Global Information Assurance Certification Paper

Mitigating the Dangers of Browsing the Web in a Corporate Environment – A Theoretical System Described

GIAC Security Essentials Certification (GSEC)
Practical Assignment
Version 1.4b (amended August 29, 2002)

Jeffrey D. Rogers

August 21, 2003

Abstract

The object of this paper is to describe a theoretical system - a proxy browser system - which could be used to mitigate the dangers of Web browsing. Unlike a traditional HTTP protocol proxy that ultimately passes the HTTP protocol back to the user's browser this theoretical system would isolate the user's browser from the Web. Isolation would be accomplished using an "out-of-band" protocol between the user's client and a device, perhaps a server running a service, which would act as a proxy browser for the client. The proxy browser would browse the Web for the user's client and pass the results – audio and video, not the HTTP protocol - back to the client by way of an "out-of-band" protocol.

Introduction

This paper focuses on an idea which is an extension of the traditional HTTP protocol proxy[1]. The idea also borrows from the theory and application of an opto-isolator[2] [3], which is a device used in electronic design to isolate one circuit from another. The goal is to describe a system which would help mitigate the dangers of Web browsing. The system would accomplish this by refining the security perimeter - pushing it out, away from the client, to where it belongs – presumably at the firewall.

The Opto-Isolator

An opto-isolator is an electro-optical component that is commonly used in communications systems. When used in a protective role, its purpose is to filter out potentially harmful voltage spikes from electrical signals traversing from one circuit to another and, perhaps, vice versa. In order to electrically isolate the two circuits from each other while maintaining the ability to convey a signal from one circuit to the other the isolator uses an "out-of-band" medium – light – to transmit the signal.

To protect the receiving circuit from a voltage spike the transmitting circuit converts an electrical signal to an optical signal and then transmits the optical signal. The receiving circuit receives the optical signal and converts it back to an electrical signal. This process is the function of the opto-isolator – conversion from an electrical to an optical signal, transmission/reception of the optical signal and conversion back to an electrical signal.

This optical isolation limits the voltage spike vulnerability of the receiving circuit because the optical receiver will not distinguish a brighter light, past a particular threshold, than it's properties allow – i.e. a brighter light does not necessarily mean a higher voltage received ergo the risk of a voltage spike damaging the receiving circuit is mitigated.

The "defense-in-depth[4]" methodology of security suggests that any system which adds protective layers to the security endeavor typically adds protection to the secured. That is the goal of the system described herein – to add a layer of security to Web browsing using a system that is based on the principle of "out-of-band" signaling and which happens to expand on the idea of the traditional HTTP protocol proxy.

The Web Browsing Risk

[1] Netscape Marketing
[2] Unknown
[3] ARC Electronics
[4] National Security Agency

Browsing the Web is a security risk to the corporation[5] [6]. The risk is primarily due to the marriage of two elements of browsing. These elements are the ability of the client to download (or receive) code from the Internet and the ability to execute that code once it is on the client.

Primary Threat to Web Browsing

The threat associated with the risk is the possibility of a client downloading malicious code and then executing it. "Malicious code", being any type of data which, when downloaded, instructs the client to act in a particular way, one that is unforeseen and unwanted by the user and/or corporation. Malicious code could include an executable, a virus, a JPEG file, or even a web page which causes a buffer overflow in the browser. To complicate matters, the downloading and/or execution process could be done knowingly or unknowingly by the user.

Associated Vulnerabilities

Some of the vulnerabilities associated with Web browsing are: browser software complexity, clients being not well protected and even user ignorance. A number of factors exist that add to the vulnerability of browsing.

Not only is Web browser software very complex but many browsers also contain one or more engines which allow them to execute different types of code other than basic HTTP or HTTPS. Engines such as Sun's Java Virtual Machine or Microsoft's ActiveX and Visual Basic are some examples. They facilitate the delivery of an enhanced browsing experience but concurrently provide the client the ability to execute downloaded code, malicious or not. Unfortunately, for the user, complexity breeds vulnerabilities.

