Malware Distribution via Widgetization of the Web

By Ameet Ranadive (ameetr@dasient.com), Tufan Demir (tufan@dasient.com), Shariq Rizvi (shariq@dasient.com), and Neil Daswani (neil@dasient.com)

Abstract

The Web 2.0 transformation has in part involved many sites using third-party widgets. We show how web-based malware and scareware is propagated via such widgets, provide data and case studies on how a web-based malware attack takes place against the web sites via widgets, and present the "widgetized web graph" showing the structure of high traffic web sites from the standpoint of widgets.

Fundamental Shift in Malware Distribution

Modern enterprise websites are highly interactive and often combine “best of breed” software and content to create a rich user experience. An enterprise website may now be using third-party “widgets” (such as analytics, polls, video players, or the ability to share with friends); sourcing in ads from a third-party network; or may be running third-party software to power various applications on their sites.

Attackers take advantage of the interactivity, interconnectedness, and interoperability of the web to exploit website vulnerabilities to increase the footprint and effectiveness of their attacks. One of the favorite attack methods of hackers over the last few years has been to distribute malware from legitimate websites. In these attacks, the hackers will compromise a legitimate website and silently place a piece of malicious code that is presented to all visitors of the website. Unsuspecting users will have a virus downloaded to their personal computer by visiting an infected web page. Sometimes, the virus is downloaded without any user interaction (“drive by download”); in other cases, the user is prompted to click a button (“social engineering malware”) or to download what appears to be legitimate file (“dangerous download”), and then receives a virus. Enterprises who depend on their website for customers, transactions, and brand are at great risk of loss from the emergence of web-based malware attacks.

To provide a sense of the scale and scope of malware attacks on websites, consider the following statistics:

- Every 1.3 seconds a new web page is getting infected
- As of 2009, every month almost 2,000,000 web pages across more than 210,000 websites are infected with Malware. This is almost double the number of web pages reported for Q4 2008.[1]
- 77% of Web sites with malicious code are legitimate sites that have been compromised [2]
- The number of malware-infected web pages has grown 12x in 4 years.
- 57% of data-stealing attacks are conducted over the Web.

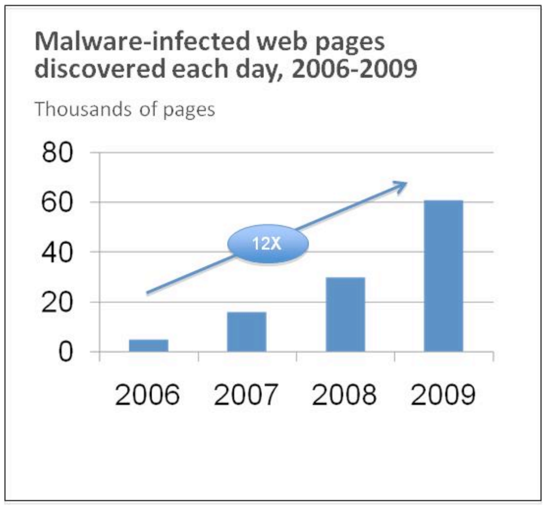


Figure1: Malware-infected Web pages discovered each day, 2006-2009

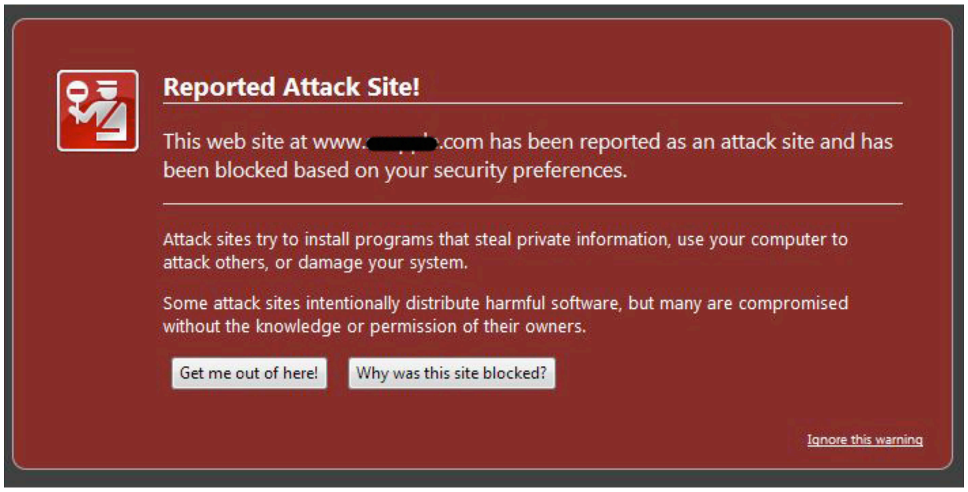
1 Microsoft Security Intelligence Report, April 2009

2 Websense State of Internet Security, Q1-Q2 2009

The Impact of Web Malware Infections

The impact on enterprise websites that suffer a malware attack is enormous. If undetected, the website will now infect any visitors with a virus. This can severely damage the website's reputation with its existing and potential customers, as well as create downstream liability issues.

If an enterprise were to suffer a malware attack, it is an extremely high visibility security incident. Any single user that is running anti-virus software on their PC may encounter a warning upon loading an infected URL from the bank's website in their browser. Users that are warned away by their anti-virus may silently churn; or, they may report the event on their blog or social network. News of malware infections travels extremely fast on Twitter and Facebook. The malware infection could get picked up and reported by the media, in which case the PR impact of the malware breach is now magnified substantially. There is also the risk that the website would be partially or fully blacklisted by a major search engine or browser.



