ISaPWN — исследование безопасности ISaGRAF Runtime

Александр Ночвай
Артем Зиненко
Евгений Гончаров

23.05.2022 Version 1.0

Краткое содержание

В начале 2020 года мы уведомили команду Rockwell Automation Product Security Incident Respose Team (RA PSIRT) о нескольких обнаруженных нами уязвимостях в среде выполнения ISaGRAF Runtime.

ISaGRAF Runtime используется в разнообразных отраслях промышленности, и его применение не ограничивается системами автоматизации промышленных предприятий. Согласно публичным источникам информации ISaGRAF Runtime используется также в транспортной, энергетической и других отраслях.

Статья содержит анализ фреймворка ISaGRAF, описание его архитектуры, протоколов IXL и SNCP, используемых для программирования устройств и управления ими, и разбор найденных уязвимостей.

В ходе исследования было выявлено несколько уязвимостей. Были продемонстрированы следующие возможные векторы атак на устройства, основанные на ISaGRAF:

- Удаленный злоумышленник может выполнять привилегированные команды сервиса IXL без прохождения аутентификации на устройствах, которые работают на ISaGRAF Runtime версии до 2010 года.
- В ISaGRAF Runtime не реализована защита от перебора пароля — удаленный злоумышленник может легко провести брутфорс-атаку.
- Если атакующий сможет провести MitM-атаку, он получит возможность перезаписать состояние тегов, загружаемую программу или данные аутентификации. Поскольку аутентификационные данные зашифрованы на фиксированном симметричном ключе, атакующий сможет расшифровать перехваченный target-пароль к устройству.
- Атакующий может использовать уязвимости для получения удаленного доступа к устройствам с ISaGRAF Runtime и исполнения произвольного кода внутри виртуальной машины ISaGRAF Runtime.
- Атакующий может использовать уязвимости для побега из ограниченного окружения (песочницы) ISaGRAF Runtime и предотвращения в дальнейшем обнаружения вредоносного кода на устройстве.

В статье приводятся детальные описания уязвимостей и анализ возможного влияния их успешной эксплуатации, а также рекомендации по дополнительным мерам для снижения возможных рисков.

К концу 2021 года все выявленные уязвимости были либо исправлены разработчиком, либо к ним были предложены меры по уменьшению

возможных рисков при их эксплуатации — разработчиком, CISA или Kaspersky ICS CERT.

К марту 2022 года об уязвимостях в своих продуктах сообщили Rockwell Automation, Schneider Electric, Xylem, GE и Moxa.

По любым вопросам, связанным с исследованием, пишите нам по адресу ics-cert@kaspersky.com.

Технология ISaGRAF

ISaGRAF — это технология программирования и среда исполнения программ для ПЛК. Она включает три компонента:

- **ISaGRAF Runtime** — адаптируемая среда исполнения. Среда, портированная и кастомизированная для определенного ПЛК, называется ISaGRAF Target.
- **ISaGRAF Workbench** — среда разработки приложений для ISaGRAF Runtime. Позволяет разрабатывать, компилировать, отлаживать и загружать на контроллер ресурсы[1] (обладает необходимыми для этого функциями управления контроллером).
- **ISaGRAF Runtime Toolkit** — инструмент для разработки драйверов и адаптации среды исполнения ISaGRAF Runtime под целевую программно-аппаратную платформу (ОС и ПЛК).

В отличие от CODESYS (см. нашу статью «Исследование безопасности: CODESYS Runtime — фреймворк для управления ПЛК» в трех частях: 1, 2, 3), где приложение компилируется в машинный код контроллера, ресурсы ISaGRAF скомпилированы в TIC (target-independent code) и запускаются на выполнение средой ISaGRAF Runtime. Можно утверждать, что ISaGRAF Runtime — это виртуальная машина, для которой TIC является виртуальным кодом.

Среда разработки приложений может быть также кастомизирована и расширена.

