JUNE 2011

virus BULLETIN

Fighting malware and spam

CONTENTS

ISSN 1749-7027

IN THIS ISSUE

2^{128} ADDRESSES

'Under IPv6, spammers could send out one piece of spam per IPv6 address, discard it and then move on to the next address for the next 10,000 years and never need to re-use a previous address.' Terry Zink explains why mail providers are not thrilled about using IPv6 to handle email.

page 4

CLASSES OF BROWSER MALWARE

Aditya Sood and Richard Enbody propose a taxonomy of browser malware with the aim to provide a better insight into the techniques and tactics used.

page 8

20 YEARS OF EICAR

Eddy Willems reports on some of the topics, debates and research presented at the EICAR 2011 conference.

page 16

'In the fight against cybercrime knowledge can be a very powerful weapon.'

Helen Martin, Virus Bulletin

EDUCATION, EDUCATION, EDUCATION

The end of May marked my tenth anniversary as Editor of *Virus Bulletin*. In some ways it seems like only yesterday that I was cautiously taking my first steps in the anti-malware industry, yet in other ways it's hard to believe that so much has changed in just a decade. There are a couple of things that have not changed: the warmth and friendliness of the members of this industry and the age-old debate over user education.

The importance of IT security in today's super-connected world seems to become ever more apparent on almost a daily basis as we hear reports of the servers of multinational companies being hacked, personal data being stolen, financial losses through phishing, targeted attacks, cyber espionage and so on.

PWC's 2011 Global State of Information Security Survey (which questioned more than 12,000 executives responsible for their organization's IT and security investments) reported that 20% of organizations had suffered financial losses as a result of cybercrime, 15% of organizations had suffered intellectual property theft, and 14% said that their brand or reputation had been damaged as a direct result of cybercrime. Meanwhile, digital investigations firm *Guidance Software* found that an astonishing 64% of employees are given no instruction about IT security in the workplace.

While the perpetual debate about the efficacy of user education runs on, a clear IT security policy in the workplace, along with guidance on how to adhere to such a policy, is surely one of the most basic steps an organization can take to help safeguard its systems.

Ensuring that employees understand their responsibilities – and that they are fully aware of the ramifications for any breaches of the policy – are also important factors. It is unlikely that many end-users within an organization will have a direct interest in IT security, so an effective way to get the message across is to educate in a way that relates to their jobs and which illustrates how their actions can play a role in safeguarding the company's assets (and ultimately their own job security).

Education at the IT administrator level is also important – in the fight against cybercrime knowledge can be a very powerful weapon. Learning events that provide solid, meaningful content from respected industry researchers can be an excellent resource for IT security admins – helping them to keep up to date with emerging issues and the latest defensive procedures. Indeed, providing such an educational forum is one of the key aims of *VB*'s one-day UK seminars (the second of which was run at the end of last month).

However, a big question mark remains over the education of the general public. One VB Seminar delegate asked last month: 'What efforts are AV vendors making to provide education for the masses?'. Most (if not all) AV firms make concerted efforts to raise general awareness of the threat landscape and the need for defensive measures. Within the industry we see frenetic blogging, tweeting and issuing of white papers in an attempt to spread the security message, and we often see company spokespersons talking to the media when a big cybercrime story breaks. But could vendors put more effort into providing education for the masses – to reach those who are not likely to be perusing technology blogs on a regular basis and who are not inclined to follow the latest IT security *Twitter* feeds?

Could AV firms look upon education programmes for the general public in the same way as free products? A growing number of security firms now provide versions of their products that are free for home use – the purpose of which is to enhance the security of the community as a whole, while also benefiting the company in question by building trust in the brand name. If AV firms were to invest more in education at the most basic level, finding ways to capture the imagination of the masses rather than sticking to their comfort zones of the online technology pages, could they both raise their own commercial profiles as well as benefiting the community?

NEWS

STANDALONE COMPARATIVES

From this month, *VB* will be taking a slightly different approach to the publication of VB100 and VBSpam comparative reviews. The comparatives will henceforth be published as standalone articles with a publication date of the middle of the month. As with the magazine, subscribers will be notified by email when the comparatives are available to download. This will allow more time for the preparation of the reviews, the checking of results and so on, without delaying the publication of the magazine.

APPLE ADDS DAILY DEFINITION CHECKS

After a recent increase in malware targeting *Mac OS X*, *Apple* has stepped up its security offerings by adding functionality to its *XProtect* program that will check for new malware definitions on a daily basis.

The daily updating feature will be added to *XProtect* when the most recent security update – which adds detection for the MacDefender trojan – is applied. Until now, *Apple* has dealt with security incidents on a case-by-case basis, pushing OS changes as and when needed to address specific issues. The new move towards automatic updates should give *Mac* users better protection – at least until new *Mac* malware starts to appear on a more frequent basis.

MOBILE INSECURITY

A report compiled by *McAfee* and Carnegie Mellon University has revealed lax security when it comes to the use of mobile devices in the workplace.

A survey of more than 1,500 users of mobile devices and senior IT decision-makers found that, while 95% of organizations have a mobile security policy in place, only one in three employees are aware of such policies.

The survey found that almost half of users store sensitive information on their mobile devices – whether personal or business-related – and that 40% of organizations have had mobile devices lost or stolen. Almost 60% of the lost or stolen devices were said to contain business-critical data.

Meanwhile, four in 10 organizations allow employees to access the Internet and download mobile apps freely using their mobile devices – a risky strategy given the potential for third-party apps to harbour malicious code.

Some of the risks associated with the use of mobile devices will be highlighted at VB2011 in Barcelona in presentations looking at *Android* malware (specifically the dangers associated with the *Android Market* and third-party apps) as well as cell phone money laundering. See http://www.virusbtn.com/conference/vb2011/ for details.

Prevalence Table – April 2011 [1]

Malware	Type	%
Autorun	Worm	9.73%
VB	Worm	7.10%
Heuristic/generic	Virus/worm	7.06%
FakeAlert/Renos	Rogue AV	6.16%
Sality	Virus	5.28%
Conficker/Downadup	Worm	5.12%
LNK	Exploit	3.71%
Downloader-misc	Trojan	3.47%
Adware-misc	Adware	2.71%
Zbot/Zeus	Trojan	2.49%
Ircbot	Worm	2.41%
Slugin	Virus	2.15%
ArchSMS/Pameseg	PU	2.07%
BHO/Toolbar-misc	Adware	1.90%
AutoIt	Trojan	1.74%
StartPage	Trojan	1.74%
Heuristic/generic	Trojan	1.65%
Qakbot	Trojan	1.63%
Virut	Virus	1.61%
Cycbot	Trojan	1.50%
WinWebSec	Rogue AV	1.50%
Crypt	Trojan	1.46%
Agent	Trojan	1.40%
Iframe	Exploit	1.32%
FakeAV-Misc	Rogue AV	1.28%
Virtumonde/Vundo	Trojan	1.28%
Dropper-misc	Trojan	1.21%
Crack/Keygen	PU	1.17%
Redirector	PU	1.12%
Exploit-misc	Exploit	1.11%
OnlineGames	Trojan	1.06%
Ramnit	Trojan	0.99%
Others [2]		13.87%
Total		100.00%

[1] Figures compiled from desktop-level detections.

[2] Readers are reminded that a complete listing is posted at http://www.virusbtn.com/Prevalence/.

FEATURE 1

DIGGING THROUGH THE PROBLEM OF IPv6 AND EMAIL

Terry Zink
Microsoft, USA

Over recent months, anti-spam bloggers have written about IPv6 and the challenges it poses for the email industry. John Levine, an author of numerous RFCs and a couple of books about spam fighting, wrote the following [1]:

> 'We will eventually figure out both how people use IPv6 addresses for mail, and how to manage and publish v6 reputation data, but until then, running a mail server on v6 will be a lot harder than running one on v4. And since you'll be able to handle all the real mail on v4, why bother [running a mail server to handle IPv6]?'

Barry Leiba, another email security writer, writes the following on *Circle ID* [2]:

> 'John Levine has one approach: leave the email system on IPv4 for the foreseeable future. Even, John points out, when many other services, customer endpoints, mobile and household devices, and the like have been – have to have been – switched to IPv6, we can still run the Internet email infrastructure on IPv4 for a long time, leaving the IP blocklists with v4 addresses, and a system that we're already managing fine with.
>
> 'Of course, some day, we'll want to completely get rid of IPv4 on the Internet, and by then we'll need to have figured out a replacement for the IP blocklist mechanism. But John's right that that won't be happening for many years yet, and he makes a good case for saying that we don't have to worry about it.'

Both writers are saying the same thing, and I have been on discussion threads where the consensus was similar: there is no agreement on how to handle IPv6 over email in the short term, but eventually it will have to be figured out. There are some who believe that mail will never move to IPv6 and some who think that it will go there one of these days. In the meantime, we just use IPv4 to send mail.

To expand a bit on what both writers are saying, the biggest reason why mail providers are not thrilled about using IPv6 to handle email is because there is currently no way to deal with the problem of abuse. Today, spammers make extensive use of botnets. Each day, they compromise new machines and start using them to spew out spam. Each of these bots uses a different IP address, and the IP addresses change all of the time. If you had 10,000 IP addresses that were sending out spam today, then tomorrow there would also be 10,000, but at least 9,700 of them would be different IP addresses from those used today.

