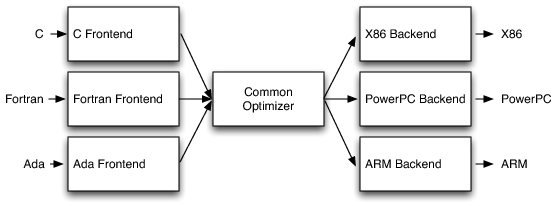
**IoT 디바이스 펌웨어 코드 난독화**

팀원 : 2015104175 박우진

2017110275 이한솔

**개요**

현재 다양한 IoT 디바이스의 사용이 증가되고 있으며 이로 인한 보안 위협도 증가되고 있다. Baremetal 디바이스의 펌웨어는 펌웨어 업데이트 파일 및 Flash De-soldering등으로 추출이 가능하며 이를 정적 분석을 통해 실행 흐름을 분석할 수 있다. 이를 통해 IoT 디바이스의 취약점을 찾을 수 있게 된다. 이 프로젝트 에서는 정적 분석을 어렵게 하기 위해 LLVM을 이용하여 Baremetal디바이스의 펌웨어를 난독화 하는 방안을 제시하며 이를 실제로 구현하고 평가한다.

1. **서론**  
     
   **1-1 연구 배경**  
     
    최근 들어 IoT 디바이스의 사용이 증가되고 있어 보안의 위협도 증가되고 있다. 하지만 IoT 디바이스 제조사들의 보안을 고려하지 않은 개발로 인해 IoT 디바이스에 대한 공격이 증가하고 있다. 따라서 IoT 디바이스 보안의 중요성이 강조되고 있다. IoT 디바이스들은 역공학을 통한 펌웨어의 정적 분석을 통해 코드의 흐름을 파악할 수 있으며 이를 위한 도구도 존재한다. Ghidra, IDA Pro와 같은 프로그램은 바이너리 파일을 손쉽게 정적 분석할 수 있도록 해주며 디컴파일도 가능하게 해주어 누구나 쉽게 펌웨어의 분석이 가능하다. 따라서 우리는 IoT 디바이스 펌웨어의 정적 분석을 어렵게 하기 위해 펌웨어의 코드를 난독화할 수 있는 방안을 알아보고 이를 구현하여 실제 펌웨어를 난독화 해보기로 한다.  
     
     
   **1-2 연구 목표**  
     
    여러 난독화 기법을 분석하고 이를 LLVM 컴파일러의 pass를 이용하여 구현할 것이다. 그 후에 LLVM 컴파일러를 이용하여 펌웨어를 컴파일할 것이다.
2. **기존 연구**  
     
   **2-1 정적분석 도구**  
     
    일반 소프트웨어 및 Baremetal 펌웨어는 정적 분석을 통해 실행 흐름 등의 소스코드 분석이 가능하다. 이러한 분석을 가능하게 해주는 도구로는 Ghidra [3], IDA Pro [4]등의 도구가 존재한다. IDA Pro와 Ghidra는 소프트웨어의 정적 분석을 편리하게 해 줄 수 있는 도구이다. Assembly 레벨에서의 분석뿐만 아니라 다양한 Architecture에서의 Decompile도 가능해 누구나 쉽게 실행 흐름의 분석이 가능하며 이를 통해 취약점 분석도 가능해진다. 또한 Ghidra는 무료 오픈소스 소프트웨어로 누구나 사용이 가능하다.   
     
   **2-2 LLVM [1]**  
     
    LLVM(Low Level Virtual Machine)은 컴파일러이다. 일반적인 Compile후 Link를 수행하는 컴파일러와 달리 LLVM은 Front-end, Middle-end, Back-end로 나누어져 있다. Front-end에서는 실제 C, C++, Go, Rust와 같은 언어 들의 코드에서 LLVM의 핵심 표현인 IR(Intermediate Representation)으로 변환 해준다. 이 IR은 Assembly와 비슷한 저급 프로그래밍 언어이다. 그 후 Middle-end에서는 LLVM의 Pass를 이용해 코드 최적화, 코드 삽입, 코드 변경 등과 같은 일을 할 수 있으며 모두 IR에서 IR로 변환된다. Back-end에서는 IR 코드를 Target Architecture에 맞는 기계어로 번역해준다. 이 단계에서 MachineFunctionPass를 적용하여 기계어 코드에 대한 최적화도 수행할 수 있다.   
      
     
     
     
   **2-2 Obfuscator-LLVM [2]**  
      
    Obfuscator-LLVM에서는 LLVM의 Pass를 이용하여 일반 소프트웨어를 난독화 할 수 있는 방안을 제시하였다. Bogus control flow 삽입 등의 다양한 난독화 방법을 LLVM Pass로 구현하여 오픈소스로 제공하였다. LLVM의 IR 단계에서의 난독화 수행으로 대부분의 Architecture에서 적용될 수 있다.   
     
