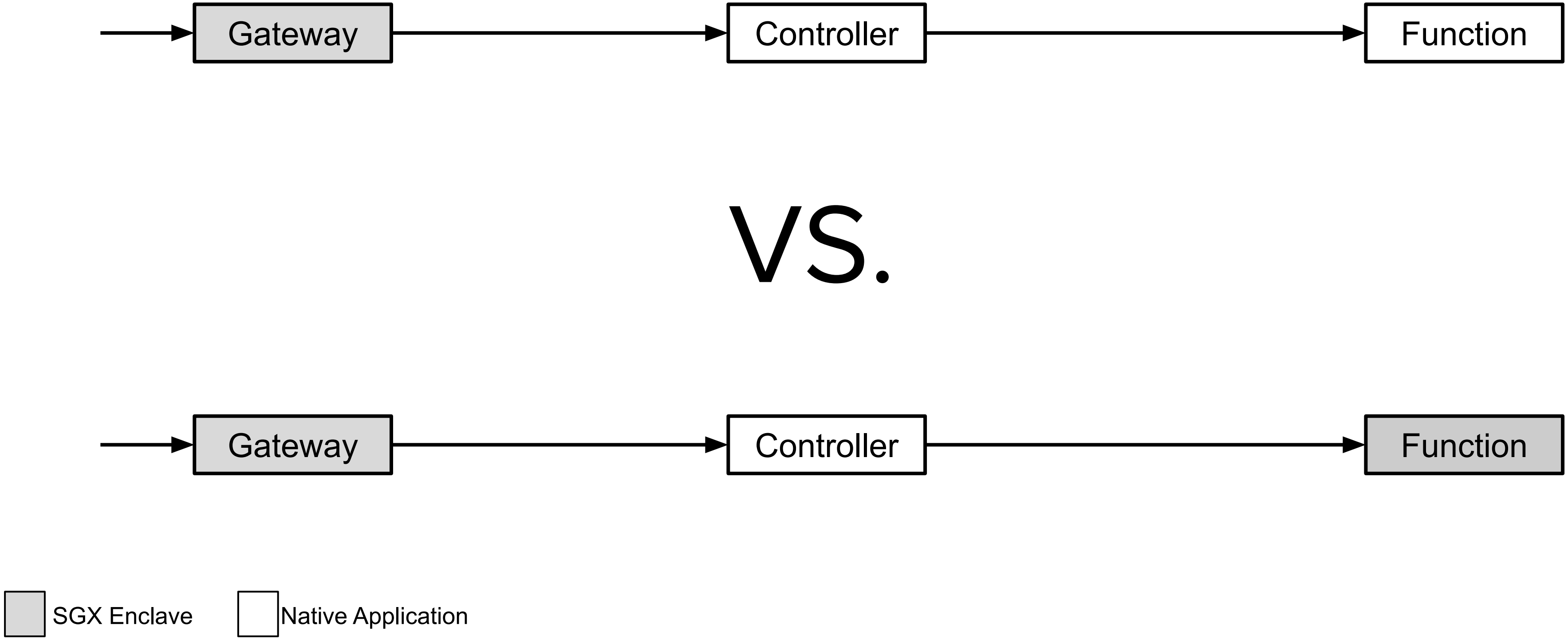
Gateway Overhead

lower → better



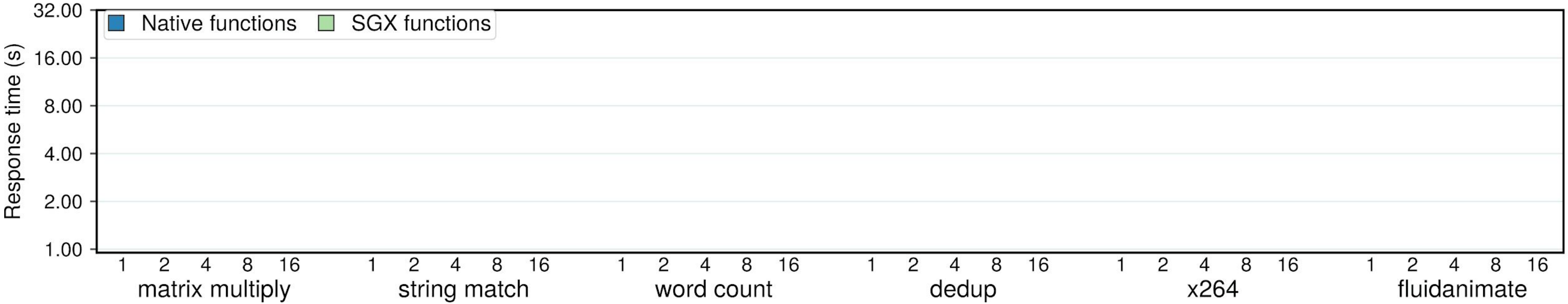
Minimal overhead (~1-5%) over native API Gateway

