Global Information Assurance Certification Paper

The Information Security Risks and Implications of the Open Source and Proprietary Software Development Models

Alexander Hidalgo, GSEC v1.4b, Option 1

Abstract

The battle between open source and proprietary software has long been raging. There are many different issues and concerns being examined and scrutinized by both sides of this war. In the most recent phase of this zealous dispute, both camps have inspected the issue of security with serious rigor. The open source and proprietary proponents of software development both have excellent points to raise: about their own strengths as well as their opponent's weaknesses.

In this paper I will be examining some of these claims, as well as providing examples of well known vulnerabilities, exploitations and attacks that have a direct correlation to the debate at hand. I will ultimately point out that in the spirit of competition, it is the very presence and rivalry between both parties that in the end provides us with the most secure software, regardless of which development model is used to achieve the end result.

Introduction

Although passwords and firewalls have been in use within the Internetworking world for years, new technologies such as intrusion detection systems, honey pots and honey nets, and other various security related hardware and software solutions are an example of how rapidly changing the information security domain is today. The recent advent and rise in the popularity of wireless networking adds even more new technology to be researched, developed and implemented. It is easy to realize that many aspects of the information security world are still in their relative infancy.

Network security as a profession provides a relatively new approach to information security. Although security implementations have long been present in the world of computer technology, they have often existed as a subset of another specific realm or career. Not too long ago, a network engineering team would have been responsible for the security of a network. Now, more and more, we are seeing not only individuals being assigned specific information security duties, but entire teams within organizations are being structured to address security related issues. Information Security has thus become a career path within the computer science realm. Some companies and organizations are currently even contemplating the implementation of a CISO (Chief Information Security Officer) position to handle the broad range of topics and concerns regarding information security policy execution throughout an

establishment or corporation.

There is a war that has been raging for quite some time within the world of computer software. It is a war that some describe as freedom versus tyranny. Others describe it as accountability and service versus the lack thereof. And as the information security world emerges as a respected sphere of the computer world, it too has been dragged into this battle. It is a struggle of epic proportions that seemingly has no real end or resolution in sight. It is the clash of the open source software development model and the proprietary (or closed source) software development model.

Open source has long claimed to have bragging rights to superiority in the realm of security. We have all heard about the idea that more eyes looking at software equals more secure software. We've also heard plenty from various sources regarding the insecurity of and number of errors in Microsoft products. From the proprietary camp we have heard plenty about the mistake of disclosing all of your source code and the dangerous side effects that this entails. Which side of the issue is correct? Is open source truly more secure? Do the benefits of proprietary software outweigh the possible negative security connotations?

The answer may not be as clear as night and day. It might not be possible to determine if one software development model is truly superior to the other. It should be noted that proprietary and open source software both exist for a specific reason, and the arguments made by both sides of this battle are generally valid and worthy of consideration. It is of utmost importance that the information security world understand each case and each contention, even if it is difficult or impossible to prove the superiority of one over the other.

History of Open Source

Open source software has been around for a very long time, although perhaps not exactly as we know it today. The very first software ever developed was open source, and all software remained this way for many years. When the first computer programmers at universities and government centers began working on projects it was a generally accepted practice to distribute the source code and allow anyone to look at it. It was not considered inappropriate at all to take any piece of software and edit it to suit your own specific purposes. If you found a problem or a bug in the program, it was assumed that you would make the appropriate changes and pass the information along to the original developer of the code. Licensing was not an issue at the time, the concept of software as product wasn't even a fully developed thought and the commercial aspect of computing appeared to rely purely on hardware sales and production.

There was an inherent sense of freedom in regards to information within the world of computer users of the time. There was an impression of unity among these code developers. An impression that important world-changing work was being done by a select and elite few, and that as a group more could

be accomplished than as individuals. Those that developed the code were expected to share knowledge; not sharing that knowledge was considered taboo. Back then, code was free, and anyone could change it to do whatever they wanted or needed; the hacker mentality was born and it was considered a badge of honor.

Security measures were minimal in many systems, because there simply did not appear to be a need for it. Sharing resources was as important as sharing knowledge, and it seemed perfectly acceptable to go into your professor's office late at night to use his terminal; he wasn't there to use it himself anyway. When government regulations regarding passwords came into action, Richard Stallman at MIT's AI Lab even went as far as to encourage all of his student to use a single carriage return as a password.