Существуют несколько различных сред разработки. ACP (Automation Collaborative Platform) является основной средой разработки, поставляемой Rockwell Automation (RA) разработчикам контроллеров на основе их фреймворка. ISaGRAF Workbench входит в состав ACP. RA разрешает производителям контроллеров предоставлять её бесплатно своим конечным пользователям (например, интеграторам и промышленным

[1] В технологии ISaGRAF вместо термина «приложение» используется термин «ресурс»

предприятиям). Дополнительно существуют специализированные среды разработки для определенных семейств устройств — например, AADvance Workbench для семейства AADvance и SCADAPack Workbench для семейства SCADAPack.

Версия ISaGRAF Runtime для операционной системы Windows распространяется вместе с ACP.

ISaGRAF Runtime

На сегодня существует три основных линейки ISaGRAF Runtime: v1, v3 и v5. Каждая из них привязана к своей версии ISaGRAF Workbench. Последняя версия ISaGRAF Workbench v6 распространяется вместе с поддержкой ISaGRAF Free Runtime v3 и v5 в качестве эмуляторов ISaGRAF Runtime.

Объект исследования

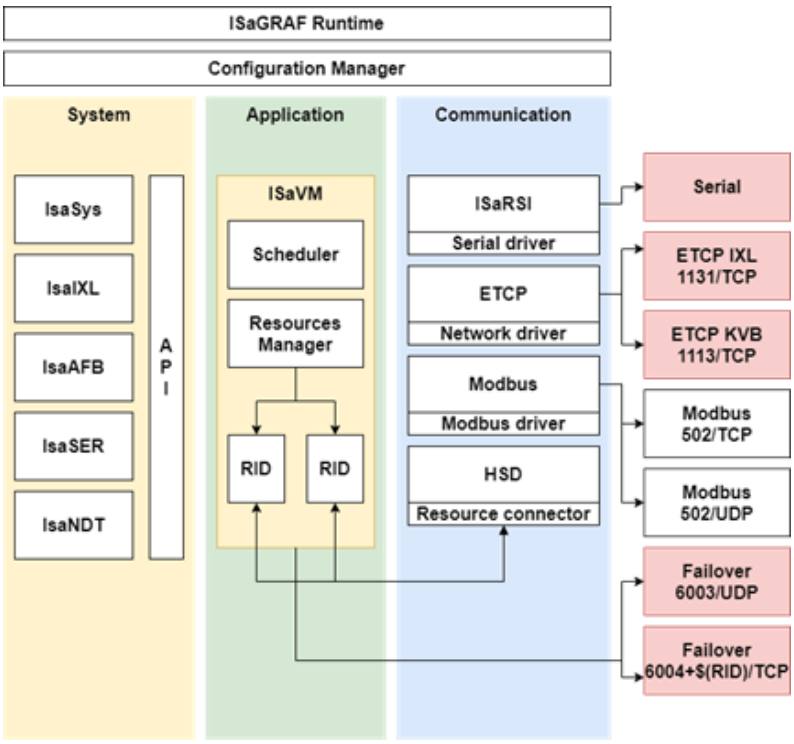
В качестве объекта исследования мы использовали файлы, полученные после установки ACP и демо-версии ISaGRAF Runtime 5.70.32 от 2016 года для Linux и Windows.

Дополнительно мы изучали следующие версии ISaGRAF Runtime:

- ISaGRAF Runtime в прошивке устройств IoPAC 8500 — 5.40.148_(09/11/2013);
- ISaGRAF Free Runtime в AADvance Workbench — (Build 2009.1.10.501_(090416));
- ISaGRAF Free Runtime в ISaGRAF Open — Build 2006.5.10.000_(060301);
- ISaGRAF Free Runtime в ICS Triplex ISaGRAF — Build 2009.1.10.501_(090416).

Компоненты ISaGRAF Runtime

Каждый компонент ISaGRAF Runtime является либо динамической библиотекой, либо исполняемым файлом. Все исполняемые объекты конфигурируются через одноименные файлы конфигурации или аргументы командной строки.



Компоненты ISaGRAF Runtime

Основные компоненты и группы компонентов:

- Configuration Manager — главный компонент, который запускает другие исполняемые компоненты;
- System Components — группа библиотек, которые предоставляют API для работы с конфигурационными файлами, памятью, строками, файлами и т.д.;
- Application — компонент виртуальной машины ISaGRAF;
- Communication components — исполняемые компоненты, обеспечивающие коммуникации с ISaGRAF Runtime по Ethernet или RS232/RS485. Здесь важно отметить, что коммуникация со средой разработки осуществляется непосредственно компонентом Configuration Manager.