Function Overhead



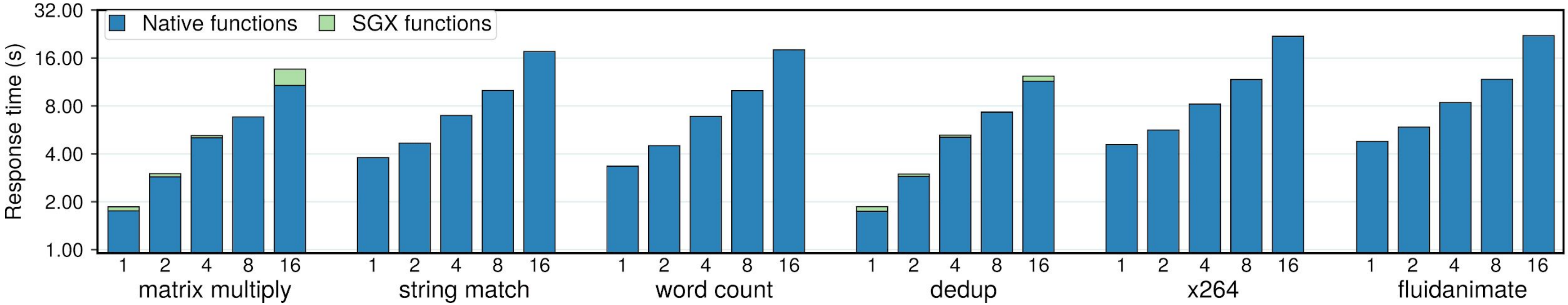
Function Overhead

lower → better



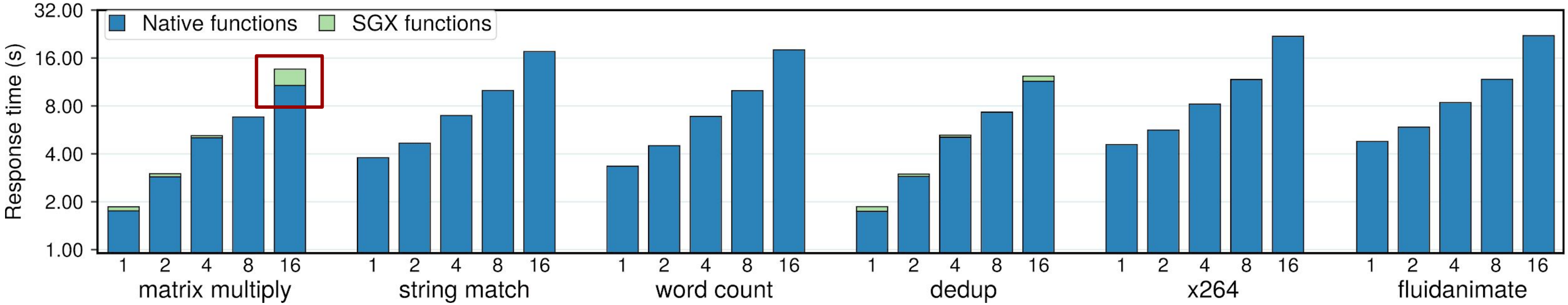
Function Overhead

lower → better



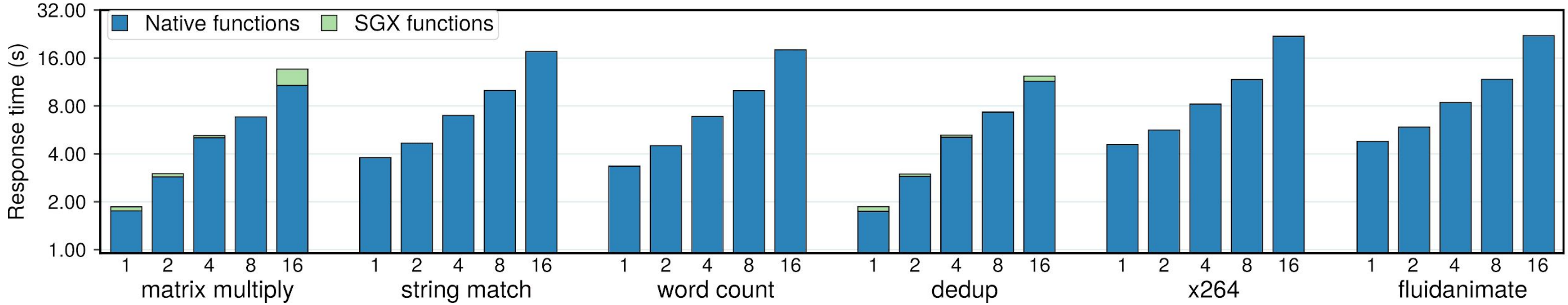
Function Overhead

lower → better



