



Как мы Grab'нули проблем при embedded разработке

16 апреля 2024



Кузьмичева Ольга

Разработчик встраиваемых систем



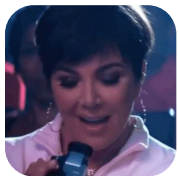
Кирпичников Антон

Разработчик встраиваемых систем

Содержание

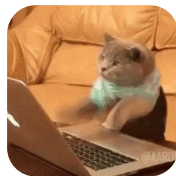
- Стандарт обеспечения подлинности данных
- Как мы писали драйвер крипточипа для видеокамеры
- На какие грабли мы наступили
- Наш подход к юнит-тестированию кода устройств

О проекте



Устройство

Видеокамера, фиксирующая подлинность произведенного контента



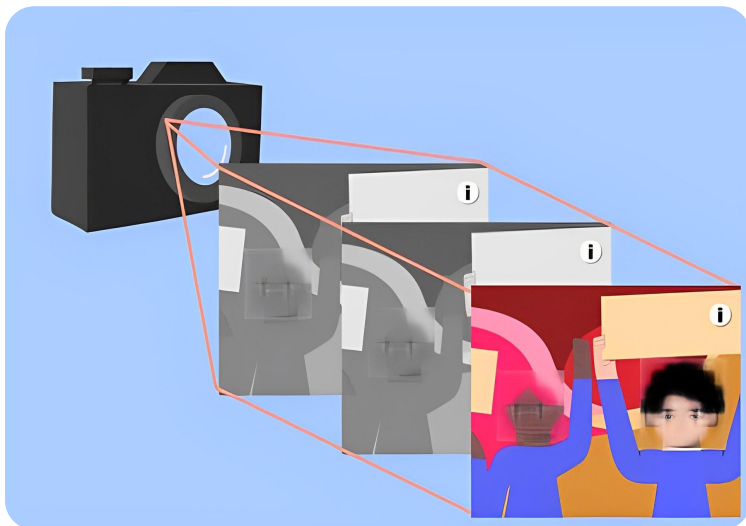
Мы занимались

Написанием драйвера для чипа безопасности, обеспечивающего подлинность контента



Подлинность контента

Важно доказать, что события, которые происходили у меня на глазах, были именно такими, какими я их видел.



Стандарт, использующий метаданные и криптографические методы, для предоставления проверяемой информации о происхождении, истории и модификациях части контента.



<https://c2pa.org>

C2PA Membership

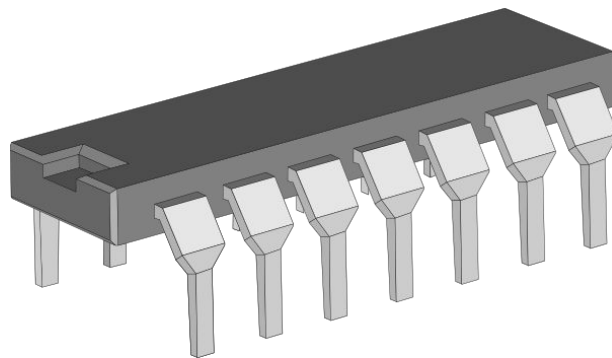
The logo for BBC, consisting of the letters 'B', 'B', and 'C' in white, each inside a black square, arranged horizontally.The Adobe logo, featuring a stylized red 'A' icon followed by the word 'Adobe' in red text.The Google logo, with the word 'Google' in its characteristic multi-colored font (blue, red, yellow, green, blue, red).The Intel logo, with the word 'intel' in a lowercase, bold, black sans-serif font, featuring a small blue square above the 'i'.The Microsoft logo, featuring the four-pane Windows icon (red, green, blue, yellow) followed by the word 'Microsoft' in a grey sans-serif font.The Sony logo, with the word 'SONY' in a bold, blue, uppercase sans-serif font.

Security controller

Цель: Подпись контента

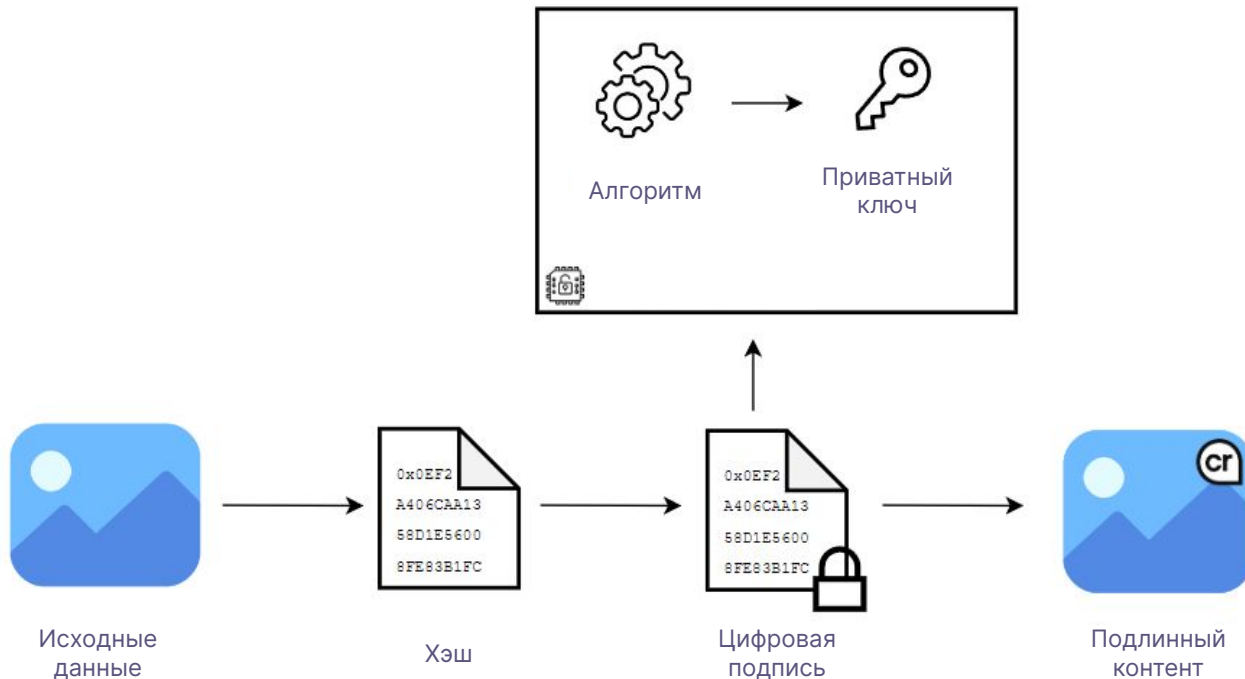
Описание

- Надежное хранение приватного ключа
- Поддержка различных криптоалгоритмов (ECC, RSA, AES, TLS)



