Global Information Assurance Certification Paper

Microsoft Internet Explorer (IE) Script Vulnerability

By: Frederick P. Williams

SANS GIAC Level 2 – GCIH Practical
SANS Network Security 2000
Monterey, CA

Exploit Details

Name:	Microsoft IE Script Vulnerability
Variants:	None
Operating System:	Windows 95, 98, NT 4.0, 2000
Protocols/Services:	Windows File Sharing: SMB & CIFS—Windows 2000 HTTP and HTML
Brief Description:	An error within Internet Explorer (IE) improperly downloads and opens MS Access databases via MS Access-type ActiveX Controls referenced in HTML documents, regardless of the configuration of IE to handle ActiveX Controls. If the database identified in the ActiveX Control contains VBA (Virtual Basic for Applications) code, the code will automatically be executed when the database is opened. This vulnerability in essence enables an Attacker to embed malicious VBA code within a MS Access Database and have it executed whenever an IE browser interprets an HTML document referencing the corrupted MS Access Database as an ActiveX Control.

Protocol Description

The core of the vulnerability described in this paper is the inability of MS Internet Explorer web browser to properly handle MS Access-type ActiveX Controls referenced in HTML (HyperText Markup Language) documents. MS Internet Explorer processes HTML documents via two prominent Internet applications: (a) documents downloaded from Web Servers via the HTTP (HyperText Transfer Protocol) protocol, and (b) e-mail messages from Mail Servers that process HTML-formatted e-mail. As a consequence, the two prominent protocols involved with this vulnerability are HTTP and mail service protocols. Of these two, HTTP is intimately germane to this vulnerability. However, although the vulnerability associated with HTML documents can also be accessed through HTML-formatted e-mail, the actual mail service protocols that are responsible for delivering HTML-formatted e-mail messages are not germane to this vulnerability. As a consequence, a discussion of mail service protocols is not relevant to the scope of this document.

HTTP is a simple, stateless client-server protocol, which involves two elements: a web client and a web server. In essence, the web client establishes a TCP connection to the web server, issues a request, and reads back the server's response. The web server signifies the completion of the response to the web client by closing the TCP connection.

HTTP web servers are vastly simple as compared to web clients. Within HTTP, there are two message types, requests and responses. The format of a request message consists of a request line, optional header(s), and a body (under certain circumstances). The request line consists of request URL and version of HTTP being used by the requestor. Under HTTP/1.0, there are only three types of requests: the GET request, the HEAD request, and the POST request. When a web server receives a GET request, it simple returns the file specified by the URL contained in the GET request. The HEAD request is similar to the GET request except that the web server only returns the header information contained in the HTML document. This request is generally used by web clients to test a hypertext link for validity, accessibility, or for modifications. The POST request is used for posting e-mail, news, or forms intended for fill out by an interactive user. The POST request is the only request message that actually has a body associated with it. The format of the response message consists of a status line, optional headers, and a body. The response status line contains the version of HTTP being used by the responder, a response code, and a response phrase.

In HTTP/1.0, both the request and response message types can contain a variable number of header fields. A header field consists of a field name followed by a colon, a single space, and the field value. Headers are divided into three categories, those that apply to request, those that apply to responses, and those that apply to describing some characteristic of a body section contained in a message. Of particular importance to HTTP are the headers corresponding to the data type of the body section of a message: the Content-Type, Content-Length, and Content-Encoding Fields. Through this simple mechanism, files and data are transferred between web clients and servers. The HTTP protocol is discussed in detail in reference [Berners-Lee, Fielding, and Nielsen 1995].

HTML documents are the principle documents returned by web servers via the HTTP (HyperText Transfer Protocol) protocol. HTML is a language derived from SGML (Standard Generalized Markup Language) and is discussed in detail in [Berners-Lee and Connolly 1995]. An HTML document is an ASCII-based language utilizing the 8-bit ISO Latin 1 character set. The HTML language largely consists of command entities referred to as tags that direct the formatting of text to be rendered in the main frame of a web browser window. Most HTML commands form a start-end pairing, which forms the basis for encapsulation. For example, a complete HTML document begins with the <HTML> tag and ends with the <HTML/> tag, encapsulating the entirety of the HTML document. The typical HTML document also contains a header section and a body section. These sections are demarcated with the <HEAD> / <HEAD/> and <BODY> / <BODY/> tags, respectively. All in all, HTML has the characteristics of a document formatting language like Troff or PostScript. However, HTML is much more than a formatting language in its ability to specify both the data and structure of a document.