The reason there is so much rotation in IP addresses is because modern spam filters make use of IP blocklists. When a blocklist service detects that an IP is sending spam, it adds it to the blocklist and rejects all mail from it. There are exceptions to this listing process such as a legitimate IP that sends a majority of good mail (such as a *Hotmail* or *Gmail* IP address), but in general, mail servers reject all mail from blocklisted IPs. The reasons they do this are:

1. 90% of all email flowing across the Internet (not including internal mail within an organization) is spam. If a sending IP is on a blocklist, a mail server can reject it in the SMTP transaction and save on all of the processing costs associated with accepting the message and filtering it in the content filter. Most mail servers today would topple over and crash if they had to handle all of the mail coming from blocklisted IPs because it would increase the number of total messages by a factor of 10.
2. Spam filters get slightly better anti-spam metrics by using IP blocklists. Content filters are good, but rejecting 100% of mail from a spamming IP address means that there is no possibility of a false negative from that IP address. By contrast, if a spam filter does not use an IP blocklist, the content filter has to learn to recognize the spam coming from that IP address, update the filter and then replicate out the changes. This is slower than pulling down a blocklist and then using it as the first line of defence. Without an IP blocklist, a spam filter will filter between 80% and 99% of the mail coming from a blocklisted IP. While many spam filters get close to that 99% range, it's still not 100%.

Blocklists are populated in a number of different ways. Some use spam traps to capture mail sent to email addresses that have never been used publicly, while others use statistical algorithms to judge that a sender is malicious (or compromised). Once the data is acquired, blocklist operators publish their lists in two ways:

1. They list individual IP addresses of all the servers that are sending mail.
2. They make use of CIDR (Classless Internet Domain Routing) notation. CIDR is a way to group large blocks of IP addresses. A provider would list a larger group of IP addresses in CIDR notation in order to save on space in the file so they don't have to list them one by one. For example, the *Spamhaus* Exploits Block List (XBL) is about 7 million entries (lines of text) and around 100MB in size. By contrast, the *Spamhaus* PBL (which lists IP ranges which should not be delivering unauthenticated SMTP mail to any Internet server) contains 200,000 lines of text (without

exceptions in ! notation) and is 6MB. However, the PBL is represented mostly in CIDR notation. If all of these ranges were expanded, it would be over 650 million individual IP addresses. That's a whole heck of a lot more IPs in the PBL for a whole lot less file size.

Today in *Forefront Online*, we run the XBL in front of the PBL and it blocks about four times as much mail as PBL (I don't know how much would be blocked if we ran them in reverse). The XBL is better at catching individual bots that are sending out spam but are not listed anywhere (they are new IPs), whereas the PBL is better at pre-emptively catching mail servers that should never send out spam (probable bots but it doesn't matter because they shouldn't be sending mail anyhow). However, if we had to list every single PBL IP singly instead of compressing it into CIDR ranges, and using the same ratio of 7 million IPs to ~100MB, then the PBL would be 9.4GB in total size. 9.4GB is a large file. It isn't completely unmanageable but it changes from being a minor inconvenience to being a major one. It takes a long time to download, upload and process a 9.4GB file. It's also easier to store the file entries in a database if there are only 500,000 entries (or even 7 million) vs 650 million of them. Databases that are as large as that run into the problem of scale.

The PBL and XBL are examples of why different styles of IP blocklists are required. The PBL lists 650 million IPs and we *still* have over 7 million IPs on the XBL that *aren't* on the PBL. Clearly, spamming bots can move around such that they are not published on the lists that have large address spaces listed. Bots are very good at hiding in places that are not blocked yet. Given enough space, spammers will hide because if they didn't they would not be able to stay in business. The problem that the industry faces is that as soon as we find a spammer's hiding space, we can block it for a while but the spammer will vacate it, relocate elsewhere and continue to spam[1].

And therein is the problem of IPv6. An IPv4 IP address consists of four octets and each octet is a number running from 0–255. This means that there are 256 x 256 x 256 x 256 possible IP addresses, which is 4.2 billion possible IP addresses (in reality, there are fewer than this because there are many ranges of IPs that are reserved and not for public consumption). Using our formula above, if you *had* to list every single IP address singly in a file, then the size of the file would be 61GB. There are few pieces of hardware that can handle that size of file in memory (whether you are doing IP blocklist look ups in rbldnsd or some other in-memory solution on-the-box). Processing the file and cleaning it up would take a very long time; you simply couldn't do it in real time where IP blocklists need to be updated frequently (once per hour at a bare minimum).

IPv6 multiplies this problem. We have seen that spammers already possess the ability to hop around IP addresses quickly. They do this because once an IP gets blocked, it is no longer useful to them. However, in IPv4 there are only so many places they can hide – 4.2 billion. In IPv6, though, there is virtually unlimited space in which to hide. To put it one way, there are 250 billion spam messages sent per day. Under IPv6, spammers could send out one piece of spam per IPv6 address, discard it and then move on to the next address for the next 10,000 years and never need to re-use a previous address. A mail server could never load a file big enough even for one day's IPv6 blocklist if spammers sent every single spam from a unique IPv6 address. Because spammers could hop around so much, IP blocklists would encounter the following problems:

1. They would get to be too large for anyone to download, process and upload.
2. They would be latent since by the time an IP was listed, spammers would have discarded it and moved onto the next IP address.

This is why no mail receivers are thrilled about the idea of using IPv6 to send mail[2]. They have to allow for the worst case scenario, which is that spammers will overwhelm their mail servers and drain processing power by having to deal with a tenfold increase in traffic.

HOW DO WE DEAL WITH IT?

One idea is to use whitelists instead of blocklists – block all mail from everyone and then maintain a central whitelist of good mail servers that send legitimate mail. The weakness here is that it defeats the whole purpose of email – that you can receive messages from people you haven't heard from before. This is known as the introduction problem. New mail servers are brought up all of the time. There's no way for you to know about it and the process of having to opt people in is a pain. This idea could be centralized, but what are considered legitimate mail servers for some people will not be legitimate for others.

Another idea is to take an unmanageable problem and break it down into a manageable one. Let's go back and take a look at how CIDR notation works and how blocklists take advantage of them. Consider the IP 216.32.180.16. This can be broken down into four eight-bit octets, and then combined to make one 32-bit number:

[1] This is the origin of the term 'whack-a-mole', a term the anti-spam industry borrowed from the carnival game. As soon as you whack one mole (or spamming bot), it hides and another pops up.

[2] Other readers will point out that the major reason it won't work is because a server could never cache that many IP addresses. While true, not every mail server looks up IPs on a blocklist via a DNS query.

A CIDR range operates on the bits of an IP address. An IP address is said to fall within a CIDR range if the first n-bits of an IP address are the same as the first n-bits of the range (where n is a number between 0 and 32). For example, let's take the range 216.32.180.0/24. If we convert this down to the bits that it represents, then this says include the range of IPs of any IP address that contains the first 24 bits; the '/24' says to take the first 24 bits:

216.32.180.16 is said to fall within the range 216.32.180.0/24 because the first 24 bits of the 32-bit representation of 216.32.180.16 is the same as the first 24 bits of 216.32.180.0/24:

```
216.32.180.0/24: 11011000001000001011010000000000
216.32.180.16:   11011000001000001011010000010000
```

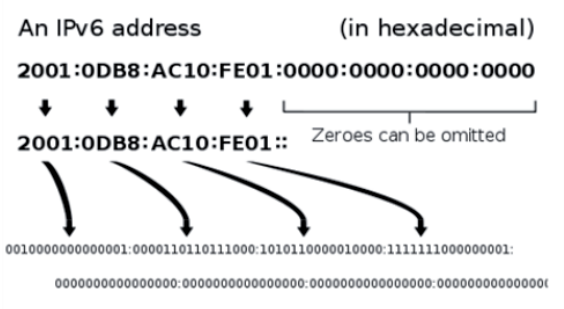
The first 24 bits match, but the last eight do not (illustrated by the '1' in green). However, this doesn't matter because we only need to match the first 24 bits. The red and blue parts match up and therefore 216.32.180.16 falls within the range of 216.32.180.0/24. If we take a slightly different IP address, 216.32.181.16, that will have a different 32-bit representation. It will not fall into the /24 range because the last bit does not match:

```
216.32.180.0/24: 11011000001000001011010000000000
216.32.181.16:   11011000001000001011010100010000
```

You can see that specifying things in CIDR notation is a very quick and easy way to list IPs on a blocklist. It makes sense to us humans reading it because we can interpret the numbers 'naturally', and it works from a technical perspective because it translates into bit-mapping. This is how PBL and other lists are able to manage so many IPs. The IP range 65.55.0.0/16 lists any IP that matches 65.55.xx.xx (65,536 IP addresses). They all fall into a logical range.

The number of IPs that fall within a CIDR range is evaluated as 2^(32-n) where n is the CIDR range (the number after the slash). A /24 (pronounced slash 24) is 2^(32-24) = 2^8 = 256 IPs, a /12 is 4,096 IPs, and so forth. The larger the CIDR range number n, the smaller the range of IPs it covers. To newbies, this is counterintuitive and takes a bit of time to get used to, but after a while you pick up the lingo. The smallest IP range is a /32 (one IP) whereas the largest is a /0 (every single IP).

