플래시 TAG Frequency를 이용한 악성 플래시 탐지 기술

정욱현* · 김상원** · 최상용** · 노봉남***

*전남대학교 정보보호협동과정

**한국과학기술원 사이버보안 연구센터

***전남대학교 시스템보안연구센터

Flash Malware Detection Method by Using Flash Tag Frequency

Wookhyun Jung* · Sangwon Kim** · Sangyong Choi** · Bongnam Noh***

*Interdisciplinary Program of Information Security, Chonnam National Univ.

**KAIST CSRC

***System Security Research Center, Chonnam National Univ.

E-mail : pplan5872@naver.com · bestksw@kaist.ac.kr · csyong95@kaist.ac.kr · bbong@jnu.ac.kr

요 약

다수 사용자들이 이용하는 범용 어플리케이션에 존재하는 원격권한 획득이 가능한 취약점을 악용한 공격이 증가하고 있다. 특히, 플래시 플레이어는 브라우저, 오피스 등 다양한 플랫폼에서 사용이 되고 있어, 공격의 주요 대상이 되고 있으며 이를 대응하기 위한 연구가 진행되고 있다. 하지만 기존의 연구는 ActionScript에 대한 사전 분석된 특성을 이용해 탐지하기 때문에 신종/변종 탐지에 한계가 있다. 한계점의 개선을 위해 본 논문에서는 사전 분석된 특성을 사용하지 않고 플래시의 Tag빈도를 기계학습을 적용해 악성/정상 플래시에 대한 분류방법을 제안하며, 실험을 통해 제안한 방법이 기존 연구의 한계점을 극복하고 신종/변종 악성 플래시를 효과적으로 탐지 할 수 있음을 검증한다.

ABSTRACT

The vulnerabilities related to Flash player which is widely used in internet browsers and office programs are gradually increased. To detect Flash malwares, previous work focuses on predefined features of ActionScript. However above work cannot detect new/mutated Flash malwares, since predefined features could not cover the new patterns of new/mutated Flash mawares. To solve this problem, we propose a Flash malware detection method that uses machine learning to learn Flash Tag patterns and classify Flash by using machine learning.

키워드

플래시 · 취약점 · 유사도 · 기계학습 · 파일구조

Ⅰ. 서 론

어도비 플래시(Adobe Flash)는 벡터 그래픽, 애니메이션, 게임 등 멀티미디어를 제공해주는 소프트웨어 플랫폼이다.

다양한 멀티미디어 제공이 가능하기 때문에 영상 스트리밍, 광고, 대화형 멀티미디어 콘텐츠 제작을 할 수 있으며 특정 OS 플랫폼에 영향을 받지 않고 인터넷 브라우저가 지원만 하면 어디서든 실행이 가능하기 때문에 다수의 사용자들이 사용하고 있다. 어도비 플래시를 실행시켜주는 어도비 플래시 플레이어는 10억 개 이상의 데스크톱에서 사용[1]되고 있기 때문에 원격권한 획득 취약점을 이용하는 공격의 대상이 되기 쉽다.

실제 그림 1과 같이 전체 플래시 취약점의 81%는 원격권한 획득 취약점[2]이며 2006년부터 증가하여 2010년부터는 매년 60건 이상 발견되고 있으며 2014년에는 76건으로 점점 증가하는 형태를 나타내고 있다.

악성 플래시를 이용한 실제 사례로 2014년에

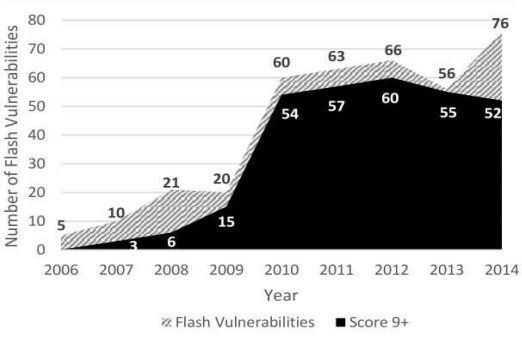

그림 1. 플래시 취약점 발생 현황(CVE기준)

발견된 “캐스퍼” 악성코드는 당시에 플래시의 알려지지 않은 취약점(CVE-2014-0515)을 이용하여 시리아 정부 웹페이지에 접근하는 사용자를 악성코드에 감염시켰다[3]. 국내에서도 악성 플래시 취약점을 Word문서파일 내부에 삽입하여 악성코드를 감염시키는 APT공격[4]을 수행하기도 했다.