Reported Attack Site!

This web site at www.[redacted].com has been reported as an attack site and has been blocked based on your security preferences.

Attack sites try to install programs that steal private information, use your computer to attack others, or damage your system.

Some attack sites intentionally distribute harmful software, but many are compromised without the knowledge or permission of their owners.

Figure 7: Firefox Malware Warning

When your site gets attacked, this is what your Firefox visitors see

Google, Yahoo!, Firefox, Internet Explorer, Chrome, and Safari all blacklist legitimate sites—including enterprise sites—that have been infected with malware. The blacklisting has an immediate impact on the enterprise website's traffic and revenues, as well as heightens the damage to an enterprise's brand and reputation.

In addition to the loss of brand, reputation, and customers, websites that suffer malware attacks may also experience downstream liability issues such as data theft. As previously noted, 57% of data-stealing attacks now occur on the web. In another study published by Google[3], the researchers discovered that, after a virus was successfully downloaded to the user's personal computer from visiting an infected web page, one of the primary activities of the virus was "data exfiltration" (or data stealing). The study reported, "Many responder sessions contained signs of data exfiltration, including browser history files and stored passwords, usually captured by keyboard loggers or browser hooks… SMTP is one method of achieving this goal. We observed several emails sending back stored passwords from the compromised machine."

Suppose that an enterprise website suffers a malware attack and is now serving a drive-by-download from its homepage. Any visitor to the homepage will be infected with a virus on their PC. This virus logs keystrokes and allows the hackers to steal the user's passwords. Suppose after visiting the homepage, these visitors subsequently enter their login information to the enterprise website. The attackers have successfully harvested the login credentials for the enterprise website. If the malware attack is undetected for a period of time, potentially thousands or even tens of thousands of the enterprise's customers can have their credentials compromised in this way. Forrester estimated[4] in a study that the cost of data breaches can be as high as $305 per record. The cost of data theft due to a web-based malware attack for a bank can easily be in the millions or even tens of millions of dollars.

[3] "Ghost Turns Zombie: Exploring the Lifecycle of Web-based Malware" (http://www.usenix.org/event/leet08/tech/full_papers/polychronakis/polychronakis.pdf)
[4] "Calculating the Cost of a Security Breach" (http://www.forrester.com/rb/Research/calculating_cost_of_security_breach/q/id/42082/t/2)

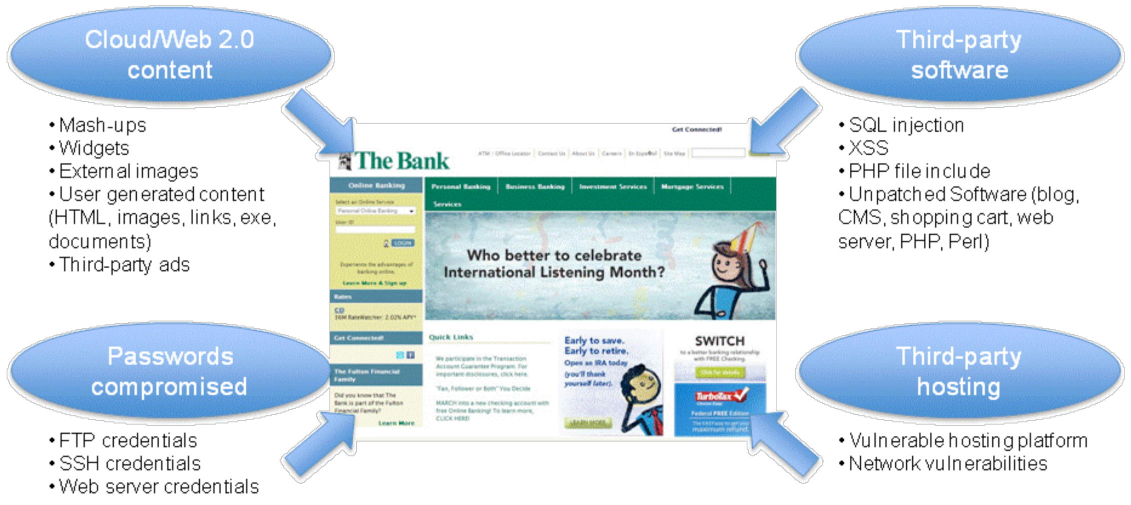
Structural Vulnerabilities Are Exploited During Malware Attacks

Enterprise websites typically have relatively tight control over their own software development lifecycle (SDLC) and security practices. However, they very often rely on third-party partners to provide content, advertisements, or software applications that power the enterprise's websites. Since the partners may not have as tight control over their software development or security practices as the enterprise, they are more vulnerable to getting compromised and infected with malware. If an enterprise's partner gets compromised, the enterprise itself can now be serving malware to its users because it relies on the partner for content, advertisements or software. The enterprise's customers will perceive that the malware is being served by the enterprise, even though it's the enterprise's partners that had been compromised.

These vulnerabilities are often "structural" in nature—they cannot be "remedied" very easily, and exist because of the interconnected, interdependent nature of the web.

Examples of structural vulnerabilities include:

- Third-party widgets, mash-ups, scripts, or iframed content
- External advertisements
- Third-party software that may have web application vulnerabilities



Cloud/Web 2.0 content
• Mash-ups
• Widgets
• External images
• User generated content (HTML, images, links, exe, documents)
• Third-party ads
Third-party software
• SQL injection
• XSS
• PHP file include
• Unpatched Software (blog, CMS, shopping cart, web server, PHP, Perl)
Passwords compromised
• FTP credentials
• SSH credentials
• Web server credentials
Third-party hosting
• Vulnerable hosting platform
• Network vulnerabilities
The Bank
Who better to celebrate International Listening Month?