Как это работает

Создание подлинного контента



Как это работает

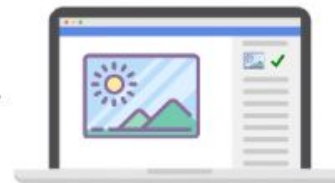
Проверка подлинного контента



Подлинный
контент



Публичный
ключ



Верификация
контента

content credentials

Выберите другой файл на своем устройстве или перетащите куда угодно



Поиск возможных совпадений ⓘ



Sep 24, 2023



Jun 18, 2023



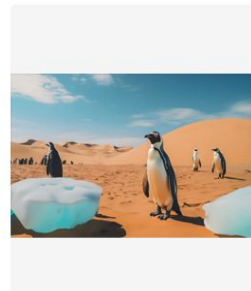
Oct 1, 2023



Oct 4, 2023

max_DSC7386-16x9 gen ai.jpg

5 окт. 2023 г.



Сводная информация о контенте

Этот изображение состоит из нескольких фрагментов контента, минимум один из которых сгенерирован с помощью инструмента или алгоритма на основе ИИ.

Благодарность и использование

Автор поделился следующей информацией:

Кем создано

obidigbo nzeribe

Учетные записи в социальных сетях

obidinzeribe
Instagram

obidigbo nzeribe
Behance



<https://contentcredentials.org/verify>

Изменить язык ▾



Сравнить

Как была построена работа

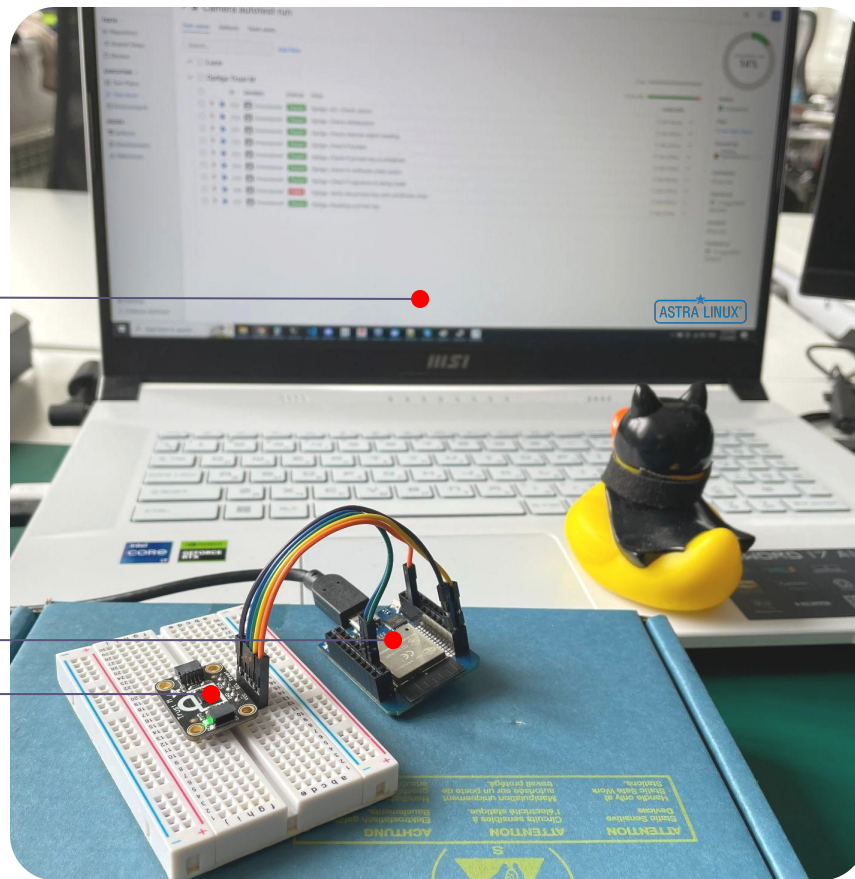
- 1) Изучение документации
- 2) Разработка драйвера
- 3) Внедрение в устройство
- 4) Полноценное тестирование
- 5) Исправление багов

С самого начала у нас была какая-то тактика, и мы ее придерживались



Тестовый стенд

- ПК
- Контроллер ESP32
- Security chip



Наши приоритеты тестирования

Интеграционное тестирование

Тестирование безопасности

Нагрузочное тестирование

Приемочное тестирование

Интеграционное тестирование

← ⚙️ Security videocam autotest run

Test cases Defects Team stats

Search... Status: Passed × Add filter

✓ Crypto chip status 🕒 4m 57s

<input type="checkbox"/>	ID	MEMBER	STATUS	TITLE	DURATION
<input type="checkbox"/>	↑ ⚙️ 199	🟢 Ilya Sapronov	Passed	I2C. Check status	🕒 29s 771ms ...
<input type="checkbox"/>	↑ ⚙️ 200	🟢 Ilya Sapronov	Passed	Check initialization	🕒 29s 755ms ...
<input type="checkbox"/>	↑ ⚙️ 201	🟢 Ilya Sapronov	Passed	Check internal object reading	🕒 29s 781ms ...
<input type="checkbox"/>	↑ ⚙️ 202	🟢 Ilya Sapronov	Passed	Check if locked	🕒 29s 766ms ...
<input type="checkbox"/>	↑ ⚙️ 203	🟢 Ilya Sapronov	Passed	Check if private key is initialized	🕒 29s 784ms ...
<input type="checkbox"/>	↑ ⚙️ 204	🟢 Ilya Sapronov	Passed	Check if cetificate chain exists	🕒 29s 790ms ...
<input type="checkbox"/>	↑ ⚙️ 205	🟢 Ilya Sapronov	Passed	Check if signature is being made	🕒 29s 780ms ...
<input type="checkbox"/>	↑ ⚙️ 206	🟢 Ilya Sapronov	Passed	Verify the private key and certificate chain	🕒 29s 801ms ...
<input type="checkbox"/>	↑ ⚙️ 207	🟢 Ilya Sapronov	Passed	Reading a private key	🕒 29s 790ms ...