악성 플래시를 탐지하기 위하여 기존 연구 [5, 11]에서는 플래시 ActionScript 내 코드분석을 통해 쉘코드, 코드 난독화, 외부 자바스크립트 호출, 블랙리스트 URL 접속 등의 특성들을 탐지하는 기법과 취약점이 있는 플래시 Tag 무결성 검사를 통해 탐지하는 기법 등이 연구되었다. 하지만 이와 같은 탐지 방식은 사전에 분석을 통해 정의된 속성의 경우 에만 탐지가 가능하다.

따라서 본 논문에서는 정상/악성 플래시 파일의 구조 속 Tag 정보를 기계 학습에 적용하는 새로운 탐지 방식을 제시한다. Tag에는 실제 플래시가 동작하기 위한 RAW정보를 담고 있다. 모든 플래시는 원하는 동작을 실행하기 위해서 그에 맞는 Tag를 포함해야 한다. 이 때문에 플래시의 Tag만을 통해 악성 플래시 탐지가 가능하다.

이 논문에서는 다음과 같이 구성된다. 2장에서는 플래시 파일의 구조와 동작 방식, 기존 악성 플래시 탐지 방법에 대한 관련 연구들을 살펴본다. 3장에서는 악성 플래시가 기존 연구들에 대한 탐지 우회를 가능케 하는 공격 방법 2가지를 설명한다. 4장에서는 악성 플래시 탐지를 위해 플래시 Tag 빈도를 기계학습에 적용하는 방법을 제시한다. 5장에서는 우리가 제시한 모델이 정확하게 탐지되는지 확인하고 왜 이것이 가능한지 고찰한다. 6장에서는 제안한 방법과 실험 결과에 대한 요약을 하고 결론을 짓는다.

II. 배경지식

2.1 플래시 구조

사용자는 웹 브라우저의 플러그 인 형태로 존재하는 플래시 플레이어를 통해 플래시 파일에 담긴 애니메이션을 볼 수 있다. 플래시 파일[6]은 그림 2와 같이 헤더와 여러 개의 블록형식으로

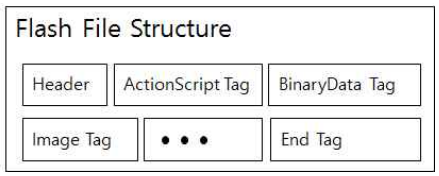

그림 2 플래시 파일 구조

나누어진 Tag로 구성되어 있다. 이러한 Tag들은 각각의 기능에 맞는 정보를 저장하고 있으며 플래시 플레이어는 이러한 태그정보를 읽어 실행한다.

특히 ActionScript Tag는 벡터 애니메이션, 스트리밍 서비스, 자바스크립트 등과의 연동 및 제어 기능을 제공하며 플래시 플레이어는 ActionScript를 실행하기 위해 AVM(ActionScript Virtual Machine)을 가진다.

2.2 악성 플래시 탐지 기술

기존 논문의 악성 플래시 탐지 방식 [5, 11]은 ActionScript에 대한 동적(디버깅, 후킹 기술) 및 정적(Decompiler [10]) 분석을 하여 악의적인 플래시 특성을 파악하여 탐지하는 방식을 이용한다. 표 1은 기존 논문에서 정리한 특성이며, 이들이 있을 경우 악성 플래시로 탐지한다.