Компоненты коммуницируют друг с другом через разделяемую память (shared memory). Для передачи сообщений используется проприетарный протокол.

Configuration Manager

Как уже было сказано, Configuration Manager (**далее CM**) — главный компонент, который инициализирует работу всех остальных компонентов. Запуск CM выполняет исполняемый файл `ISaGRAF` (`ISaGRAF.exe`). CM управляется с помощью файла конфигурации ISaGRAF.ini и аргументов командной строки.

Основные задачи CM:

- Инициализировать shared memory для всех остальных компонентов;
- Инициализировать коммуникационные драйверы (HSD, ETCP и ISaRSI);
- Запустить коммуникационные компоненты (ISaRSI, ETCP);
- Запустить виртуальные машины (ISaVM) и указать им загружаемые ресурсы;
- Инициализировать сетевой сервис IXL, работающий на порту 1131/TCP.

Компилируемая программа и виртуальная машина

Формат скомпилированных приложений сравним по сложности с форматом исполняемого файла Windows — в скомпилированном приложении также есть секции инициализации константных данных, данных по умолчанию, секции переменных, кода, ресурсов и описание требований к виртуальной машине, которая сможет запустить скомпилированный файл.

CM может одновременно запускать несколько виртуальных машин для запуска нескольких приложений — по одной виртуальной машине на приложение. В случае необходимости виртуальные машины могут также коммуницировать друг с другом через разделяемую память (shared memory).

Протокол IXL

Устройство с ISaGRAF Runtime программируется по Ethernet или последовательному интерфейсу. Коммуникацию по последовательному интерфейсу обеспечивает компонент ISaRSI[2]. Коммуникацию по Ethernet — CM.

Среда *разработки* удаленно выполняет базовые операции с ПЛК, такие как программирование, загрузка приложения, отладка и изменение конфигурации. Для выполнения этих базовых операций по Ethernet ACP коммуницирует с ISaGRAF Runtime по порту 1131/TCP. В качестве протокола коммуникации используется проприетарный протокол IXL — eXchange Layer — протокол для обмена информацией между средой разработки и CM.

Для установки соединения серверный узел посылает несколько служебных сообщений клиенту IXL. Первое служебное сообщение содержит пакет с четырьмя нулями (`00 00 00 00`). Второе сообщение содержит версию протокола IXL.

[2] Есть предположение, что акроним ISaRSI расшифровывается как ISaGRAF Remote Serial Interface.

ISaGRAF Free Runtime передает вторую версию протокола (`02 00 00 00`), эмулятор AADvance T900 первую (`01 00 00 00`), а эмулятор SCADAPack — нулевую версию протокола (`00 00 00 00`).

