



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К НАЧУНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ
НА ТЕМУ:

Анализ существующих реализаций доверенных сред
исполнения (ТЭЕ)

Студент группы ИУ7-32М

(Подпись, дата)

А. В. Романов

(И.О. Фамилия)

Руководитель НИР

(Подпись, дата)

Д. Е. Бекасов

(И.О. Фамилия)

2023 г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	4
1 Анализ предметной области	5
1.1 Кольца привилегий	5
1.2 Доверенная среда исполнения	6
2 Существующие реализации ДСИ	7
2.1 Intel SGX	7
2.1.1 Processor Reserved Memory	8
2.1.2 Жизненный цикл анклава	9
2.1.3 Проверка целостности	10
2.2 Keystone	10
2.3 ARM TrustZone	10
2.3.1 Обычный и безопасный мир	11
2.3.2 Режимы работы процессора	13
2.3.3 Разделение памяти	14
2.3.4 Проверка целостности	14
3 Сравнение реализаций ДСИ	15
3.1 Критерии сравнения	15
ЗАКЛЮЧЕНИЕ	21
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	22

ВВЕДЕНИЕ

Необходимость повышения безопасности исполнения приложений, работающих в системах безопасности и обрабатывающих защищаемую информацию, привела к разработке программно-аппаратных решений, создающих доверенные среды исполнения (англ. TEE – Trusted Execution Environment [1]) на базе аппаратных средств, доверенных загрузок или аппаратно-программных модулей доверенной загрузки. Intel [2] и ARM [3] являются лидерами в этой области. Целью данной работы является анализ и сравнение существующих реализаций доверенных сред исполнения (ДСИ).

Для достижения поставленной цели необходимо решить следующие задачи:

- провести обзор существующих реализаций ДСИ;
- описать плюсы и недостатки каждой из реализаций;
- сформулировать критерии сравнения;
- сравнить существующие реализации.

1 Анализ предметной области

В этом разделе будут проведен анализ предметной области. Описаны методы обеспечения защиты информации на современных процессорах. Дано понятие и характеристика доверенной среды исполнения.

1.1 Кольца привилегий

В целях безопасности, компоненты любой системы разделены на уровни привилегий – кольца защиты, за реализацию которых отвечает разработчик процессора. Во всех современных системах, реализована кольцевая система уровней привилегий. От внешнего кольца к внутреннему идёт увеличение полномочий для инструкций кода, выполняемых на процессоре в данный момент (рис. 1).

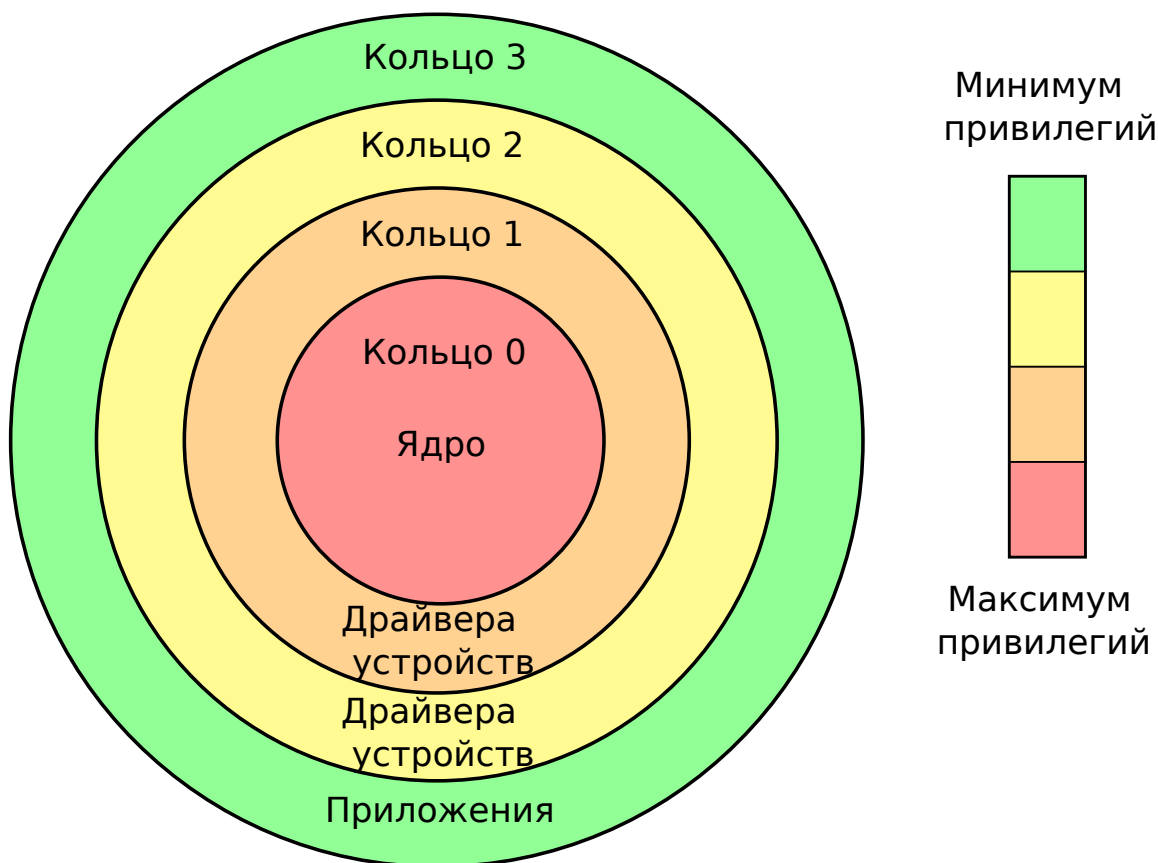


Рисунок 1 – Концептуальная схема представления колец защиты в современных системах