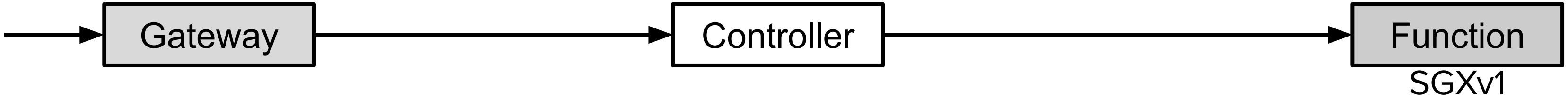
Function Overhead

lower → better

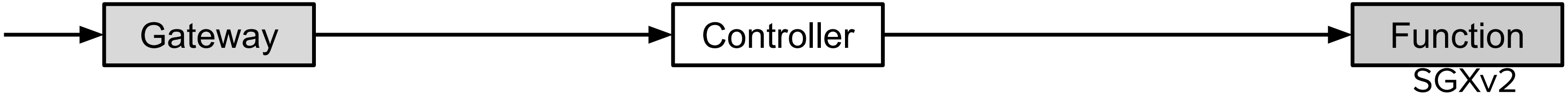


Minimal overhead over native functions (up to 25%)

SGXv2 Optimizations

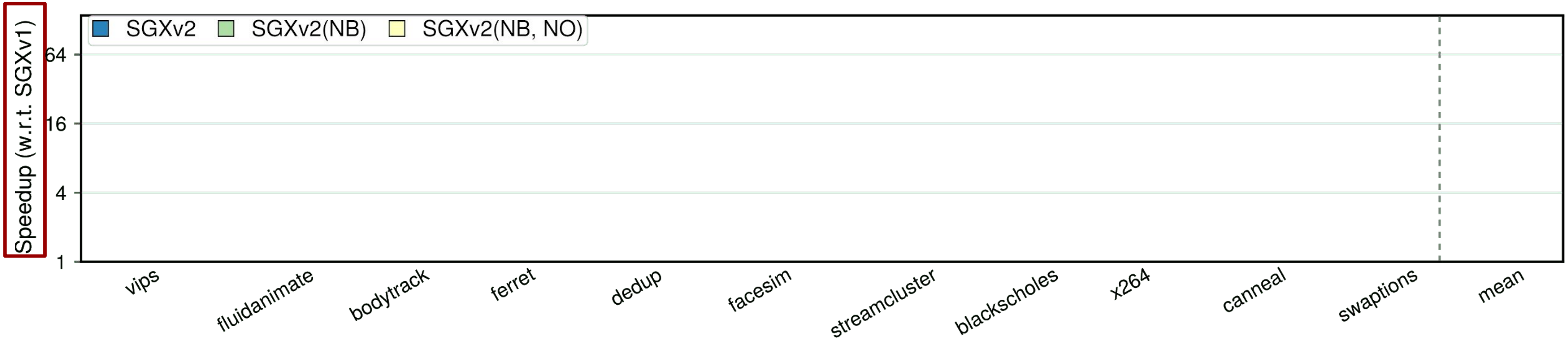


VS.