An HTML file can also contain script code that executes when the file is referenced from the operating web browser. An example of script code that is executable from the Internet Explorer web browser is Microsoft Visual Basic Scripting Edition (VBScript), which is a subset of the Microsoft Visual Basic for Applications (VBA) development environment and macro language. VBScript is implemented in Internet Explorer and other applications

that utilize ActiveX Controls and Java Applets. Within an HTML document, the <SCRIPT> and <SCRIPT/> tags are used to demarcate script code. As in the following example:

```
<HTML>
…
<SCRIPT>
function f1()
{
…
VBScript commands
…
}
<SCRIPT/>
…
<HTML/>
```

Particular to the vulnerability described within this paper, scripts may contain one or more <OBJECT> tags. Object tags are used to load ActiveX Controls, where data property of the <OBJECT> tag specifies the actual ActiveX Control to be loaded. ActiveX Controls normally take the form of executables. However, for Windows platforms, all Microsoft Office documents are defined as ActiveX Controls.

As previously indicated, the vulnerability described in this paper also applied to HTML-formatted e-mail. However, the Internet Explorer Script Vulnerability is not realized through delivery of HTML-formatted e-mail via the available mail service protocols, but actual viewing of HTML-formatted e-mail through the currently popular e-mail programs such as Outlook, Outlook Express, and Eudora when the interpreter for HTML-formatted e-mail is the Internet Explorer engine.

Description of Variants

In essence, the vulnerability described in this paper is a generic VBA (Virtual Basic for Applications) code execution vulnerability. This vulnerability lies uniquely within Microsoft Internet Explorer in its inability to properly handle Microsoft Access Database-type ActiveX Controls. This particular vulnerability is unique as far as the author has been able to determine. However, exploits based on this vulnerability are only limited in the extent of what can be accomplished with VBA code and the level of access the victim computer has to other computers in and out of its local environment. One candidate exploit of this vulnerability is described in this paper, to execute a binary executable (i.e., BackOriface2K) accessed from a remote computer on the victim computer. However, many exploit variants are possible with this vulnerability.

How the Exploit Works

Given that the primary focus of this paper is on the vulnerability afforded by this error in Internet Explorer, the discussion will be divided into two parts: (a) the characteristics of the vulnerability (this section), and then (b) a potential exploit for this vulnerability (next section). With regard to this vulnerability, the minimum criteria for this vulnerability to be realized on a host computer is installation of the following:

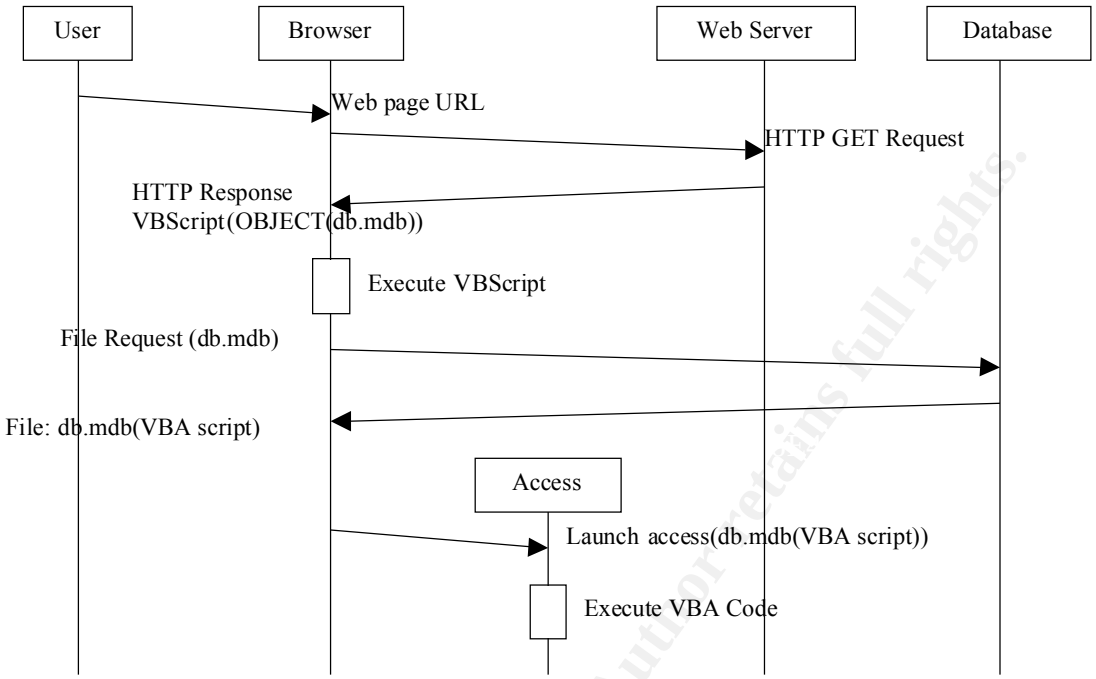
- Windows OS: Windows 95, 98, NT 4.0, 2000
- Internet Explorer v 4.01 SP2 or higher (up through Internet Explorer 5.5)
- Microsoft Access 97 or 2000