UNIX was the operating system of choice by the mid 70's. Although developed mostly by Bell Labs, it was a prime example of an open source software project. Developers from universities and institutions around the world had made countless contributions to the project as the operating system grew and changed. Other things were changing as well, though. As the government was dividing AT&T and Bell Labs in the late 70's, UNIX became privately held by Bell Labs. It was listed as an asset, and was given a price tag. No longer were changes by outsiders welcomed with open arms, and no longer was there a widely used and supported operating system that could be acquired for free.

Making the best out of a bad situation, developers from around the world began work on their own operating system projects. At Berkley University a very strong UNIX support and development group had already emerged, and over the years their Berkley Software Distribution (BSD) version of UNIX replaced all of the original Bell Labs code. Andrew Tannenbaum in Amsterdam wrote a complete UNIX clone from scratch for his university students and named it MINUX. Soon there after, Linus Torvald, who was both unhappy with the capabilities of MINUX and unaware of the presence of BSD, began working on his own built-from-scratch UNIX clone named Linux. All over the world developers quickly resolved the issue of their free operating system being stolen away from them, and the true movement behind open source software had begun.

Is Open Source More Secure?

Proponents of the open source model will openly claim that an open source format always has been and always will be superior to a proprietary system. The thinking is this: the more people that look at a piece of code and the more people that you have working on this code, the better the end product will invariably be. On the other side of the coin however, proprietary software enthusiasts will claim the same is true of their model. They believe that a more focused group of programmers, required security checks and accountability for all implemented code will yield both a more stable and a more secure product.

Both parties appear to have valid arguments, but which can users trust more?

Open source projects have the potential to be more secure than closed source projects. With the number of developers some of these projects have, and the fact that the source code is open and available to the world, open source projects could in theory surpass proprietary forms of software in virtually all aspects. Unfortunately, it does not appear that this theory holds true in the real world.

First, the very backbone and selling point of open source must be scrutinized. Is the complete and total disclosure of a product's source code conducive to a more secure product? It is perhaps important to note that it is not only the actual security of a product that is important, but also the knowledge of the insecurities as well. If a program has several major security flaws that are never discovered, it does not, in the end, make this program any less secure. Compare this with a program that has but one minor, yet highly documented and recognized, security blemish. Allowing the whole world to peek in on your development can yield positive results, but in with the good come the bad. For every trusted developer you have examining your code, you must only assume you have one person viewing it with malicious intent.

With the disclosure of your project's source code, you are inviting everyone to search through your code and find errors and security leaks. The motivation for searching for security vulnerabilities ranges across a wide gambit of possibilities. There are those with selfless intentions and only the good of the community at heart. There are also has those that are specifically looking for a security lapse to further their own agenda. This could be as simple and seemingly harmless as wishing to gain recognition within the industry, or it could be due to a desire to use the discovered exploit for malicious intent.

Since open source software is generally free, you can by and large procure these products by downloading them from a website. Many open source projects have a centralized "official" web page, which might be located on their own web server (i.e. www.redhat.com or www.apache.org) or on a community-based development site (i.e. www.sourceforge.net or www.freshmeat.net). Often CVS (Concurrent Version System) repositories are also available, allowing users to download up to the minute and prerelease versions of these projects.

Although many users will download their open source software from official websites and servers, many also download their copies from mirror sites. A mirror site is merely a server that has offered to make the specific software available for download to the public. There are a countless number of mirror servers out there distributing a countless number of open source software packages. For example, Linux.Com is a website that hosts lists upon lists of these mirror sites for the many different Linux distributions. One visit to http://www.linux.org/dist/download.html?ID=30, for example, will list websites and FTP servers that have RedHat Linux available for public download. Can the integrity and veracity of these sites be fully confirmed? How do we know that what we are downloading is exactly what we believe we are downloading?

The thought behind open source software is to allow contributions from any source. Any single programmer who believes that they might have

something of worth to add to the project is encouraged to do so. Even programmers with good intentions often let security glitches slip past, and therefore it is not surprising to learn that implementing a vulnerability of some sort is not an amazing or difficult feat. What do you know about your contributor? This contributor could be a child prodigy from Eastern Europe or a grizzled Internet veteran. This contributor could also be a highly skilled cracker with malevolent intents, an American teenager with an anti-establishment outlook, or even a trained official from a non-neutral or hostile foreign nation.

As a whole, open source software is free. It is free for legitimate users, and it is free to malicious hackers. Its source code can be read by bug trackers, and its source code can be read by malicious hackers. It is used by the network security industry, and it is used by malicious hackers. The open source model may have many benefits, but from a security stand point, it is difficult to determine if these out-weigh the possible negatives.