Furthermore, corporate users browse the Web using workstations which are generally trusted by the user and by the corporation. This means that the malicious code, perhaps a downloaded trojan, would most likely be executed in a trusted environment, thereby inheriting the trust allotted to the client. Inherited trust may put valuable assets at risk. Assets may include databases or any number of other information sources and these assets may have no way of determining if a client has a legitimate or malicious intent. The problem is exacerbated because the client is trusted.

Current Solutions

There are other technologies, security-related and not, that help mitigate the risks associated with browsing. These technologies include anti-virus systems, proxies, firewalls and user education as well as others. All of these solutions

[5] Obscure^
[6] Religious Society of Friends (or Quakers) in Britain

have associated pros and cons as does the proposed system. However, some are better than others. A few are discussed next.

Anti-virus Systems

Anti-virus systems are certainly a must-have for today's corporations. They are warranted because if the anti-virus system can detect and contain or eradicate the malicious code, whether it is a virus or of some other form, before it has had a chance to do harm then, we have won a battle and we can continue to contain the virus. Anti-virus systems do just that – they are very effective against known, and some unknown malicious code.

Anti-virus systems are, however, not 100% effective against all malicious code. New viruses, with no known signature, are an especially high threat[7]. Anti-virus is a mandatory layer, albeit an important one in our security model, but it is a reactive technology not a proactive solution. It adds a security layer to our "defense-in-depth" security model and it is an effective layer, but Anti-virus should not be the only layer.

Firewall Systems

Basic firewalls are packet filtering firewalls and generally provide little, if any, assistance to blocking infection of a client relative to Web browsing. This is due to the nature of browsing and the way a packet filtering firewall inspects traffic. In most circumstances the firewall is implemented to allow browsing to occur by allowing the protocol's TCP/IP port to pass unrestricted from the client to the Internet. This allows the client to browse in an unrestricted manner. If browsing is unrestricted then the firewall cannot restrict unwanted code from being downloaded because its inspection technique cannot discern malicious code from non-malicious code.

A firewall may however reduce the overall effectiveness of the malicious code once it executes on the client by reducing its ability to communicate and infect other clients or by reducing its ability to gather data to further its attack. Again, the firewall is not a solution. But a firewall is another very important layer for our defense model.

HTTP Proxy System

An exception to the simplistic overview of the protection provided by a packet filtering firewall system is the protection afforded by an application layer firewall. For our purposes an application layer firewall will be considered the same as a traditional HTTP protocol proxy which is discussed in this section. A traditional proxy may or may not help mitigate the risks relative to browsing. It depends greatly on the capabilities of the proxy and how it is configured and implemented.

[7] Sha Sha Chu, Brendan Dixon, Peter Lai, Darren Lewis, and Camila Valdes

Furthermore, a proxy is not generally considered a security system, although it provides security related features. Proxies in their generic form are caching systems used to speed Web retrieval and help minimize recurring traffic from the corporate network to the Internet. The basic proxy simply acts as a relay on behalf of the user's Web browser. If the browser asks the proxy to retrieve content from the Web the proxy will simply perform the retrieval and then forward it, in HTTP protocol form, to the client as requested. As a relay there is little if any protection to the browser against the download of malicious code.

However, if a proxy examines the application layer of the connection and has some built in intelligence and is configured to disallow certain Web browsing content, it may be able to restrict the browser's ability to request and/or receive certain types of material. If the material blocked, in a particular Web browsing instance, is malicious code then the proxy solution has provided security effectively in that particular case.

Unfortunately, restricting content may have an undesirable effect for non-malicious content accidentally filtered. A proxy induced restriction of acceptable content may come at a steep cost to the browsing experience. Even basic restrictions may hobble many Web sites whether the content is malicious or not. This is because it is difficult to determine good code from malicious code – the proxy would most likely disable a particular function by removing content such as all Java applets, not just the malicious ones. A proxy may be an effective solution, but the hobbled experience may not be acceptable for all environments.