표 1. 탐지 특성

특성	탐지 인자	분석
쉘코드, 난독화	특정 문자열/함수	정적
환경 검사	악성 URL, 외부 연결	정적
실행 동작	시간, opcode	동적
알려진 취약점	특정 Tag 무결성	정적

하지만 ActionScript에 대한 사전에 악성 플래시의 분석을 통해서 탐지 인자들을 정의하였기 때문에 이들을 사용하지 않거나 회피 기술을 이용하는 신종/변종 악성 플래시의 경우에 탐지가 어렵다.

2.3 기계 학습

기계학습은 입력되는 데이터로부터 컴퓨터가 학습할 수 있도록 하는 방법이다. 최근에 각광받는 Deep Learning은 Neural Network 알고리즘 일종으로, 다른 알고리즘들(Support Vector Machine, Nive Bayes 등)의 경우 입력된 데이터에 대해서 linear 함수로만 분류하였으나, 방대하면서 복잡한 데이터(이미지 등)들을 Deep Learning 내부의 수많은 히든 레이어(linear 분류 함수)의 조합으로 생성된 Neural Network(non-linear 분류 함수)를 통해 이미지 분류에서 뛰어난 성능[12]을 보여주었다. 하지만 연산 량이 많다는 단점이 존재한다.

III. 악성 플래시 탐지 우회 기법

3.1 ActionScript 은폐

ActionScript 코드 정보 분석을 이용한 악성 및 정상에 대한 탐지 방법이 증가하면서, 이를 우회하기 위해 악성 ActionScript를 다른 Tag로 숨기는 탐지 회피 방식이 나타나고 있다. 예를 들어 쉘코드를 그림 3의 0x100 Tag1 영역에 숨긴 후, 그림 3의 0x200를 실행하면 Tag1의 주소로 jump 하여 숨겨진 쉘코드가 실행된다.[7]

```
0x100 : Tag1 header with Shellcode
0x104 : Shellcode in Tag1
.....
0x200 : DoAction Tag
0x204 : jump -0x100
```

그림 3. Tag의 위치를 변경하여 쉘코드 은폐

이와 같이 원본 ActionScript 영역을 다른 주소로 이동 시킬 경우 정적분석 도구를 이용하여 이를 탐지하기 어렵다. 그 이유는 정적 분석 도구는 사전에 정의되어 있는 코드 영역만 분석을 수행하기 때문이다.

3.2 Non-ActionScript 기반 취약점

ActionScript를 사용하지 않고 특정 Tag의 취약점을 이용한 공격이 가능하다. 예들 들면 DefineSceneAndFrameLabelData Tag의 negative scene count value 취약점을 통해 공격이 가능한 CVE-2007-0071 취약점이 발견되었다.[14]

기존 연구에서는 이를 해결하기 위해 해당 Tag만을 위한 검사를 추가하였다. 하지만 이와 유사한 신종 취약점이 발견되면 신규 취약점 Tag에 대한 검사를 하지 않으므로 탐지가 불가능하다.

IV. 제 안

우리의 디자인 목표는 앞서 언급한 연구들의 한계점을 극복하기 위해서 플래시가 가진 모든 Tag빈도를 기계학습에 입력하여 정상/악성 플래시 Tag 빈도를 학습하게 하고, 정확한 분류가 되도록 하는 것이다. ActionScript 특성 정보를 사용하지 않고 플래시 Tag 빈도를 기계 학습에 적용하므로 ActionScript를 은폐하더라도 악성 플래시 탐지가 가능하다. 또한 신규 취약점 Tag가 사전에 학습되지 않더라도 이전 학습된 악성 플래시에서 빈번하게 쓰이는 Tag의 빈도을 사용하여 탐지가 가능하다. 만약 신규 취약점 Tag가 학습이 되면 이를 악성인자로 포함하며, 더욱 정확한 탐지가 가능해진다.