The War Against Microsoft

Microsoft: the proprietary software giant, and the sworn enemy of most of the open source movement. Alleged destroyers of information freedom, and supposed purveyors of insecurity. Bill Gates, Microsoft's CEO, himself appears to have much to do with the advent of software as product. As early as 1976 people can recall Gates sending messages across networks warning people against using and/or changing his programs and source code without authorization. At the time, this shocked people. Today it would appear wholly normal.

Although there are obviously many developers of proprietary software besides Microsoft, it is generally Microsoft that we speak of when regarding an open source versus proprietary debate. Especially from a security stand point, this is a war of Linux versus Windows. One of the most used protocols on the Internet is HTTP, and there is a competition raging between Apache and Microsoft IIS. Interactive web content is becoming more and more necessary, therefore a battle between PHP and Microsoft's ASP is on the horizon. Databases are always of major importance, and hence we have mySQL taking on MicrosoftSQL. It is not only Microsoft that might make security mistakes in the proprietary software world, but it is Microsoft's dominance in products that are more security sensitive (operating systems and Internetworking software primarily) that has made them the target of scrutiny concerning their security practices.

In the aftermath of both the Code Red and Nimda worms that ravaged Microsoft IIS systems during the summer of 2001, Microsoft has emerged with a new focus on security with the announcement of it's Strategic Technology Protection Program in August of 2001. However, there are still many that are doubtful that Microsoft can produce products as secure as they are promising. This is mostly due to Microsoft's poor track record of security and response to

security threats. Microsoft will have to gain perception as a much more security conscious company if partners and users are to trust its .Net initiative. This initiative is an all-encompassing plan for ubiquitous, secure online services. There has been much publicity made of Microsoft's resolution of focus on the topic of security, but only time will tell if they will be successful.

Microsoft has long carried with it a reputation of developing insecure software. Evidence to support this resides in the long list of security patches for every version of Windows that has been developed. However, it should be pointed out that the fact that these patches exist in the first place is saying that somebody is making sure that fixes are being made. Although a rather flawed record is generally associated with Microsoft, they have also been known to respond to security flaws and issue patches within hours of being notified of a problem. Their Security Response Team is operational seven days a week. Conversely, open source software projects may have had an excellent track record with developing and releasing patches quickly, but in the end it all relies on the hopes that someone, somewhere out there will decide to take time out of their day to develop a fix.

The vast majority of computers in this world are running Microsoft Windows as an operating system. These Microsoft operating systems are used nearly ten times as much as all other operating systems combined, it is only common thinking that more security breaches would come to light merely due to this increased usage. However, on the SANS/FBI Twenty Most Critical Internet Security Vulnerabilities List it might surprise many open source advocates that seven of the twenty listed problems are Linux related issues, while only six correspond to Windows systems (the last six are general and apply to all operating systems). While this does not necessarily denote that Microsoft products are more secure than open source projects, it is stating something in relation to the common thinking of many of today's computer users and experts. The familiar concept of open source simply being more secure may be misplaced.

It must also be pointed out that the open source camp isn't all wrong its accusations on Microsoft and proprietary software's security connotations. In normal closed source development projects, programmers have very minimal risk to personal reputation for errors that they create in a commercial product. An error might affect a performance review done within the company, but rarely does it affect their personal and professional standing. Mistakes are generally kept in house if the project has not yet reached production, and even if a massive mistake is found at a later date after the product has been fully released, the chances of one person being publicly blamed in a closed source environment are very small indeed. Conversely, with every line of code written by a developer in an open source project, they are putting their own reputation in danger and their code under severe amounts of scrutiny.

In addition to this, questions of innovation and creativity over technique and practice can be raised. Those participating in open source projects are more often than not participating for their love of coding, and a desire to help. Generally speaking, they are not paid; coding is a hobby to open source

developers. They spend their free time discussing and reading up on new programming techniques and exchange ideas with fellow programmers from many different educational backgrounds on a regular basis. Can large corporations such as Microsoft provide programmers with the same sort of zeal for their projects? Although creative programming probably cannot be proven logically to be better or worse, it is difficult to see how having contributors that are doing their work for fun would have a negative impact in regards to the quality of the product.

As stated earlier, we have all heard about the security problems Microsoft has had in the past. Although it does appear that some of these claims and accusations might be unjustified, or at least exaggerated, we cannot completely dismiss the problems that exist in Microsoft's products. We can only look to the future to see if Microsoft successfully develops an improved and secure product as they have promised us.

