유해 코드 탐지를 위한 소프트웨어 설계 및 구현

마복순*O, 신대철*, 임채호**, 원유헌*

*홍익대학교, **한국정보보호센터

The Implementation of Software for Hostile Mobile Code detection

Boksoon Ma*, Daechoul Shin*, Chaeho Lim**, Yoohun Won*

* Hongik University, ** Korea Information Security Agency

요약

유해한 이동 코드는 웹 환경에서 흔히 발견될 수 있으며 시스템에 적재되어 시스템 자원을 독점한다거나 부적절한 방법으로 자원을 탐닉하는 등의 불법적인 활동을 한다. 따라서 이러한 유해 이동 코드를 차단하기 위한 여러 가지 방법 즉, Firewall이나 Code Signing, SandBox, Playground등과 같은 보안 메커니즘이 제안되어졌으나, 이러한 보안 메커니즘들은 모든 이동 코드들에 대한 유해성 여부를 판단할 수 없고 또한 제한적인 실행으로 이동 코드의 효율성을 저하시키는 단점들을 내포하고 있다. 본 연구에서는 유해한 이동 코드를 진단·분석하고 유해 이동 코드 실행을 탐지·차단할 수 있는 탐지 소프트웨어를 개발하고자 한다.

1. 서론

이동 코드는 자바 애플릿(applet)이나 스크립트(script)와 같이 원격지에서 실행 가능한 코드로서 웹 브라우저를 통하여 코드를 다운로드 받아 클라이언트에서 수행이 가능한 코드이다[4]. 코드 자체가 이동을 하기 때문에 호스트에 도착한 이동 코드가 유해한 코드를 포함하고 있다면, 호스트에 악영향을 미칠 수 있다. 현재 일반 사용자들이 주로 사용하는 윈도우95와 같은 운영체제에서는 이러한 이동 코드를 이용한 공격을 막을 보호 대책이 없고 심지어 UNIX에서도 사용자의 권한을 가지고 이동 코드가 수행되기 때문에 사용자의 파일을 조작하거나 정보가 유출될 수 있다. 따라서 이동 코드를 사용할 경우 보안에 많은 관심을 가져야 한다. 기존에 연구된 방법에서는 이동 코드의 내용을 점검할 수가 없고 일부는 유해여부 판단을 사용자에게 부과하기 때문에 완벽한 이동 코드에 대한 보안을 한다고 볼 수가 없다.

본 연구에서는 유해 코드에 존재하는 특정 패턴들을 이용하여 이동 코드의 내용을 점검함으로써 유해 코드를 판별하고 자동으로 대응 할 수 있도록 하였다.

2. 관련 연구

유해한 이동 코드의 유형으로는 서비스 거부, 사생활 침해, 사용자를 귀찮게 하는 이동 코드 등이 있으며, 이동 코드를 수행한 사용자의 의도에 반하여 수행되는 것 등이 있다[7]. 이렇게 유해한 이동 코드를 차단하기 위한 방법으로는 [표 1]과 같은 방법들이 사용되었다.

유해 애플릿 차단 종류	유해 이동 코드차단 방식
Firewall	자바 필터링 기능을 통해 유해 이동 코드를 차단하지만, 일단 유입된 코드에 대해서는 차단할 수 있는 방법이 없음[2,3,8].
Code Signing	원시 코드에 특정 원칙에 따른 암호 내용을 삽입하여 서명된 코드에 따라 권한을 부여.
SadnBox	SandBox내에서만 이동 코드가 수행되어 실행 환경에는 피해를 줄 수 없도록 제한 JDK에서 구현된 방식[5,6].
Playground	사용자의 시스템 자원을 접근할 수 없는 곳에서 코드를 실행하도록 하는 방법. 하지만, 이러한 물리적 분리는 코드의 이동성에 따른 서버의 부하와 코드서버의 로드를 감소시키는 노력이 요구됨[1].
Java Filter	클래스 로더와 애플릿 사이에 "Guard"라는 필터를 위치시켜 신뢰할 수 없는 애플릿을 방지하고자 함. 이러한 필터는 URL 리스트형태로 존재하며 사용자가 관리하도록 함[3].

[표 1] 유해 이동 코드 차단 방법

본 연구에서는 Java Filter를 바탕으로 기존의 Guard가 URL 리스트를 통해 유해 코드를 차단하던 방식을 패턴 매칭과 패턴 메커니즘으로 분류하여 유해 코드의 내용을 통해서 유해 코드를 차단하는 새로운 방식을 시도하였다.