그림 4는 디자인 목표에 따라 설계된 탐지 시스템 모식도이며 3단계 처리(추출, 변환, 분류)를 거친다. 추출 단계는 플래시 파일에서 Tag 정보를 리스트에서 추출한다. 변환 단계에서는 앞서 생성된 리스트의 Tag값의 유무에 따라 값이 있으면 1 없으면 0으로 리스트를 업데이트 한다. 분류 단계에서는 앞서 업데이트된 리스트를 기계학습 모델에 입력한다.

특히, 분류 단계에서 Deep Learning 알고리즘의 하나인 DFNN(Deep Feed-forward Neural Network)을 적용하였다. Deep Learning은 방대하면서 복잡한 데이터들에 대한 학습이 가능하므로, Tag들에 대한 학습을 통하여 non-linear 분류 함수를 찾을 수 있다. 하지만 Deep Learning은 연산 량이 많아 GPU 이용 가능한 Theano[13]를 통해 속도를 최적화하였다.

V. 실험 결과

5.1 정상/악성 플래시 수집

온라인 통합 백신 검사 기능이 있는 Google's Virustotal Intelligence에서 악성으로 진단된 333개의 악성 플래시(2007.1.~2015.3.)를 수집했다. 이를 표 2에 따라 CVE 연도별 5그룹으로 나눴다.

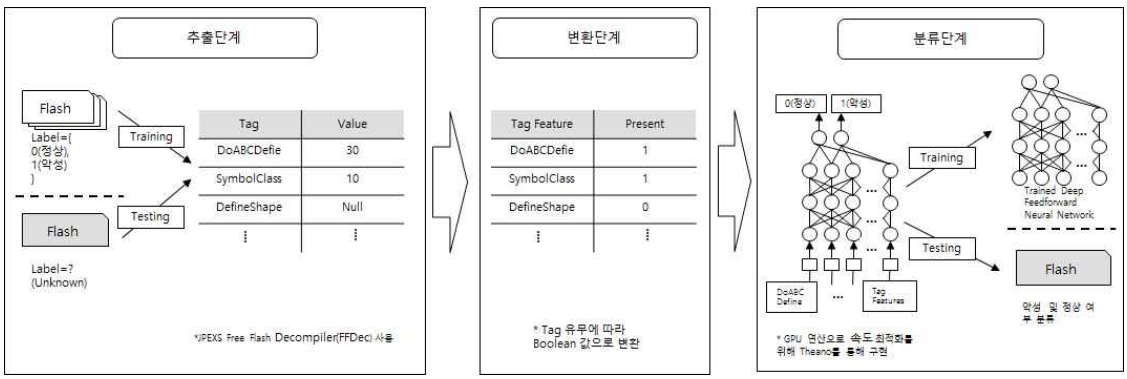

그림 4. 악성 플래시 탐지 시스템 모식도: 정상/악성 라벨이 있는 다수의 플래시 파일을 하나씩 학습시켜 학습된 DFNN 모델을 생성하고 임의의 플래시 파일을 테스트해 정상/악성을 분류한다.

표 2. CVE 연도별 샘플 수집

연도	7~11	12	13	14	~15.3.
수량	132	15	65	89	7

정상 플래시는 구글 검색(Filetype: swf)을 통해 무작위로 30000개를 수집하고 그 중에서 무작위로 308개를 추출하였다.

5.2 실험 방법

Tag를 이용한 새로운 탐지 방법이 ActionScript 은폐 문제점을 해결할 수 있으므로 제시한 시스템에 대한 탐지 성능이 충분한 지 검증을 하면 된다. 이를 위해 모든 Tag들에 대해 학습이 완료된 DFNN 모델의 정상/악성 가중치를 수식 (1)에 따라 측정하고, Tag유무별 가중치 측정값을 검증하기 위해 정상/악성 지수 알고리즘 표 3에 따라 통계 수치를 계산하여 확인한다.

$$Input\ Tag\ Weight\ T(i) = \sum_{h=1}^{h=k} W_{ih} \qquad (1)$$

W_{ih} : $Weight\ i \rightarrow h$

k : $The\ Number\ of\ Hidden\ Nodes$

Non-ActionScript 기반 취약점은 CVE-2007-0071 악성 플래시들을 신규 취약점이라 가정하고 이를 학습하지 않고도 잘 탐지하는지 보여줌으로써 문제점이 해결됨을 검증할 것이다. 이는, 새로운 Non-ActionScript기반 취약점이 출현하더라도 해당 취약점 Tag가 아닌 악성 플래시에서 빈번하게 쓰이는 Tag빈도를 통해 탐지할 수 있다.

