코드 복잡도와 가독성 분석을 통한 악의적인 리디렉션 탐지

최경미○ 홍충선*
경희대학교 컴퓨터공학과
{mi9149○, cshong* }@khu.ac.kr

Malicious redirection detection through code readability and complexity analysis

Kyungmi Choi ○ Choong Seon Hong*
Department of Computer Science and Engineering, Kyung Hee University

요 약

웹의 취약점을 악용한 코드 주입(code injection)을 통해 웹페이지의 리디렉션이 가능하고, 이를 통해 사용자가 인식하지 못하는 사이 랜섬웨어와 같은 악성코드들이 사용자의 컴퓨터에 다운로드 된다. 본 논문에서는 시그니쳐 기반의 리디렉션 탐지를 피하기 위해 쓰는 대표적인 우회 방법인 문자열 split와 난해한 문법 사용을 통한 코드 난독화를 탐지하기 위해 코드의 가독성과 복잡도를 평가하여 이를 통한 악의적인 리디렉션 탐지를 제안한다.

1. 서 론

최근 다양한 악성코드와 변종 유형이 많이 발견되고 있다.[1] 악성코드의 변종이 많이 생겨날수록 해당 악성코드에 대한 대응책을 일일이 마련하기 힘들어진다. 또한 현재 탐지된 악성코드와 변종 유형을 개별적으로 대응하기에는 그 수와 종류가 많아 어렵다. 따라서, 악성코드에 감염되기 전에 감염 여부를 탐지하여 사전에 감염되는 것을 방지한다면, 다양한 악성코드에 대해 일일이 대응책을 세우지 않으면서도 사용자의 컴퓨터를 악성코드로부터 보호할 수 있다.

Redirection은 악성코드를 유포하는 주된 방식 가운데 하나이다. 예를 들어 사용자가 웹페이지에 삽입되어있는 광고를 클릭하여 해킹된 웹 사이트에 접속하게 되면, 사용자는 해당 광고 페이지로 이동하는 것이 아니라 크래커가 유도한 페이지도 이동하게 된다. 이렇게 이동하게 되면 크래커가 유도한 페이지에서 drive-by download와 같은 방법으로 사용자가 인지하지 못한 채 사용자의 컴퓨터가 악성코드에 감염될 수 있다. [2]

따라서 본 연구에서는 악성코드가 유포되는 주된 방식 중 Redirection을 통한 악성코드 감염 방지를 다루고자한다. 본 논문에서 제안하는 악의적인 Redirection 탐지방법은 코드의 가독성과 복잡도를 이용하여 악의적인 redirection을 검출하고, 이를 통해 사용자의 컴퓨터가 악성코드에 감염되는 것을 사전에 막는다. 악의적인 사용자 및 크래커가 Redirection을 통한 악성코드를 구현하기 위해서는 관련 스크립트/태그 및 의도한 코드를 웹 페이지의 취약점을 이용하여 삽입하는 방식인 code injection을 해야 한다.[6]크래커들은 여러 우회기법을 이용하여 Redirection과 관련된 특징적인 태그를 숨겨 특정 String을 이용하는 방식인 시그니쳐 기반 악성코드 탐지를 피한다.[7].[9]

우회 기법 중 대표적인 방법으로는 코드 난독화를 꼽을 수 있다. 코드 난독화 기법에는 대표적으로 escape()함수를 이용한 암호화, 난해한 문법 사용 또는 string split를 꼽을 수 있는데, 코드 난독화를 거친 코드는 사람이 이해하기 어렵거나 아예 불가능하다. 따라서 코드의 복잡도 및 가독성을 이용하여 코드 난독화를 이용한 악의적인 redirection을 검출하는 방법을 제안하고자 한다. 2장에서는 악의적인 Redirection 코드들의 특징과 특징들을 식별하기 어렵도록 난독화 하는 기법의 대표적인 방법을 다룬다. 3장에서는 본 논문에서 제안하는 악의적인 Redirection 탐지 기법에 대해 설명하고, 4장에서는 테스트 환경을 구축하여 악의적인 Redirection 탐지 기법의 성능을 평가한다. 마지막으로 5장에서는 결론 및 향후 연구에 대해 논한다.