Figure 3: Bank Partners, Compromised Passwords are often the source of malware infections

Structural Vulnerability #1: Third-party widgets

Many websites use third-party widgets for counting traffic, tracking users, sharing content, video, polls, and other user functionality. The use of third-party widgets has enabled rich user functionality and analytics. However, in a security context, websites that use third-party widgets are allowing arbitrary executable code, supplied by a third party, complete access to the web page DOM and the user's session information. This could, of course, be used to infect the website's users with malware.

In a research paper published by Google titled "The Ghost in the Browser," researchers claimed that Third-Party Widgets were one of the primary vectors of attack for a website to get infected with malware.

> We identified a free statistics counter that operated fine for almost four years, "when the nature of the counter changed and instead of cataloging the number of visitors, it started to exploit every user visiting pages linked to the counter… In this particular case, the user visited a completely unrelated web site that was hosting a third-party web counter. The web counter was benign for over four years and then drastically changed behavior to exploit any user visiting the site. This clearly demonstrates that any delegation of web content should only happen when the third party can be trusted."

Unfortunately, it is virtually impossible for an enterprise to know that it can "trust" a third-party widget provider from a security standpoint.

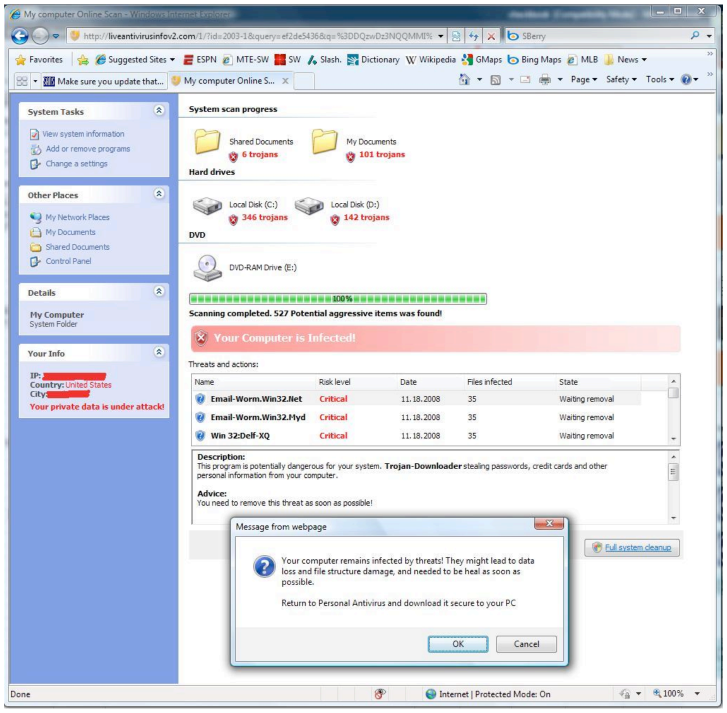
- The widget provider could be an attacker masquerading as a legitimate provider.
- A legitimate widget provider could get compromised, and now all of the websites that are using the widget could be serving malware to its users.
- The attackers can use DNS cache poisoning to redirect a legitimate domain

Structural Vulnerability #2: Third-party advertising

Malicious advertisements (also known as "malvertising") are another way for a website to experience a malware attack. Many enterprises use third-party advertising partners on some of the content pages of their website. The advertising partner who provides the ads to the website to display on its website may in turn be sub-syndicating their ad inventory to other partners. In many cases, an advertisement travels through several tiers of ad network providers before reaching its final destination—the enterprise's website.

Some of the upstream ad network providers may be second- or third-tier networks that do not screen advertisers at all and therefore may accept ads from attackers who pose as advertisers. Attackers can insert an advertising creative into an upstream ad network that looks like a legitimate banner ad, but contains malicious code that will infect users when displayed on the bank's website. Customers who visit the bank's website may receive the malicious ad and experience an intermittent or a fast flicker of the flash malware ads which seem harmless. However, the ad would install a trojan on the user's PC.

Even top-tier ad networks sub-syndicate their ad inventory, which may expose them to malicious ads inserted into dubious upstream ad networks. Thus, the mere presence of advertising on the bank's website, even if it is supplied by a top-tier network, means that it is vulnerable to a malvertising attack. In a recent study by a security firm[5], Yahoo, Google, and Fox Audience Network were all implicated as serving malicious ads through their network. The study found that the malicious ads contained JavaScript code dubbed "JS:Prontexi," which is a drive-by-download Trojan in script form that targets the Windows operating system. The malware exploits vulnerabilities in Adobe Reader and Acrobat, Java, QuickTime, and Flash and launches fake antivirus warnings. The study also reported that it registered more than 2.6 million instances of this particular malware due to malicious ads on consumer's computers in a period of less than three months.



My Computer
System Folder

IP: [redacted]
Country: United States
City: [redacted]
Your private data is under attack!

System scan progress

Shared Documents 6 trojans
My Documents 101 trojans

Hard drives

Local Disk (C:) 346 trojans
Local Disk (D:) 142 trojans

DVD

DVD-RAM Drive (E:)

100%

Scanning completed. 527 Potential aggressive items was found!

Your Computer is Infected!