Можно создавать ещё более сложную систему – формировать ещё больше колец защиты, для ограничения каждого из компонентов системы. Однако, чем сложнее архитектура системы и чем больше количества кода в ней, тем проще злоумышленнику найти уязвимость и эксплуатировать её [4].

Главной задачей злоумышленника является получение доступ к привилегиям, которые бы позволили получить доступ к необходимым ресурсам системы. Может показаться, что архитектурно верным является решение размещать код, который отвечает за управление пользовательским данным, и конфиденциальные данные нужно исключительно на последнем кольце защиты – ведь получить доступ туда сложнее всего. Но у такого подхода есть свои недостатки [4]. Данный подход был переосмыслен – в настоящее используется схема, когда и код, и конфиденциальные данные хранятся на одном и том же уровне, что и пользовательские предложения, однако, доступ к ним имеет только лишь процессор. Такой метод защиты информации называется анклавом или доверенной средой исполнения.

1.2 Доверенная среда исполнения

Доверенная среда исполнения (ДСИ) – специальная изолированная область, предоставляемая процессором, которая позволяет вынести из системы часть функциональности приложений и ОС в отдельное окружение, содержащее память и выполняемый код в которой будут недоступны из основной системы, независимо от уровня текущих привилегий. Так, например, в ДСИ выполняется код отвечающих за реализацию различных алгоритмов шифрования, обработки закрытых ключей, паролей, процедур аутентификации и работы с конфиденциальными данными.

В случае, если система была скомпрометирована, информация хранящаяся в ДСИ не может быть определена, и доступ к ней будет ограничен лишь внешним программным интерфейсом. В отличие от других методов защиты информации, таких как например гомоморфное шифрование, аппаратная реализация ДСИ практически не влияет на производительность системы и

уменьшает время разработки программного обеспечения [1].

С другой стороны, аппаратная реализация ДСИ имеет свои недостатки:

- некоторые современные процессоры имеют лишь частичную поддержку ДСИ либо не имеют её вовсе;
- нет возможности программно исправить уязвимость. Найденные уязвимости в реализации ДСИ могут быть исправлены лишь в новых ревизиях процессора, т.е. без его физической замены, злоумышленник сможет эксплуатировать уязвимость.
- увеличивается издержки производства на разработку таких процессоров – их конечная стоимость возрастает.

Таким образом, в некоторых случаях, появляется необходимость в программной реализации ДСИ. Стоит отметить, что в конечном счёте, программная реализация всё равно использует другие аппаратные механизмы предоставляемые процессором [?].

2 Существующие реализации ДСИ

2.1 Intel SGX

Intel Software Guard Extension (SGX) – реализация ДСИ от компании Intel, включена в большинство современных процессоров Intel Core. Конфиденциальность и целостность в этой технологии достигается с помощью использования анклава – специальной, зашифрованной области кода и данных. Это достигается с помощью различных компонентов и протоколов, одним из которых является специальная область памяти, называемая Processor Reserved Memory (PRM), обеспечивающая безопасное хранилище, к которому не может обращаться никто, кроме самого процессора. Для выполнения кода, после череды его проверок, он загружается извне в PRM. Процессор с помощью специальных инструкций переходит в режим анклава (enclave mode) и выполняет загруженный код. В технологии Intel SGX именно анклав является доверенной средой

исполнения.

2.1.1 Processor Reserved Memory

Processor Reserved Memory (PRM) – защищенная часть оперативной памяти, к которой не имеет доступ код, который исполняет в режиме non-enclave. Это реализуется аппаратно, с помощью специальных контроллеров доступа к памяти.

Данный участок памяти подразделяется на дополнительные разделы:

- Encalve Page Cache (EPC) – разделенная страницы размером 4Кб область памяти, которые хранят код конкретного анклава, к которому относятся; это позволяет использовать в системе несколько анклавов одновременно. EPC управляется программным путём с помощью ОС. Доступ к нему может быть получен только программным обеспечением входящим в данный анклав;
- Enclave Page Cache Map (EPCM) – массив с одной записью для каждой страницы, хранимой в EPC; запись содержит в себе метаданные этой страницы: владелец, виртуальный адрес и т. д;
- SGX Enclave Control Structure (SECS) – содержит в себе метаданные соответствующего анклава; хранится в специальной страничной части EPC. Страница, ассоциированная с SECS не отображается в память напрямую и доступна только для реализации SGX – это сделано в целях дополнительной безопасности.

Схема представления памяти с использованием Intel SGX представлена на рисунке 2.