Borland's Backdoor

A backdoor, in the simplest of terms, is an undocumented way of gaining access to a computer system. On December 8th, 2000 a very serious backdoor was discovered in Borland Software's InterBase database software. Hard-coded into the very source code of the software itself, the username "LOCKSMITH" and the password "LOCKSMITH" could be used to gain full access to the database. This could allow a user to change any information they wanted, or even insert other programs that could allow for even more compromising attacks at a later time. The backdoor was traced back to version 4 of the software, meaning it had been present since 1994. At first it was unclear as to whether or not this backdoor had an actual purpose, or if it was added as a purely malicious act by an insider. Soon, however, it was discovered that it had originally been added to allow separate modules of the program to communicate correctly.

The discovery of the flaw took place approximately 6 months after Borland decided to release its software in an open source format. It was even an open source developer at IBPhoenix by the name of Frank Schlottman-Godde that found the error as he was working on an open source version of InterBase entitled Firebird. Chances are, the bug would not have been found if Borland hadn't released InterBase as an open source project. Within hours of the discovery of the backdoor, patches were available and systems were being brought back up in a healthy state. Companies such as Motorola, Nokia, Boeing and the Boston Stock Exchange had all been vulnerable to this overlooked problem for years and years.

A trusted member of the development community discovered the security lapse, and patches were released within hours. However, the source code had already been publicly available for more than six months. Although there is no documentation nor any evidence that anyone had discovered the backdoor before Schlottman-Godde did, it must be generally accepted that the instant the

source code for InterBase was allowed to be transmitted freely, malicious hackers all around the world were scourging through the code looking for security flaws. It was later noted in press releases and articles covering the subject that the actual segment of code that contained the backdoor login information was ridiculously easy to find, and was probably only never found because no one had been looking for it. What these press releases and articles fail to mention, is that it is likely people were already looking for it.

The Code Red Problem

Like any well-orchestrated attack, the Code Red worm incorporated the element of surprise. The Black Hat Security Briefings in Las Vegas were taking place on Thursday July 12th, where a researcher giving a lecture predicted that worms would continue to be a cause for concern on the Internet. Most attendees agreed with the statement. On this same day, a vicious worm had started infecting computers across the world.

Ken Eichman at Chemical Abstract Services first noticed something out of the ordinary on July 13th, when 611 attacks from 27 sources appeared on his over night intrusion detection system logs. Although this was more than he was accustomed to, it was not severe enough for him to raise a major alarm. By Saturday July 14th, the number of servers attacking his system jumped to more than 1000 individual systems, and he knew that it had to have been more than a random fluke. Eichman contacted the Dshield.org mailing list, which compiles intrusion detection system logs for analysis, and the hunt for Code Red had begun.

That Monday evening, analysts at eEye Digital Security were able to obtain a copy of the worm's code and began to analyze the data. They discovered that the worm would deface the web page documents of any infected server, and then begin to scan the Internet for other vulnerable systems. Once identified and found, the worm would infect the new system and the process would begin again. By Tuesday evening the worm had infected an estimated 10,000 systems.

The worm relied on a security hole that had been found about a month before the infections started. eEye Digital Security had discovered the problem and had notified Microsoft and the world about the vulnerability in Microsoft's Internet Information Server (IIS). What was known as the Printing ISAPI Flaw had been discovered and Microsoft had already released solutions and patches. Unfortunately, as is all too common in the computing world, not everyone had patched themselves right away. Microsoft themselves even labeled the security flaw as "urgent", but this apparently did not do much to help the world protect themselves from the attack that would occur a month later.

One of the biggest issues Microsoft has had regarding the reporting of security problems with their own software is exactly how this reporting commonly takes place. When eEye discovered the flaw, they contacted

Microsoft and even worked with them to help correct the problem. They also released a security bulletin, which not only detailed the vulnerability, but gave specific examples on how one would exploit it. Would Code Red have ever taken place if eEye had contacted Microsoft privately about the problem, and allowed them to fix it in more secrecy? Microsoft seems to think so. There are those that believe that the information should be released immediately so that individuals and corporations can take measures upon themselves to effect appropriate protections until a commercial fix is designed and distributed. One could even make the case that eEye released the information to gain notoriety for their firm. In the case of the backdoor in Borland's InterBase, the concept of sharing information seemed to work out for the best when their security flaw was found and fixed. In the case of Code Red, it appears that the disclosure of vulnerability information helped lead to a devastating attack instead.

The Past, Present and Future

Over all, the computer industry has grown tremendously in almost all facets over the last few decades. It has been an unbridled, unplanned and perhaps chaotic growth, but new advances and new technologies continue to surface and make the industry better as a whole. Not many people could say that they are truly disappointed with how the computer technology industry and community has expanded over the years. Our lives are continuously made better by both home and business-based computing improvements.