IPv6 changes things because there are 128 bits in an IP address. Here's an example from *Wikipedia*[3]:



A /32 is no longer the smallest IP range, it is now /128. The size of a standard subnet is 2^64 IP addresses, the square of the size of the number of IPs in IPv4. While the planners of IPv6 don't think that the entire address space will be used, it will make network routing and management much more efficient.

One idea to make the problem of mail more manageable is to restrict the address space that is allowed to send mail. Ideally, we'd restrict where mail servers could send mail from. If we say that the number of individual mail servers in the world will never exceed 32 million (not unreasonable), or 2^25, then what if the 25 least significant bits were reserved for mail servers?

Right off the hop, any IP address that tried to connect to a mail server to send email that was outside the range (in hexadecimal) of 0:0:0:0:0:0:0:0 to 0:0:0:0:0:0:0200:0000 (or, :: to ::0200:0000) could automatically be rejected. This would be a PBL in reverse. Whereas PBL lists IPs that should never send mail, this algorithm would state that mail should only be accepted from IPs that are allowed to send it, and everything else should be rejected.

[3] Image taken from http://upload.wikimedia.org/wikipedia/commons/7/70/Ipv6_address_leading_zeros.svg.

This is similar to the idea of moving to a whitelist solution – in which mail is only accepted from the servers from which you want to receive mail. It solves the introduction problem because new people who you might want to hear from will be sending mail from a permitted set of IP addresses. All of the standard reputation tracking applies and the amount of space that spammers can hide in is restricted. If they want to send spam from servers that traditionally never send mail, they won't be able to do it because all of the good guys have already set up an agreement that says 'If you want to send mail to us, you must do it from this set of IP addresses.'

Randomizing the IP to send from a mail server that is outside the pre-agreed range will not make it easier for a spammer to hide because they wouldn't have been able to send mail from it anyhow. To make an analogy, if you send mail from an IP on the PBL and then switch to another IP on the PBL, it doesn't matter because in either case, your email would still be rejected.

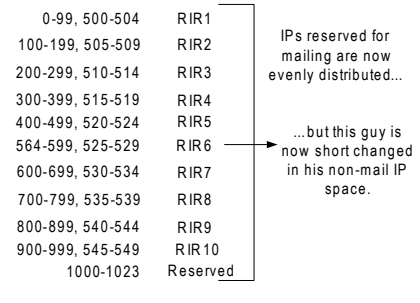
As it turns out, the least significant 64 bits are reserved in IPv6. The first 64 bits of the IPv6 address are the network address (48 bits routing prefix and 16 bit subnet id), and the last 64 bits are the interface identifier. The 64-bit interface identifier may be generated automatically from the interface's MAC address using the modified EUI-64 format, obtained from a DHCPv6 server, automatically established randomly, or assigned manually[4]. Using the least significant 64 bits will be problematic because an IP address is what we use to identify a device attached to the Internet and if they are already predefined by some algorithm, then we can't use them. The least 25 bits in an IPv6 address are already spoken for. But, we *could* allocate some other 32 million or so IP addresses (a /103) *somewhere* to be used for sending mail.

This would have to be managed to avoid it spiralling out of control – we need to know which block of IP addresses are reserved for sending mail and then how to share that range across millions of customers. For example, suppose we had 1,024 IP addresses to allocate and we decided to reserve 500–564 (1/16 of the Internet) for sending mail. How do we share it? Let's suppose that there are 10 major regional Internet registries (RIRs) who hand out the IPs to their customers (ISPs, people with their own home Internet permanent connections, businesses, etc.). Let's further suppose they decide to divide it up manually. RIR 1 gets addresses 0–99, RIR 2 gets 100–199, and so forth up to RIR 10 who gets 900–999 with the final 24 IPs being reserved for special functions. However, RIR 6 has all of the IPs that are permitted to send mail. That's not fair and nobody would agree to that.

0-99	RIR1
100-199	RIR2
200-299	RIR3
300-399	RIR4
400-499	RIR5
500-599	RIR6 → he has all the mail server IPs!
600-699	RIR7
700-799	RIR8
800-899	RIR9
900-999	RIR10
1000-1023	Reserved

Instead, we decide to divide things up more equitably. RIR 1 gets addresses 0–99 plus 500–504 (five IP addresses used to send mail). RIR 2 gets 100–199 plus 505–509 (also five IP addresses). Thus, each of the registrars has to 'logically' manage both its allocated range and its special email range. Instead of using CIDR ranges to allocate everything sequentially, there has to be a big table of who owns what. This gets very messy when you have to manage a lot of different IP ranges, particularly when the universe is as vast as IPv6. On the other hand, we're going to have to manage lots of IP addresses anyhow and this is just one more set of IPs that must be managed.

If IANA were to publish the rules and say that these are the designated IP ranges that are to be used to send mail, then everyone would be playing by the same set of rules right from the beginning. Not only that, but it's really not all that different from today. Regional Internet registries already allocate space to local Internet registries (LIRs), who then distribute the blocks down to their customers. When IANA provisions space, it would have to ensure that it provisions it such that it takes the special reserved range for mail into account. This is something that it already does today when it provisions IP space as well as geo-allocates it. Smarter people than me will need to figure out the necessary algorithms.



From these examples, it's clear that using an even distribution based upon numerical order is not going to work

[4] Because the MAC address of the machine is used to generate the interface identifier in some cases, this makes it easier to reject mail from these servers. You are no longer blocking an IP address that is subject to change in the case of DHCP, but instead blocking the actual piece of hardware which cannot change its MAC address. It's a more granular level of block that is more reliable… if we can determine that the IP was generated using the MAC address.

but reserving IP ranges and then mapping them out would. Even today, we have reserved IP address space that nobody is supposed to use (224.0.0.0 upwards is reserved for multicast, 10.0.0.0/8 is part of RFC 1918's internal address space, and so forth). The work that needs to be done is the following:

- A committee of people must figure out how many IP addresses should be reserved for sending mail – such that we are not likely to run out of space in a couple of decades – and then reserve an appropriate range for it.
- IANA must then reserve that space and come up with rules for how to hand that out to the RIRs. The RIRs must then come up with rules for how to allocate it to the LIRs, who then have to figure out how to allocate it to their customers. They then have to manage the infrastructure necessary to maintain the mappings of who owns what.
- Next, RFCs need to be written on how to send and receive mail over IPv6.
- Then, software vendors need to write code to perform IPv6 email transactions that are able to implement these rules.
- Finally, IP blocklist maintainers need to start populating their lists in IPv6 notation pursuant to the restrictions that are built into the RFCs.

It's a ton of work – years of it – but if we want to start receiving mail over IPv6 then that's what needs to be done.

This restriction of IP space for mail solves one problem but it doesn't solve others. On the one hand, it makes management of IPs scalable for machines that are bots. Today, most spam is sent from botnets. However, botnets do not always send out all of their spam directly – many bots compromise legitimate mail hosts or email accounts and send out spam that way, or create a throwaway account at a free email service and send out small amounts of spam from it before discarding it. This technique is used today but on a smaller scale than spamming directly. If we successfully solve the problem of direct-to-spam botnets, spammers will simply shift the bulk of their spamming to compromised or throwaway accounts.

I guess that means those of us in the e-security industry will always have a job. There's a silver lining to everything!

REFERENCES

[1] Levine, J. A Politically Incorrect Guide to IPv6, part III. http://jl.ly/Internet/v6incor3.html?seemore=y.

[2] Leiba, B. IP Blocklists, Email, and IPv6. http://www.circleid.com/posts/ip_blocklists_email_and_ipv6/.

FEATURE 2

A BROWSER MALWARE TAXONOMY

Aditya K. Sood, Richard J. Enbody
Michigan State University, USA

In this paper, we propose a taxonomy of browser malware. We classify browser add-ons, emphasizing their privileges. Since privileges impact the capability of malware, we use the resulting classification as a basis for our taxonomy. We hope that this taxonomy will provide better insight into the techniques and tactics used by browser malware, and assist in the development of defences.

BROWSER MALWARE – SUBVERTING DESIGN PRINCIPLES

Browsers are becoming a prominent medium for spreading malware infections because they are the interface to the web. Browser malware thrives by exploiting flaws in browser design principles. The design shortcomings most extensively exploited are:

- Insufficient isolation of components in existing browser designs. Browser malware exploits the principle of isolation to run arbitrary code in the context of a running browser to modify other components and corrupt the normal functioning of running components.
- Unrestricted communication among browser components. Browser malware manipulates the principle of integrity by eavesdropping on the inter-component communication interfaces and performing malicious hooks on the HTTP interfaces to control the traffic flow.
- Flaws in same origin policy implementation. Browser malware exploits the principle of same origin policy because persistent browser state is not effectively partitioned. This weakness allows the malware to conduct attacks by exploiting flaws in the same origin policy.
- Insufficient browser customization restrictions. Browser design permits extensive customization to enhance flexibility in the browser, but does so with insufficient restrictions. Browser malware exploits the principle of flexibility by executing malicious code as a part of extensible code.
- Vulnerable privacy models. Existing browser designs have adopted a privacy model which is not robust. Website cookies, user history, browser storage and so on are the foundation of users' privacy. Browser malware overcomes the privacy model by exploiting the

internal browser state to manipulate the interfaces that control the privacy components.