▨ SGX Enclave □ Native Application

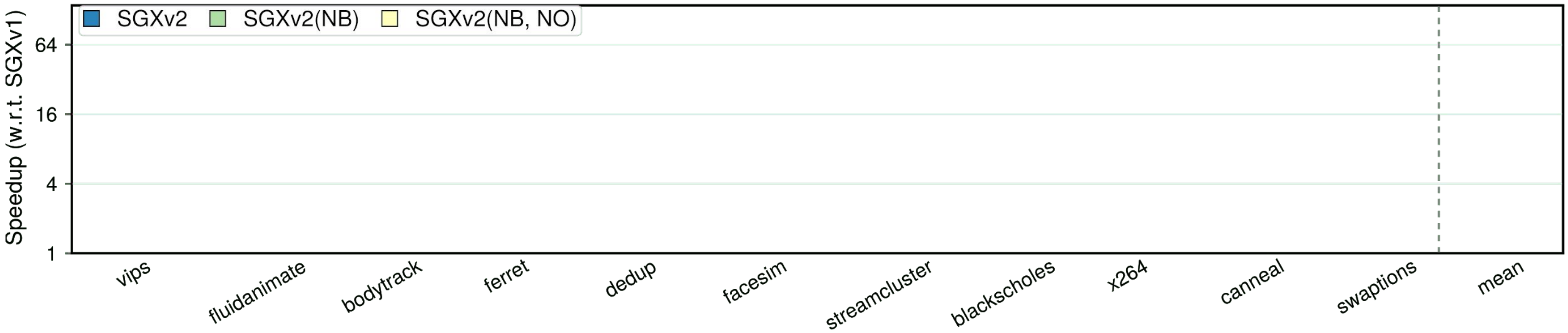
SGXv2 Optimizations



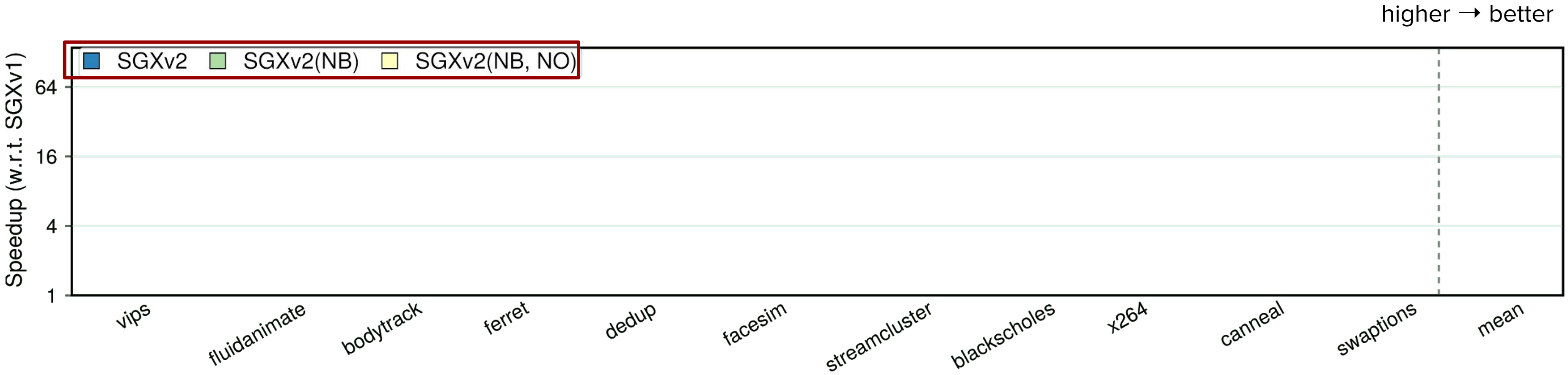
Speedup normalized by the SGXv1 function run time

SGXv2 Optimizations

higher → better



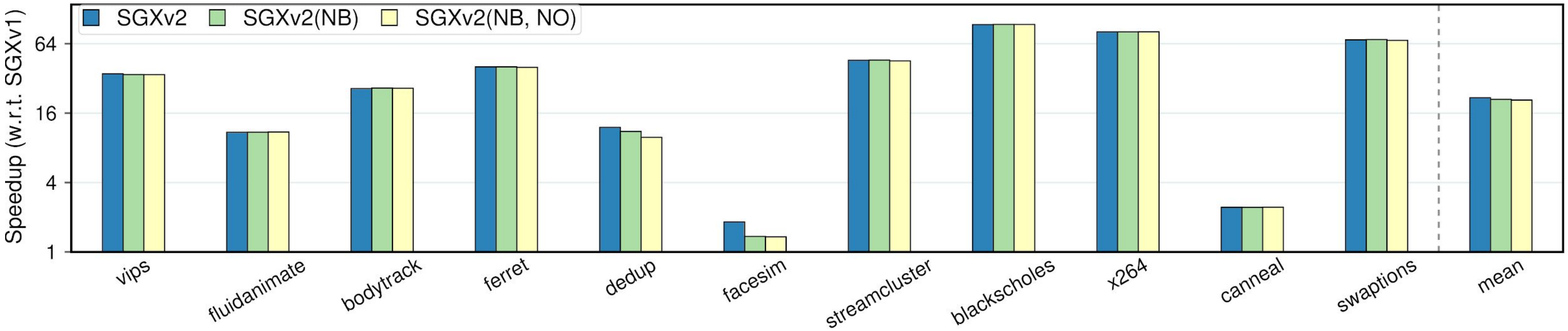
SGXv2 Optimizations



- SGXv2 - all optimizations
- SGXv2(NB) - no batched augmentation
- SGXv2(NB,NO) - no batched augmentation and memory zeroing on deallocation

