

Проблема безопасности программного обеспечения микроконтроллеров, используемых в АСУ ТП

В. В. БАРДАКОВ
ООО “Диджитал Секьюрити”
Россия
v.bardakov@dsec.ru

КЛЮЧЕВЫЕ СЛОВА

Безопасность, АСУТП, микроконтроллеры, AVR.

1 ВВЕДЕНИЕ

В последнее время особое внимание уделяют безопасности НМІ составляющей SCADA.

Однако микроконтроллеры в составе SCADA тоже являются потенциально уязвимыми устройствами.

В большинстве случаев, защитой от эксплуатации уязвимостей в них, является труднодоступность каналов взаимодействия и программного обеспечения, заложенного в микроконтроллеры.

Каналы взаимодействия с микроконтроллерами, при использовании их в составе SCADA, концентрируются в определенных местах, или же вовсе выводятся за пределы контролируемой территории (удаленное управление тех. процессом), что частично нивелирует первую проблему эксплуатации уязвимостей.

Проблема доступности программной части контролера решается множеством способов, в том числе технических. Часть подобных методик крайне сложны и дорогостоящи, но позволяют добиться гарантированного успеха.

Микроконтроллеры разных производителей различаются векторами атак и имеют свои, не описанные в литературе, особенности эксплуатаций уязвимостей и методов защиты от них.

2 ОСНОВНАЯ ЧАСТЬ

2.1 Архитектура AVR и структура памяти.

Среди множества микроконтроллеров, используемых в АСУТП, объектом анализа были выбраны 8-bit микроконтроллеры AVR фирмы Atmel, получившие широкое распространение, благодаря небольшой стоимости, развитой периферии и доступности.

Первоначальной задачей было определить, какие аппаратные ограничения реализованы, чтобы избежать или затруднить эксплуатацию переполнений буфера и стэка, как наиболее вероятных при разработке сложного программного обеспечения[1].

Структура памяти AVR микроконтроллеров указана в документации[2]. Для анализа был выбран контроллер AtTiny87 из-за его периферии.

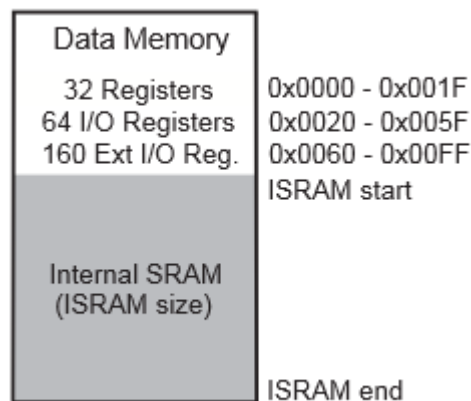


Рис. 1: Структура памяти

Однако все разграничения в структуре SRAM условны. Подобную архитектуру памяти и контроллера в целом, можно объяснить лишь необходимостью сокращения количества команд. Единственным препятствием для выполнения вредоносного кода через эксплуатацию переполнений буфера, в таком случае, остается неисполняемый стек. Однако, злоумышленник может перейти к исполнению кода заложенного в микроконтроллере.

2.2 Проведение атак.

Дальнейшие исследования проводились в двух разных направлениях. Обусловлено это относительной недоступностью программного обеспечения, заложенного в AVR.

Злоумышленник, не имеющий доступа к программной части устройства, переполнив буфер, может перейти к выполнению участков кода, местоположение которых, относительно постоянно. Такими инструкциями в AVR являются:

- прерывания
- вспомогательные инструкции, генерируемые компилятором

Вектор прерываний в AVR статичен[2,5], что позволяет злоумышленнику вызвать произвольный обработчик(и) прерываний. Подобные действия способны привести к отказу в обслуживании(DoS) микроконтроллера и проведению атак типа Fault Injection.

Каждый компилятор добавляет свои вспомогательные инструкции. Для анализа, в данном исследовании, был выбран официальный компилятор от фирмы Atmel. Компилятор C в Atmel Studio 6.X добавляет ряд инструкции в конец программной части, для обработки прерывания RESET.

```
.text:00000049 loc_49:
.text:00000049 cli
.text:0000004A
.text:0000004A loc_4A:
.text:0000004A rjmp loc_4A
```

Рис.2: Инструкции позволяющие провести DoS

Адрес данных инструкций может быть подобран при условии установленного флага прерываний. Перейдя к исполнению данных инструкций, злоумышленник может вызвать отказ в обслуживании(DoS) контроллера.

Имея доступ к программной части, злоумышленник может определить адреса всех инструкций. Это дает возможность злоумышленнику, использовать дополнительные вектора атак.

Злоумышленник может обойти проверки, заложенные в программное обеспечение устройства.

Злоумышленник может использовать технику ROP[4,5], выполнив, тем самым, произвольный код, составленный из частей программного обеспечения, заложенного в AVR. В процессе подобных атак, злоумышленник может изменить содержимое FLASH памяти,

воспользовавшись функцией самопрограммирования. В таком случае происходит нарушение целостности программного обеспечения устройства. Прерывания значительно упрощают ROP, благодаря наличию инструкции `reti`, выгружающей адрес возврата из стека и устанавливающей флаг глобального прерывания.

2.3 Защита.

Учитывая структуру памяти в AVR, и труднодоступность исходного кода. Достаточно эффективной защитой, помимо white-box анализа перед запуском в производство, является отказ от использования компилятора фирмы Atmel версии 6.0 и 6.1. Дополнительной защитой может служить рандомизация адресного пространства Flash памяти посредством самопрограммирования и реализация динамического вектора прерываний.

3 ЗАКЛЮЧЕНИЕ

Основной защитой от эксплуатации уязвимостей в AVR является неисполняемый стек и относительная закрытость программного кода. Несмотря это, злоумышленник может осуществить DoS или реализовать Fault Injection атаки. Имея доступ к техническим средствам, для получения программной составляющей микроконтроллера злоумышленник может получить полный доступ к AVR? Изменив программную составляющую посредством самопрограммирования. Исходя из перечисленных векторов атак, AVR микроконтроллеры нуждаются в дополнительной защите, которая, пока, не реализована в виде коммерческих продуктов.

ЛИТЕРАТУРА

- [1] Viega, John, McGraw, Gary, Building Secure Software, Addison-Wesley, Boston, Mass., 2002.
- [2] 7728G–AVR–06/10
- [3] Barnett, Richard H., et al. Embedded C programming and the Atmel AVR. Delmar Pub, 2007.
- [4] Roemer, Ryan, et al. "Return-oriented programming: Systems, languages, and applications." ACM Transactions on Information and System Security (TISSEC) 15.1 (2012): 2.
- [5] Kornau, Tim. "Return oriented programming for the ARM architecture." Master's thesis, Ruhr-Universitat Bochum (2010).