Our focus is on the *Mozilla Firefox* and *Microsoft Internet Explorer* browsers.

BROWSER MALWARE TAXONOMY (BMT)

In developing our taxonomy we define browser malware as a malicious piece of code that results in the circumvention of browser functionality or which uses a browser as a platform to infect operating system (OS) layers thereby. Our taxonomy, BMT, is based on the following critical elements of security that are exploited by the browser malware:

- Browser malware exploits security vulnerabilities in the components, plug-ins and OS layers.
- Browser malware contains malicious extensions that reside in the browser itself and exploit the characteristics of the default browser architecture.
- Browser malware uses the OS as a base to hook and hijack critical browser functions in order to take control of the browser communication channel.

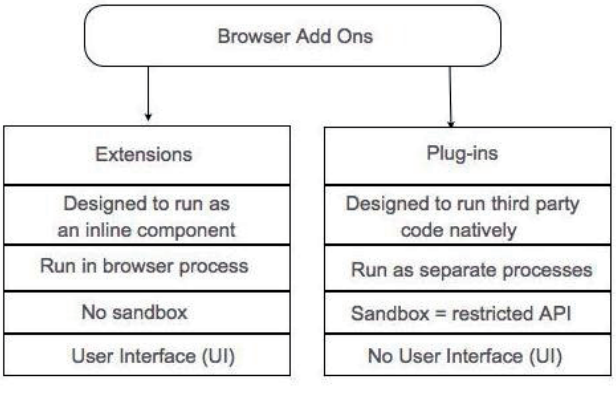


Figure 1: Generic browser extensibility model.

In order to discuss browsers we need to define some terms. To complicate this task, different browsers use different terminology. Browsers depend on a variety of 'add-ons' for extensibility, flexibility, and customization. With respect to browser malware we find it useful to divide add-ons into two categories: extensions and plug-ins (see Figure 1). Extensions run in the browser's context so they have the same rights as the browser itself. An example of an extension would be *NoScript* (for *Firefox*). In contrast, plug-ins run as separate processes and interact with the browser through an API that is more restricted than that used by an extension (some refer to this restriction as a 'sandbox'). *Adobe Flash* is an example of a plug-in. We have borrowed *Firefox* terminology, but the definitions fit other browsers. One can think of extensions as being part of the browser whereas plug-ins are separate, but intimately connected to the browser. It is the difference in privileges that differentiates add-ons with respect to malware.

Let's briefly take a look at *Microsoft*'s browser terminology. Browser Helper Objects (BHO) are treated as a part of the *Microsoft Internet Explorer* extension model [1]. In our taxonomy we are treating BHOs as extensions and ActiveX Objects as supporting programs for proprietary plug-ins. An Active X Object has a wide variety of functionality in the way it is used. For example, when installing a BHO from a remote location an ActiveX Object can be used to download that BHO. If an ActiveX Object is allowed to run from a browser, it can perform malicious functions by directly calling OS objects. Custom designed or proprietary plug-ins require an Active X Object to run dynamically if the plug-in is not permanently enabled in the browser.

A BHO is a DLL (Dynamic Link Library) that runs automatically when *Internet Explorer* is loaded. Extensions in *Internet Explorer* use the COM interface to design inline components that run in exactly the same manner as other proprietary components. BHOs extend the browser functionality to a great extent, but can also be used for nefarious purposes. Most of the time Active X Objects and BHOs are installed as DLLs in the operating system because *Microsoft* makes extensive use of DLLs for all types of operations.

Our proposed BMT uses this extension and plug-in distinction along with their different privileges to differentiate between the types of malware.

CLASS A BROWSER MALWARE

Class A browser malware exploits the default monolithic architecture of browsers. It installs itself as a browser component and utilizes the browser model to conduct attacks. This type of malware is quite dangerous because it functions as an inline component. Malicious extensions and browser rootkits fall into this category. Also, class A browser malware can exploit inherent vulnerabilities in the browser native components to download binaries into the operating system. Figure 2 shows the class A browser malware model.

As a browser component, class A browser malware has the full access rights of the browser and runs in the same memory context (address space) as other extensions, so it can perform the following operations:

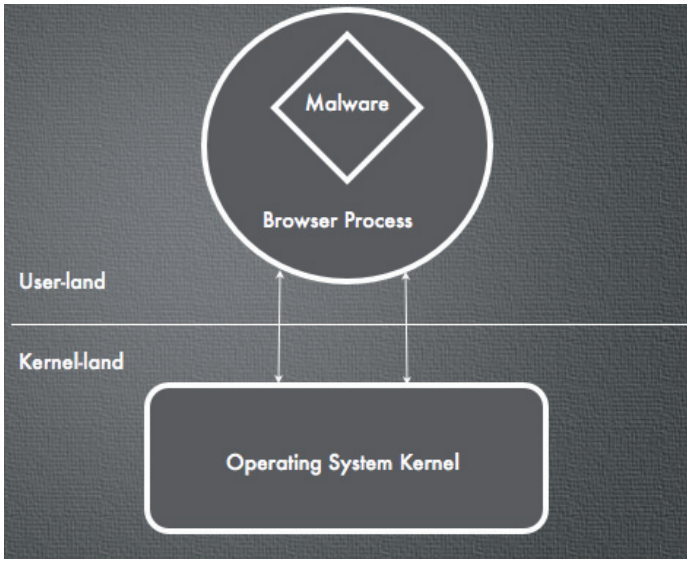


Figure 2: Class A browser malware.

- It is capable of reading and writing to disks, controlling network sockets, tampering with the browser's user interface, stealing stored data, altering the registry and modifying other extensions.
- It can exploit other installed extensions by using JavaScript wrapper functions to change their functionality. It can interact with installed plug-ins such as PDF or *Flash* in order to launch malicious code. In this way it can act as a carrier for trojans.
- Like a rootkit it can hide itself in the browser. For example, *Firefox* extensions can be hidden by manipulating parameters in the install.rdf file and using CSS [2] to install malicious extensions with transparent style metrics. Basically, the aim is to remove entries in the extension manager so that malicious extensions cannot be enumerated. In *Internet Explorer*, it is possible to hide extensions by hooking objects in Security Manager [3] and creating new objects such as invisible tags.
- Class A browser malware can also exploit the security vulnerabilities present in the different browser components such as the rendering engine, JS interpreter, browser engine, XML parser, networking modules and user interface. Exploitation of these vulnerabilities can result in successful execution of drive-by download attacks [4]. JavaScript heap corruption [5, 6] plays a critical role in the successful execution of these types of attacks.
- Class A browser malware can make explicit use of asynchronous HTTP requests via AJAX with associated events to generate listener functions for communication with third-party servers. It can use encrypted protocols for data transfer in a secure manner.

Examples of class A browsers include Firefox FormSpy [7], FFSniff [8], Sothink Web Video Downloader 4.0 spreading Win32.LdPinch.gen [9], Master Filer spreading Win32.Bifrose.32.Bifrose Trojan [9], Download.ject [10, 11] and MyWay Searchbar [12, 13].

Note that browser exploit packs such as the Phoenix [14] and BlackHole [15, 16] do not install any extensions, but do exploit browser vulnerabilities to download malicious executables in the system. Therefore, they display some of the typical behaviour of Class A browser malware.

CLASS B BROWSER MALWARE

Class B browser malware exploits vulnerabilities in the plug-in interface in order to cause an infection. Since most plug-ins originate from third-party vendors, inherent vulnerabilities in plug-ins play a critical role in determining the success of class B browser malware. Generally, plug-ins run as separate processes and are usually placed in a sandbox so the interface with the browser is restricted.

Plug-ins are platform-independent code so vulnerabilities in third-party code such as *Adobe Flash* can broadly impact browser security. The impact is significant because the risk of exploitation in third-party software is transferred to the browser. This interdependency results in hybrid vulnerabilities that exist when the plug-in code is run simultaneously with the browser. A malicious plug-in can use JavaScript in a web page (loaded in a sub window) to

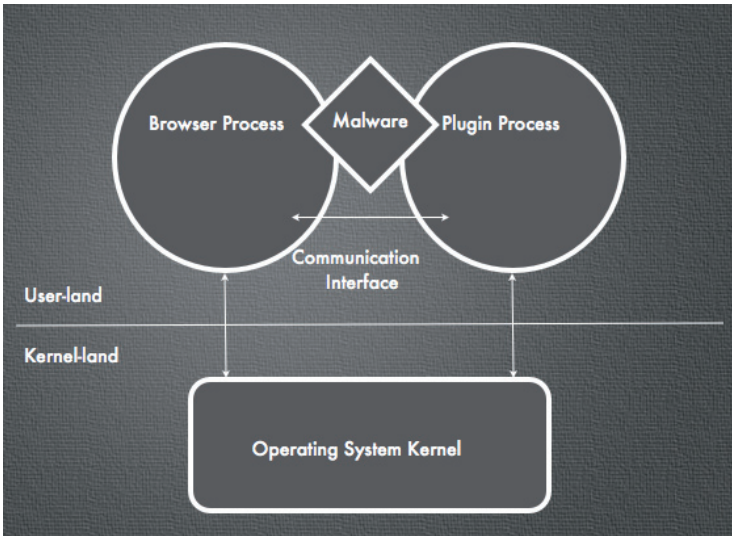


Figure 3: Class B browser malware.

perform various functions. A web page's JavaScript runs under restricted privileges (sandboxed) so it cannot interact much with the internal browser components. Of course, vulnerabilities in the sandboxing can allow modification of the internal browser components. In any case, plug-ins operate outside of browsers and can act as a parasite to use browser resources to spread infections.