SGXv2 Optimizations

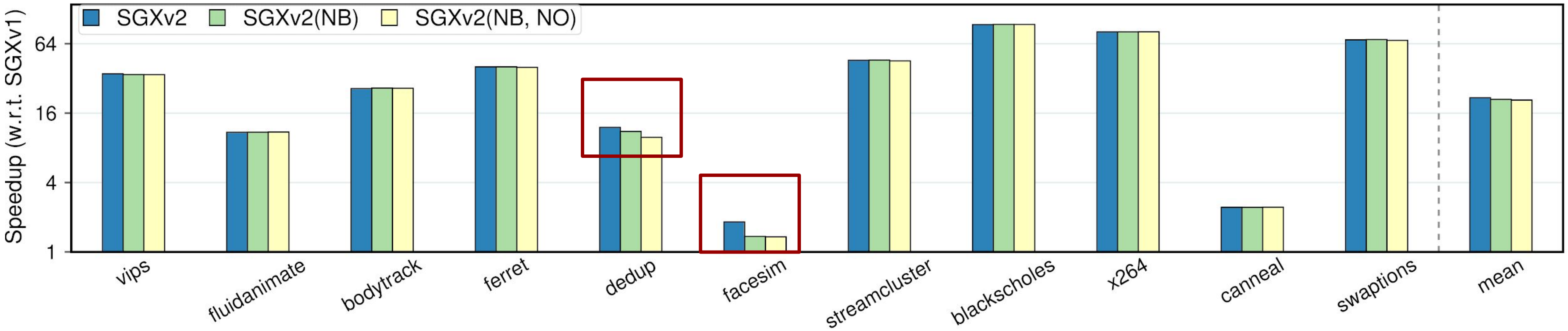
higher → better



- SGXv2 - all optimizations
- SGXv2(NB) - no batched augmentation
- SGXv2(NB,NO) - no batched augmentation and memory zeroing on deallocation

SGXv2 Optimizations

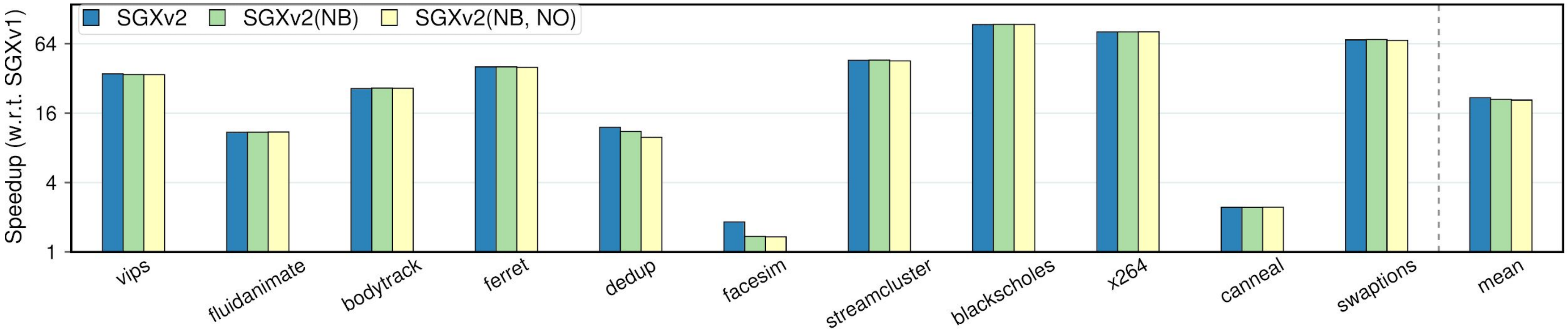
higher → better



- SGXv2 - all optimizations
- SGXv2(NB) - no batched augmentation
- SGXv2(NB,NO) - no batched augmentation and memory zeroing on deallocation

SGXv2 Optimizations

higher ➝ better



10 times lower latency on Phoenix benchmarks with SGXv2
10% lower latency from additional optimizations on a few benchmarks

Summary

Clemmys is:

- **Secure** - protects functions using enclave
- **Fast** - achieves near-native performance
- **Flexible** - does not restrict workloads

Summary

Clemmys is:

- **Secure** - protects functions using enclave
- **Fast** - achieves near-native performance
- **Flexible** - does not restrict workloads

Thank You for your attention!

bohdan.trach@tu-dresden.de

Funding

This project was funded by the European Union’s Horizon 2020 program under grant agreement No. 690588 (Selis), and BMBF No. 03ZZ0517A (FastCloud)