The Problem with These Solutions

The primary disadvantage with the aforementioned solutions is that ultimately the client performs the actual Web browsing regardless of whether content is filtered or not. The client runs Web browser software that interacts directly with the Internet. This provides a direct path for malicious content to travel from the Internet to the client. Even if the content is filtered for malicious code prior to being received by the client there is the possibility that the filter will miss something and it will be passed on to the client. A "solution" may be to break this direct connection - the main concept of the browser proxy system.

The "Out-of-Band" Browser Proxy System

Now that the problem and some current "solutions" have been described consider a different approach: a Web browser where it is impossible to either download (or receive) code. In the case of traditional Web browsing it would most likely be an exceptionally boring browsing experience; however, the security risk to the corporation would be greatly reduced compared to unrestricted client browsing.

If the vulnerability is substantially reduced then the perimeter, with respect to browsing, would effectively be pushed away from the client and back toward the firewall where it belongs. Specifically and most important, the perimeter with respect to Web browsing would no longer be at the client. This alone is a worthwhile accomplishment because it would go a long way toward mitigating the risk of Web browsing by severing the direct client-to-Internet path of traditional browsing.

This is the focus of this paper – an "out-of-band" browser proxy system that would effectively refine the security perimeter by moving it away from the client. Most other approaches as described earlier, if they are to be effective, have a negative affect on the browsing experience. Typically they diminish the effective experience by disallowing functionality that webmasters utilize to "deliver" the Web to the user. Functions not allowed may include Java, ActiveX, Flash and others. These are all delivery methods that provide an enhanced browsing experience; however, they usually require the download and execution of software at the client which are the two elements we explicitly want to deny – the Web browsing vulnerability.

To accomplish this we need to stop, take a step back and think for a moment about what the Web's primary functionality is and then we can understand how an "out-of-band" browser proxy system could possibly operate. The Web's primary functionality is to make it easier to get around the Internet, and find things so that ultimately you are provided with what you are looking for, or sometimes what you're not looking for. The Web browser is an interactive portal to the sights and sounds of the Internet. Maybe a little more or less, but that's basically what it is.

To facilitate the portal process, like a traditional HTTP protocol proxy, the "out-of-band" proxy browser would browse for the user's client, but instead of simply relaying the HTTP protocol back to the client's Web browser, as the traditional proxy does, client software would communicate with the "out-of-band" browser proxy via an out-of-band communications protocol. The client software would act like a "remote control" controlling the browser proxy. Such a system would enable the user to experience the Web on their client, but the browser proxy would be doing the actual browsing thereby offloading the risk from the client to the proxy browser.

Browser Proxy System Design

There are several factors to consider in designing such a system. Some factors are usability, functionality, scalability and system security and will be described herein. Overall, the system must be simple for the user, perhaps even seem almost identical to traditional browsing. It must provide a higher level of security than traditional browsing and it must deliver as close to the full experience of the

Web as possible. And, these features must be maintained for tens, hundreds and perhaps even thousands of simultaneous users.

Whether or not all of these features or any of them could be provided is not within the scope of this paper. Such a system is presumably theoretical and each of these features will be outlined on this premise. Scalability may be an especially elusive element considering the use of "remote control" technology. Historically, "remote control" applications are processor intensive. Delivery of the Web in an unabridged format may be equally elusive. Many other problems arise such as legitimate file downloading, printing and perhaps many other issues relative to unforeseen Web content. "Heightened security" is a feature that only time and practice can attest to.

The browser proxy system described herein would be comprised of three main components: the browser proxy, an "out-of-band" protocol, and a software client on the user's computer. The software client would allow the client to communicate via the "out-of-band" protocol to the browser proxy, probably via "remote control" technology, thus allowing the browser proxy to browse for the client while allowing the user to experience the Web as intended – regardless of the existence of malicious content.

"Out-of-Band" Protocol

A protocol/application such as X could be used for the "remote control" function. Other protocols or methods could be used as well such as VNC, Microsoft Terminal Services or even a protocol and system designed specifically for application to this system. In fact, the specific protocol used is not important. What is important is that "out-of-band" communications are utilized, i.e. there is no chance of simply relaying the HTTP protocol back to the client. The purpose is to create a separation between the client and the browser proxy which would presumably provide added security in a way like the opto-isolator provides protection to electronic circuit interconnectivity.