Figure 3 shows the high-level view of class B browser malware.

Class B browser malware shows the following characteristics:

- Custom designed plug-ins can run malicious scripts in the browser and exploit DOM to carry out XSS for stealing sensitive information, phishing, malvertisements and social engineering attacks. In *Firefox*, malicious plug-ins make use of the Netscape Plug-in Application Programming Interface (NPAPI), which is used to design platform-independent code. NPAPI plug-ins are specifically used by class B browser malware for malicious Internet functionality which is a potential playground for spreading infections. In *Internet Explorer*, custom designed plug-ins use Active X Objects to execute arbitrary code that can manipulate browser resources.
- Malicious plug-ins can carry exploit code in order to drop binaries in the OS by exploiting vulnerabilities in the plug-in interface – so-called drive-by download attacks. Examples include the exploitation of vulnerabilities in the *Adobe Reader*, *Flash* and *Silverlight* plug-ins.
- Apart from JavaScript heap corruption, class B browser malware also uses evasive techniques such as RC4 encryption, splitter modules including variables and arrays, multiple level compressions and encoding as well as cross referencing of objects in order to evade detection. However, it supports both JavaScript and non JavaScript-based exploits.

Real-world examples of class B browser malware include Trojan.Pidief [17], Exploit.PDF-JS.Gen [18], SWF AdJack Gnida [19], Trojan:SWF/Redirector.I [20] and TrojanDownloader:SWF/Nerner.A [21].

Note that browser exploit packs such as BlackHole and Phoenix show characteristics of class B malware because these packs exploit vulnerabilities in third-party support plug-ins such as *Adobe Flash* and Java.

CLASS C BROWSER MALWARE

Class C browser malware typically resides in the operating system and exploits the browser interface. It also exploits OS layer vulnerabilities by using the access rights of the user of the operating system. In general, class C browser malware uses system-level APIs to attack the browser or plug-in processes in order to control the browser communication interface. This malware also contains system-level rootkits specifically aimed at exploiting the browser interface with the OS. Class C browser malware establishes itself in either the user-land or kernel-land layers of the OS, or some hybrid of the two. As noted earlier, both class A and class B browser malware play a crucial role in dropping class C browser malware in the OS by exploiting certain vulnerabilities in browsers and plug-ins. This observation indicates that dependencies exist between all classes of browser malware that are presented in this taxonomy. Figure 4 shows a high-level view of class C browser malware.

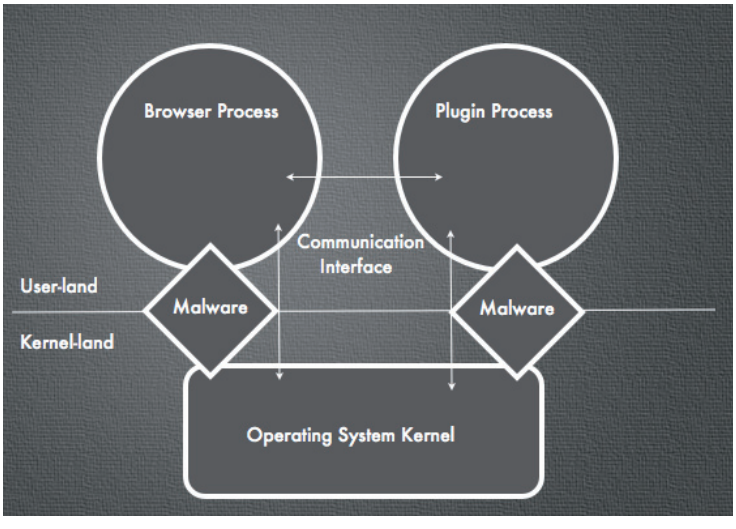


Figure 4: Class C browser malware.

Class C browser malware shows the following characteristics:

- This class of malware typically behaves like a rootkit and hides itself in the OS so that the possibility of detection is reduced. Class C browser malware basically aims to control the browser's communication interface with the Internet to manipulate the traffic flow. Class C browser malware implements hooking [22] in order to take control of the execution flow. In user-land space, this class of malware uses Import Address Table (IAT) hooking, inline hooking and DLL hijacking using APIs. In kernel-land space, this class of malware primarily performs hooking through the System Service Descriptor Table (SSDT) in order to control the browser-related functions in the kernel.

- Class C browser malware is capable of hooking the browser communication channel, traffic monitoring, injecting malicious traffic into the browser, circumventing OS firewalls, generating fake website pages and keylogging POST requests and all system-related activities. This ability is critical because class C browser malware has complete control of the standard HTTP functions in OS libraries which are used heavily by browsers for all types of work.

Real-world examples include the Zeus bot [23], SpyEye bot [24], IRC bots [25] and PRRF [26].

Note that browser exploit packs such as BlackHole and Phoenix actually exploit issues in browsers and plug-ins to drop class C browser malware into the system.

CONCLUSION

In this browser malware taxonomy we have presented three different classes of browser malware classified on the basis of a browser architectural model. In particular, differences in privileges are critical in the classification. Class A browser malware resides in the browser process and acts as an inline component of the browser. Class A browser malware exploits vulnerabilities in the browser process. Class B browser malware takes the form of malicious plug-ins and also exploits vulnerabilities in third-party vendor software. Class C browser malware resides in the operating system and controls the browser communication channel from outside.

We hope that this taxonomy will provide better insight into the techniques and tactics used by browser malware, and assist in the development of defences.

REFERENCES

[1] Microsoft. Verified Security for Browser Extensions. http://research.microsoft.com/pubs/141971/tr.pdf, 2010.

[2] Devaux, C.; Lenoir, J. Browser Rootkits. Hack.lu, 2008. http://archive.hack.lu/2008/lenoir_presentation.pdf.

[3] Wueest, C.; Florio, E. Firefox and Malware: When your browser bytes you. Virus Bulletin International Conference 2009. http://www.virusbtn.com/pdf/conference_slides/2009/Wueest-Florio-VB2009.pdf.

[4] Barwinski, M.; Irvine, C.; Levin, T. Empirical Study of Drive-by-Download Spyware. http://www.cisr.us/downloads/papers/06paper_spyware.pdf.

[5] Sotirov, A. Heap Feng Shui in JavaScript. Black Hat Europe 2007. http://www.blackhat.com/presentations/bh-europe-07/Sotirov/Presentation/bh-eu-07-sotirov-apr19.pdf.

[6] Chennete, S.; Joseph, M. Detecting Web Browser Heap Corruption Attacks. Black Hat USA 2007. https://www.blackhat.com/presentations/bh-usa-07/Chenette_and_Joseph/Presentation/bh-usa-07-chenette_and_joseph.pdf.

[7] Oswald, E. Trojan Hides Itself as Firefox Extension. BetaNews. http://www.betanews.com/article/Trojan-Hides-Itself-as-Firefox-Extension/1153934797.

[8] Costoya, J. Malicious Firefox Extensions. Trend Micro blog. http://blog.trendmicro.com/malicious-firefox-extensions/.

[9] Claburn, T. Mozilla Removes Two Malicious Firefox Add-Ons. Information Week. http://www.informationweek.com/news/security/vulnerabilities/222700171.

[10] Schneier, B. Schneier Micros. Schneier on Security. http://www.schneier.com/blog/archives/2004/10/schneier_micros.html.

[11] Wikipedia. Download.ject. http://en.wikipedia.org/w/index.php?title=Download.ject&oldid=423105396.

[12] Leyden, J. Dell rejects spyware. The Register. http://www.theregister.co.uk/2005/07/15/dell_my_way_controversy/.

[13] Wikipedia. MyWay Searchbar. http://en.wikipedia.org/w/index.php?title=MyWay_Searchbar&oldid=427757462, 2005.

[14] SecNiche Security. Phoenix Exploit Kit (2.4) – Infection Analysis. Malware At Stake Blog. http://secniche.blogspot.com/2010/10/phoenix-exploit-kit-24-analysis.html.

[15] Websense Blog. Black Hole Exploit Pack. http://community.websense.com/blogs/securitylabs/pages/black-hole-exploit-kit.aspx.

[16] SecNiche Security. Java OBE + Blackhole Dead Man Rising. Malware At Stake Blog. http://secniche.blogspot.com/2011/02/java-obe-tookit-exploits-blackhole-dead.html.

[17] Symantec Security Report. The Rise of PDF Malware. http://www.symantec.com/content/en/us/

enterprise/media/security_response/whitepapers/the_rise_of_pdf_malware.pdf.