With regard to the exploit described in this paper, the further minimum criteria for it also be realized on the host computer is the enabling of both NetBIOS protocols to the Internet and Windows File Sharing (SMB for Windows 9x and NT 4.0, CIFS for Windows 2000) to the Internet.

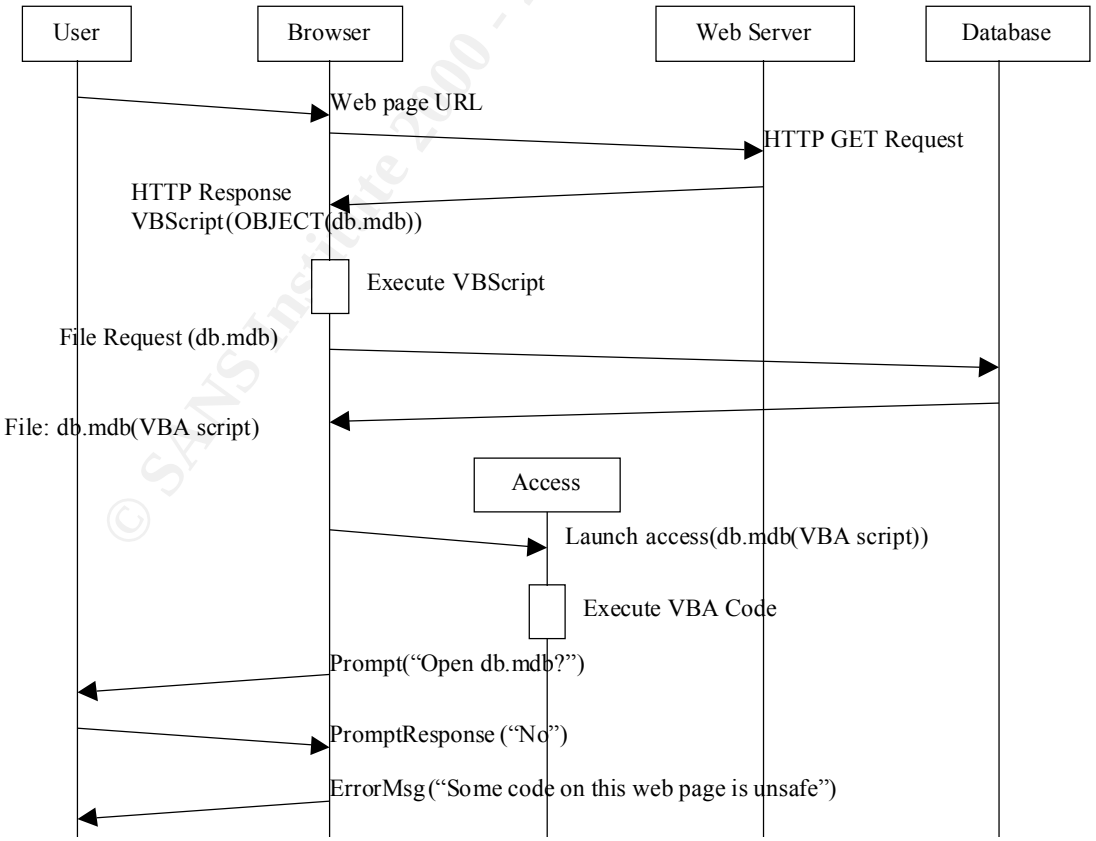
The MS Internet Explorer Script vulnerability is a generic vulnerability from which many different exploits can be accomplished. The essence of this vulnerability is MS Internet Explorer. In general, MS Internet Explorer allows the execution of locally or remotely hosted MS Access databases that are referenced from a web page containing VBScript code. This is generally accomplished through the use of <OBJECT> Tags in VBScripts within HTML formatted documents to load ActiveX controls. By default, MS Access files are treated as unsafe for scripting. However, if an <OBJECT> tag is utilized that references an MS Access file (*.mdb) in a script, execution of the MS Access file is accomplished regardless of the Internet Explorer browser settings. This error in essence causes Internet Explorer to improperly download and open the MS Access Database file, regardless of the configuration within Internet Explorer to handle all ActiveX Controls.