Furthermore, connections should be initialized from the client on an internal network, to the proxy browser which would then open connections to the destination HTTP server. This should complete the connection initiation and there should be no return connections initiated from the browser server to the client. This "outbound" flow of connections suggests that the proxy browser be situated on an untrusted or semi-trusted network such as a DMZ (see figure 1) and that a connection policy be enforced by a firewall or a router to only allow this outbound flow of connection initiation. The DMZ network configuration will help contain the Browser Server should it be compromised[8].

[8] Danielrm26

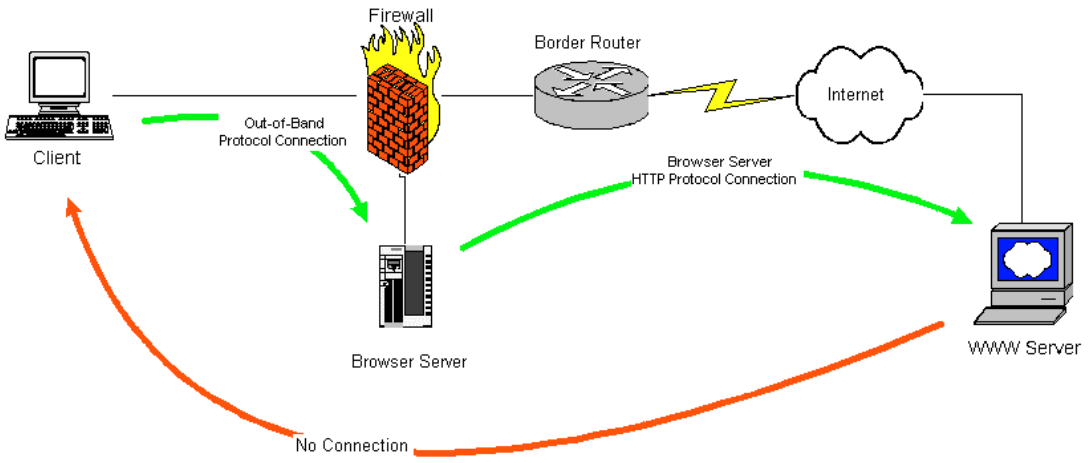

Figure 1 - Outbound Connection Flow

Software Client

The software client could be a common "remote control" application if an off-the-shelf solution is used to implement the system or it could be an application engineered specifically for this system. Regardless of its origin, the software client would need the ability to provide sight to the user as well as sound in the case where a Web site contains such content.

If the client is engineered specifically for this system then there are many possibilities. One such possibility is a client written to a browser-embedded engine. This may be advantageous as it could be integrated in such a way that its use is transparent to the user. For example, if the client software were written as a Java applet it may be possible that when the user opens their browser, such as Netscape and attempts to connect to a Web page, the request is intercepted by the proxy browser system. Once intercepted, a Java client could be invoked in Netscape which could work with the proxy browser as the user's client to display the page on the client. This entire process could be completely transparent to the user.

Obstacles for the client may include user printing, software downloading, as well as unforeseen others. Legitimate software downloading is a problem specific to this system because downloading software is explicitly what we want to deny – how would it determine legitimate software from illegitimate?

Proxy Browser

The proxy browser is probably the most likely area where development could reign in such a system as this. The technologies and methods applied in this component would most likely dictate the technologies used in the other components of the complete system as well.

The proxy browser would essentially have to be a "complete" Web browser, minimally containing a browsing engine (see figure 2). Also, some processing would certainly need to be performed on the proxy browser so that the Web page could be rendered properly for users without returning the original HTTP protocol to the client. Processing related to the "remote control" functionality would also need to be performed – most likely, independently for each user. Furthermore, to accommodate scalability one proxy browser would most likely need to spawn many sub-proxy browsers to accommodate multiple simultaneous users as shown in figure 2.