Анклав получает доступ к EPC, выделяя часть своей виртуальной памяти под линейный диапазон адресов анклава (Enclave Linear Address Range, ELRANGE), который содержит адреса сопоставленные с EPC. Другие адреса виртуальной памяти отображаются на память расположенную за пределами EPC. Процессор проверяет что в результате трансляции физического адреса страницы, находящейся в EPC, виртуальный адрес совпадает с адресом храня-

щимся в EPCM – это позволяет предотвратить атаки на трансляцию адресов.

Каждая страница обладают индивидуальными правами доступа, которые устанавливаются при выделении соответствующей страницы и определяются автором анклава, что так же является дополнительной мерой безопасности. Страницы разделены на те, которым разрешено чтение, запись и выполнение кода анклава. Эта информация так же находится в метаданных страницы, хранящихся EPCM.

State Save Area (SSA) – ещё один компонент Intel SGX, отвечающий за сохранение текущего состава анклава. Это специальная область памяти, которая используется для хранения контекста выполнения кода анклава. Эта область памяти необходима, например, когда в системе произошло прерывание – процессору необходимо перейти в нормальный режим исполнения для его обработки. После этого, процессор может загрузить состояние анклава из SSA и возобновить выполнение кода.

Ещё одной особенностью с точки зрения безопасности и производительности, является возможность вытеснения страниц из PRM в обычную (не-PRM) память. Большинство современных компьютеров поддерживают избыточное использование оперативной памяти, выгружая некоторые страницы во вторичные устройства. Технология Intel SGX поддерживает это, добавляя меры, которые гарантируют целостность и конфиденциальность вытесняемых страниц: вытесняемая страницы шифруется с помощью симметричного шифрования, а ключ хранится в специально отведенных для этого страницах EPC.

2.1.2 Жизненный цикл анклава

Для управления анклавами Intel предоставляет набор специальных инструкций:

- ECREATE – создаёт новый анклав и сохраняет его метаданные в SECS;
- EADD – используется для добавления новых страниц в анклав; ОС загружает новые страницы в EPC, заполняя необходимые метаданные. Вызывать эту инструкцию можно только после создания анклава, попытка до-

бавить страницы после этого этапа приведёт к ошибке;

- с помощью инструкции `EINIT` и специального токена от `Launch Enclave`, процессор начинает выполнять код анклава. `Launch Enclave` – специальный анклав, проходящий все те же стадии инициализации, что и другие. Его основная цель является выдача токена другим анклавам на основе списка одобренных анклавов;
- приложения могут выполнить команду `EENTER` для входа в режиме анклава и выполнения там свой код, и по окончании выйти оттуда используя команду `EEEXIT`. В виртуальном адресном пространстве этих приложений должны иметься соответствующие страницы EPC;
- команда `AEX` используется при возникновении прерывания или исключения, после её выполнения процессор переходит в обычный режим исполнения, сохраняя контекст в `SSA`;
- `ERESUME` – возобновить выполнение анклава с контекстом, сохраненным в `SSA`. Стоит отметить, что анклав может иметь более одного `SSA` в случаях, если при выполнении одного и того же блока происходит несколько прерываний.

После выполнения кода, в метаданных страниц, ассоциированных с этим анклавом, выставляется пометка, что они невалидны и очищается TLB кэш. Это позволяет защитить Intel SGX от атак на память. На рисунке 3 представлена схема жизненного цикла анклава с использованием команд, описанных выше.

2.1.3 Проверка целостности

2.2 Keystone

RISC V

2.3 ARM TrustZone

ARM TrustZone – технология аппаратного обеспечения ДСИ, разрабатываемая компанией ARM. Большинство процессоров разработанных ARM имеют поддержку TrustZone [5]. Данная технология основана на разделении режимов работы процессора на два ”мира”: обычный мир (Normal World) и без-

опасный мир (Secure World). Процессор переключается в безопасный мир по запросу (с помощью специальной инструкции), при работе с конфиденциальными данными. Всё остальное время, процессор работает в режиме обычного мира. Процессоры с поддержкой данной технологии имеют способность разделять память, независимо от её типа, на ту, которая доступна только в безопасном мире, и ту, которую можно использовать в обычном мире. ARM предоставляют открытый исход программного обеспечения для поддержки данной аппаратной технологии – ARM Trusted Firmware [6].

2.3.1 Обычный и безопасный мир

Ключевой особенностью ARM TrustZone является способность процессора переключаться между обычным и безопасным миром. Каждый из этих миров управляется собственной операционной системой, которые обеспечивают необходимую функциональность. Основное различие между этими ОС заключается в предоставляемых гарантиях безопасности. В один момент времени, процессор может находиться только в одном из двух миров, что определяется значением специального бита NS (Non-Secure), бит является частью регистра Secure Configuration Register (SCR). Этот регистр доступен для периферии только для чтения, изменять его значение может лишь сам процессор. Когда процессор находится в обычном режиме исполнения кода, значение бита NS равно 1, и наоборот, когда процессор находится в безопасном мире, значение бита NS равно 0.