[18] BitDefender. Exploit.PDF-JS.Gen. http://www.bitdefender.com/VIRUS-1000487-en--Exploit.PDF-JS.Gen.html.

[19] Lindner, F. Defending the Poor – Countering Flash Exploits. http://www.recurity-labs.com/content/pub/DefendingThePoor_26C3.pdf.

[20] Microsoft. Trojan:SWF/Redirector.I. http://www.microsoft.com/security/portal/Threat/Encyclopedia/Entry.aspx?Name=Trojan%3aSWF%2fRedirector.I.

[21] Microsoft. TrojanDownloader:SWF/Nerner.A. http://www.microsoft.com/security/portal/Threat/Encyclopedia/Entry.aspx?Name=TrojanDownloader%3ASWF%2FNerner.A.

[22] Wikipedia. Hooking. http://en.wikipedia.org/w/index.php?title=Hooking&oldid=428889341.

[23] Leyden, J. Zeus bot latches onto Windows shortcut security hole. The Register, 2010. http://www.theregister.co.uk/2010/07/27/zeus_exploit_shortcut_hole/.

[24] Coogan, P. SpyEye Bot versus Zeus Bot. Symantec Blog. http://www.symantec.com/connect/blogs/spyeye-bot-versus-zeus-bot.

[25] Honeynet. Tracking Bots and Botnets. http://www.honeynet.org/book/export/html/50.

[26] Tereshkin, A. Rootkits: Attacking Personal Firewalls. Black Hat USA 2006. http://blackhat.com/presentations/bh-usa-06/BH-US-06-Tereshkin.pdf.

[27] Bugzilla. Bug ID- 483086 non-http[s] SearchForm URIs should be ignored. https://bugzilla.mozilla.org/show_bug.cgi?id=483086.

[28] BitDefender. Trojan.PWS.ChromeInject.B. http://www.bitdefender.com/VIRUS-1000451-en--Trojan.PWS.ChromeInject.B.html.

[29] Nightangle, J. Firefox Malware. http://blog.johnath.com/2008/12/08/firefox-malware/.

[30] Louw, M.; Lim, J.; Venkatakrishnan, V.N. ExtensibleWeb Browser Security. http://www.cs.uic.edu/~venkat/research/papers/extensible-browser-dimva07.pdf.

FEATURE 3

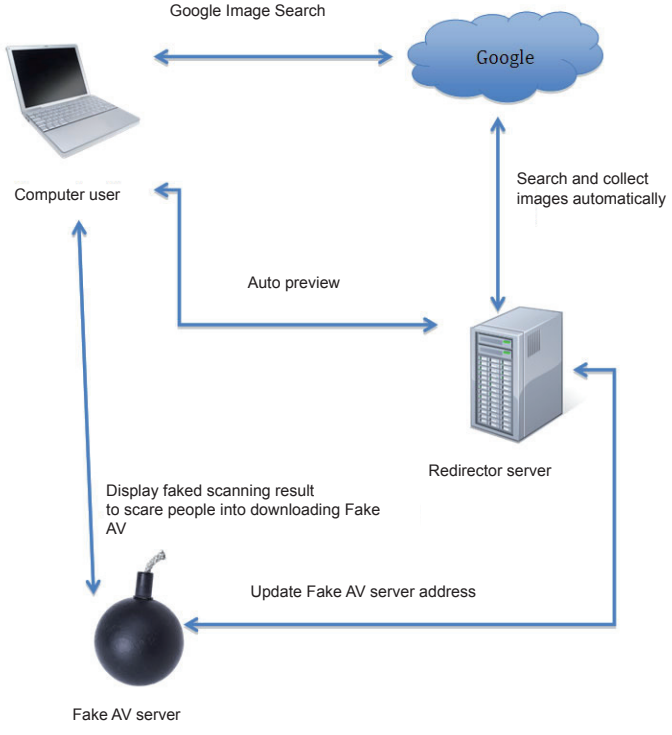
NEW TARGETED ATTACK VIA GOOGLE IMAGES

Robert X Wang
iSIGHT Partners, USA

Fake AV is a term used to describe threats which masquerade as real security products, fooling victims into believing that they have discovered infections on their machines. The 'products' then insist on payment for an activation/registration code in order to clean the supposed threats from the system.

As with many other threats, social engineering techniques are often used to spread fake AV. The malicious file may have a tempting name and arrive as an email attachment, a download link, or a shared folder. However, such methods require a considerable amount of user interaction – if the user doesn't download and execute the file, the fake AV will not be able to fool the user into believing their system is infected and subsequently generate revenue for the attacker.

A new method of targeted attack has recently been discovered in which the auto preview feature of *Google Images* is utilized to lure the user into downloading and purchasing the fake AV application. This method has proved to be very effective.



METHODOLOGY

After the user has entered a keyword and clicked the 'search' button in *Google Images*, the search engine displays all the relevant images it has found. When the user clicks on an image, *Google* both displays the original image and previews the web page on which the image is hosted.

When an image from a malformed URL (redirector) is clicked, the redirector will examine some of the following parameters and then decide what to do next:

- The referer URL in the HTTP header
- The *Google Image Search* keyword
- The User-Agent string in the HTTP header

If the referer URL is not from *Google Image Search* and/or the *Google Image Search* keyword is not detected, the redirector will display a clean web page. This tactic is used by the fake AV to prevent it from being spotted.

If the User-Agent string is not supported, the redirector will display a clean web page or else redirect to another clean page. Different types of browser will display different pages.

If all checks are passed, the redirector will send the browser to another malformed page, which lures the user into downloading the fake AV application or its downloader.

The popularity of *Google Image Search* makes this method extremely effective.

EXAMPLE

If any of the images shown in Figure 1 are opened, 'hxxp://oliviercassab.com/samuri-harvest-autumn-wallpapers/' will be loaded and previewed. If the same link is opened directly in a browser, a clean page will be displayed (Figure 2). The clean page will also be displayed if the referer URL is not from *Google Images* or the search keyword does not appear to be correct.

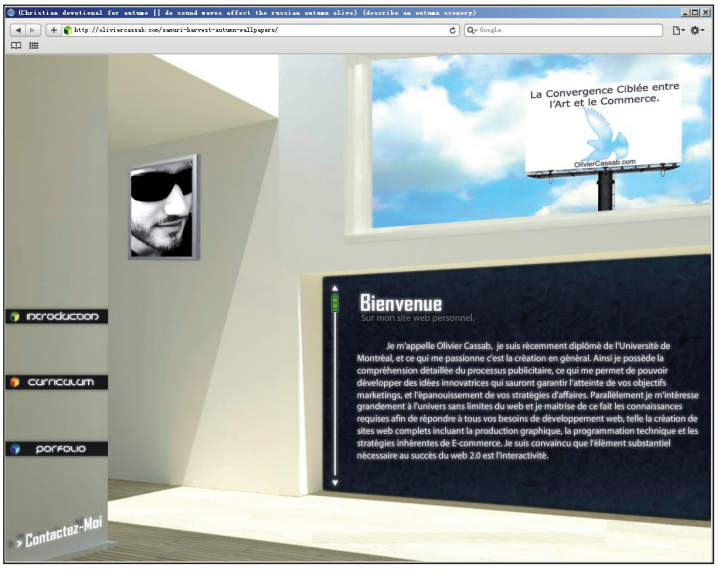


Figure 2: If the referer URL is not from Google Images or the search keyword does not appear to be correct, the redirector will display a clean page.

If the browser User-Agent string is not supported, another clean page will be displayed and the search keyword used in *Google Images* will be passed to the new URL (Figure 3).

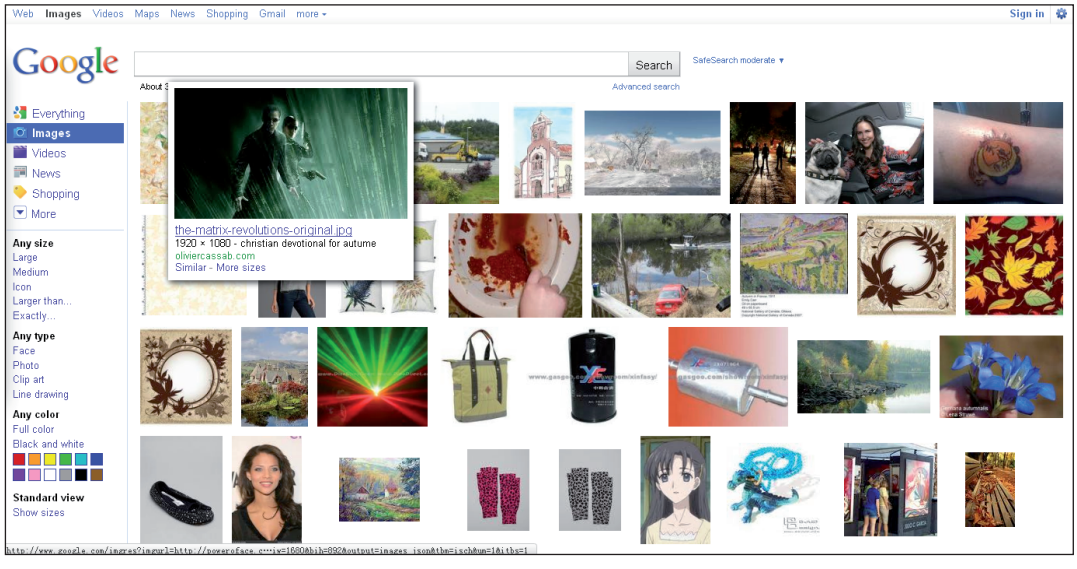


Figure 1: If any of these images are opened, hxxp://oliviercassab.com/samuri-harvest-autumn-wallpapers/ will be previewed.