   **2-3 기존 연구의 문제점**  
     
    손쉽게 사용가능한 정적 분석 도구로 인해 수많은 IoT 디바이스 들에 대한 실행 흐름 파악이 가능해지고 이를 이용한 취약점 탐지 공격 등이 증가하고 있다. 이를 방지하기 위해 Obfuscator-LLVM은 LLVM을 이용한 난독화 방안을 제시하였다. 하지만 이 난독화 방안은 IR 단계 에서의 난독화만 제시하며 실제 Target Architecture 에서의 기계 코드의 난독화 방안은 제시하지 않았다.
3. **프로젝트**  
     
   **3-1. 기존 연구와의 차이점 및 해결방안**  
    기존 연구인 Obfuscator-LLVM은 basic instruction substitution, Bogus control flow삽입 등 control flow에 대한 난독화만을 수행한다. 그러나 정적 분석 도구를 통해 소프트웨어의 문자열 데이터를 손쉽게 파악할 수 있으며 이를 이용해서도 실행 흐름, 네트워크 연결 등을 파악할 수 있다. 이러한 문자열 데이터와 같은 정적 데이터 등도 난독화하여 알아 볼수 없도록 할 예정이다. 또한 기계어 코드에 대한 난독화도 수행하지 않으므로 기계어 코드에 대한 난독화도 수행할 예정이다.  
     
    정적 데이터에 대한 난독화의 경우에는 xor연산을 이용한 난독화를 통해 사용자가 알아 볼 수 없도록 할 수 있으며 기계어 코드에 대한 난독화는 LLVM에서 Target Architecture으로의 코드 변환시 수행되는 MachineFunctionPass를 작성하여 수행할 수 있다.   
     
   **3-2. 프로젝트 내용**  
     
    LLVM의 FunctionPass 및 ModulePass와 같은 Pass를 작성하여 가짜 컨트롤 플로우 삽입 등의

|  |  |  |
| --- | --- | --- |
| 진행 주차 | 담당 | 내용 |
| 1~3 | 전원 | 난독화 방법 조사 및 분류 |
| 4 | 전원 | 조사 보고서 작성 |
| 5-7 | 전원 | LLVM 소스코드 획득 및 Pass작성법 연구 |
| 7~10 | 박우진 | Control Flow기반 난독화 및 기계어 코드 난독화 Pass 작성 |
| 7~10 | 이한솔 | Static 데이터 난독화 Pass 작성 |
| 11~13 | 전원 | LLVM Pass 컴파일 및 Arduino Due 펌웨어 난독화 |
| 14 | 전원 | 펌웨어 난독화 결과 확인 및 성능 분석 |
| 15 | 전원 | 발표 및 최종 보고서 작성 |

Control Flow단계에서의 난독화를 할 수 있도록 한다. 또한 정적 코드 난독화를 수행하는 Pass도 작성한다. 그 후에 LLVM의 MachineFunctionPass를 이용하여 IoT 디바이스에서 주로 사용되는 ARM Architecture에 대한 Assembly Code 레벨 에서의 난독화를 수행하는 Pass를 작성한다.   
  
 그 후 펌웨어를 컴파일하고 ARM Cortex-M3 보드인 Arduino Due에 펌웨어를 Flashing 하고 Ghidra Tool을 이용하여 난독화에 대한 테스트 및 성능을 분석한다.

1. **진행 일정**

1. **결론**  
     
    본 프로젝트는 기존의 IoT 디바이스의 펌웨어 코드 분석을 어렵게 하여 실행 흐름 및 취약점의 분석을 어렵게 하기 위해 LLVM을 이용하여 펌웨어 코드의 난독화 방안에 대해 제안한다. 이를 LLVM Pass로 작성하며 실제 Arduino의 펌웨어에 적용하여 난독화 결과를 확인하고, 성능에 대한 테스트도 진행해본다. 이를 통해 IoT 디바이스의 Baremetal 펌웨어의 정적 분석이 더욱 어려워져 실행 흐름을 파악하는 것이 어려워질 것이다.

**참고 문헌**

[1] LLVM - <https://llvm.org>

[2] P. Junod, J. Rinaldini, J. Wehrli, and J. Michielin. “Obfuscator-LLVM - software protection for the masses”. In B. Wyseur, editor, Proceedings of the IEEE/ACM 1st International Workshop on Software Protection, SPRO'15, Firenze, Italy, May 19th, 2015, pages 3--9. IEEE, 2015.

[3] U.S.A. National Security Agency, Ghidra - <https://ghidra-sre.org/>

[4] Hex-Rays, IDA-Pro - <https://www.hex-rays.com/products/ida/>