Other publicly accessible security conscious implementations such as some ftp and e-mail servers utilize a sandbox technique[9] to limit the effectiveness of an attack on their service. This same technique could be used for the many sub-proxy browsers to limit the ability of "crosstalk" between browsing sessions, limit the possibility of escape into the underlying browser server operating system, as well as provide on-the-fly tear-down and construction of new compartmentalized sub-proxy browser sandboxes.

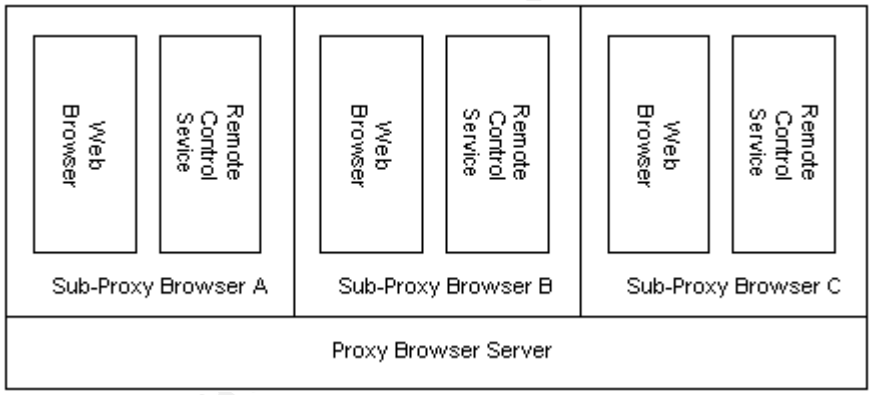

Figure 2 - Proxy Browser Server Block Diagram

Advantages of this approach

The primary advantage of such a system, over the other systems described, is that the computer actually browsing the Web is no longer the user's computer. Instead it is now a system which, hopefully, is under the control and protection of corporate IT staff. Perhaps a security related group may oversee the security of the system as well.

As a single computer or small set of computers the browser proxy could be configured and secured as a single function system. Extraneous services and software could be removed. Security related patches could be rolled-out in a

[9] Brent

shorter time-frame than required for every Web browser in the organization. Patches and settings could be applied with specificity. Also, if the browser system is patched prior to the clients, it may provide the time necessary to roll-out the patch to all clients without exposing the corporate network to a higher level of risk. The need for every client in the corporate network to be patched for all Web browsing related vulnerabilities may be dissolved as well – this could be a cost benefit to the organization.

Another advantage could be gained if the browser were relocated to a DMZ network rather than being on a network and system that is trusted by the user and by the presiding IT staff. By placing the browser proxy in the DMZ a breech of security on the browser proxy would be less likely to be a security breech for the internal private network. In a DMZ configuration a firewall would be more likely to provide protection whereas it would probably be less effective if the browser were on the internal network and became "infected."

Unresolved Issues

There would most likely be shortcomings of the proposed system which may include implementation hurdles to be overcome. Specific problems could include printing and/or legitimate downloads of software and data by the user. The problem is with respect to legitimate user downloads. How could a user still perform legitimate downloads of trusted software in a simple manner yet be protected from illegitimate and unknowing downloads of malicious code?

Unfortunately, any download could be potentially harmful and there may be no easy way to distinguish the good from the bad - layers of security measures designed to do so seem to be the only protection. This download dilemma is one that could prove to be very difficult to overcome in a security conscious manner. A partial rescue may come from corporate policy which could dictate that no downloading is allowed. If so, this type of a system could easily facilitate such a policy – perhaps much easier than enforcing the same policy in an environment that incorporates traditional browsing. By providing the user only sight and sound via "remote control" there may be no conventional method implemented which would allow the user to download. Such a security "feature" may be necessary to help ensure a "no download" policy.

Possible Security Risks of Browser System

Unfortunately, due to the complexity of computers in general and the never-ending threat of attack, no functional system to browse the Web can be completely secure. The proxy browser system sounds secure, but an attack could be crafted to target possible vulnerabilities this system might have specifically.