For normal trusted web sites with trusted MS Access databases, this vulnerability generally presents no problem. However, if the web site cannot be trusted and if the MS Access Database identified in the ActiveX Control contains VBA (Virtual Basic for Applications) code, the code will automatically be executed when the database is opened, ignoring the unsafe for scripting safeguard. If the web site was being operated or controlled by an Attacker, the victim computer could execute the malicious VBA code resulting in malicious activity or system compromise. The extent of damage caused by this vulnerability is only limited to the scope of the VBA code executed by MS Access.

The error within Internet Explorer is manifested in the following explanation. In general, Internet Explorer can be configured to handle ActiveX Controls contained within web pages downloaded from web servers in three different manors. In the first manor of configuration (i.e., default installation), Internet Explorer allows all ActiveX Controls. As a consequence, all ActiveX Controls encountered are loaded silently, without any prompting of the user. In the case of the Active Control referencing the MS Access database with the malicious VBA code, the exploit would be automatically executed without knowledge of the user. The session would be as follows:

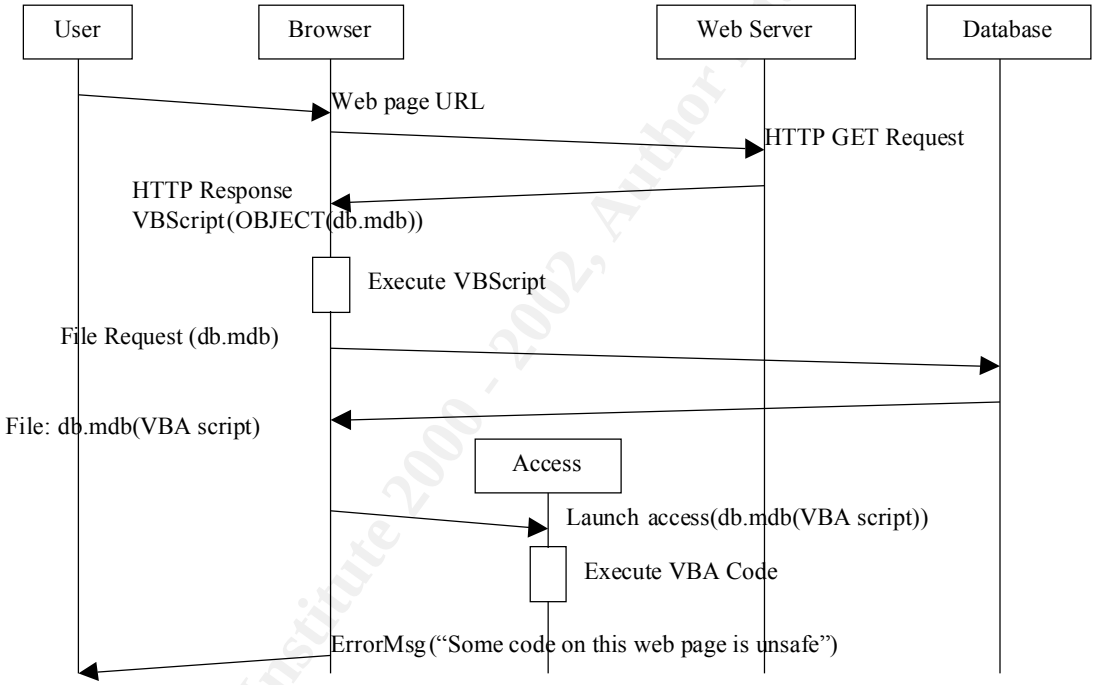


As indicated by the diagram, the VBA code is executed without warning to the user. In the second manor of configuration, Internet Explorer can be set to prompt the user for confirmation or denial of each ActiveX Control as it is encountered. Under this configuration, the above session would be as follows:



Under this configuration, Internet Explorer warns the user that some of the code on the requested web page is unsafe, but only after it downloaded the database file, and used it to launch MS Access. At this point, the VBA code internal to the database file is executed within MS Access, BEFORE the user was even prompted to confirm whether or not the database file embedded in the web page script should opened. Only after a negative response to that query is the user warned that the downloaded web page contained unsafe code. It is also significant to point out that even after the negative confirmation to the open database prompt, the user is still not informed that the database has already been downloaded and opened by MS Access. This is the first manifestation of the underlying error within Internet Explorer.

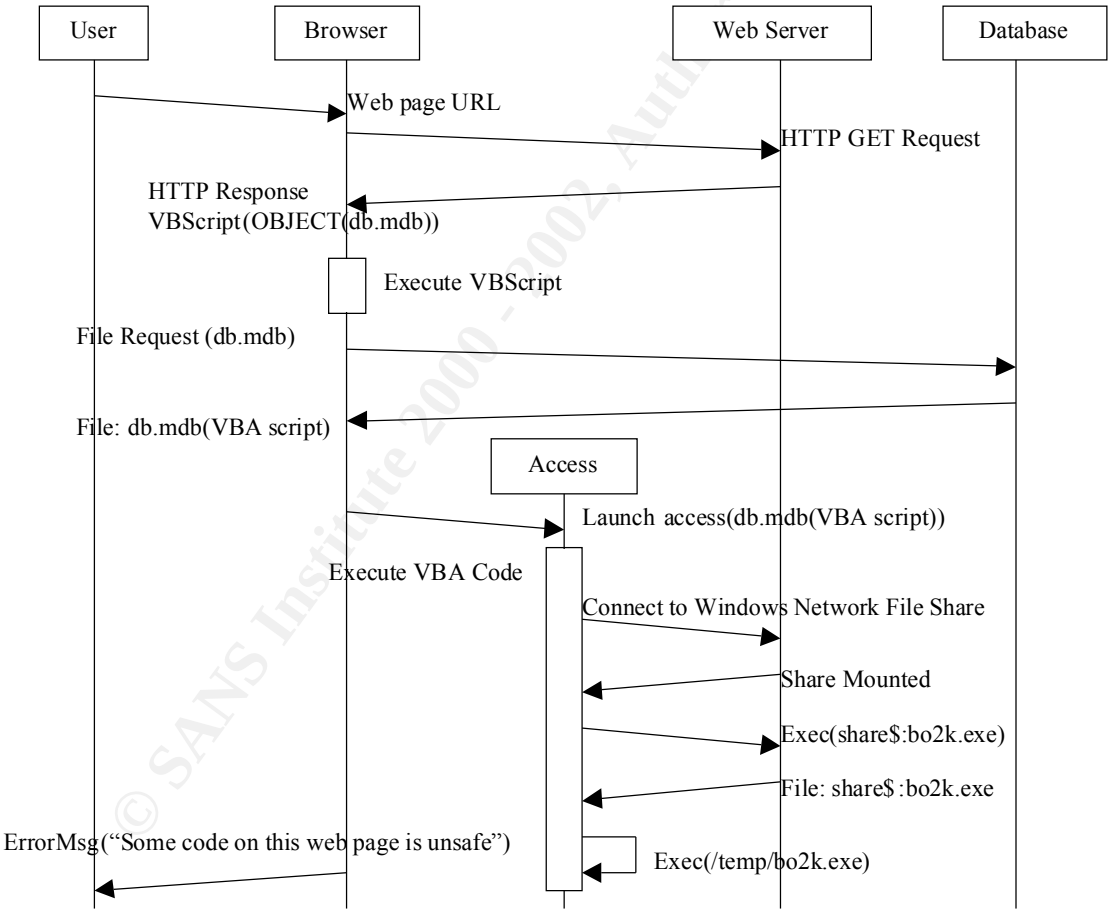
In the third manor of configuration, Internet Explorer can be set to disable execution of ActiveX Controls entirely. The session under this configuration would be as follows:



Under this configuration, Internet Explorer again warns the user that some of the code on the requested web page is unsafe. In spite of this and being configured to disable execution of all ActiveX Controls, Internet Explorer still downloaded the database file, and used it to launch MS Access. At this point, the VBA code internal to the database file is executed within MS Access. Also, the user is not even warned of the fact that the database file was downloaded and opened in spite of the configuration of Internet Explorer, which was in fact ignored.