Completion rate
15%

Status
✔️ Completed

Plan
[Cover test cases](#)

Started by
Dmitriy Starodubtcev (inactive)

Estimation
🕒 23m 22s

Started at
📅 21 Aug 2023 11:12:53

Duration
🕒 4m 57s

Finished at
📅 21 Aug 2023 11:12:54

Наши приоритеты тестирования

Нагрузочное тестирование

```
while(1)
{
    crypto_chip.send_message("sign_data:data1");
    script.msleep(1);
    crypto_chip.send_message("sign_data:data2");
    script.msleep(1);
    crypto_chip.send_message("sign_data:data3");
    script.msleep(1);
    crypto_chip.send_message("sign_data:data4");
    script.msleep(1);
}
```

Тестирование безопасности

300

Security. Brute Force

[General](#) [Properties](#) [Runs](#) [History](#) [Defects](#)

Pre-conditions
Chip is initialized

Steps

- 1 Enter command to toolbox command line

Input data

```
send_message("brute_force;100000")
```

Expected result
Logs with 100000 messages "Wrong password"

301

Security. Change Password

[General](#) [Properties](#) [Runs](#) [History](#) [Defects](#)

Pre-conditions
Chip is initialized

Steps

- 1 Enter command to toolbox command line

Input data

```
send_message("write_password_object;lololo")
```

Expected result
Message "Changing this object is not allowed"

302

Security. Sniff password

[General](#) [Properties](#) [Runs](#) [History](#) [Defects](#)

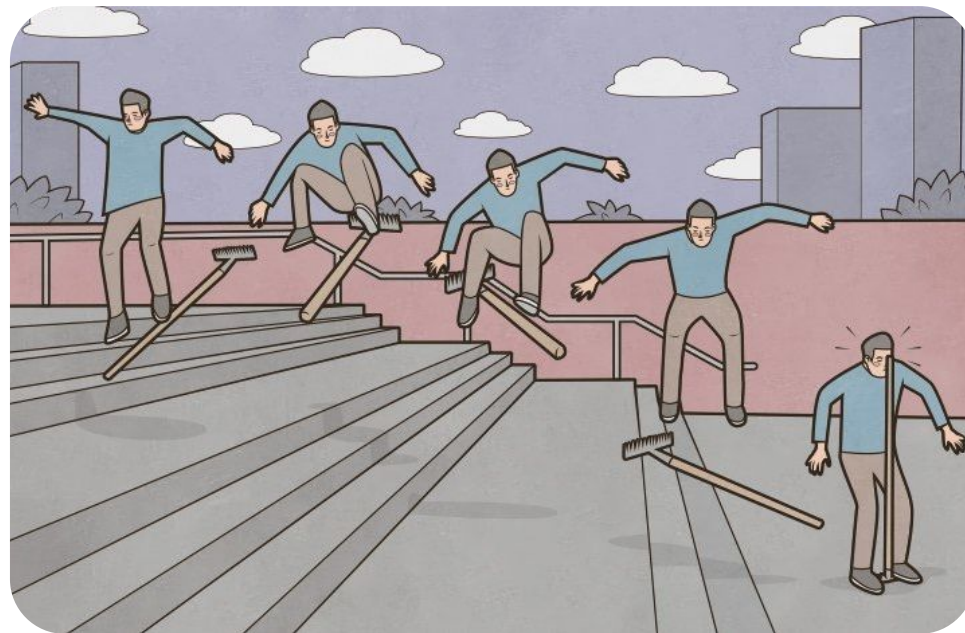
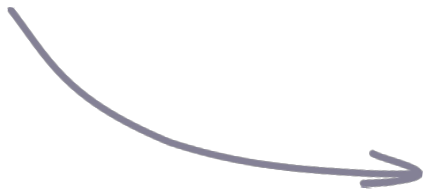
Pre-conditions
Chip is initialized

Steps

- 1 Solder to the I2C bus
- 2 Start signing
- 3 Process
- 4 Search for password in raw data

Expected result
Password was not found in raw data

Коротко о нашей разработке



Наши приоритеты тестирования

Приемочное тестирование

content credentials

[Выберите другой файл на своем устройстве](#) или перетащите куда угодно

f5ba0ad3-7e5f-4f67-b909-ea6ff1...
12 февр. 2023 г.

12 февр. 2023 г.

Процесс

Данные, зарегистрированные приложением или устройством, с помощью которого создан этот контент:

Использованное приложение или устройство

Truepic Lens SDK libc2pa C++ Library 2.5.1

Сведения о параметрах захвата камеры

Дополнительные данные, полученные с камеры, с помощью которой было снято изображение или видео. Данные EXIF могут редактироваться производителем контента.