표 3. Tag별 가중치를 검증하기 위한 알고리즘

```
1:  In samples, total_tag_cnt
2:
3:  tag_benign_cnt[total_tag_cnt] = {0}
4:  tag_mal_cnt[total_tag_cnt] = {0}
5:  for (flash in samples) {
6:    for (tag in flash) {
7:      if (flash is benign)
8:        tag_benign_cnt[tag.id]++;
9:      if (flash is malicious)
10:       tag_mal_cnt[tag.id]++;
11:   }
12: }
13: tag_id = 0
14: while (tag_id++ < total_tag_cnt) {
      Tag정상지수[tag_id] = tag_benign_cnt
    [tag_id]/(tag_benign_cnt[tag_id]+tag_mal
    _cnt[tag_id])
15:   Tag악성지수[tag_id] = tag_mal_cnt[tag
    _id]/(tag_benign_cnt[tag_id]+tag_mal_cnt
    [tag_id])
    }
16:
17: Out Tag정상지수, Tag악성지수
```

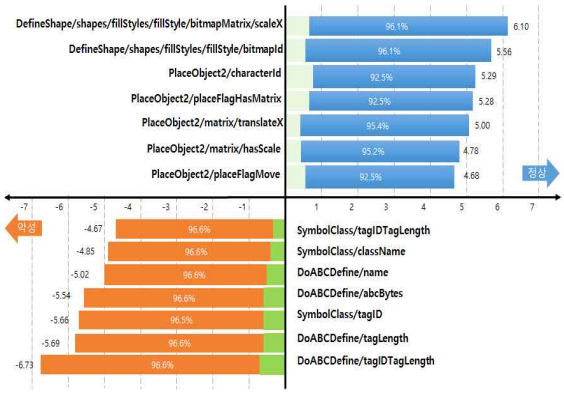

그림 6. DFNN의 Tag 가중치 측정 (정상/악성 Top 7) 및 정상/악성지수의 Tag 가중치 검증

5.3 실험 결과

수집한 모든 플래시 샘플을 이용하여 학습된 DFNN 모델을 생성하였다. 이 모델의 TAG가중치를 수식 (1)의 식에 따라 측정한 결과 그림 6에서처럼 ActionScript에 관련된 DoABCDefine Tag와 SymbolClass Tag에서 가중치가 악성으로 나왔다. 악성 플래시가 ActionScript를 이용하여 취약점 및 쉘코드를 실행하기 때문이다. 반대로 DefineShape Tag와 PlaceObject2 Tag는 실제 플래시 UI와 관련된 Tag로써 실제 상세한 UI 모양이나 벡터이미지에 관련된 Tag들이 가중치가 정상으로 나왔다. 이것은 악성 플래시와는 달리 정상 플래시는 실제 벡터그래픽 관련 서비스를 제공하는 영역이 상세하게 제공되기 때문으로 분석된다.

수집한 플래시 샘플에 대한 Tag 정상/악성 지수를 검증한 결과 정상으로 분류된 Tag는 평균 94.3% 정상지수, 악성에서는 96.6% 악성지수를 확인했다. 이는 Tag빈도만으로도 악성 플래시를 탐지하는데 유용하다는 것을 보여준다. 또한 표 4의 CVE 연도별 분류 실험 결과 또한 Tag 빈도를 통한 악성 탐지 방법이 신종/변종 악성 플래시에 대해서도 정확히 탐지할 수 있음을 보여준다.