How to Use the Exploit

In order to illustrate the severity of this vulnerability, a hypothetical exploit has been chosen which can make use of this vulnerability for an Attacker to obtain remote access. This exploit presumes that the target victim can be tricked or manipulated into setting their Internet Explorer web browser to download the Attacker's malicious web page is located. All that is required is to fashion the VBA Code that is executed within MS Access to access a remote share on an Internet host (the Attacker's computer) and run a program form that share, namely Back Orifice 2000. All that is required from the Attacker's host is to open a share to the Internet containing the Back Orifice 2000 executable. The VBA code is then fashioned to mount that remote share and explicitly execute the required executable. The executable is then loaded from the remote share onto the victim computer and hence Back Orifice 2000 is installed. The session for this series of events would be as follows (assuming that the Internet Explorer browser was configured to disable execution of ActiveX Controls entirely:



As the diagram illustrates, the Back Orifice 2000 executable is downloaded from the mounted share an is executed as a local program on the victim host, just like any other executable could be run from a locally mounted share.

Signature of the Attack

A possible signature for this attack would be to log and trap all file transfer activity involving MS Access database file types (i.e., ADE, ADP, MDA, MDB, MDE, and MDW). Another potential indicator is to log and trap the use of scripts (as identified by the use of <SCRIPT> / <SCRIPT/> tags) in Internet Explorer HTML documents downloaded from web servers external to the host local network.

How to Protect against the Exploit

In reference to the vulnerability itself, Microsoft indicates that a patch is available that removes the <OBJECT> tag vulnerability whenever it references a Microsoft Access file. The patch is available for Internet Explorer v4.01 SP2 and higher at:

http://www.microsoft.com/windows/ie/download/critical/patch11.htm

In regards to this specific vulnerability in the event that the aforementioned patch cannot be applied, the only workaround that eliminates or reduces this vulnerability is to disable the file mapping to msaccess.exe for the following file types: ADE, ADP, MDA, MDB, MDE, and MDW. As a consequence, any time MS Access Database is run from the mouse, the user must manually map the file to the MS Access application.

In regards to the sample exploit described in this paper in the event that the aforementioned patch cannot be applied, the following workarounds should be applied to decrease exposure to and/or limit damage resulting from this or similar exploits:

- Block all outgoing Windows File Sharing (CIFS) ports at the network border router / firewall. In particular, block all outgoing traffic to ports UDP 138 (NetBIOS Datagram Service), UDP and TCP 139 (NetBIOS Session Service), UDP and TCP 445 (MS CIFS for Windows 2000). This ensures that the exploit cannot run malicious programs on the Internet though Microsoft shares.
- Block the transfer of all Microsoft Access file types (ADE, ADP, MDA, MDB, MDE, and MDW) across all border routers to the Internet. However, this must be done on all ports through the router, since an attacker can designate any port as the destination for a file request. This also minimizes the possibility of having an MS Access file transferred from the Internet to a host on the internal network.

Source Code / Pseudo Code

No known exploit code is publicly available for this vulnerability.

Additional Information

Microsoft Security Bulletin MS00-049 [MSSB MS00-049] documents the IE Script vulnerability. This bulletin is available at:

http://www.microsoft.com/technet/security/bulletin/MS00-049.asp

Background information on the vulnerability as well as the other vulnerabilities that the Internet Explorer repairs is available in the following Microsoft Knowledge Base Article [MSKB Q269368] "Patch Available for Vulnerabilities in Internet Explorer." The article may be referenced at:

http://www.microsoft.com/technet/support/k.b.asp?ID=269368

Microsoft has also posted a FAQ for Security Bulletin MS00-049 [MSFAQ MS00-049]. This FAQ is available at:

http://www.microsoft.com/technet/security/bulletin/fq00-049.asp

References

[Stevens 1996] "TCP/IP Illustrated, Volume 3: TCP for Transactions, HTTP, NNTP and the UNIX Domain Protocols," Addison-Wesley, Reading, Mass.