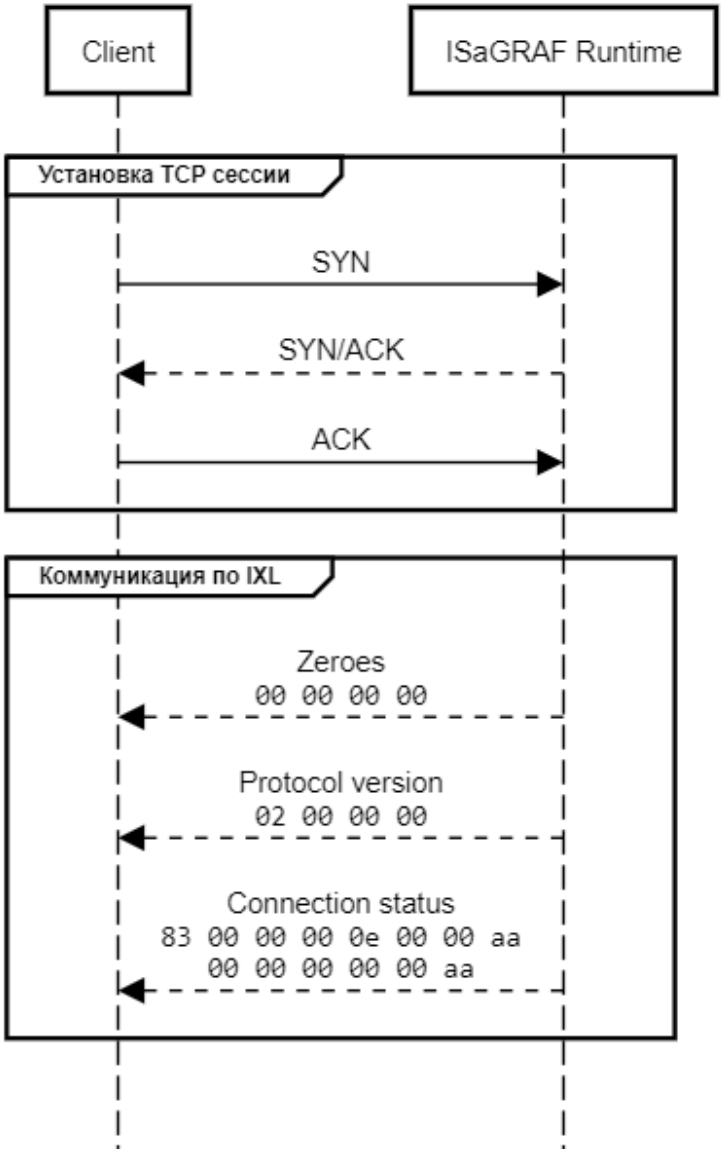


Схема коммуникации между ACP и ISaGRAF Free Runtime по IXL протоколу

Формат сообщений IXL максимально простой — каждая команда содержит поля header и body. В конце каждого поля body и header размещается константа `0xAA`. Контрольной суммы нет.

Формат поля header

Header содержит идентификатор команды, тип сообщения, флаг о наличии ошибок, размер всего сообщения и идентификатор ресурса. Формат header следующий:

Field	Type	Description
Direction	1 bit	Указывает направления сообщения: 0 — это запрос, 1 — ответ
Status	1 bit	Флаг о наличии ошибки при обработке запроса
Command	6 bit	Идентификатор команды
Packet size	4 byte	Размер всего пакета
Resource ID	2 byte	Идентификатор ресурса
Ending	1 byte	Окончание заголовка header константой `0xAA`

Поле Resource ID определяет, к какому именно ресурсу будет применена команда. По этому значению CM понимает, какой именно виртуальной машине передать запрос. Запрос виртуальной машине передаётся через драйвер ISaGRAF HSD, для передачи данных используется shared memory.

Поле Command содержит идентификатор команды. Протокол IXL определяет следующие команды:

- `READ` (`0x1`) — получить значения переменных
- `WRITE` (`0x2`) — изменить значения переменных
- `CONNECT` (`0x3`) — завершить установку соединения, отправляется только в качестве ответа
- `DISCONNECT` (`0x4`) — прервать соединение
- `SUBCMD` (`0x6`) — выполнить команду CM
- `START_DIALOG` (`0x7`) — установить диалог для выполнения системных команд CM
- `STOP_DIALOG` (`0x8`) — завершить диалог для выполнения системных команд CM

Ниже представлен пример разбора данных для команды `CONNECT`, которые были отправлены от ISaGRAF Runtime клиенту:

```
tcp.payload:
0000  83 00 00 00 0e 00 00 aa 00 00 00 00 00 aa

header:
0000: 83 = 0b100000011
            1.............. - Direction response
            .0.............. - Status OK
            ..............11 - Command CONNECT
0001: 00 00 00 0e = 14 - packet size
```

```
0005: 00 00 = 0 - resource id
0007: aa - mask ending

body:
0008: 00 00 00 00 00 aa
```

Получение такого пакета означает, что соединение по IXL протоколу было успешно установлено.

Формат поля body

Формат поля body зависит от значения поля Command и флага Direction.

Пример формата поля body для ответа CONNECT

Значение body в примере из главы «Формат поля header» (см. выше), может быть разложено следующим образом:

```
tcp.payload:
0008: 00 00 00 00 00 aa

body:
0008: 00 00 00 00 - error number
0012: 00 - Internal ID status
0013: aa - mask ending
```

Поле body команды `CONNECT` будет содержать сразу два идентификатора ошибки и константу окончания поля.