За связь между обычным и безопасным миром отвечает специальный механизм – Secure Monitor. Он соединяет оба мира и является единственной точкой входа в безопасный мир. Для того, чтобы из обычного мира перейти в безопасный, существует специальная инструкция процессоров ARM – Secure Monitor Call (SMC). При вызове данной инструкции процессор передает управление Secure Monitor. Тот в свою очередь готовит систему к переходу из одного мира в другой и передает управление соответствующей ОС. Инструкция SMC используется как для перехода из нормального мира в безопасный, так и для перехода

из безопасного в нормальный. Некоторые прерывания или исключения могут быть настроены так, чтобы они так же проходили через Secure Monitor и были обработаны в безопасном мире. ARM предоставляет спецификацию Secure Monitor Call Calling Convention (SMCCC) [7], которая является стандартом при реализации вызовов SMC.

На рисунке 4 представлена цонепутальная схема взаимодействия двух миров.

Доверенная операционная система, так же как и обычная, может запускать приложения. Они, как и в обычном мире, обращаются к доверенной ОС при необходимости получения каких-либо ресурсов или обработки прерываний и исключений. Таким образом, Secure Monitor передаёт управление одному из доверенных приложений, и уже те, в свою очередь, обращаются к доверенной ОС.

ARM предоставляют открытый исходной код эталонной доверенной операционной системы, которая называется OP-TEE [8]. Global Platform предоставляет спецификацию для реализации API взаимодействия доверенных приложений [9] с доверенной ОС [10].

Физически миры разделены таким образом, что часть регистров доступны только в безопасном мире. Периферия, например память, может быть настроена так, что она может быть доступна лишь в определенном мире. Технология TrustZone в нормальном режиме работы процессора не позволяет программному обеспечению получить доступ к аппаратным средствам, которые могут быть доступны лишь только в безопасном мире.

При сборке компонентов системы, производитель устройства должен позаботиться о конфигурации периферии для работы с TrustZone:

- если предполагается, что периферия может получать доступ к безопасному режиму исполнения, процессор и внешнее устройство должны быть соединены (помимо различных шин) линией NS. Получение сигнал NS=0 от процессора к периферии означает, что команда является доверенной

(например операция чтения или записи).

- в обратном случае, линия NS может быть опущена. Предполагается, что такая периферия не имеет никаких привилегий, т.е. NS=1 всегда.

На рисунке 5 представлена схема взаимодействия процессора и периферии для поддержки TrustZone. Периферийные устройства №1 и №3 соединены линией NS, а устройство №2 нет.

Стоит отметить, что чаще всего не вся периферия соединена сигналом NS с процессором. Например, тот факт, что производитель устройства не соединил сигналом NS камеру и ядра процессора, полностью исключает возможность предоставления пользователю доступа к устройству с помощью технологии распознавания лица.

2.3.2 Режимы работы процессора

Современная архитектура ARM поддерживается три режима работы процессора:

- EL0 – непривилегированный режим работы, предназначенный для исполнения обычных программ;
- EL1 – привилегированный режим работы – исполняется кода ОС, обработчиков прерываний и исключений;
- EL2 – режим работы гипервизора.

Для того, чтобы программа исполняемая на уровне EL0 могла перейти в EL1 (например, обратиться к ресурсам доступным только ОС), в архитектуре ARM существует команда Supervisor Call (SVC). Аналогично, команда Hypervisor Call (HVC) предназначена для перехода из режима EL1 в EL2.

Каждый из этих уровней могут исполняться в нормальных (Non-Secure) так и безопасных (Secure) режимах (рис. 6)

- обычные приложения исполняются на уровне Non-Secure EL0, а приложения доверенной ОС на Secure EL0;
- обычная ОС выполняется на уровне Non-Secure EL1; все прерывания и исключения произошедшие в нормальном мире так же обрабатываются

на этом уровне; доверенная ОС и прерывания произошедшие в безопасном мире исполняются на уровне Secure EL1;

- Secure Monitor всегда исполняется на уровне Secure EL1;
- гипервизор исполняется в режиме Non-Secure EL2.

Бит NS, который был описан в предыдущей главе, определяет то, в каком режиме сейчас исполняется код: обычном (NS=0) или безопасном (NS=1).

Благодаря дублированию нормального и безопасного режима архитектуры ARM позволяет запустить сразу две ОС: обычную и доверенную.

2.3.3 Разделение памяти

В целях безопасности в ARM TrustZone память разделяется на защищенную и незащищенную область. Благодаря контроллеру адресного пространства (TZASC – TrustZone Address Space Controller) незащищенная область памяти может использоваться только из обычного мира, а защищенная из безопасного. Кэш памяти так же разделяется на защищенную и незащищенную. Необходимо отметить, что данный контроллер не является обязательным в архитектуры ARM TrustZone, поэтому разработчики могут принять решение отказаться от них в пользу более компактных и менее энергоемких устройств.

2.3.4 Проверка целостности

Проверка целостности ДСИ является важным механизмом, который не позволит злоумышленнику изменить исходный код доверенной ОС или доверенных приложений. В ARM TrustZone данный механизм реализован на аппаратном уровне как для ОС, так и для приложений: каждый раз, при загрузке доверенной ОС в память, с помощью цифровой подписи проверяются её целостность. Аналогичная схема используется и при загрузке доверенных приложений. Запустить ОС может только подписанные приложения, подпись формируется разработчиками, на стадии компиляции в исполняемый файл.

На данный момент, доверенная среда исполнения от компании ARM не поддерживает процедуру удалённой проверенной проверки (например, с помощью сервера). Существуют лишь программные решения этой проблемы от сто-

ронных разработчиков [5].