Threats and actions:

Name	Risk level	Date	Files infected	State
Email-Worm.Win32.Net	Critical	11.18.2008	35	Waiting removal
Email-Worm.Win32.Myd	Critical	11.18.2008	35	Waiting removal
Win 32:Delf-XQ	Critical	11.18.2008	35	Waiting removal

Description:
This program is potentially dangerous for your system. **Trojan-Downloader** stealing passwords, credit cards and other personal information from your computer.

Advice:
You need to remove this threat as soon as possible!

Message from webpage

Your computer remains infected by threats! They might lead to data loss and file structure damage, and needed to be heal as soon as possible.

Return to Personal Antivirus and download it secure to your PC

Figure 6: Screen shot of fake antivirus warning common with malvertising

Structural Vulnerability #3: Third-party applications

One way in which hackers can compromise a website is via attacks against vulnerable web applications running on the site. According to a recent SANS study[6] , "Attacks against web applications constitute more than 60% of the total attack attempts observed on the Internet. These vulnerabilities are being exploited widely to convert trusted web sites into malicious websites serving content that contains client-side exploits. Web application vulnerabilities such as SQL injection and Cross-Site Scripting flaws in open-source as well as custom-built applications account for more than 80% of the vulnerabilities being discovered."

In these attacks, the attackers identify websites that are running vulnerable versions of website software, including blogging/ content management software, web server software, and development tools. Poor input sanitization or output escaping result in SQL injection or cross-site-scripting (XSS) vulnerabilities in these web applications. The attackers exploit the vulnerabilities in these web applications in order to plant malicious code onto the website. For example, in a SQL injection attack, the hackers send database commands into form fields (like login or comment forms) instead of legitimate user input. The database commands are constructed in a way that they trick the web application into executing the commands and planting malicious code into the

[5] CNET, "Malware delivered by Yahoo, Fox, Google ads" (http://news.cnet.com/8301-27080_3-20000898-245.html)
[6] SANS Top Cybersecurity Risks Sept 2009 (http://www.sans.org/top-cyber-security-risks/)

database. If the web application calls on the database to generate dynamic web pages (for example, calling the database to generate parts of the header or footer), the malicious code planted in the database could be presented to users, resulting in infections.

One key challenge for enterprises is that they often rely on third-party software to power various parts of their websites. The enterprise is depending on the software provider's SDLC to ensure that the applications are secure. However, many providers may not regularly test their software or employ secure coding practices to ensure that their software does not contain web application vulnerabilities. Even if the software partner did regularly test, fix, and update their software to patch known vulnerabilities, there is always the risk of unknown, zero-day vulnerabilities that may be exploited. In addition, it is difficult for enterprises to keep their third-party web applications up-to-date with the latest versions provided by their partners. Existing applications may be reliant on the web server or scripting applications (such as PHP, ASP, and Perl)—an upgrade of the web server or scripting applications may be a very extensive project that would involve upgrading multiple other applications at the same time, and may require significant planning, coordination, and time.

Websites Across Verticals Have Significant Structural Vulnerabilities

Background

In Q2 2010, Dasient performed a study to assess which verticals were most at risk of having websites infected with web-based malware. Dasient ran automated, passive malware risk assessments against the websites of Fortune 500 websites, and other highly trafficked websites in a number of verticals, including:

- Publisher/media
- Financial
- Ecommerce
- Traditional retail
- High-tech manufacturers
- Travel, entertainment, and leisure
- Consumer packaged goods (CPG)
- Business services
- Manufacturing
- Healthcare

The following sections report the results of this study.

Third-party widgets

Across all verticals, 75% of websites used some third-party JavaScript widgets on their sites. Although all verticals used JavaScript widgets, some verticals had higher propensity to use third-party JavaScript than others. Travel, entertainment, and leisure sites were the top vertical using third-party JavaScript, with a whopping 99% of sites using widgets. Publishers, high-tech manufacturing companies, financial websites, and CPG companies also had a higher than average percentage of websites using JavaScript widgets. Many of these sites used JavaScript widgets for analytics, sharing content, mashing-up content from other sites, or offering videos using embedded video players.

Vertical	**Percentage of sites using third-party JavaScript**
Travel, entertainment, and leisure	99%
Publisher	95%
High-tech	94%
Financial	89%
CPG	82%

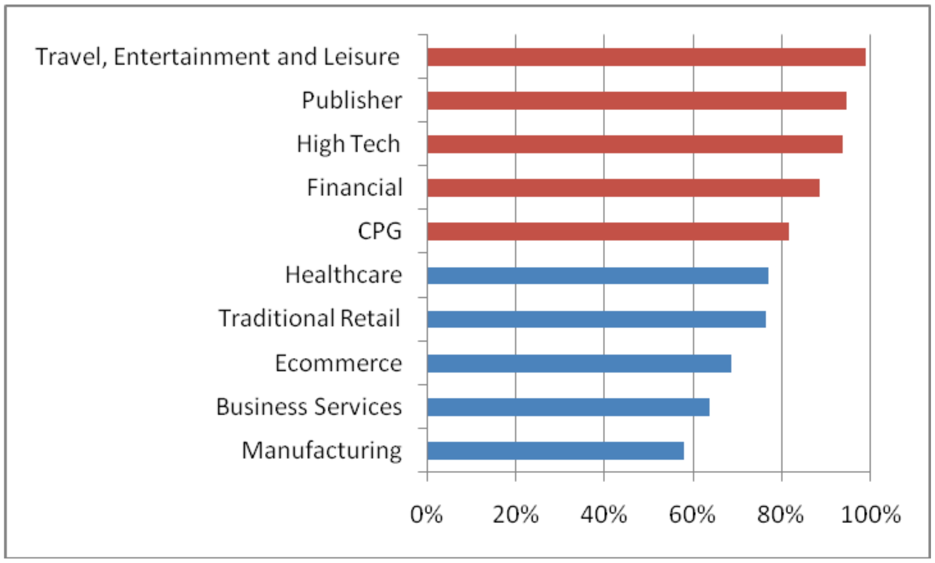


Figure 6: Percentage of sites in each vertical that used third-party JavaScript widgets.