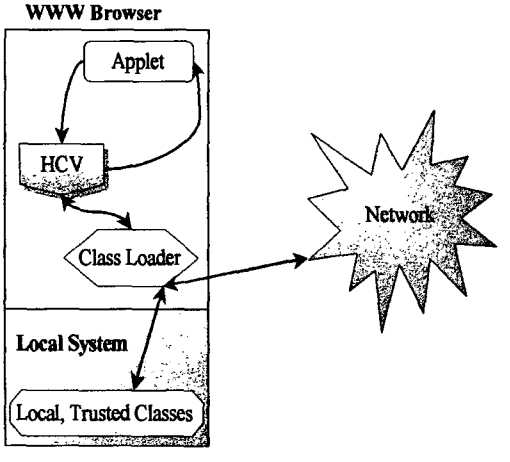
3. 시스템 설계 및 구조

앞에서 살펴본 Java Filter는 URL을 이용한 패턴 매칭이므로 모든 구문에 대해 정교하지 못할 뿐만 아니라 URL 리스트를 사용자가 직접 관리해야 하는 부담을 안고 있다.

본 연구에서는 유해한 패턴들을 데이터베이스로 저장하여 시스템에 접근한 유해 이동 코드의 내용을 분석하여 코드 내에 존재하는 유해 패턴을 검출해 이에 대해 자동적으로 대응할 수 있도록 하였다.

3.1 전체 시스템 구조

자바 바이트 코드의 수행도중 새로운 클래스를 요구할 때마다 클래스 로더(class loader)가 클래스 파일을 가져오기 위해 불려진다. 클래스 로더를 통해 애플릿이 생성되면 지역 시스템으로부터 또는 네트워크를 통해 요구되어지는 클래스를 가져온다. 이 때 HCV(Hostile Code Verifier)가 클래스의 내용을 분석하여 이동 코드의 유해 여부를 판단한다.



[그림 1] 전체 시스템 설계

3.2 HCV(Hostile Code Verifier)의 구성

HCV의 구성요소와 동작과정은 다음과 같다.

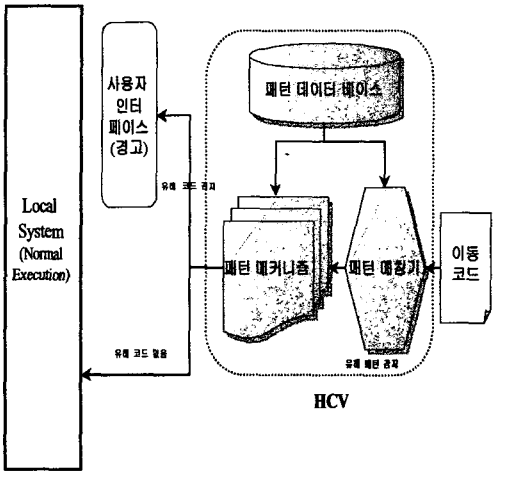
3.2.1 HCV의 구성 요소

HCV의 구성요소는 크게 세 가지로 이루어진다.

- 패턴 매칭기 : 패턴 데이터베이스에 저장된 유해 코드 패턴의 키워드가 이동 코드내에 존재하는지를 점검한다. 이렇게 점검된 내용을 패턴 메커니즘에 전달한다. 패턴 매칭기를 통과함으로써, Byte Code의 내용을 일단 분석하여, 복잡한 패턴 메커니즘을 빨리 수행할 수 있도록 하는 준비과정이라고 할 수 있겠다.
- 패턴 데이터베이스 : 유해한 이동 코드가 가지는 유해 패턴을 정형화하여 저장한다.
- 패턴 메커니즘 : 패턴 매칭기에서 전달 받은 결과와 패턴 데이터베이스에 저장된 유해 패턴들을 사용하여 이동 코드 내에 유해 코드가 존재하는지를 검사한다.

3.2.2 HCV의 동작 과정

HCV의 동작 과정은 [그림 2]와 같다.