3 Сравнение реализаций ДСИ

3.1 Критерии сравнения

Для сравнения ранее описанных реализаций ДСИ были выделены следующие критерии:

- К1 - безопасность;
- К2 - производительность;
- К3 - надежность механизма аттестации ДСИ;
- К4 - наличие открытого исходного кода.

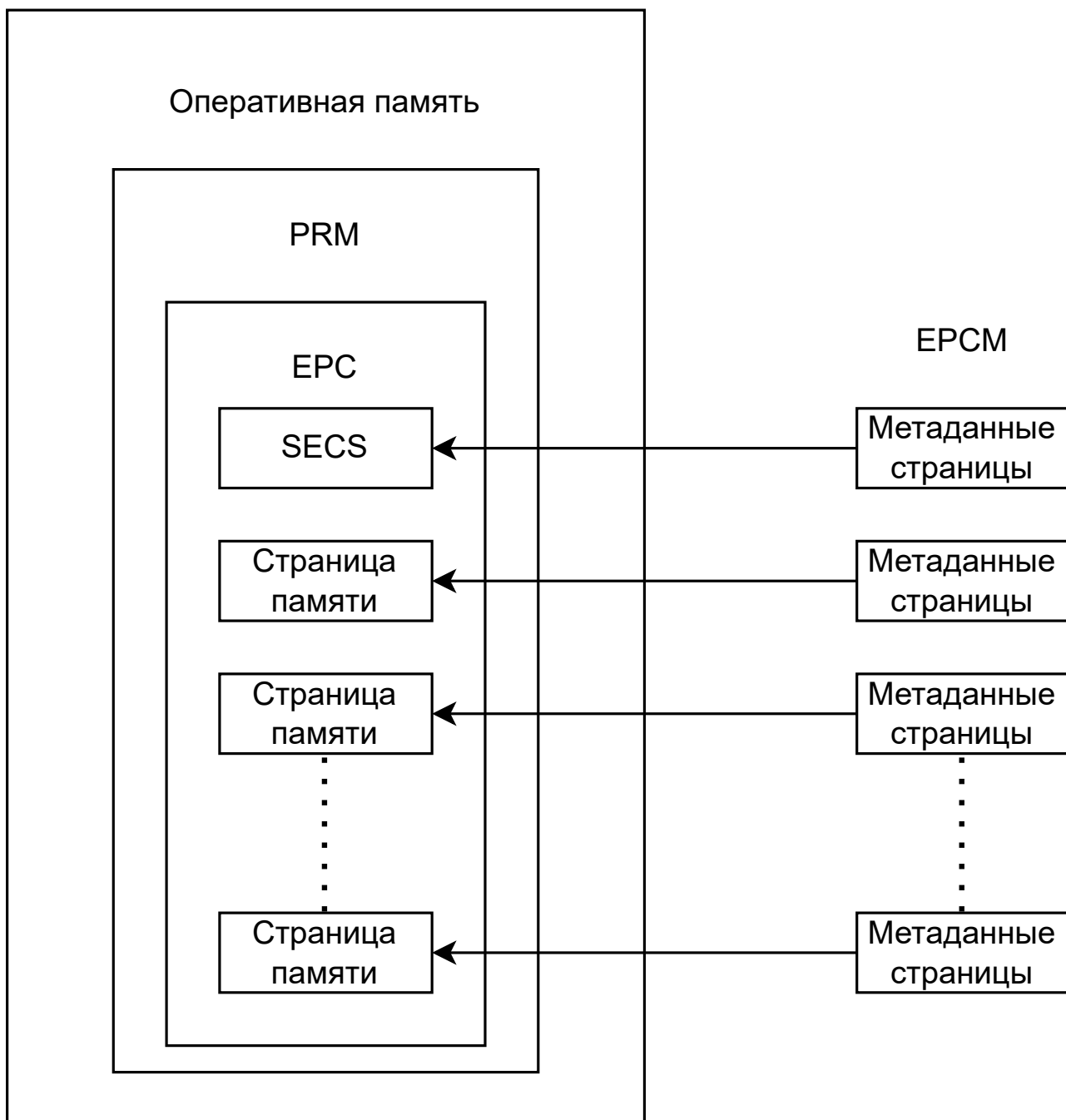


Рисунок 2 – Схема представления памяти Intel SGX

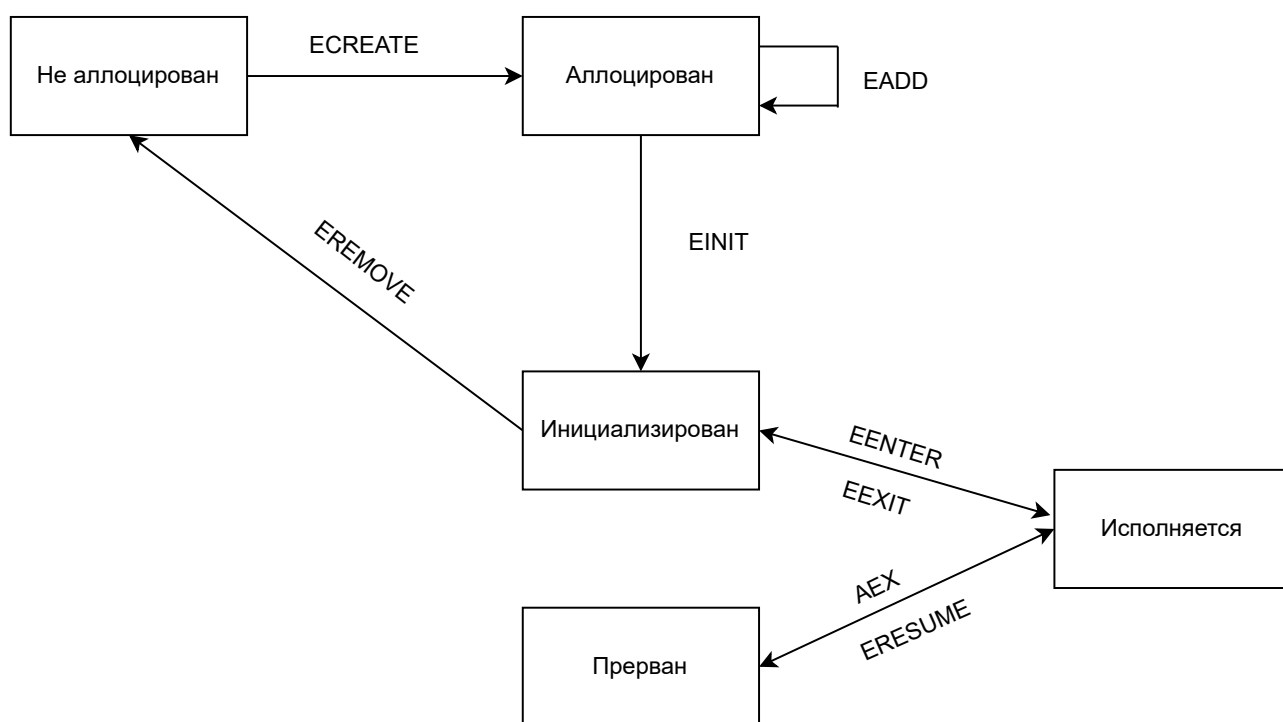


Рисунок 3 – Жизненный цикл анклава

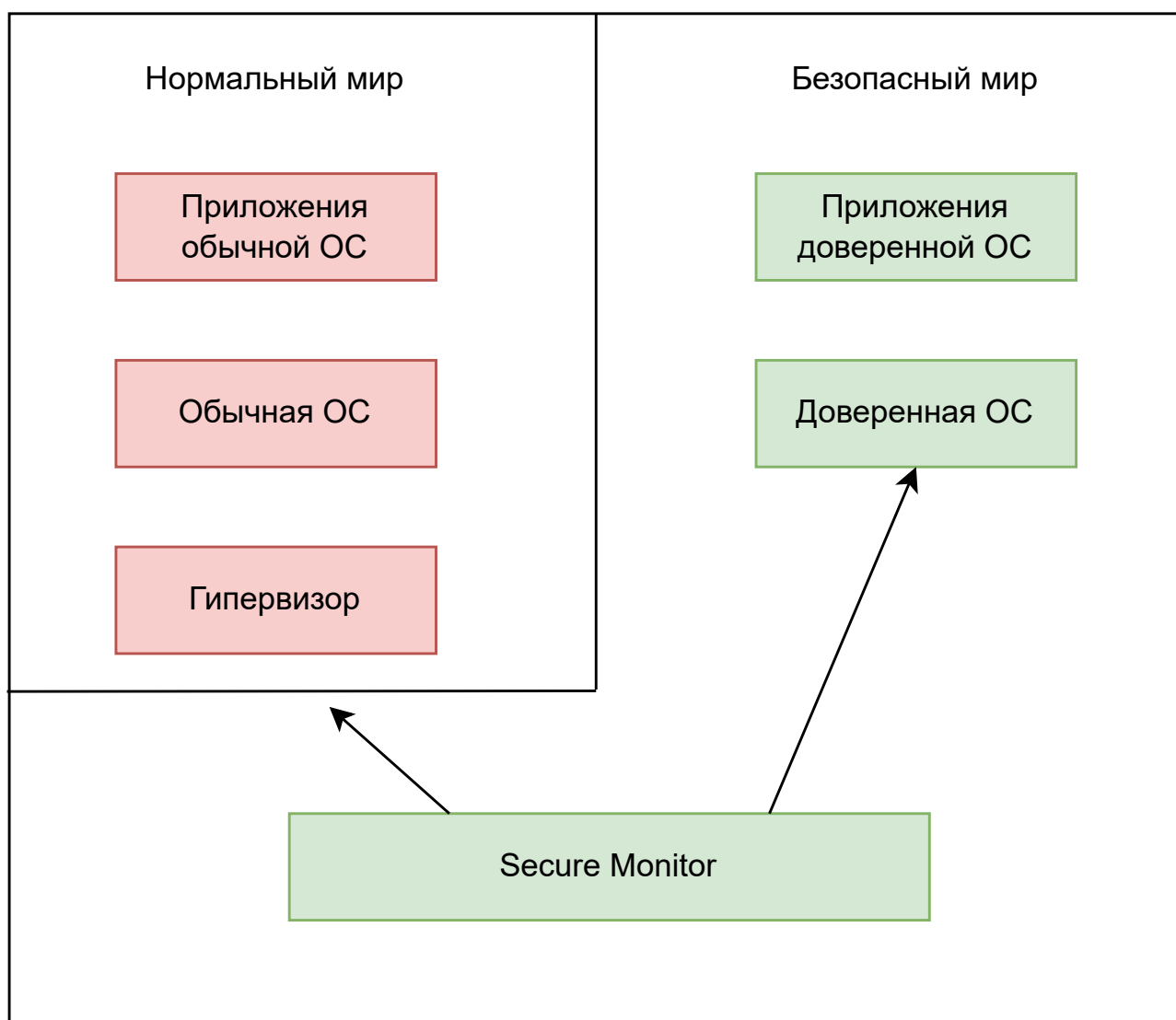


Рисунок 4 – Концептуальная схема взаимодействия двух миров для процессоров ARM Cotrex-A