Third-party advertising

Across all verticals, 42% of websites used some third-party advertising on their sites.

As expected, publishers were the top vertical using third-party advertising, with 82% of sites. Given that many publishers are dependent on advertising to support their business model online, it is no surprise that so many of them would host third-party ads. What was a bit more surprising is that a high percentage of travel, entertainment, and leisure sites; traditional retail sites; and high-tech sites also offered third-party ads on their site. The travel, entertainment, and leisure sites offered ads for complementary products—if a user is searching for hotels, why not also show them ads for rental cars, flights, and other related products? Traditional retail and high-tech sites also offered ads for complementary and related products, but as a much smaller percentage. Also unexpectedly, financial websites ranked relatively high for showing advertising on their websites.

Vertical	**Percentage of sites using third-party advertising**
Publisher	89%
Travel, entertainment, and leisure	83%
Traditional retail	56%
High-tech	53%
Financial	41%

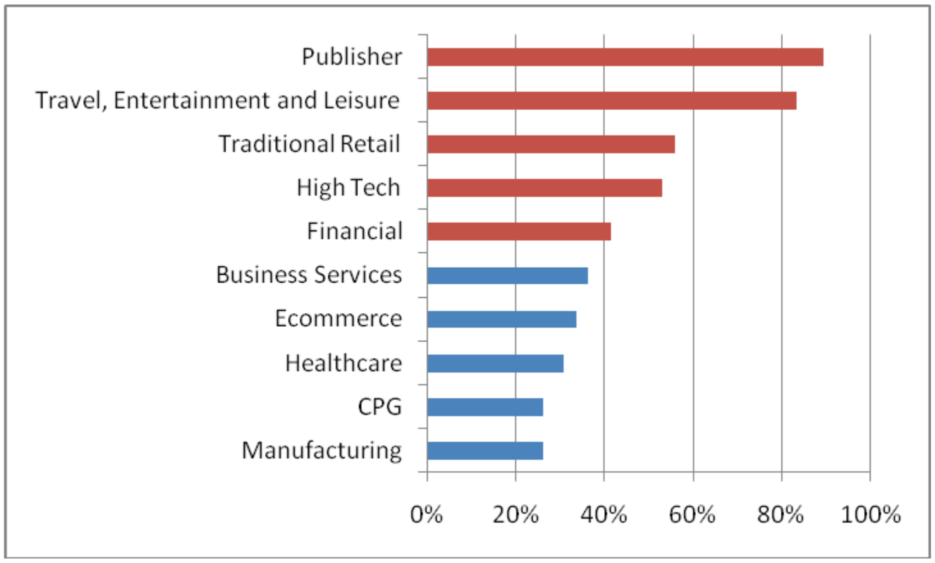


Figure 6: Percentage of sites in each vertical that used third-party advertising.

Third-party applications

Across all verticals, a surprising 91% of businesses had some outdated software powering their websites. Three verticals were tied for having the highest percentage of websites with outdated software applications: CPG, publisher, and high-tech websites.

Interestingly, some of the verticals that had a lower percentage of sites with external JavaScript or ads, actually ranked higher for having outdated software. For example, manufacturing companies' websites tended to not have as much external JavaScript or advertising as other verticals. However, manufacturing sites had the 4th highest percentage of outdated software.

Financial and healthcare companies' websites were among the lowest percentage of having outdated web applications (although still quite high at 87% and 85%, respectively).

Vertical	**Percentage of sites using third-party advertising**
CPG	97%
Publisher	97%
High-tech	97%
Manufacturing	95%
Traditional retail	94%

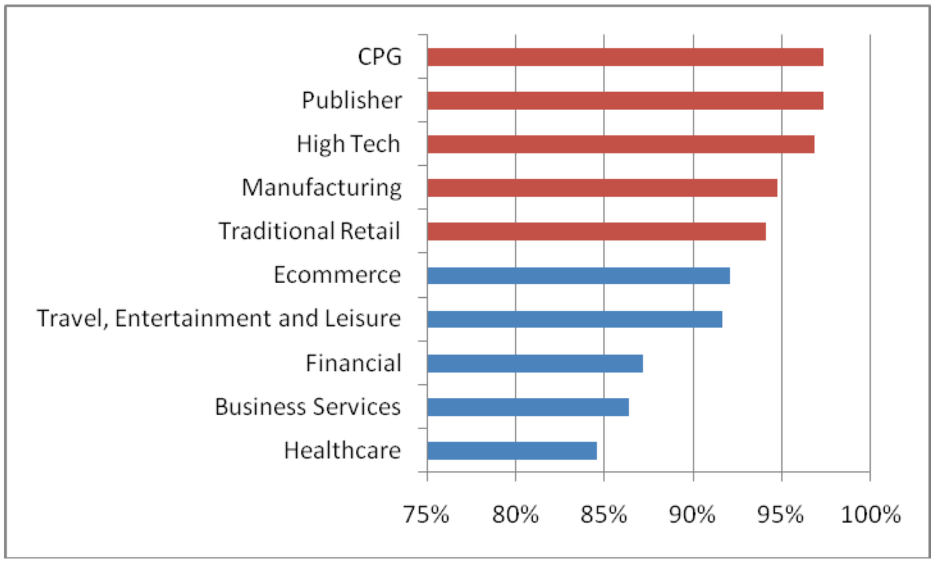


Figure 6: Percentage of sites in each vertical that had outdated web applications.