2. 관련 연구

2.1 Redirection의 악의적인 이용

공격자가 악의적인 Redirection을 통해 사용자의 컴퓨터에 악성코드를 유포하기 위해서 취약점이 있는 웹페이지에 <frame>, <javascript>등을 이용하여 자신이 유도한 페이지로 이동하는 코드를 주입한다.[1] 그림 1은 meta의 요소 중 http-equiv를 이용하여 Redirection을 구현한 코드의 예시

이 논문은 2017년도 정부(미래창조과학부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임(2015-0-00274,(ICBMS-2세부) ICBMS 플랫폼 간 정보모델 연동 및 서비스 매쉬업을 위한 스마트 중재 기술 개발)
*Dr. CS Hong is the corresponding author

이다. 코드 가운데 'content = 0' 으로 표기된 부분은, 대기 시간 없이 공격자가 유도한 URL로 이동한다는 것이다.[3] 따라서Client는 지연 없이 바로 유도한 URL로 이동하기 때문에 사용자는 자신이 들어가고자 했던 웹 페이지에서 Redirection 되었다는 사실을 인지하기 어렵다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title>The Tudors</title>
    <meta http-equiv="refresh" content="0;URL='http://thethdors.example.com/'" />
</head>

<body>
    <p>This page has moved to a <a href"http://thetudors.example.com/">
      theTudors.example.com</a>.</p>
</body>
</html>
```

그림 1 meta refresh를 이용한 client side redirection.[3]

2.2 난독화 종류

앞서 다루었던 악의적인 Redirection 방법은 공격자가 Redirection을 위해 추가적인 코드를 웹 페이지의 취약점에 삽입한다. 그러나 이러한 악의적인 코드들은 비슷한 특징들을 가질 수 있다. 따라서 공격자는 이러한 특징들을 은폐하기위해 코드를 식별하기 힘들도록 난독화 기법을 사용 한다.

난독화 기법에는 여러 종류가 있다. 가장 대표적인 난독화 기법은 암호화 하는 방법과 Redirection의 특징적인 태그들을 한 눈에 알아볼 수 없도록 난해하게 코딩하는 방법이다.

가장 대표적인 코드 난독화 방법은 문자 split이다.

```
var myStr = "document.write('Malicious Code')";
eval(myStr);
```

그림 2 (a) original code[4]