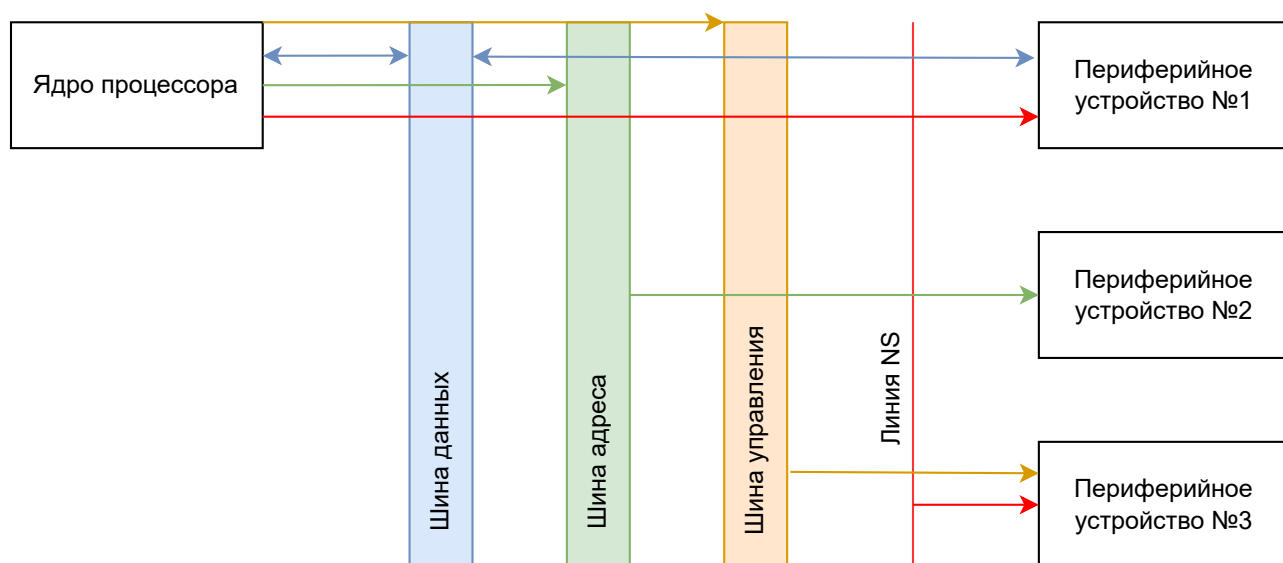


Рисунок 5 – Пример взаимодействия процессора и периферии для поддержки TrustZone

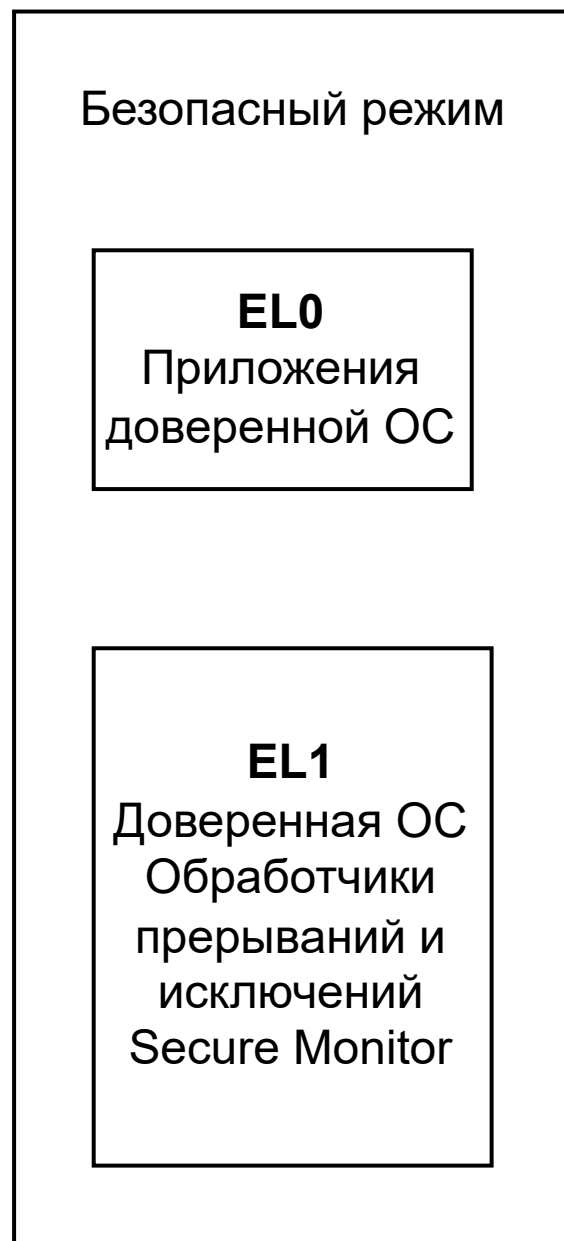
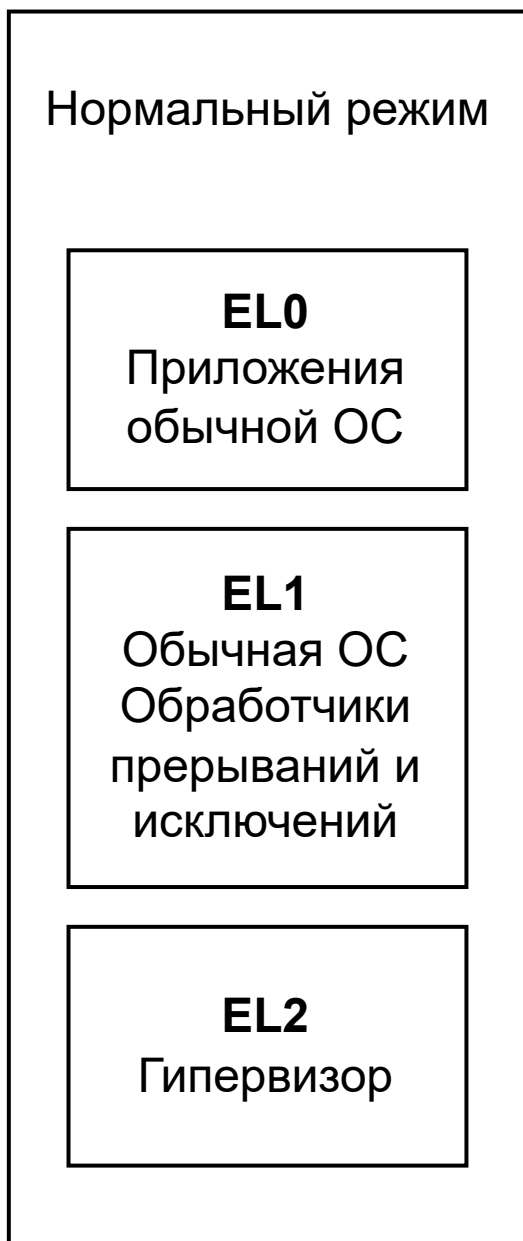


Рисунок 6 – Режимы работы процессоров архитектуры ARM

ЗАКЛЮЧЕНИЕ

В ходе выполнения научно исследовательский работы была достигнута ее цель – проведен анализ и сравнение существующих реализаций ДСИ.

Для достижения данной цели были решены следующие задачи:

- проведён обзор существующих реализаций ДСИ;
- описаны плюсы и недостатки каждой из реализаций;
- сформулированы критерии сравнения;
- проведено сравнение существующих реализации.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Introduction to Trusted Execution Environments – Global Platform [Электронный ресурс]. – Режим доступа: <https://globalplatform.org/wp-content/uploads/2018/05/Introduction-to-Trusted-Execution-Environment-15May2018.pdf>, свободный – (09.10.2023)
2. Intel | Data Center Solutions, IoT, and PC Innovation [Электронный ресурс]. – Режим доступа: <https://www.intel.com/>, свободный – (10.11.2022)
3. Building the Future of Computing – Arm® [Электронный ресурс]. – Режим доступа: <https://www.arm.com>, свободный – (09.10.2022)