Дата съемки

12 февр. 2023 г. в 23:40 GMT+5

Изменить язык

Fit

Сравнить

Автоматизация приемочных тестов

Проверка метаданных
на устройстве с помощью
сторонних библиотек

mbedTLS

openSSL

Генерация контента
и его валидация
вне устройства

cypress

Особенности нашего проекта

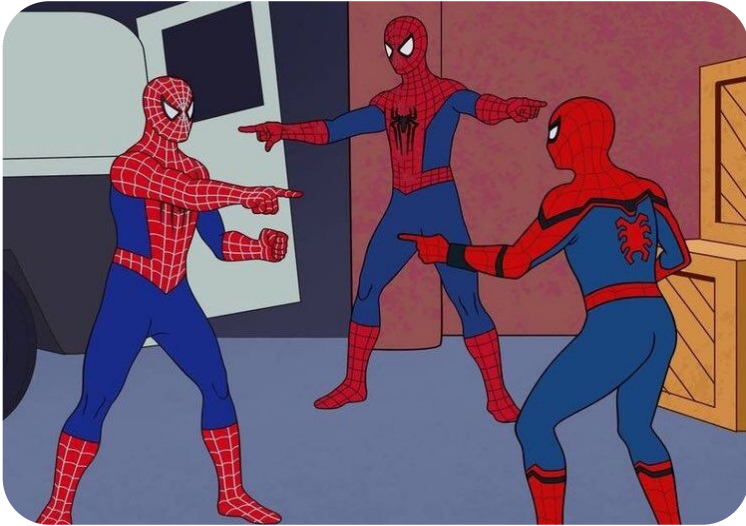
Чистый C

Криптография

Много Legacy

Real Time Operating System

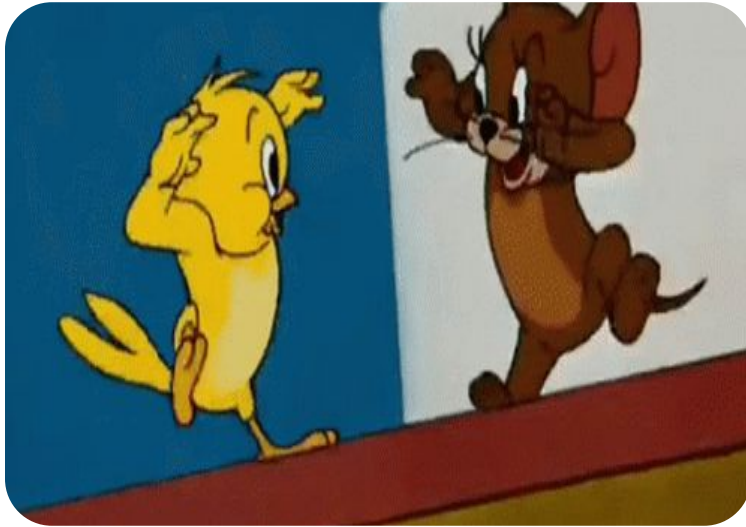
Грабля #1



А кто будет тестировать?

- Отсутствие embedded специализации у тестировщиков
- Разработчики вынуждены сами заниматься тестированием

Грабля #1: как не наступить



Тестировщики

Разработчики

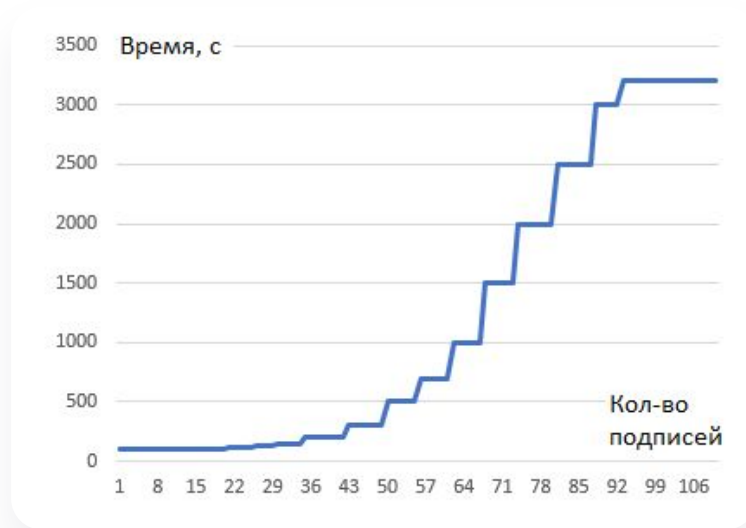
Вынужденное тесное сотрудничество


- Повышение квалификации тестировщиков
- Разработка проходит с мыслями о тестировании

Грабля #2

Почему так медленно?

То, что хорошо работает при низкой нагрузке, может сломаться при частых запросах.





Подозрительно
частые запросы.
Буду отвечать
медленнее.

Я не хакер,
клянусь..

Грабля #2: как не наступить

Своевременное нагрузочное тестирование

Поможет выявить неочевидные
детали работы устройства!



Грабля #3

Дыра в безопасности

Кажется мы предусмотрели всё...
...или нет?



Обращение к «белым хакерам»

- Свежий взгляд на безопасность проекта
- Команда с опытом взлома устройств



Грабля #3: итог теста безопасности

Все хорошо!

“

”

Пароль не удалось обнаружить
в eMMC памяти

”

“

Грабля #3: итог теста безопасности

Или нет..