The advantages of open source are evident, as discussed. More individuals working on and reviewing the product means a higher chance of a secure product. You are not limited to relying upon a small team of software developers, most of whom have probably been assigned a very specific task within the project. Conversely, one also must account for code and contributor trustworthiness, as well as full disclosure of how your system and its services are currently running.

Proprietary software also comes with it's own host of pros and cons. Large corporations often implement smaller groups of more focused programmers – developers that theoretically are much more familiar with their own portions of the code. You are also not giving anyone the blueprint to your system. However, history teaches us that security holes can be found even in closed-source programs, so perhaps not allowing the general public to search for these errors themselves is a detriment. Microsoft does not have an impressive track record to convince us that closed-source development leads to more secure programs.

Although original software intentions may have focused upon freedom of information and a cultural well being, we must remember that it is no longer only "computer-geeks" that sit at the keyboards of our world. Because the user base of the computing world has changed, we cannot expect the ideologies to have remained unscathed. Open source may have been the beginning, but proprietary

software appears to have its place as well. This is not to say that the open source approach is an outdated model. This is merely to point out that we need to take a look at where we are today as an industry and as a community.

We have proprietary developers and we have open source developers. Microsoft claims to improve security, open source junkies refute this, and statistics suggest something in the middle. In our work places and our homes, we use both closed source and open source software. There are proponents and opponents for each, and probably even more people operating somewhere in the middle. Both sides raise good points, and both sides are making good efforts to better themselves. Perhaps this is exactly how it should be. The chance of this changing any time soon is very small. Competition breeds creativity and quality, and until one camp can truly prove a superior product, perhaps the mix of both is the most secure of all.

List of References

Broersma, Matthew. “Study: Open source poses security risks”. 31 May 2002. URL: http://zdnet.com.com/2100-1104-929669.html (5 August 2002)

Kaster, John. “A Little History on the InterBase Security Hole”. 18 January 2001. URL: http://community.borland.com/article/0,1410,26611,00.html (5 August 2002)

Lemos, Robert. “’Code Red’: What went wrong?”. 27 July 2001. URL: http://www.zdnet.com/filters/printerfriendly/0,6061,2799069-2,00.html (26 August 2002)

Lemos, Robert. “Flaws turn security spotlight on open source software”. 4 April 2002. URL: http://techupdate.zdnet.com/techupdate/stories/main/0,14179,5105053,00.html (5 August 2002)

Lemos, Robert. “Microsoft’s security push lacks oomph”. 11 January 2002. URL: http://news.com.com/2100-1001-808010.html?legacy=cnet (26 August 2002)

Lemos, Robert. “Microsoft tries to cage security gremlins”. 6 November 2001. URL: http://news.com.com/2100-1001-275430.html?tag=rn (26 August 2002)

Perens, Bruce. “Open source advocate says, ‘’We’re better at security’’”. 4 April 2002. URL: http://techupdate.zdnet.com/techupdate/stories/main/0,14179,2859555,00.html (5 August 2002)

Rosenberg, Donald K. Open Source: The Unauthorized White Papers. John Wiley & Sons, January 15, 2000.

The SANS Institute. “The Twenty Most Critical Internet Security Vulnerabilities”. 2 May 2002. URL: http://www.sans.org/top20.htm (25 August 2002)

Shankland, Stephen. “Borland InterBase backdoor detected”. 11 January 2001. URL: http://zdnet.com.com/2100-11-527115.html?legacy=zdnn (5 August 2002)

Stiennon, Richard and John Pescatore. “No software will ever be totally secure”. 21 March 2002. URL: http://techupdate.zdnet.com/techupdate/stories/main/0,14179,5105156,00.html (5 August 2002)

“Linux Online – Where to Download RedHat Linux”. URL: http://www.linux.org/dist/download.html?ID=30 (25 August 2002)

Last Updated: July 23rd, 2017

Upcoming Training

SANS Boston 2017	Boston, MA	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Prague 2017	Prague, Czech Republic	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 Adelaide 2017	Adelaide, Australia	Aug 21, 2017 - Aug 26, 2017	Live Event
Community SANS Trenton SEC401	Trenton, NJ	Aug 21, 2017 - Aug 26, 2017	Community SANS
Virginia Beach 2017 - SEC401: Security Essentials Bootcamp Style	Virginia Beach, VA	Aug 21, 2017 - Aug 26, 2017	vLive
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
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 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 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
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
Community SANS Sacramento SEC401	Sacramento, CA	Oct 02, 2017 - Oct 07, 2017	Community SANS
Mentor Session - SEC401	Arlington, VA	Oct 04, 2017 - Nov 15, 2017	Mentor