```
var a = "Code')";
var b = "doc";
var c = "ument";
var d = "('Malicious";
var Str = b+c+".write"+d+a;
myFunc = eval;
myFunc(myStr);
```

그림 2 (b) string split를 통한 난독화 된 code[4]

문자 split기법을 이용하여 redirection의 특징적인 코드를 숨기면 시그니쳐 또는 패턴 기반 탐색을 회피 할 수 있다. 이 외에도 사용자가 보기에 해석하기 어려운, 잘 쓰지 않는 문법들을 이용하여 난독화 하는 방법이 있다. 예를 var a=("aaa" , "bbb")와 같이 결과가 한눈에 안 보이는 문법이나, 삼항 연산자에 문자 split기법을 적용하는 방법 등이 있다.

3. 제안사항

본 연구에서는 코드의 가독성 및 복잡도의 분석을 통해 해당 javascript 코드의 악의적인 Redirection 코드 포함 여부를 미리 탐지하는 방법을 제안하고자 한다.

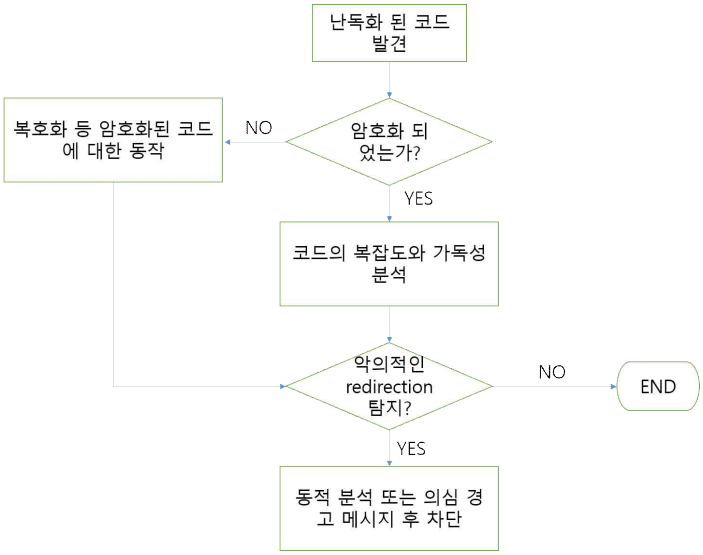

그림 3 악의적인 redirection 탐지 순서도

제안하는 탐지기법의 flow는 그림 3과 같다.

1) 난독화 된 코드를 발견하면 암호화의 여부를 판단한다. 난독화의 순기능은 해당 코드가 아무에게나 공개되는 것을 방지 하는 것이다. 만약 이러한 목적을 위해 코드를 난독화 하였다면 대부분 escape()함수 등을 이용한 암호화를 통해 난독화 한다. 위에서 소개한 난독화는 악의를 가진 코드를 숨기려는 의도 외에는 다른 것을 생각해 보기 힘들다. 때문에 난독화된 코드를 발견하였다면 해당 코드가 암호화 되었는지의 여부를 먼저 확인한다.
2) 암호화 되지 않고 난독화 되었다면 이에 대한 가독성과 복잡도를 계산하여 악의적인 redirection 코드인지를 분별한다.
3) 악의적인 코드로 판단된다면 이를 동적 분석의 대상으로 삼거나 또는 바로 차단한다.

기존의 탐지 방법은 난독화된 코드를 복호화 하고 이를 시그니쳐를 기반으로 하여 탐지한다. 이러한 탐지방법은 새로운 우회방법으로 공격할시 탐지할 수 없다는 문제점을 갖고 있다.

본 논문에서 제안하는 기법은 코드의 복잡도와 가독성을 이용하여 탐지하는 방법이다. 코드 복잡도는 McCabe의 Cyclomatic complexity(MCC) 모델을 사용하였고, 가독성은 P.L Buse의 가독성 모델을 중 난독화된 redirection 코드에 부합하는 특징을 선별하여 계산한다. 대부분의 난독화된 redirection 코드는 낮은 복잡도와 낮은 가독성의 특징을 띄기 때문에 이러한 특징을 이용하여 탐지 한다면 새로운 우회방법으로 공격한다 할지라도 해당 악성코드를 탐지할 수 있다.

그러나 가독성 및 복잡도 분석과 같은 정적분석으로는 정밀하게 악성코드를 탐지 할 수 없다. 확실하게 탐지하기 위해서는 동적 분석을 해야 한다. 하지만 모든 코드들을 동적으로 분석한다면 많은 시간이 소요되기 때문에 반응시간이 중요한 웹의 특성에 부합하지 않게 된다.[8] 본 연구에서는 악의적인 redirection을 탐지하기 위해 먼저는 코드의 복잡도와 가독성을 기반으로 하여 1차

적으로 동적으로 분석할 대상을 걸러낼 것을 제안한다.

4. 성능평가

본 논문에서는 복잡도를 계산하기 위해 정적분석 도구인 오픈소스 complexity-report를 사용하였다. complexity-report는 javascript의 복잡도를 평가하기 위해 McCabe의 Cyclomatic complexity(MCC) 모델을 사용한다. MCC 모델은 분기 및 decision point를 기반으로 복잡도를 계산하는 모델이다. MCC의 공식은 그림 4와 같다.

$$M = E - N + X$$

M : McCabe Cyclomatic Complexity
E : The number of edges
N : The number of nodes or decision point(conditional statements)
X : The number of exits(return statement)

그림 4 MCC formula[10]

테스트 결과 Redirection은 코드 자체는 매우 간단하므로 암호화를 쓰지 않고 난독화한 코드 MCC 값이 1이 나왔고 그 화면은 그림 5다.