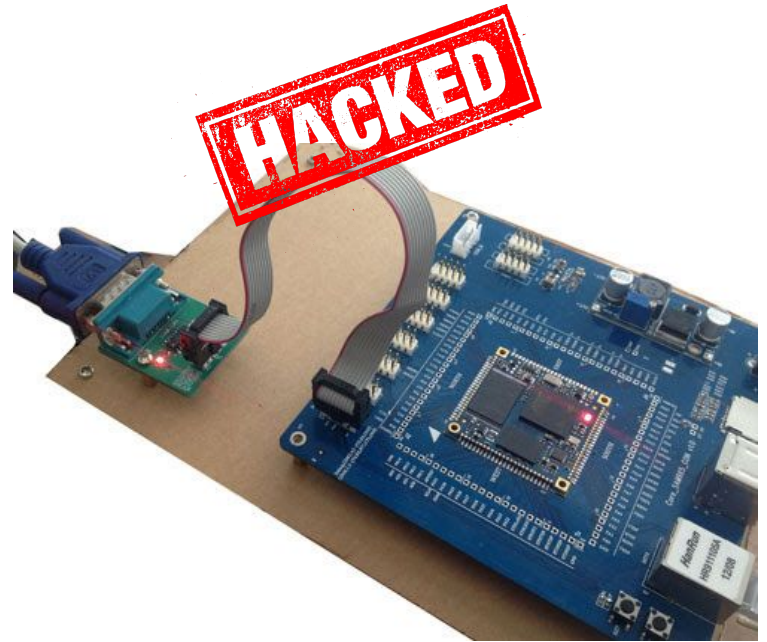
Открытый дебаг интерфейс



Незашифрованная память



Отладочная информация



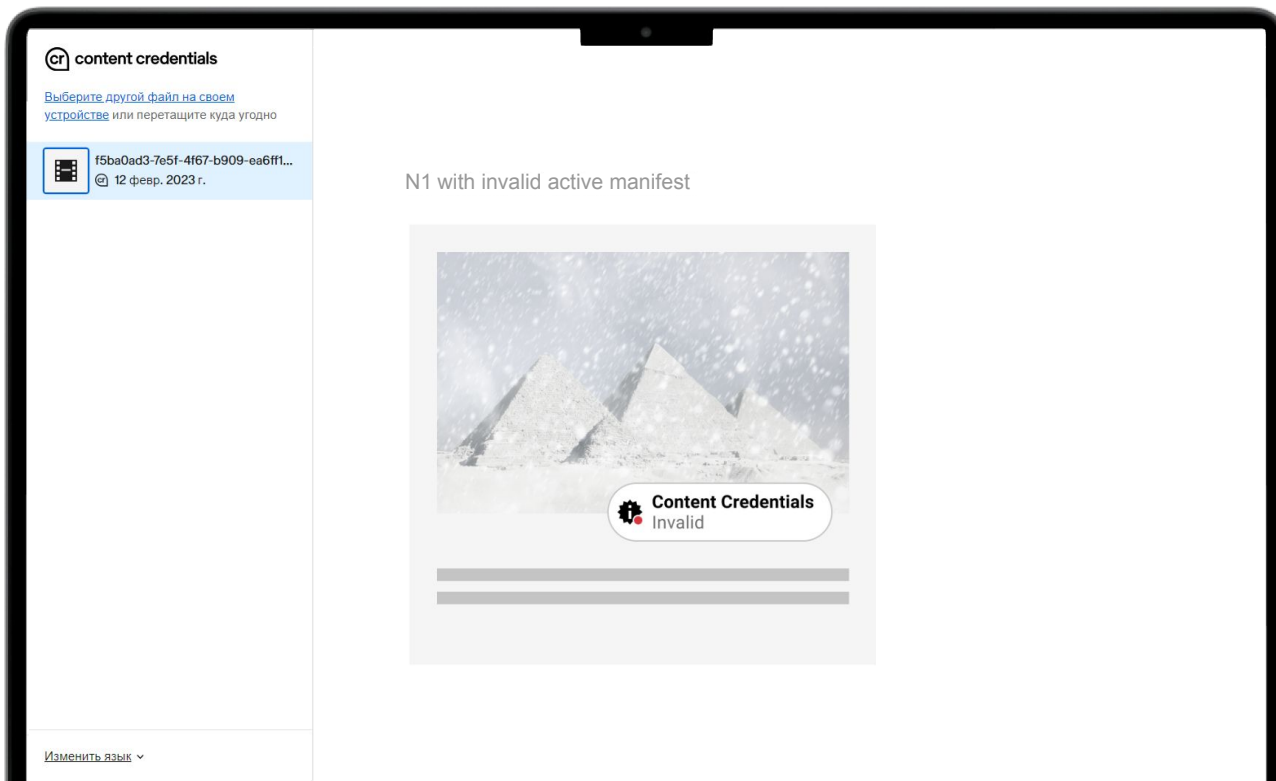
Грабля #4

Принес домой – а оно не работает

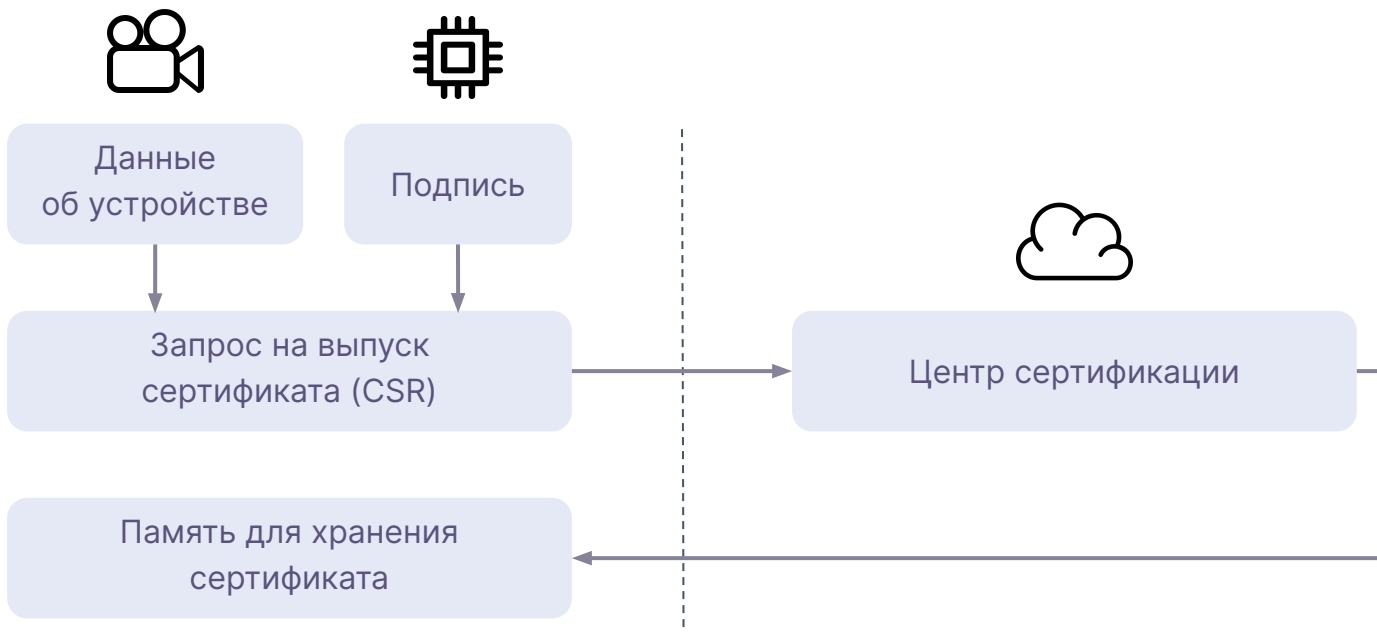
- Крипчип был не инициализирован
- Отсутствует сертификат
- Неверный сертификат



Сломанная верификация



Не всем сертификатам можно доверять



Грабля #5

Как мы чуть не потеряли
много 

Некоторые изменения конфигурации
могут оказаться необратимыми.



Грабля #4, #5: как не наступить

Тестирование на производстве

- Скрипт проверки валидной работы крипточипа
- Приемочные тесты



Грабля #6

Third-party библиотеки

У любой Third-party библиотеки
могут проявиться особенности работы
на target платформе.