표 4. CVE 연도별 분류에 따른 탐지결과 및 CVE-2007-0071에 대한 탐지결과 (실험대상에서 악성 플래시와 동일한 수만큼 정상 플래시를 사용)

학습대상	실험대상	FN (악성미탐)	FP (정상오탐)
2007-2011	2012-2015	0.86%	0.86%
2007-2012	2013-2015	0.62%	0.93%
2007-2013	2014-2015	0%	1.10%
2007-2014	2015	0%	0%
CVE-2007-0071 제외한 전체샘플	CVE-2007-0071 (8EA)	0%	0%

Non-ActionScript 기반 취약점도 잘 탐지되는지 검증하기 위해 표 4 마지막 행처럼 CVE-2007-0071 탐지 성능 실험을 진행하였다. 악성 미탐율(FN)은 0%, 정상 오탐률(FP)도 0%로 나타났다. 이는 학습되지 않은 Non-ActionScript 기반 취약점이라도 탐지가 가능하다는 것을 보여준다.

Ⅵ. 결 론

본 논문에서는 기존 연구의 한계점인 ActionScript를 은폐하는 탐지회피 기술, Non-ActionScript 기반 취약점 악용 등 탐지 회피 기술에 대응하여 신종/변종 악성 플래시를 탐지하기 위해 플래시 Tag빈도에 대한 기계학습을 적용하는 방법을 제안했다. 제안한 탐지 방법에 대해 정상/악성 플래시 파일을 대상으로 실험한 결과 정상/악성 플래시에서 자주 쓰이는 Tag들을 확인할 수 있었으며, 신종/변종에 대한 탐지도 가능함을 확인하였다.

참고문헌

[1] Adobe사 Flash Player 사용자 통계 (2015. 4. 13.), http://www.adobe.com/products/flashruntimes/statistics.html

[2] Adobe Flash Player CVE Detail, http://www.cvedetails.com/product/6761/Adobe-Flash-Player.html?vendor_id=53

[3] ESET Malware Blog Syria Targeted by CartoonMalware (2015. 3. 5.), http://www.eset.com/int/about/press/articles/malware/article/syria-targeted-by-cartoon-malware/

[4] Zdnet 어도비 플래시 취약점 이용 APT공격 '주의', http://www.zdnet.co.kr/news/news_view.asp?artice_id=20120509154839&type=det

[5] Timon Van Overveldt, Christopher Kruegel, Giovanni Vigna, "FlashDetect: ActionScript 3 malware detection", Springer Berlin Heidelberg, In Research in Attacks, Intrusions, and Defenses, pp. 274-293, 2012

[6] Adob사 Swf File Format Specification(version19), http://www.adobe.com/devnet/swf.html

[7] Fukami and Ben Fuhrmannek, "SWF and the Malware Tragedy.", OWASP, Application Security Conference, 2008

[8] DComSoft사 SWF Protector, http://www.dcomsoft.com

[9] DoSWF사 Professional Flash SWF Encryptor, http://www.doswf.org/

[10] JPEXS Free Flash Decompiler, https://www.free-decompiler.com/flash/download.html

[11] Sean Ford, Marco Cova, Christopher Kruegel, Giovanni Vigna, "Analyzing and Detecting Malicious Flash Advertisements", IEEE, Proceedings of the Annual Computer Security Applications Conference (ACSAC), 2009

[12] Classification datasets results, http://rodrigob.github.io/are_we_there_yet/build/classification_datasets_results.html

[13] Frederic Bastien, Pascal Lamblin, Razvan Pascanu, James Bergstra, Ian Goodfellow, Arnaud Bergeron, Nicolas Bouchard, DavidWarde-Farley, Yoshua Bengio, "Theano: new features and speed improvements". Cornell University Library , Presented at the Deep Learning Workshop(NIPS), 2012

[14] CVE-2007-0071 취약점 정보, http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2007-0071