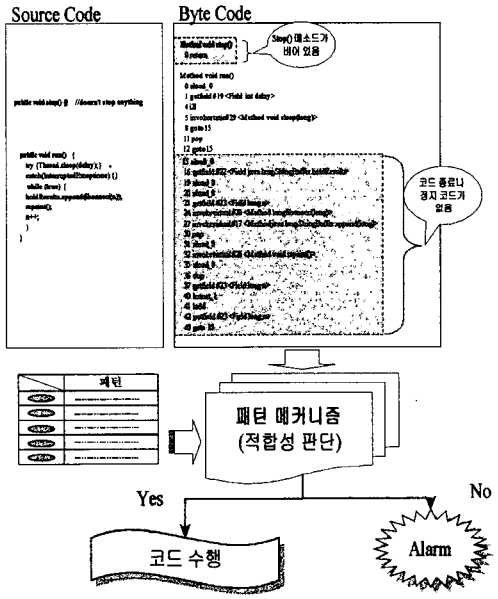
[그림 2] HCV의 동작 과정

① 패턴 매칭기는 이동 코드를 입력으로 받아 패턴 데이터베이스의 키워드와 비교를 통해서 유해 코드의 포함 가능성을 일차로 판단한다.
② 패턴 매칭기에서 나온 결과를 바탕으로 패턴 데이터베이스의 유해 패턴을 이용하여 패턴 메커니즘을 적용시켜 유해 코드 여부를 최종 판단한다.
③ 유해 코드가 아니라고 판정이 되면, 로컬 시스템에서 이동 코드의 내용을 실행시키도록 한다.
④ 패턴 메커니즘을 통해 유해 코드로 판정되는 경우 유해 코드를 처리하기 위한 동작을 취하도록 한다. (현재는 사용자 인터페이스를 통해서 경고를 한다.)

4.실험

[그림 3]에서와 같이 일단 이동 코드의 내용을 패턴 매칭기를 통해 유해 패턴의 키워드를 통해 검사한다. 코드 내용 중에서 루프가 존재함을 검출하고, 패턴 메커니즘으로 단계가 이동한다. 패턴 메커니즘을 이용하여 이동 코드의 유해성을 판단할 때는, 패턴 매칭기를 통해서 전달받은 결과값에 대한 비교가 우선한다. 또한, 추가적으로 패턴 메커니즘을 통한 검사로 코드 내의 stop()메소드가 비어 있어있음을 발견하게 된다. stop()메소드가 비어있는 것은 유해하다고 판단하게된다.

패턴 데이터베이스의 유해 코드 패턴과 패턴 메커니즘을 통해서 [그림 3]의 이동 코드 내에 존재하는 루프에는 코드를 종료시키거나 루프를 벗어 날수 있는 조건이 명시되어 있지 않기 때문에 무한루프라고 판단을 내릴 수 있다.



[그림 3] 패턴 메커니즘과 패턴 데이터베이스를 통한 유해 이동 코드 판정 과정

5. 결론

기존의 Java Filter에서는 유해한 이동 코드 방지를 위해 이동 코드의 URL을 검사하여 유해한 이동 코드를 필터링을 하였으나 본 연구에서 구현한 HCV는 이동 코드의 내용을 분석하여, 코드 내에 존재하는 유해 코드의 패턴을 점검하여 유해 여부를 판단함으로써, 좀더 정확하게 유해한 이동 코드를 검출해 낼 수 있다.

향후 연구 과제로는 우선 데이터 베이스에 저장된 유해 패턴의 내용을 더욱 추가하는 것이다. 그리고, 입력받은 이동 코드와 패턴 데이터베이스를 비교하여 새로운 유해 패턴을 유추할 수 있는 방법을 고찰해 볼 예정이다.

<참고문헌>

[1] Dahlia Malkhi, Michael Reiter, IEEE Symposium on Security and Privacy, May 4 1998, Secure Execution of Java Applets using a Remote Playground

[2] David Martin (Boston University), S. Rajagopalan (Bellcore), and Aviel Rubin (Bellcore) exploring the idea of using a firewall to protect against hostile applets., Blocking Java Applets at the Firewall

[3] Dirk Balfanz, Ed Felten. Technical Report 567-97, De partment of Computer Science, Princeton University, October, 1997. , A Java Filter

[4] Ellen Latz, Security Risks in Java and Javascript, http://www.rvs.uni-bielefeld.de/lecture/wwwdesign/ss96/ha-usarbeiten/elatz/

[5] L. Gong, M. Mueller, H. Prafullchandra, and R. Schemers. Published in Proceedings of the US-ENIX Symposium on Internet Technologies and Systems, Monterey, California, December 1997. , Going Beyond the Sandbox: An Overview of the New Security Architecture in the Java™ Developement Kit 1.2

[6] L. Gong and R. Schemers. Published in Proceedings of the Internet Society Symposium on Network and Distributed System Security, San Diego, California, March 1998, Implementing Protection Domains in the Java™ Development Kit 1.2

[7] Mark D. LaDue, Hostle Applets on the Horizon

[8] The original IEEE Java Security paper by the Princet on Team. An excellent reference.,Java Security: From HotJava to Netscape and Beyond