Case Studies

In the following, we described three case studies – one corresponding to each major type of structural vulnerabilities from real, live cases "in the wild."

Compromised Widgets

Case Date: Friday, June 4, 2010

We identified a free statistics counter that operated fine for almost four years, when the nature of the counter changed and instead of cataloging the number of visitors, it started to exploit every user visiting pages linked to the counter. In this particular case, the user visited a completely unrelated web site that was hosting a third-party web counter. The web counter was benign for over four years and then drastically changed behavior to exploit any user visiting the site. This clearly demonstrates that any delegation of web content should only happen when the third party can be trusted."

Just this past weekend, the Dasient security research team identified a third-party JavaScript widget that was responsible for infecting web users at a large Quantcast 100 website. The third-party widget in question was from a reputable market research and analytics firm, and the widget was used for traffic analysis and audience demographics. (Our team has been in contact with the Quantcast 100 website, and is also reaching out to the widget provider in order to help resolve this problem.)

This third-party JavaScript code was included among a number of other tracking tags present on several thousand URLs of the Quantcast 100 website. The JavaScript code (after being anonymized) is as follows:

```
// xxxxxx tagging
XXXX.require('//secure-us.xxxxxxxxxxxx.com/xxx.js', function () {
   var trac = nol_t({
      cid: 'xx-xxxxxxx',
      content: '0',
      server: 'secure-us'
   });
   trac.record().post();
});
```

In turn, http://secure-us.xxxxxxxxxxxx.com/xxx.js served the following complicated JavaScript code:

```
function NolTracker(b,a){this.pvar=b;this.mergeFeatures(a)}function
nol_t(b,a){return new
NolTracker(b,a)}NolTracker.prototype.version="6.0.9";NolTracker.prototype.scr
iptName=(function(){try{var b=document.getElementsByTagName("script");var
c=b[b.length-1].getAttribute("src").match(/[^\/]*$/)}catch(a){}return
c||"xxx.js"})...
```

At the end of the complex JavaScript was a malicious iframe sourcing in content from:

```
http://94. 75. 210. 6/measure/
```

What is notable about the attack above is that the JavaScript code is so complex, it would be difficult for even a technical person to parse the code quickly and identify the malicious iframe at the end. Furthermore, the attackers have used the pathname "measure" on the malicious domain in an effort to further obfuscate their attack. As a result, a technical person who was investigating the cause of the malware might not pay attention to the iframe; he or she could easily assume that this was part of the legitimate JavaScript code that was measuring user traffic on the website.

The attackers compromised this third-party analytics provider's JavaScript code and embedded the malicious iframe. A quick search on Google for the JavaScript code showed over 19,000 results of websites which contained this provider's analytics code. Thus, the attackers were able to stripe their web-based malware over thousands and thousands of legitimate websites (including multiple Quantcast 100 websites) by infecting the third-party analytics provider's JavaScript code with the malicious iframe.

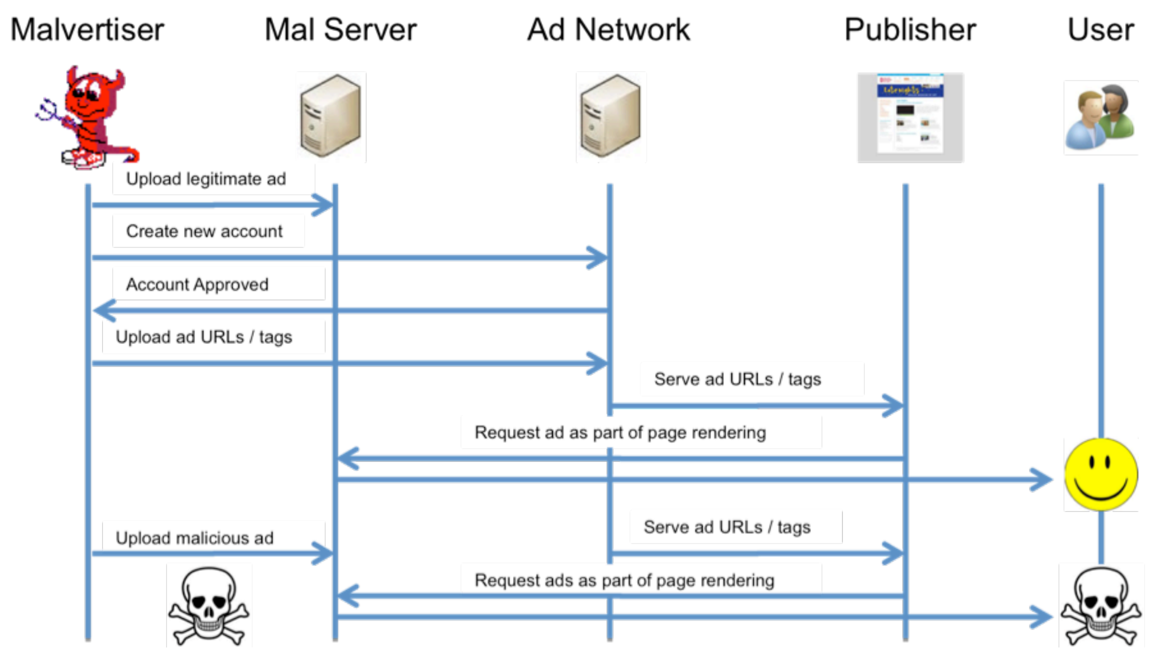
There is a significant implication for web businesses. The "widgetization" of the web will continue to create opportunities such as the one detailed in this post for attackers to infect legitimate websites with malware. Any third-party code included in a legitimate website can be compromised and exploited to serve malware. In fact, the attackers have an incentive to infect these JavaScript widgets as a way to achieve scale and get "back door access" to popular websites. The concern for web businesses is that, despite all of the security operations and software development practices that they may have in place, there are dependencies on third-parties for rendering functionality on web pages on their site. And a particular web business has no control over the security practices of the third-party partner, which can get compromised, as was evident from the attack described above.