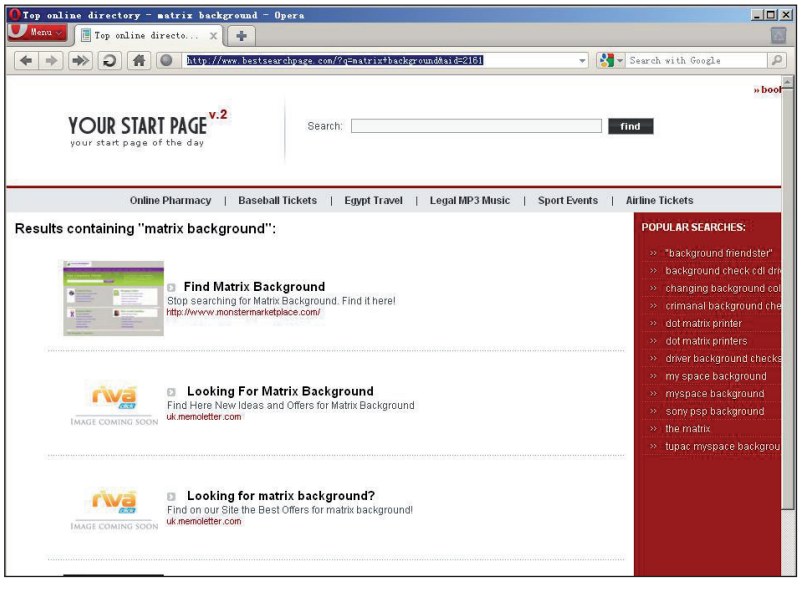


Figure 3: The search keyword used in Google Images is passed to the new URL.

Otherwise, depending on the browser User-Agent string, the user will be redirected to different sites:

All *IE*-based browsers and *Google Chrome* will be redirected to free hosting domains in the following format:

- update%decimal.number%.%random.string%.%free.domain.provider%
- scan%decimal.number %.%random.string%.%free.domain.provider%

For example:

- scan21.goff.cz.cc
- update6.ocerloh.mo.cx

All domains point to the same IP address: 212.124.119.186.

Different browsers will display different pages (see Figure 4), but will lure the user into downloading a similar fake AV downloader – the file name of which is in the following format:

InstallInternetProtection_%random.3.decimal.digits%.exe

Each downloader may have a different MD5. Each comes with an encrypted download link in its appended data. The downloaded program 'Internet Protection' (see Figure 5) then displays a series of fake alerts to lure the user into purchasing an activation code to remove the supposed threats from their machine.

Mozilla Firefox and *Apple Safari* are redirected to different domains with different pages and, instead of a downloader, the pages in *Firefox* and *Safari* lure the user into downloading a fake AV application (AntiSpy2011Setup.exe) directly. All domains point to the IP address 91.213.157.110. There are approximately 200 malicious domains on this IP address and the number is still increasing – the attacker registers several domains every day.

The malicious site oliviercassab.com has approximately 8,800 related images on *Google Images* and is far from being the only site to abuse the search service.

Another, iqsplus.com, appears to be an older version in which the redirector does not check the User-Agent string

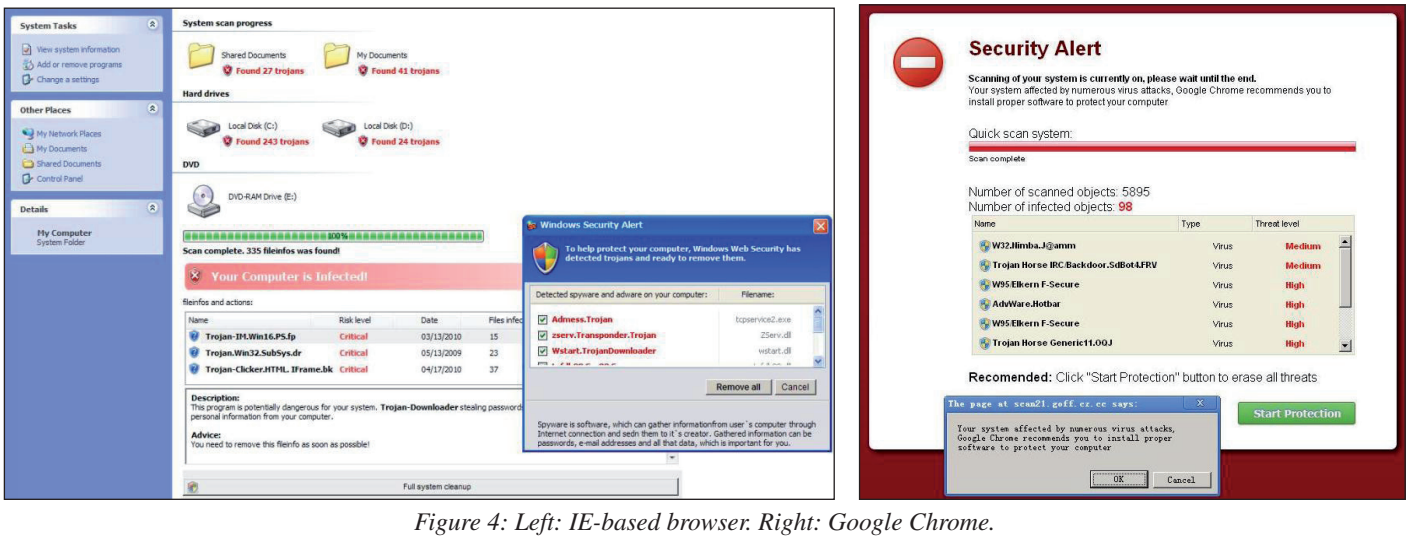


Figure 4: Left: IE-based browser. Right: Google Chrome.

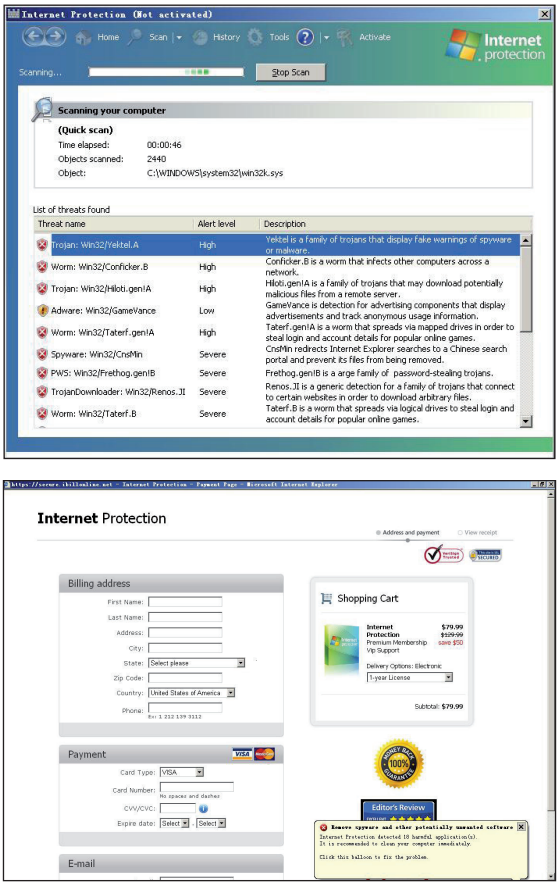


Figure 5: 'Internet Protection' displays fake alerts.

and search keyword. This uses approximately 119,000 related images on *Google Images*.

The owner of iqsplus.com has many other redirectors including:

- beyonceknowles.eu (approx. 67,300 related images)
- cinta-asia.com (approx. 33,000 related images)
- zeness.com (approx. 14,600 related images)
- peekspy.com (approx. 5,850 related images)
- quickbuildeco.com (approx. 3,350 related images)
- lynxemi.com (approx. 300 related images)

CONCLUSION

These attacks target all users of the *Google Images* service. Attackers update the redirector and final malformed URL frequently, which makes the attacks very hard to track. Computer users should always treat such search engine results with caution.

CONFERENCE REPORT

EICAR 2011: A 20TH ANNIVERSARY IN AUSTRIA

Eddy Willems
EICAR and G Data, Belgium

2011 marks the 20th anniversary of EICAR. In that time, many things have been achieved – sometimes with difficulty, but always with openness and sincerity. This year the event looked back over the past 20 years in an attempt to determine the essential facts and developments. In a growing world of poor communication, misunderstanding, hype and commercially driven interest, it is time to realign stake holders – in particular scientific research and commercial product vendors. It is time to assess the real threats and the myths in the non-transparent world of computer malware and anti-malware. As well as looking back, EICAR 2011 also looked at the present and into the future:

- What new form will malware threats take?
- What are the issues with current and future anti-malware techniques and products?
- Is the current industry-driven approach to AV still the right one?
- Should governments be more proactive?
- Are new tools and/or regulations required?
- Does law enforcement need to use malware in an offensive way?