4. The Security Paradox of Complex Systems, 2002. David Woods, Nancy Leveson, Brian Rebentisch. с. 1 - 15.
5. Comparison of Prominent Trusted Execution Environments, 2022. Xiaoyu Zhang [Электронный ресурс]. – Режим доступа: https://elib.uni-stuttgart.de/bitstream/11682/12171/1/Zhang_Xiaoyu_Prominent_Trusted_Execution_E – (22.10.2022)
6. TrustedFirmware-A (TF-A) | ARM [Электронный ресурс]. – Режим доступа: <https://www.trustedfirmware.org/projects/tf-a/> – (22.10.2022)
7. Secure Monitor Calling Convention (TF-A) | ARM [Электронный ресурс]. – Режим доступа: <https://developer.arm.com/Architectures/SMCCC> – (22.10.2022)
8. OP-TEE | ARM [Электронный ресурс]. – Режим доступа: <https://www.trustedfirmware.org/projects/op-tee/> – (22.10.2022)
9. Global Platform – TEE Client API Specification [Электронный ресурс]. – Режим доступа: https://globalplatform.org/wp-content/uploads/2010/07/TEE_Client_API_Specification-V1.0.pdf – (22.10.2022)

10. Global Platform – TEE Internal Core API Specification [Электронный ресурс].
– Режим доступа: <https://globalplatform.org/specs-library/tee-internal-core-api-specification/> – (22.10.2022)