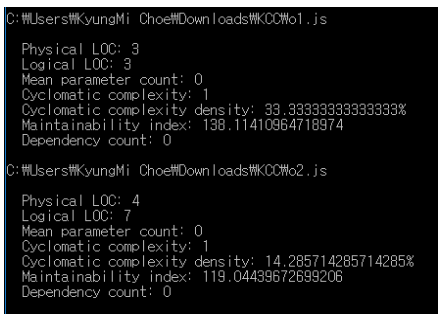

그림 5 complexity 분석 결과

	기준1	기준2	기준3	가독성
난독화 된 코드	0.04%	0.013%	0.8%	-0.2156
난독화 되지 않은 코드	0.02%	0.17%	0.09%	0.02

기준 1 : 전체 코드 중 "{","("의 차지 비율(negative)
기준 2 : 전체 코드 중 black의 차지 비율(positive)
기준 3 : 전체 코드 중 가장 긴 줄의 글자 수(negative)

그림 6 가독성 분석 결과

P.L Buse의 가독성 모델에 따르면, 가독성에 영향을 주는 변수는 코드의 '{', '(' 기호의 수의 평균, 코드 길이 등을 들 수 있다.[5] 난해한 문법으로 난독화된 코드는 Redirection이라는 간단한 기능을 구현하는 것에 비해 코드의 길이가 매우 길고 '{', '(' 기호의 개수가 많다는 특징을 갖는다. 또한 blank를 찾아보기 힘들다는 특징을 발견할 수 있다. P.L Buse의 가독성 모델을 바탕으로 가독성을 계산하면 그림 6과 같다.

5. 결론 및 향후 연구

본 논문은 코드의 가독성과 복잡도를 활용한 난해한 문법을 이용한 악의적인 Redirection 코드의 1차적 탐지 방법을 제안한다. 난독화 된 코드를 다시 원본 코드로 바꾸어 악성코드를 탐지하는 방법은 수많은 우회방법에 맞춰 대응해야 한다는 단점이 있다. 그러나 가독성과 복잡도를 기반으로 악성코드를 탐지한다면 새로운 우회기법으로 공격한다하여도 기존의 난독화 된 코드의 특징을 갖는 한 탐지될 수 있을 것이라 예상된다.

그러나 본 논문에서 제안하는 방법을 통해서 효과적으로 악성 코드를 탐지하기 위해서는 javascript 코드를 알맞게 분리시켜 크래커가 삽입한 코드만 따로 분리할 수 있어야 한다. 따라서 향후 javascript code를 적절하게 분리함으로써 탐지율을 높이는 방향으로 연구를 진행할 것이다.

참 고 문 헌

[1] 정진성, "랜섬웨어, 공격 경로 다각화로 맹공 펼친다", AhnLab, 2017.
[2] 최상용, 김용민, 노봉남, "Drive-by download 공격 및 대응 기술동향", 사이버보안연구센터, 2013.
[3] ERCIM, Keio, Beihang, "Using meta refresh to create an intant client side redirect", World Wide Web Consortium, 2016.
[4] Mehran Jodavi, Mahdi Abadi, Elham Parhizkar, "DbDHunter: An Ensemble-based Anomaly Detection Approach to Detect Drive-by Download Attacks", 2015 5th International Conference on Computer and Knowledge Engineering , 2015
[5] Raymond P.L. Buse, Westley R. Westley, "Learning a Metric for Code Readability", IEEE Transactions on Software Engineering, 2010
[6] Aditya K. Sood, Sherali Zeadally, "Drive-By Download Attacks: A Comparative Study, IT Professional, 2016.
[7] 지선호, 김휘강, "난독화된 자바스크립의 자동 복호화를 통한 악성코드의 효율적인 탐지 방안 연구", 정보보호학회 논문지, 2012
[8] "자바스크립트 정적 분석의 한계와 발전 방향", https://gist.github.com/VReality64/96a57ba698c5b44c35c7714ae8947863
[9] 지선호, 김휘강, "난독화된 자바스크립트의 자동 복호화를 통한 악성코드의 효율적인 탐지 방안 연구", 정보보호학회 논문지 제22권 제4호, 2012.
[10] Luis Atencio, "Measuring Code Complexity ", https://dzone.com/articles/measuring-code-complexity