Пример формата поля body для команды READ

Поле body для команды `READ` содержит количество элементов, значения которых запрашивается. Пример запроса:

```
body data:
0008: 00 02 00 00 01 24 02 00 00 00 01 00 00 00 04 00 00 15 c8 03 00 00 00 01 00 00 00 04 aa

body:
0008: 00 02 - number of variables is 2

first item:
0010: 00 00 01 24 02 00 00 00 01 00 00 00 04

    parsed:
    0010: 00 00 01 24 - id of item
    0014: 02 - type of item is signed integer
    0015: 00 00 00 01 - default value
    0019: 00 00 00 04 - size of item

second item:
0023: 00 00 15 c8 03 00 00 00 01 00 00 00 04

    parsed:
```

```
    0023: 00 00 15 c8 - id of item
    0027: 03 - type of item is integer
    0028: 00 00 00 01 - default value
    0032: 00 00 00 04 - size of item

0033: aa - mask ending
```

Первые два байта определяют количество запрашиваемых переменных. Элемент переменной имеет следующую структуру: идентификатор переменной, тип переменной, значение по умолчанию и размер переменной. В ответе значение по умолчанию будет заменено на текущее значение переменной.

Пример формата поля body для команды SUBCMD

Поле body для команды `SUBCMD` определяет только значение подкоманды, от которого зависит дальнейший формат данных. При этом само назначение подкоманды зависит от поля Resource ID. В случае если клиенту необходимо выполнить системную команду CM, в качестве Resource ID используется константа `0x0FFF`. Системной командой может быть, к примеру, получение количества запущенных ресурсов или загрузка конфигурационного файла. Если же необходимо загрузить, очистить или удалить ресурс, то используется константа `0xFFE`. А когда необходимо пройти аутентификацию или установить пароль, то значение Resource ID игнорируется.

Иногда поле body может целиком отсутствовать, как, например, для команды `SUBCMD RESOURCE_GET_STATE_EXTENDED` (`0x1E`):

```
body data:
0008: 1e aa

body:
0008: 1e - SUBCMD id value RESOURCE_GET_STATE_EXTENDED
0009: aa - mask ending
```

А для команды `SUBCMD RESOURCE_LAUNCH` (`0x50`) дополнительные поля нужно передать обязательно:

```
body data:
0008: 50 00 00 01 00 00 aa

body:
0008: 50 - SUBCMD id value RESOURCE_LAUNCH
0009: 00 - run mode
0010: 00 01 - resource id to launch
0012: 00 - length of argument
0013: aa - mask ending
```

Для команды `RESOURCE_LAUNCH` дополнительные поля содержат идентификатор ресурса для запуска, размер аргументов, сами аргументы (в

примере выше отсутствуют, т.к. их длина равна 0) и режим, в котором нужно будет их запускать.

Мы нашли около 65 обработчиков команд для ISaGRAF Free Runtime. Список этих обработчиков может отличаться для других ISaGRAF target. Все команды можно свести к следующим группам:

- Команды отладки приложения: выполнение одной инструкции, получение списка точек останова, получение стека вызовов, удаление и добавление точек останова;
- Команды работы с файловой системой: загрузка файла конфигурации, загрузка и удаление произвольного файла, загрузка файла-модуля и загрузка приложения через файл;
- Команды аутентификации и изменения target-пароля;
- Команды работы с ресурсом: проверка состояния, проверка контрольной суммы, остановка, запуск, получение базовой информации, получение возможностей виртуальной машины.

Протокол SNCP

Необходимо отметить, что не все устройства с ISaGRAF Runtime используют порт 1131/TCP для коммуникации по IXL. К примеру, согласно информации в документации к устройству AADvance Controller, для корректной работы ISaGRAF требуется сетевой доступ к порту 1132/TCP:

AADvance Communication Ports

Protocol	Port Number	Port Open	Purpose	Port Open When
TCP	502	When configured	Modbus TCP slave	Open if Controller is configured as a Modbus TCP slave.
TCP	1132	Always	ISaGRAF®, application downloads, debug, SoE etc.	N/A
TCP	2000	When configured	Modbus RTU slave	Open if controller is configured as a Modbus RTU slave. The default port, 2000, is given in this table.