It is unrealistic to believe that web businesses will be able to remove all third-party software and JavaScript code from their websites. The "widgetization" of the web will only accelerate, as the trend towards distributed software development, interactivity, and combining best-of-breed software and widgets continues. Despite a web business having significant preventative security measures in place, its website is vulnerable to serving malware due to the use of third-party JavaScript widgets. Therefore, it is critical that web businesses monitor their websites (and thus their third-party JavaScript widget providers) for malware on a regular basis. An attack where a reputable partner gets compromised and infected with malware could happen any time, and it is important that the web business can respond immediately if such an attack occurs. Otherwise, the web business is at risk of serving malware to its users, which would result in users getting infected with malware; significant losses of brand, reputation, and revenue; and potential liability issues.

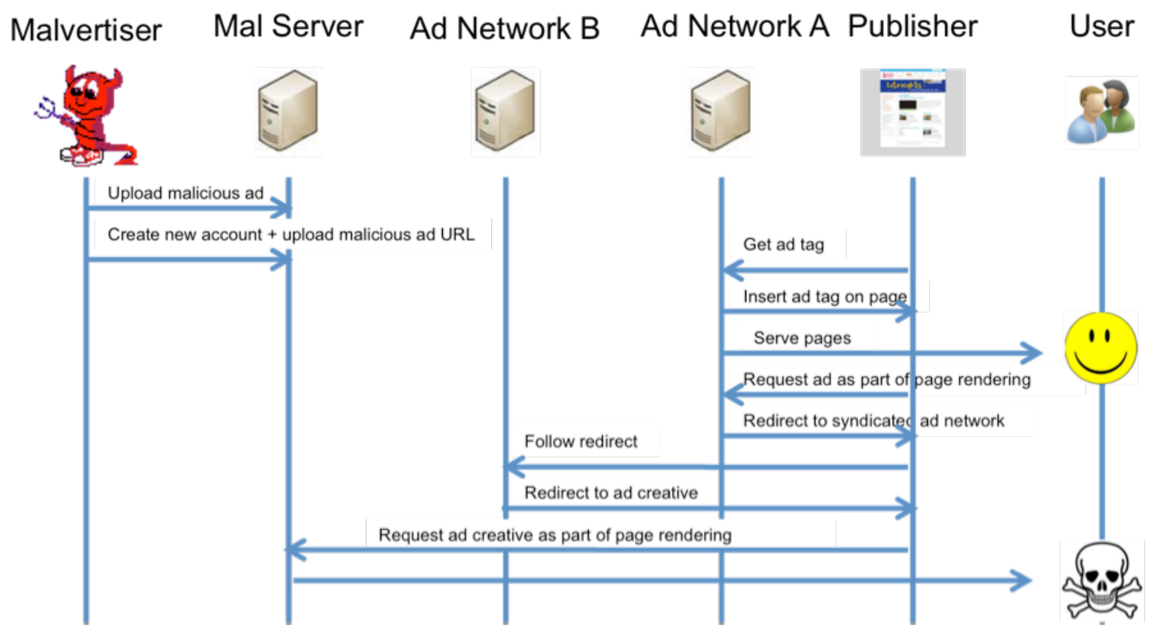
Compromised Ad Network

In the following, we show two cases in which the compromise of a legitimate ad network results in legitimate publisher sites infecting users. Most of the malvertising cases we have identified at Dasient fit into one of these two cases, and we describe these cases in the abstract instead of providing particular incidents.