An example of an attack could be where the client is forced to download code to itself by virtue of a specially engineered payload that when browsed, is passed to the end client via the "remote control" channel. Instead of experiencing sound or video, due to vulnerabilities that may exist in the browser system itself, the downloaded payload may cause something such as a buffer overflow in the sound processing subsystem which in-turn could cause additional payload to execute. This would definitely be a breach of the "no download" methodology the system strives for. It would be an exploited vulnerability with grave consequences.

Another possibility is one where malicious code "breaks out" of the sandbox. If the system is designed to operate so that the browsing software and client "remote control" functionality execute in a sandbox environment, it may be possible via an exploit to force code execution on the system, outside the sandbox environment. This could potentially lead to system compromise - if the "breakout" alone is not considered one. Once the system is compromised it could be recruited to perform a variety of unwanted tasks, including attacking internal clients that are using the system to browse the Web or perhaps it could be used to eavesdrop on browsed material as well as a myriad of other possible unauthorized activities.

Any number of attacks could be launched against the browser proxy system. Like any other system, it would not be impenetrable. However, it would be an added layer in our security model. And, since security is attained by layering security measures it could potentially slow attacks enough to where they could be caught, hopefully before damage is done. If such a system were designed and implemented it would be interesting to discover its effectiveness against attacks.

Conclusion

The paragraph above states "if such a system were designed and implemented it would be interesting to discover its effectiveness against attacks." As this statement implies, the system described herein does not exist. It is just a theoretical system. Assumptions, possibilities, designs, ideas and concepts are all provided but the system, its components, and its abilities are simply conjecture. It sounds good, but until it is designed, implemented and tested there is no way to know if it's feasible or practical.

The opto-isolator certainly performs its function in the electronics world, but whether the same concept could be effective in this application is unknown. It is probable that it would be effective especially since we are considering its use for a security application. "Secure" is not definite in the world of security. And there's no one system that can provide "secure" security. That's why we apply security systems in layers. That's also why the proxy browser system would probably be effective in a security application – it adds a layer. Standing alone it

is like any other single system security system, but incorporated as a layer it would assist with security. That's the goal of the proxy browser system. To add a layer to the security model utilizing the concept of the opto-isolator coupled with the idea of the traditional proxy system.

References

ARC Electronics. "Opto-Isolator (optical isolation) a Tutorial." URL: http://www.arcelect.com/OPTOISOL.htm (Aug. 21, 2003).

Bellovin, Steven M. and Ioannidis, Sotiris. "Building a Secure Web Browser." Jun. 21, 2001. URL: http://www.usenix.org/publications/library/proceedings/usenix01/freenix01/full_papers/ioannidis/ioannidis_html/index.html (Aug. 21, 2003).

Brent. "How To Install and Configure BIND9 to Run in a Sandbox." URL: http://brent.kearneys.ca/howto/bind9-sandbox.html (Aug. 21, 2003).

Danielrm26. "What is a DMZ?" Broadband Reports.com Security FAQ. #4545. URL: http://www.dslreports.com/faq/4545 (Aug. 21, 2003).

National Security Agency, Information Assurance. "Defense in Depth." June 8, 2001. URL: http://www.nsa.gov/snac/support/guides/sd-1.pdf (Aug. 21, 2003).

Netscape Marketing. "Netscape Proxy Server." URL: http://wp.netscape.com/proxy/v3.5/ (Aug. 21, 2003).

Obscure^. "Browsing Websites at your own risk." URL: http://eyeonsecurity.org/articles/browsing.html (Aug. 21, 2003).

Religious Society of Friends (or Quakers) in Britain. "A Five-Minute Guide Web Browsing Security." The Unofficial Internet Starter Pack for Quakers. Apr. 3, 2003. URL: http://www.quaker.org.uk/unofficial/5browse.html (Aug. 21, 2003).

Stein, Lincoln and Stewart, John. "9. Client Side Security." The World Wide Web Security FAQ. Version 3.1.2. Feb. 4, 2002. URL: http://www.w3.org/Security/Faq/wwwsf2.html (Aug. 21, 2003).