[Berners-Lee, Fielding, and Nielsen 1995] "Hypertext Transfer Protocol—HTTP/1.0," Internet Draft, Link: ftp://ds.internic.net/internet-drafts/draft-ietf-http-v10-spec-02.ps

[Berners-Lee and Connolly 1995] "Hypertext Markup Language—2.0," Internet Draft, Link: ftp://ds.internic.net/internet-drafts/draft-ietf-html-spec-05.txt

[MSSB MS00-049] "IE Script Vulnerability," Link: http://www.microsoft.com/technet/security/bulletin/MS00-049.asp

[MSFAQ MS00-049] "MS00-049 Frequently Asked Questions," Link: http://www.microsoft.com/technet/security/bulletin/fq00-049.asp

[MSKB Q269368] Microsoft Knowledge Base Article, "Patch Available for Vulnerabilities in Internet Explorer," Link: http://www.microsoft.com/technet/support/k.b.asp?ID=269368

Last Updated: July 22nd, 2017

Upcoming Training

Security Awareness Summit & Training 2017	Nashville, TN	Jul 31, 2017 - Aug 09, 2017	Live Event
SANS San Antonio 2017	San Antonio, TX	Aug 06, 2017 - Aug 11, 2017	Live Event
SANS Boston 2017	Boston, MA	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Salt Lake City 2017	Salt Lake City, UT	Aug 14, 2017 - Aug 19, 2017	Live Event
SANS New York City 2017	New York City, NY	Aug 14, 2017 - Aug 19, 2017	Live Event
SANS Virginia Beach 2017	Virginia Beach, VA	Aug 21, 2017 - Sep 01, 2017	Live Event
SANS Chicago 2017	Chicago, IL	Aug 21, 2017 - Aug 26, 2017	Live Event
Community SANS Memphis SEC504	Memphis, TN	Aug 21, 2017 - Aug 26, 2017	Community SANS
SANS Adelaide 2017	Adelaide, Australia	Aug 21, 2017 - Aug 26, 2017	Live Event
Mentor Session AW - SEC504	Milwaukee, WI	Aug 23, 2017 - Sep 29, 2017	Mentor
Mentor Session AW - SEC504	New York, NY	Aug 24, 2017 - Sep 08, 2017	Mentor
Mentor Session - SEC504	Denver, CO	Aug 29, 2017 - Oct 10, 2017	Mentor
SANS San Francisco Fall 2017	San Francisco, CA	Sep 05, 2017 - Sep 10, 2017	Live Event
SANS vLive - SEC504: Hacker Tools, Techniques, Exploits and Incident Handling	SEC504 - 201709,	Sep 05, 2017 - Oct 12, 2017	vLive
SANS Tampa - Clearwater 2017	Clearwater, FL	Sep 05, 2017 - Sep 10, 2017	Live Event
SANS Network Security 2017	Las Vegas, NV	Sep 10, 2017 - Sep 17, 2017	Live Event
SANS Dublin 2017	Dublin, Ireland	Sep 11, 2017 - Sep 16, 2017	Live Event
Mentor AW - SEC504	Santa Clara, CA	Sep 11, 2017 - Sep 22, 2017	Mentor
Mentor Session - SEC504	Arlington, VA	Sep 20, 2017 - Nov 01, 2017	Mentor
Community SANS Columbia SEC504	Columbia, MD	Sep 25, 2017 - Sep 30, 2017	Community SANS
SANS Baltimore Fall 2017	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS London September 2017	London, United Kingdom	Sep 25, 2017 - Sep 30, 2017	Live Event
Rocky Mountain Fall 2017	Denver, CO	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS SEC504 at Cyber Security Week 2017	The Hague, Netherlands	Sep 25, 2017 - Sep 30, 2017	Live Event
Mentor Session - SEC504	Boston, MA	Sep 26, 2017 - Nov 07, 2017	Mentor
SANS DFIR Prague 2017	Prague, Czech Republic	Oct 02, 2017 - Oct 08, 2017	Live Event
Mentor Session AW - SEC504	Houston, TX	Oct 02, 2017 - Dec 11, 2017	Mentor
Mentor Session - SEC504	Columbia, SC	Oct 03, 2017 - Nov 14, 2017	Mentor
SANS October Singapore 2017	Singapore, Singapore	Oct 09, 2017 - Oct 28, 2017	Live Event
SANS Phoenix-Mesa 2017	Mesa, AZ	Oct 09, 2017 - Oct 14, 2017	Live Event
Community SANS Chicago SEC504	Chicago, IL	Oct 09, 2017 - Oct 14, 2017	Community SANS