.[RegEx]*

wolfSSL

Грабля #6: как не наступить

Тестирование third-party библиотек

Отдельное тестирование
используемого функционала third-party
библиотек.



Грабля #7

Отсутствие юнит тестов



When you finally find the person
who wrote all the bad code

Грабля #7: проблема

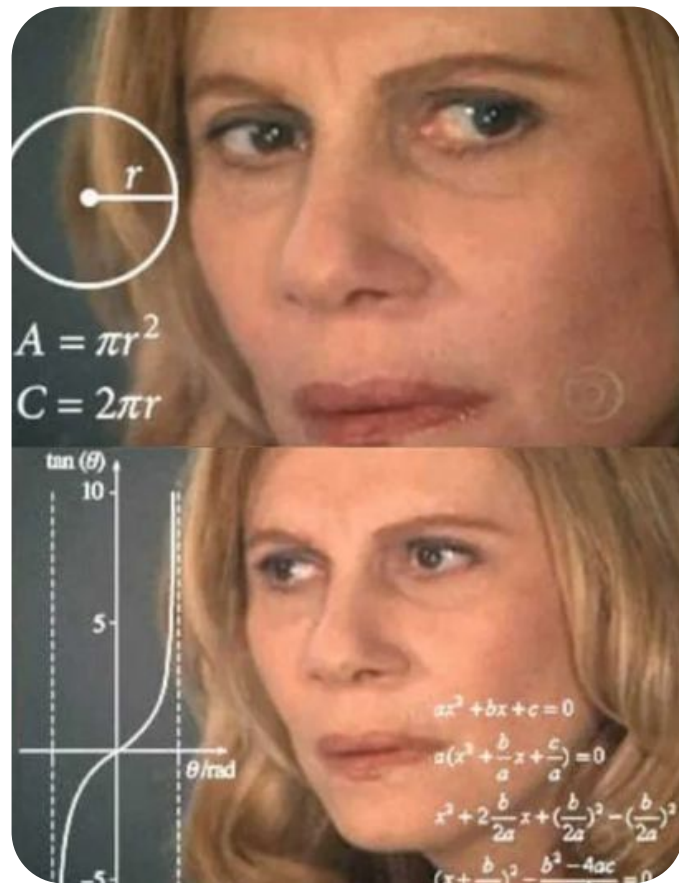
Почему писать юнит тесты сложно

- 1) Вычисления в реальном времени
- 2) Зависимость от аппаратного обеспечения

Грабля #7: как не наступить

Покрыть тестами весь код

А есть ли способ покрыть юнит тестами
как software, так и hardware код?



Два решения

- Тестирование в целевой среде
- Двойное тестирование



<https://www.amazon.com/Driven-Development-Embedded-Pragmatic-Programmers/dp/193435662X>

The
Pragmatic
Programmers

Test-Driven Development for Embedded C

James W. Grenning

Forewords by Jack Ganssle
and Robert C. Martin

Edited by Jacquelyn Carter



Первое решение

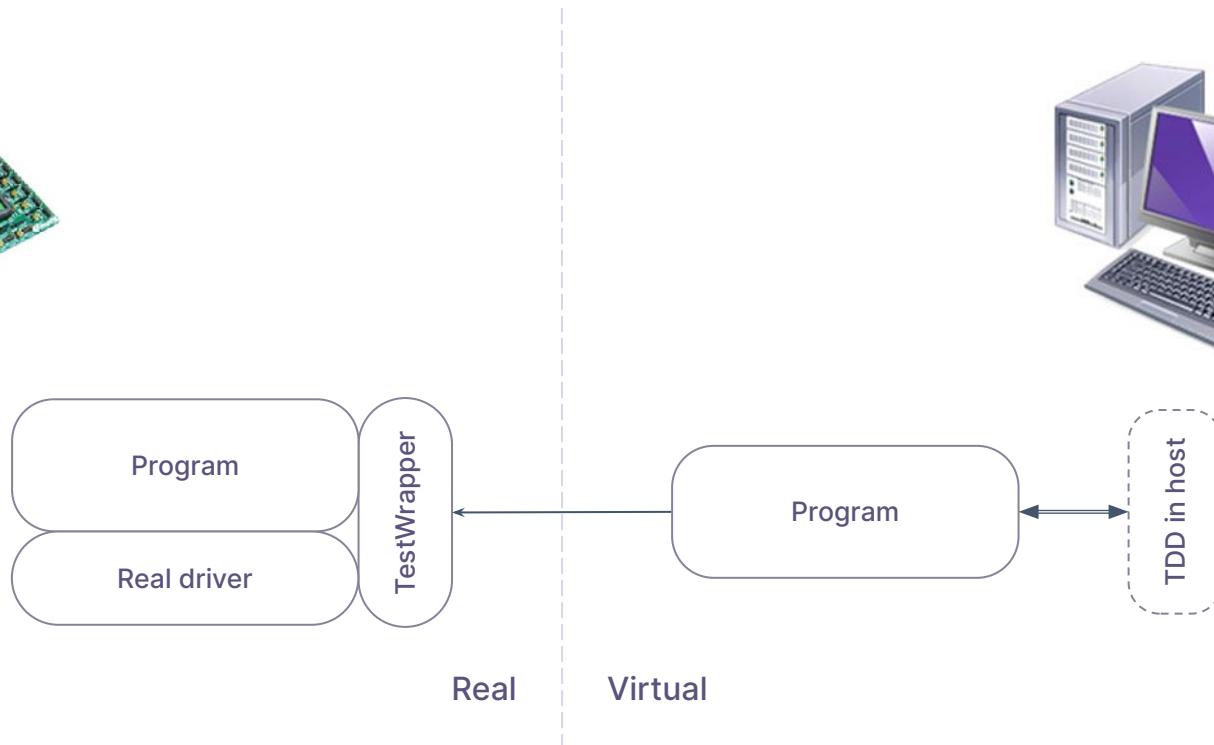
Тестирование в целевой среде



target



host



Тестирование в целевой среде

Преимущества:

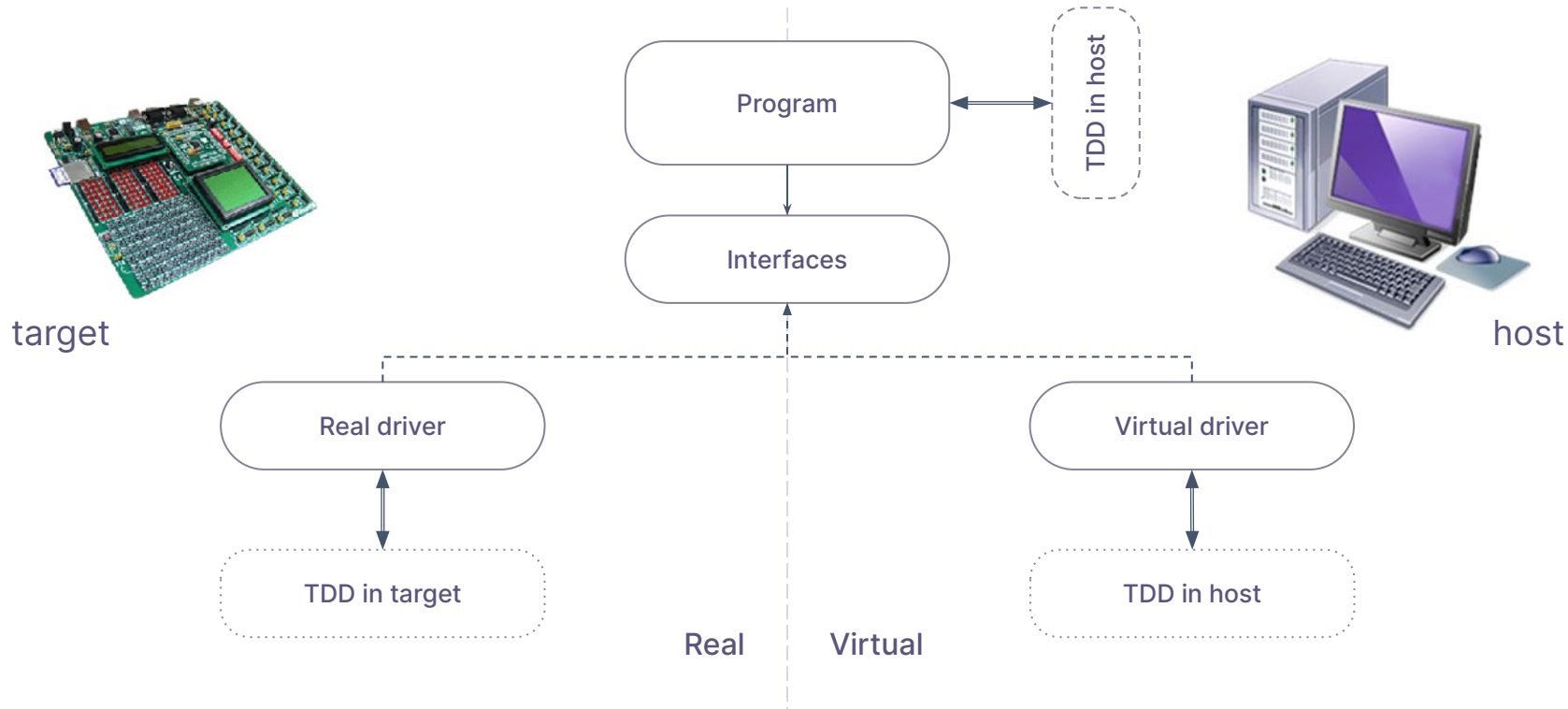
- максимально возможное покрытие кода тестами
- тесты и код устройства запускаются в одной среде

Недостатки:

- медленный запуск тестов
- сложность установки

Второе решение

Двойное тестирование (Dual Targeting)



Тестирование в хост-среде (Dual Targeting)

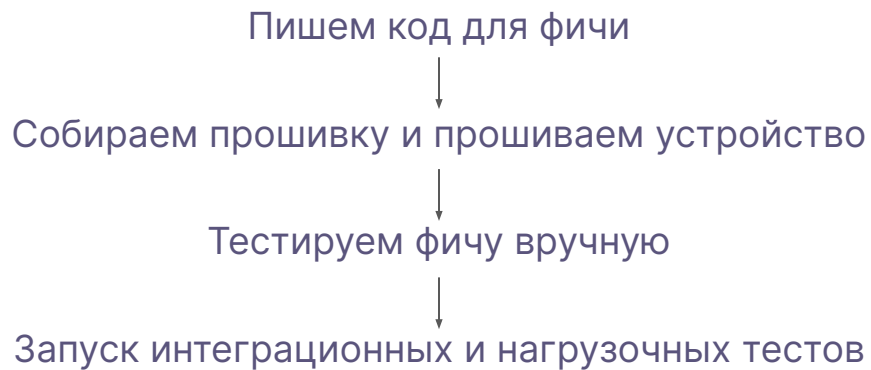
Преимущества:

- тесты запускаются быстро
- код можно писать без доступа к устройству
- разделение чистого кода и кода с зависимостями устройства

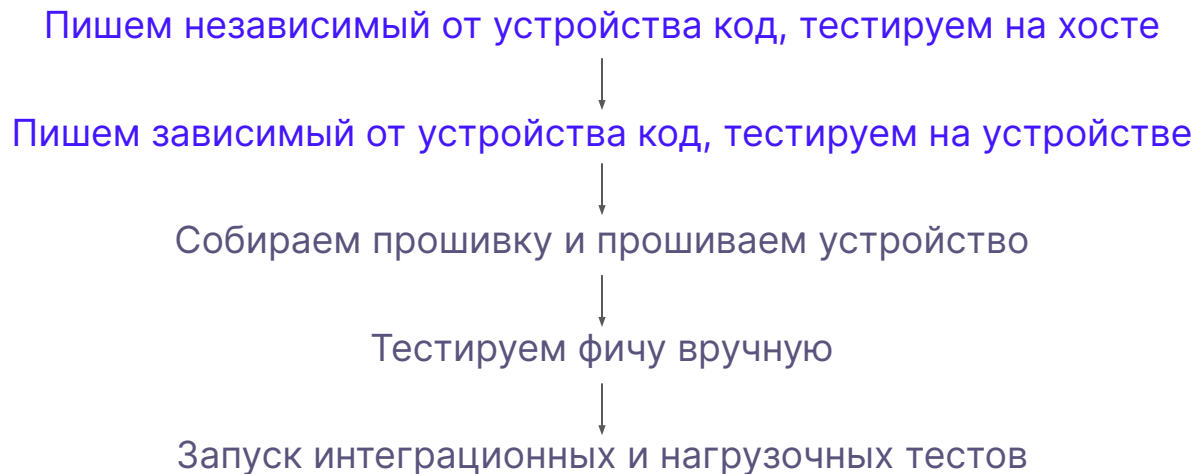
Недостатки:

- тесты и код устройства запускаются в разных средах

Как мы разрабатывали



Что изменилось с Dual Targeting



Независимый от устройства код

- Алгоритмы
- Вычисления, обработка данных
- Внешние библиотеки

● Настраиваем среду

1) Конфигурация тестовой среды

- IDE - Visual Studio
- Тестовый фреймворк для software - GoogleTest
- Тестовый фреймворк для hardware - самописный

2) Конфигурируем проект

- для x86 платформы
- для целевой платформы

Пример использования Dual Targeting. Software код.

● Пишем тест, который провалится

```
TEST(CertificateChain, HaveAtLeastOneCert)
{
    char *certificate_chain = "MIICzTCCAbUCAQAwwYcx CzAJBgNVBAYTAkdC\GP";
    uint8_t ret_code;

    ret_code = checkCertificateChain(certificate_chain);

    EXPECT_EQ(0, ret_code);
}
```

Пример использования Dual Targeting. Software код.

● Делаем тест зеленым

```
uint8_t checkCertificateChain(char *_certificate_chain)
{
    char *c_begin_certificate = "-----BEGIN CERTIFICATE-----";

    uint16_t certificate_chain_length = strlen(_certificate_chain);
    uint16_t begin_found = 0;

    if ((begin_found = getSubstringCount(
        certificate_chain,
        c_begin_certificate,
        certificate_chain_length)) == 0 )
    {
        return -1;
    }

    // do the same for END CERTIFICATE footer
    char *c_end_certificate = "-----END CERTIFICATE-----";
}
```

Пример использования Dual Targeting. Software код.

● Рефакторинг

- Делаем код более читаемым
- Оптимизируем работу

Зависимый от устройства код

- Файловая система
- Драйвера
- Обработка прерываний

Пример использования Dual Targeting. Hardware код.

● Пишем тест, который провалится

```
TEST(CertificateChain, ReadCertificateChain)
{
    char *file_name = "test/dummy.txt";
    certificate_chain = calloc(BUFFER_SIZE, sizeof(uint8_t ));
    uint8_t ret_code;
    uint8_t system_partition = SYSTEM_DATA;

    ret_code = readFileFromStorage(SYSTEM_DATA, file_name, certificate_chain);

    EXPECT_EQ(0, ret_code);
}
```

Пример использования Dual Targeting. Hardware код.

Делаем тест зеленым

```
uint8_t readFileFromStorage(uint8_t partition, char* file_full_name, char*_buffer);
{
    if (partition == null || file_full_name == null || buffer == null)
    {
        return ERROR_INVALID_PARAMETER;
    }
    if(((uint32_t)buffer % BUFFER_ADDRESS_ALIGNMENT) != 0)
    {
        return ERROR_INVALID_ALIGNMENT;
    }

    mutex.lock();
    // trigger some HW low level functions
    mutex.unlock();

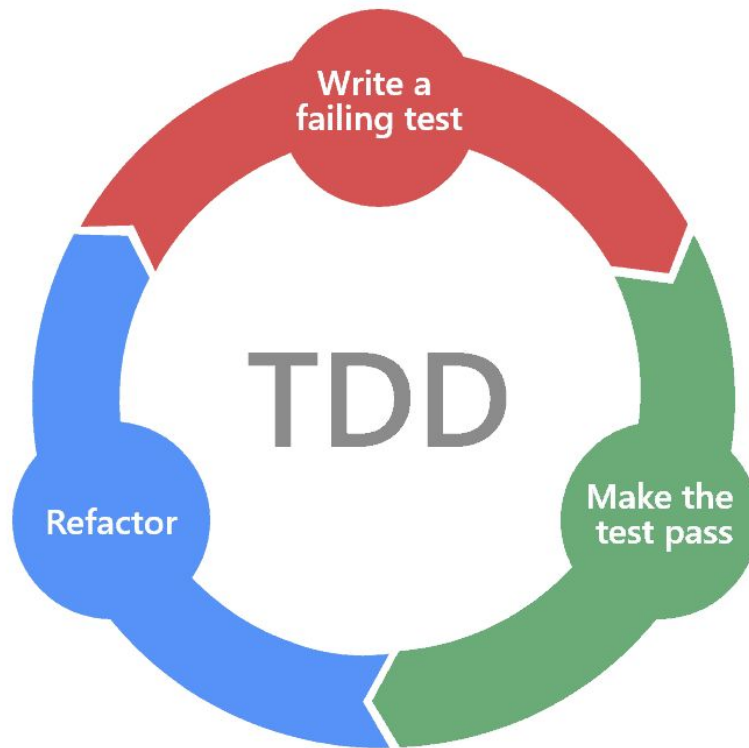
    return OK;
}
```

Пример использования Dual Targeting. Hardware код.

● Рефакторинг

- Делаем код более читаемым
- Оптимизируем работу

Да, это ТДД



Стоило ли оно того?

Плюсы

- Архитектура кода
- Легче поддерживать код
- Код служит документацией

Минусы

- Не является панацеей
- Высокий порог вхождения

Возвращаясь к граблям

- Тщательно выбирайте приоритеты тестирования
- Объединяйте тестирование и разработку
- Не для всех проблем есть решение, но всегда найдется компромисс



Чему бы грабли
не учили,
а сердце верит
в чудеса...

Даже если все
работает, всегда есть, к
чему стремиться :)



Разработчики встраиваемых систем

Кузьмичева Ольга & Кирпичников Антон



okuzmicheva@tourmalinecore.com



akirpichnikov@tourmalinecore.com



Web site

tourmalinecore.com

Vk

[tourmalinecore](https://vk.com/tourmalinecore)