Sha Sha Chu, Brendan Dixon, Peter Lai, Darren Lewis, and Camila Valdes. "Anti-Virus Software." VIRUS A Retrospective. URL: http://www-cse.stanford.edu/classes/cs201/projects-00-01/viruses/anti-virus.html (Aug. 21, 2003).

Unknown. "Opto-isolators." V2.01. Jun. 4, 2003. URL: http://homepages.which.net/~paul.hills/SpeedControl/Optos.html (Aug. 21, 2003).

Last Updated: July 25th, 2017

Upcoming Training

SANS Prague 2017	Prague, Czech Republic	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Boston 2017	Boston, MA	Aug 07, 2017 - Aug 12, 2017	Live Event
Community SANS Omaha SEC401*	Omaha, NE	Aug 14, 2017 - Aug 19, 2017	Community SANS
SANS New York City 2017	New York City, NY	Aug 14, 2017 - Aug 19, 2017	Live Event
SANS Salt Lake City 2017	Salt Lake City, UT	Aug 14, 2017 - Aug 19, 2017	Live Event
SANS Chicago 2017	Chicago, IL	Aug 21, 2017 - Aug 26, 2017	Live Event
SANS Virginia Beach 2017	Virginia Beach, VA	Aug 21, 2017 - Sep 01, 2017	Live Event
SANS Adelaide 2017	Adelaide, Australia	Aug 21, 2017 - Aug 26, 2017	Live Event
Virginia Beach 2017 - SEC401: Security Essentials Bootcamp Style	Virginia Beach, VA	Aug 21, 2017 - Aug 26, 2017	vLive
Community SANS Pasadena SEC401 @ NASA	Pasadena, CA	Aug 23, 2017 - Aug 30, 2017	Community SANS
Mentor Session - SEC401	Minneapolis, MN	Aug 29, 2017 - Oct 10, 2017	Mentor
SANS San Francisco Fall 2017	San Francisco, CA	Sep 05, 2017 - Sep 10, 2017	Live Event
SANS Tampa - Clearwater 2017	Clearwater, FL	Sep 05, 2017 - Sep 10, 2017	Live Event
Mentor Session - SEC401	Edmonton, AB	Sep 06, 2017 - Oct 18, 2017	Mentor
SANS Network Security 2017	Las Vegas, NV	Sep 10, 2017 - Sep 17, 2017	Live Event
Community SANS Albany SEC401	Albany, NY	Sep 11, 2017 - Sep 16, 2017	Community SANS
Mentor Session - SEC401	Ventura, CA	Sep 11, 2017 - Oct 12, 2017	Mentor
Community SANS Columbia SEC401	Columbia, MD	Sep 18, 2017 - Sep 23, 2017	Community SANS
Community SANS Dallas SEC401	Dallas, TX	Sep 18, 2017 - Sep 23, 2017	Community SANS
Community SANS Boise SEC401	Boise, ID	Sep 25, 2017 - Sep 30, 2017	Community SANS
Baltimore Fall 2017 - SEC401: Security Essentials Bootcamp Style	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	vLive
Community SANS New York SEC401	New York, NY	Sep 25, 2017 - Sep 30, 2017	Community SANS
Rocky Mountain Fall 2017	Denver, CO	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS London September 2017	London, United Kingdom	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS Baltimore Fall 2017	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS Copenhagen 2017	Copenhagen, Denmark	Sep 25, 2017 - Sep 30, 2017	Live Event
Community SANS Sacramento SEC401	Sacramento, CA	Oct 02, 2017 - Oct 07, 2017	Community SANS
SANS DFIR Prague 2017	Prague, Czech Republic	Oct 02, 2017 - Oct 08, 2017	Live Event
Community SANS Charleston SEC401	Charleston, SC	Oct 02, 2017 - Oct 07, 2017	Community SANS
Mentor Session - SEC401	Arlington, VA	Oct 04, 2017 - Nov 15, 2017	Mentor
SANS October Singapore 2017	Singapore, Singapore	Oct 09, 2017 - Oct 28, 2017	Live Event