Фрагмент документации к AADvance Controller

В TCP and UDP Port Configuration — Quick Reference Guide указано имя протокола, использующего порт 1132/TCP — «SNCP»:

1132	TCP	SNCP	AADvance	Safety Network Control Protocol, used by OPC, workbench debugger and binding networks

Фрагмент документа TCP and UDP Port Configuration — Quick Reference Guide

SNCP является надстройкой над протоколом IXL, добавляющей поля для проверки целостности данных и источника сообщения.

SCNP использует следующий формат сообщения:

Field	Type	Description
Magic	2 byte	Константа `0xCCCC` — идентификатор сообщения SCNP
ixl_crc_meta	2 byte	Информация о типе контрольной суммы поля `ixl_crc` — значение `0x0201` указывает на использование CRC64, иначе CRC32
message_id	2 byte	Идентификатор сообщения пакета
ixl_body_size	2 byte	Размер `ixl_body`
ip_src	4 byte	IPv4 адрес источника сообщения
ip_dst	4 byte	IPv4 адрес назначения сообщения
ixl_crc	4/8 byte	Контрольная сумма `ixl_body`
crc32	4 byte	Контрольная сумма заголовка SNCP
ixl_body	ixl_body_size	Сообщение формата IXL

Обнаруженные уязвимости и возможные векторы атак

В общей сложности мы уведомили Rockwell Automation PSIRT о 9 уязвимостях, которые могут быть проэксплуатированы удаленным или локальным злоумышленником.

Рассмотрим сценарии атак захвата устройства с ISaGRAF Runtime удаленным не аутентифицированным злоумышленником с использованием обнаруженных уязвимостей. Конечная цель злоумышленника — выход из ограниченного окружения ISaGRAF Runtime и захват устройства.

Получение административного доступа к CM

Сервис IXL определяет некоторые команды как привилегированные, и для их выполнения необходимо пройти аутентификацию с помощью target-пароля. Привилегированные команды взаимодействуют с файловой системой, приложением, конфигурацией ISaGRAF Runtime и контролем режима работы устройства.

Успешная атака на аутентификацию сервиса IXL означает захват устройства и исполнение произвольного кода внутри виртуальной машины ISaGRAF Runtime.

Отсутствие аутентификации в версиях ISaGRAF Runtime до 2010 года

Первая уязвимость — это отсутствие аутентификации в версиях ISaGRAF Runtime до 2010 года.

Однозначно сложно указать, какие именно версии ISaGRAF Runtime уязвимы, потому что многие вендоры используют свою версионность. К примеру, так делает компания Rockwell Automation для продуктов AADvance и ICS Triplex. Тем не менее, версии всех рассмотренных нами экземпляров ISaGRAF Runtime содержат год. Поэтому мы условно обозначаем уязвимые версии ISaGRAF Runtime как версии до 2010 года.

Мы подтвердили наличие этой уязвимости в ISaGRAF Free Runtime, который входит в комплект различных сред разработки. Во всех версиях, созданных до 2010 года, вместо обработчиков команды аутентификации PASSWORD_CHECK стояла заглушка.

В качестве примера ниже представлен псевдокод для команды `PASSWORD_CHECK` в AADvance Workbench и ACP:

Configuration Manager — Build 2009.1.10.501_(090416)

```
01: subcmd_id = *body_data;
02: if (subcmd_id == PASSWORD_CHECK) {
```