OPENING

The conference took place at Krems University, a lovely location not far from Vienna. The conference started on Sunday 8 May with a meet-the-experts reception on the university premises.

The event was opened the next morning with a keynote speech from EICAR chairman Rainer Fahs who addressed this year's conference theme of 'cyber war'. He pointed out that there is no agreed definition of the term. Though comprising some properties of cybercrime and cyber terrorism, 'cyber war' unfortunately is more than just a buzz word and, if not carefully analysed and addressed in the near future, could lead to unknown escalation of attacks on the Internet and its underlying infrastructure. In his book *Cyber War* (May 2010), Richard A. Clarke describes cyber warfare as 'actions by a nation-state to penetrate another nation's computer or networks for the purposes of causing damage or disruption'. This is possibly the most realistic definition we can find. Rainer brought up a number of questions including: what constitutes an attack? What are the boundaries of the battlefield? What are cyber

weapons? There are not even any regulations or treaties to regulate such a war. However, nations are re-organizing their military structures and adding cyber defence to their planning in order to be fully prepared for any such attack.

Morton Swimmer of *Trend Micro* continued the conference with an in-depth look at future threats. I particularly liked his spectrum of cyber threats, where cyber bullying was the lowest level and cyber war the highest.

After a coffee break the day continued with possibly the most controversial paper of the conference: 'Magic Lantern .. reloaded, (Anti)Viral Psychosis McAfee Case' by Eric Filiol and Alan Zaccardelle of *ESIEA Research*. They chose the *McAfee* product because it is widely used but they noted that similar issues could have been found in other AV products. They detailed a couple of problems they had found with the product. One of these was related to not detecting specific cases of the Conficker autorun files, another was related to *McAfee*'s quarantine encryption and the possibility of exploiting this to launch an attack against the centralized management system. Their conclusion was that we all have to be very careful with AV marketing in general and that it might be a good idea to have an independent body to verify such things. During the follow-up discussion several people suggested that the academic world and the AV industry should communicate better – as it seems that there are a lot of misconceptions on both sides and that each party would be able to learn from the other.

Ralf Benzmueller of *G Data* presented an excellent overview of the common AV techniques currently being used in the industry, and Boris Sharov (*Dr. Web*) described what cyber warfare means from an AV vendor's perspective. Judging by the content of these presentations, I don't think that any of us in the industry will be out of a job any time soon.

This year's panel session was moderated by Rainer Fahs and included Eric Filiol, Boris Sharov, Ralf Benzmueller and Morton Swimmer. Most of the questions related to the cyber war theme, and after hearing the answers I can only say that there is a real need for more communication between law enforcement, academics, and above all organizations. Of course, like many things, this is easier said than done.

This year's gala dinner was held at the Kloster UND restaurant, housed in the former church of an old Austrian monastery. While the venue was beautiful, we took in some interesting scenery en route – walking past a high-security prison to reach our destination!

DAY TWO

If the first day was a more general presentation day, the second day was for those who are more technically minded. It started with this year's best paper award which went to Igor Sorokin of *Dr. Web* for 'Comparing Files Using Structural Entropy'. As malware writers tend to use increasingly complex techniques to protect their code, AV vendors face the problem of increasingly difficult file scanning as well as the massive growth of AV databases. Sorokin's solution is based on the assumption that different samples of the same malicious program have a similar order of code and data areas. Files may be characterized by the complexity of their data order. Sorokin's approach consists of using wavelet analysis for the segmentation of files into segments of different entropy levels and using edit distance between sequence segments to determine the similarity of the files.

I loved Anthony Desnos and Geoffroy Gueguen's paper: 'Android malware: is it a dream?', which gave a detailed analysis of DroidDream and a look into the possible future problems of the *Android* OS. They showed the importance of new *Android* analysis tools such as *Androguard* (http://code.google.com/p/androguard/).

David Harley from *ESET* explained how difficult it can be to tell the difference between fake AV and real AV. In his paper it became clear that the marketing efforts of the bad guys and the good guys (the AV industry) can sometimes be very similar and difficult to distinguish. More education for end-users seems to be one of the key areas we must look at to help solve this problem.

During one of the breaks between sessions I gave a demonstration in which I showed a representation of the EICAR conference, created by Dr Sarah Gordon, inside *Second Life*. Walking around in this virtual environment with Sarah's avatar and explaining the *Second Life* version of EICAR to those in the real world was a strange experience. Holding a conference in a virtual environment could, and possibly *will* be the future (in fact it has already been done in some ways), but I for one would miss many human elements and can't imagine ever preferring it to a real-world venue.

In the last session of the conference a team from *BitDefender* demonstrated CDS, or Clean by Detection Shifting. The technique uses prior information stored in the cloud to pre-emptively block undetected malware before it has a chance to execute its payload. However, the team's research is still a 'work in progress' as there are still some disadvantages to the technique and so far it has only been tested in a lab environment.

NEXT YEAR

This is just an overview of some of the many interesting papers presented at the conference. This year's event was another great one and I'm already looking forward to next year's. The venue for EICAR 2012 will be announced on the EICAR website in September.

END NOTES & NEWS

The MAAWG 22nd General Meeting takes place 6–9 June 2011 in San Francisco, CA, USA. See http://www.maawg.org/.

Security Summit Rome takes place 8–9 June 2011 in Rome, Italy (in Italian). For details see https://www.securitysummit.it/.

The 2011 National Information Security Conference will be held 8–10 June 2011 in St Andrews, Scotland. Registration for the event is by qualification only – applications can be made at http://www.nisc.org.uk/.

The 23rd Annual FIRST Conference takes place 12–17 June 2011 in Vienna, Austria. The conference promotes worldwide coordination and cooperation among Computer Security Incident Response Teams. For more details see http://conference.first.org/.

SOURCE Seattle 2011 will be held 16–17 June 2011 in Seattle, WA, USA. For more details see http://www.sourceconference.com/.

The Cyber Security for Oil and Gas Conference takes place 22–23 June 2011 in London, UK. For full details see http://www.cyber-security-oilandgas.com/.

The Eighth Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA 2011) takes place 7–8 July 2011 in Amsterdam, The Netherlands. For details see http://www.dimva.org/dimva2011/.

Black Hat USA takes place 30 July to 4 August 2011 in Las Vegas, NV, USA. DEFCON 19 follows the Black Hat event, taking place 4–7 August, also in Las Vegas. For more information see http://www.blackhat.com/ and http://www.defcon.org/.

The 20th USENIX Security Symposium will be held 10–12 August 2011 in San Francisco, CA, USA. See http://usenix.org/.

The 8th Annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference (CEAS 2011) will be held in Perth, Australia 1–2 September, 2011. See http://ceas2011.debii.edu.au/.

VB2011 takes place 5–7 October 2011 in Barcelona, Spain. For full programme details including abstracts for each paper, and online registration see http://www.virusbtn.com/conference/vb2011/.

RSA Europe 2011 will be held 11–13 October 2011 in London, UK. For details see http://www.rsaconference.com/2011/europe/index.htm.

The MAAWG 23rd General Meeting takes place 24–27 October 2011 in Paris, France. See http://www.maawg.org/.

The CSI 2011 Annual Conference will be held 6–11 November 2011 in Washington D.C., USA. See http://www.CSIannual.com/.

The sixth annual APWG eCrime Researchers Summit will be held 7–9 November 2011 in San Diego, CA, USA. The summit will bring together academic researchers, security practitioners and law enforcement to discuss all aspects of electronic crime and ways to combat it. For more details see http://www.antiphishing.org/ecrimeresearch/2011/cfp.html.

The 14th AVAR Conference (AVAR2011) and international festival of IT Security will be held 9–11 November 2011 in Hong Kong. A call for papers has been issued, with a deadline for submisisons of 30 June. For details see http://aavar.org/avar2011/.

Ruxcon takes place 19–20 November 2011 in Melbourne, Australia. The conference is a mixture of live presentations, activities and demonstrations presented by security experts from the Aus-Pacific region and invited guests from around the world. For more information see http://www.ruxcon.org.au/.

Hacker Halted 2011 will take place 21–27 November in Miami, FL, USA. See http://www.hackerhalted.com/2011/.

SUBSCRIPTION RATES

Subscription price for 1 year (12 issues):

- Single user: $175
- Corporate (turnover < $10 million): $500
- Corporate (turnover < $100 million): $1,000
- Corporate (turnover > $100 million): $2,000
- *Bona fide* charities and educational institutions: $175
- Public libraries and government organizations: $500

Corporate rates include a licence for intranet publication.

See http://www.virusbtn.com/virusbulletin/subscriptions/ for subscription terms and conditions.

Editorial enquiries, subscription enquiries, orders and payments:
Virus Bulletin Ltd, The Pentagon, Abingdon Science Park, Abingdon, Oxfordshire OX14 3YP, England
Tel: +44 (0)1235 555139 Fax: +44 (0)1865 543153
Email: editorial@virusbtn.com Web: http://www.virusbtn.com/