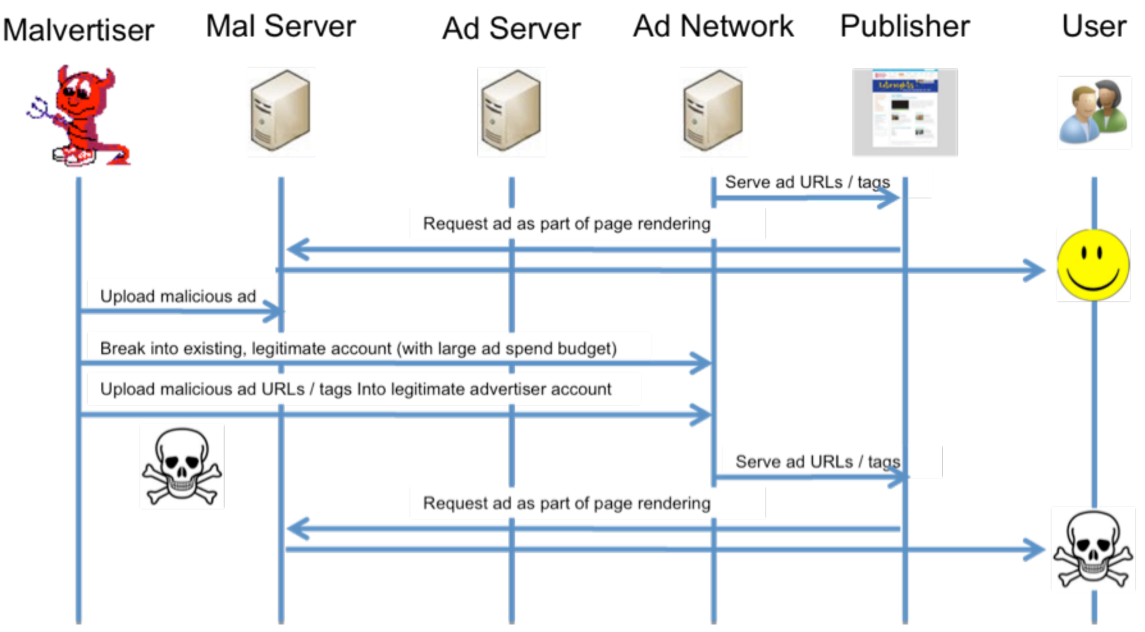
In the first case below, a “malvertiser” or malicious advertiser, creates a new account with a legitimate ad network, and submits legitimate ads to the ad network for review. Upon having the new account and legitimate ads reviewed and approved, the malvertiser then uploads malicious ads in place of the legitimate ads onto his/her own server. Many ad networks allow advertisers to host the creatives served, and as a result do not have direct control of the ads. If the ads are changed, the ad network does not have any direct way to know, unless they regularly monitor the ad creatives.



The way the malvertisers achieve scale using the attack method above is to exploit the syndication and sub-syndication of ads across various networks, as illustrated below. In the illustration below, the ad tag placed on the publisher site requests an ad from Ad Network A. However, Ad Network A may not have an ad in its inventory that is relevant to the publisher's page or to the user (e.g., based on behavioral targeting), and may redirect the request to fill the ad slot to Ad Network B. Ad Network B may then serve an ad to the user. Of course, if Ad Network B has a malvertisement uploaded to it, then the malvertisement can be served to the user even though Ad Network A may be clean. When Ad Network A syndicates its ad space to Ad Network B, malvertisments uploaded to Ad Network B can achieve more scale as compared to a scenario in which no syndication is employed at all.


Malvertiser
Mal Server
Ad Network B
Ad Network A
Publisher
User
Upload malicious ad
Create new account + upload malicious ad URL
Get ad tag
Insert ad tag on page
Serve pages
Request ad as part of page rendering
Redirect to syndicated ad network
Follow redirect
Redirect to ad creative
Request ad creative as part of page rendering

In the second case below, the malvertiser breaks into an existing account with the ad network. The attackers usually target an account at the ad network that already has a large ad spend budget such that they can achieve scale with their attack. Once the account is broken into, the malvertiser replaces existing, legitimate ad creatives with malicious ones or can also create new advertising campaigns with malicious creatives. If the account has been in use for quite some time, has become "trusted", and the ad network does not actively monitor all its advertiser's creatives, malvertisements can be injected via the case shown below.


Malvertiser
Mal Server
Ad Server
Ad Network
Publisher
User
Serve ad URLs / tags
Request ad as part of page rendering
Upload malicious ad
Break into existing, legitimate account (with large ad spend budget)
Upload malicious ad URLs / tags Into legitimate advertiser account
Serve ad URLs / tags
Request ad as part of page rendering

Compromised Application

Case Date: Wednesday, September 22, 2010

Last week we detected an infection on a Quantcast 100 publisher site which was due to a vulnerable, third-party ad server (We have also observed the same attack on 400 different sites since September 14). The exploit served from the infected page created two drive-by-downloads which infected users during a two-hour window. We were monitoring this site with Dasient's Anti-Malvertising (AMS) Service and identified the malware as soon as it hit the publisher site. Fortunately, the third-party provider was able to fix the infection quickly after being alerted.

The publisher site used a script tag to source in an external JavaScript snippet. Normally, this third party script adds image tags to the publisher page. They use an ad server application to manage such images. The application keeps these image tags in the "zones" table in its database.

In this incident, the attackers successfully injected malicious JavaScript tags into the zones table by exploiting a vulnerability found in the ad server versions before 2.8.7. When the users visited the publisher site, they received the image tags as well as the malicious JavaScript tags which silently downloaded a Java virtual machine exploit (CVE-2009-3867, CVE-2010-0886) to install two drive-by-downloads on the user's computer.

More information about the attack can be found at these links:

http://blog.openx.org/09/security-update-how-to-secure-your-openx-installation/
http://blog.sucuri.net/2010/09/openx-users-time-to-upgrade.html

As we reported earlier in this report, 75% of web sites uses external, third-party JavaScript widgets, 42% of web sites use third-party ad-related resources and 91% of web sites use third-party applications, and/or out-of-date applicaitons. In this attack, the third party application was compromised that led to the infection of the publisher site. In other words, a structural vulnerability on the publisher site was exploited. Traditional signature-based approaches, which have very low (or zero) detection of these drive-by-downloads, fail to prevent thousands of users from getting infected. Therefore we keep on emphasizing how important it is to use behavioral approaches, and to monitor ads as well as all kinds of third party content (including widgets) on websites.

Conclusion

Web Malware attacks pose a serious threat to websites and their users, and the use of widgets for a variety of third-party functionality can be leveraged to spread web malware. Although web sites can have relatively tight control over their own SDLC and security practices, they are often vulnerable to web-based malware attacks due to third-party widgets.