Configuration Manager — Build 5.72.00.142_(02/12/2021)

```
01: subcmd_id = *body_data;
02: if (subcmd_id == PASSWORD_CHECK) {
03:   dsysMemReset(recved_password, 9);
04:   dsysMemCpy(recved_password,body_data + 1, 8);
05:   dsysDecryptPassword8(recved_password, recved_password);
06:   if (IXDINFO.password[0] == '\0') {
07:     param_1->authorized = true;
08:   }
09:   else {
10:     compared = dsysStrCmp(recved_password, IXDINFO.password);
11:     param_1->authorized = compared == 0;
12:   }
13:   if (param_1->authorized == false) {
14:     error = ISA_RC_PASSWDINVALID;
15:   }
16:   else {
```

```
03:   response->command = PASSWORD_CHECK;
04:   response->error = ISA_RC_SUCCESSFUL;
05:   ixsMsgProc(param_1->field_0x0,
param_2, response, 2, 0x10);
06: }
```

```
17:     error = ISA_RC_SUCCESSFUL;
18:   }
19:   response->command = PASSWORD_CHECK;
20:   response->error = error;
21:   ixsMsgProc(param_1->field_0x0, param_2, response, 2, 0x10);
22: }
```

Из псевдокода видно, что в версии 2009 года реализована заглушка команды `PASSWORD_CHECK`, которая всегда возвращает статус `ISA_RC_SUCCESSFUL` (аутентификация успешно пройдена). В версии 2021 года CM извлекает из body-пароль (строка 04), расшифровывает его (строка 05) и затем сравнивает со значением в памяти (строка 10). В случае если значения совпадают, обработчик возвращает `ISA_RC_SUCCESSFUL` (строка 17). Если нет — `ISA_RC_PASSWDINVALID` (строка 14).

Таким образом, удаленный злоумышленник может выполнять привилегированные команды сервиса IXL без прохождения аутентификации на устройствах, которые работают на ISaGRAF Runtime до 2010 года.

Перебор target-пароля

Target-пароль — это чувствительная к регистру буквенно-числовая строка длиной всего до 8 символов. При этом ISaGRAF Runtime не реализует защиту от перебора пароля. Поэтому угроза перебора пароля на устройствах с ISaGRAF Runtime вполне актуальна.

Передача target-пароля происходит при выполнении команды `PASSWORD_CHECK`, при которой зашифрованный target-пароль передается в body. Target-пароль шифруется при помощи Tiny Encryption Algorithm (TEA) на фиксированном ключе.

На языке программирования Python функция шифрования пароля будет следующей:

```
from ctypes import c_uint32
from construct import Int32ul, Int32ub, PaddedString

K1 = <REDACTED>
K2 = <REDACTED>
KEYS = [K1, K2, -K1, -K2]
DELTA = 0x9e3779b9

def encipher(v, k):
    y = c_uint32(v[0])
    z = c_uint32(v[1])
    sum_value = c_uint32(0)
    n = 32
    w = [0, 0]
```

```
    while (n > 0):
        sum_value.value += DELTA
        y.value += (z.value << 4) + k[0] ^ z.value + \
            sum_value.value ^ (z.value >> 5) + k[1]
        z.value += (y.value << 4) + k[2] ^ y.value + \
            sum_value.value ^ (y.value >> 5) + k[3]
        n -= 1

    w[0] = y.value
    w[1] = z.value
    return w

def encrypt_password(password: str):
    if len(password) == 0:
        raise BaseException
    _ = PaddedString(8, "ascii").build(password)
    encrypted = encipher([Int32ul.parse(_[:4]), Int32ul.parse(_[4:])], k=KEYS)
    return Int32ub.build(encrypted[0]) + Int32ub.build(encrypted[1])
```

Таким образом, для выполнения атаки осталось реализовать установку соединения с сетевым сервисом CM по протоколу IXL и выполнение команды `PASSWORD_CHECK` с возможным паролем.

Шифрование пароля симметричным алгоритмом на фиксированном ключе и MiTM

Передаваемые данные по протоколу IXL не зашифрованы, и в продуктах ISaGRAF нет никакой возможности защитить их с помощью шифрования. При условии, что других средств защиты коммуникации не используется, и злоумышленник может провести MiTM, он будет способен перезаписать состояние тегов, загружаемую программу или данные аутентификации. Т.к. данные аутентификации зашифрованы на фиксированном симметричном ключе (CVE-2020-25180), то можно расшифровать перехваченный target-пароль.

На языке программирования python функция расшифровки будет следующей:

```
from ctypes import c_uint32
from construct import Int32ul, Int32ub, PaddedString

K1 = <REDACTED>
K2 = <REDACTED>
KEYS = [K1, K2, -K1, -K2]
DELTA = 0x9e3779b9

def decipher(v, k):
    y = c_uint32(v[0])
    z = c_uint32(v[1])
    sum_value = c_uint32(0xc6ef3720)
```

```
    n = 32
    w = [0, 0]

    while (n > 0):
        z.value -= (y.value << 4) + k[2] ^ y.value + \
            sum_value.value ^ (y.value >> 5) + k[3]
        y.value -= (z.value << 4) + k[0] ^ z.value + \
            sum_value.value ^ (z.value >> 5) + k[1]
        sum_value.value -= DELTA
        n -= 1

    w[0] = y.value
    w[1] = z.value
    return w

def decrypt_password(encrypted):
    if len(encrypted) == 0:
        raise BaseException
    c = decipher([Int32ub.parse(encrypted[:4]),
                  Int32ub.parse(encrypted[4:])], k=KEYS)
    return PaddedString(8, "ascii").parse(Int32ul.build(c[0]) + Int32ul.build(c[1]))
```

В качестве мер смягчения Rockwell Automation предлагает корректно сконфигурировать firewall и сегментировать сеть. Необходимо добавить, что протокол не подразумевает передачу значения сессии после успешного прохождения аутентификации, и сервис связывает признак того, что пользователь является аутентифицированным, с IP адресом клиента.

Побег из песочницы

Разработчики ISaGRAF Runtime реализовали команду по заливке произвольного файла через IXL сервис. При этом имя файла не проверятся ни по спискам разрешенных имен файлов, ни на служебные символы. Что привело к уязвимости Path Traversal (CVE-2020-25176), которая может привести к удалённому исполнению кода.

Так, злоумышленник может загрузить на устройство произвольный файл через команду `DOWNLOAD_FILE_PACKET` (`0x3b`), манипулируя его именем и указывая в нем специальные символы.

С учётом того, что CM ищет некоторые исполняемые файлы с именем, сформированным по определенной маске (CVE-2020-25182), для выполнения произвольного кода злоумышленнику остается только загрузить исполняемый файл с заданным именем. Так, можно залить файл с именем «IVMdead» и отправить команду на запуск ресурса с идентификатором 57005. В результате CM запустит исполняемый файл `IVMdead`.

Такая уязвимость может быть использована для побега из ограниченного окружения ISaGRAF Runtime и закрепления на устройстве. Возможный злоумышленник может использовать эту уязвимость для сокрытия своего пребывания на устройстве с ISaGRAF Runtime.

Для исправления уязвимости компания Rockwell Automation добавили проверку загружаемого файла на служебные символы.

Выводы

Наше исследование в очередной раз показывает, насколько серьёзны уязвимости в сторонних компонентах. Несмотря на то, что обнаружить описанные уязвимости было несложно, сроки их исправления в конечных продуктах — огромны.

В случае ISaGRAF пользователю, чтобы узнать, что в используемом им продукте есть уязвимости, нужно дождаться, пока Rockwell Automation исправит уязвимости и опубликует адвайзори, а затем вендор конечного продукта сделает то же самое. В некоторых случаях цепочка поставки ISaGRAF оказывается ещё более длинной.

К марту 2022 года об уязвимостях в своих продуктах сообщили:

- Rockwell Automation
- Schneider Electric
- Xylem
- GE
- Moxa

Мы рекомендуем в защите полагаться на подход defense-in-depth и, помимо установки обновлений, рекомендуем уделить особое внимание предотвращению несанкционированного сетевого доступа к портам 1131/TCP и 1132/TCP.

По всем вопросам, связанным с обеспечением безопасности конечных устройств, построенных на основе технологии ISaGRAF, равно как и по любым вопросам по данному исследованию, пишите нам по адресу ics-cert@kaspersky.com.

Центр реагирования на инциденты информационной безопасности промышленных инфраструктур «Лаборатории Касперского» (Kaspersky ICS CERT) — глобальный проект «Лаборатории Касперского», нацеленный на координацию действий производителей систем автоматизации, владельцев и операторов промышленных объектов, исследователей информационной безопасности при решении задач защиты промышленных предприятий и объектов критически важных инфраструктур.

Kaspersky ICS CERT ics-cert@kaspersky.com