“Critical Infrastructure Security: The Emerging Smart Grid”

Final Take Home Exam: Spring 2012

EE 582-02

Team Members:

Bhavana Muppala

Dylan Fate

Sarah Sundgren

1) Choose one real distributed application program that you are familiar with (or could become familiar with if you don't know any (!!!)). Write the following:

- *A paragraph or two giving an overview of the application. Make sure to include which generic architecture (client-server, publish-subscribe, P2P . . .) you believe best categorizes the application, and why it does so. Be sure also to describe what application-specific activities the various pieces in that architecture (clients, servers, whatever) are doing: what they are requesting, processing, etc.*
- *A paragraph or two describing whether you believe the following runtime issues are important to users of this application, and why you believe it (e.g., what happens in its absence):*
    - *Low latency network connections*
    - *High bandwidth available*
    - *Perfectly "consistent" and "correct" replies/answers/service versus an inconsistent or approximate reply/answer/service*

Make sure your answer indicates how you would desire to trade off the above runtime properties, and the "worst" for each it could reasonably tolerate.

Answer:

Massively multiplayer online games (MMOGs) are an example of a distributed application programs. Guild Wars, developed by ArenaNet is a successful MMOG that uses some unique technology to accomplish smooth and continuous operation for its clients. The game creates a persistent online world where players can interact via gameplay and chat. To create and maintain this world ArenaNet has developed many new technologies. Guild Wars uses a client-server model, where the server and each client run a parallel copy of the world at the same time. The server is boss, and has the final say of what the final world state is if the local client ever gets out of synch.

When a user wants to download and play the game, they first download a small 90 KB launcher. This launcher establishes a connection with the ArenaNet server and verifies the user has an account. The server inventories the Guild Wars assets on the client's computer. Initially, it has the client only download the character creation and introductory missions. The user can begin playing the game while the launcher continues to download additional areas and missions from the streaming ArenaNet server. This minimizes load screens, and allow for fluid, continuous play. If the game is patched, an individual revision identifier tags the necessary files for replacement in order to implement the change. This way, an update changes only the minimum number of required files on the client's machine. Once the files are on the user's computer, they don't need to be re-downloaded unless changed.

Guild Wars only stores “dumb” assets on the clients computers. These are assets not associated with the logic of the game. This includes art files, sound files, skins, pictures, etc.All “intelligent” assets that contain game logic are stored on secure servers. This limits the amount of hacking and cheating that can occur in the game as users cannot edit how the game is played locally.

Guild Wars uses multiple servers to run the game. It creates what is called a “shardless world”, that doesn’t require a user to remained confined to a single server. Each town has multiple districts, which are identical versions of the town that can support hundreds of users each. When a user enters a town they are placed in a district, but may move to any district of their choosing, provided it isn’t full. This technique limits the maximum load on each server, but still allows players to feel they are all in the same world.

The client server interaction is done in a clever way as well. Instead of transmitting the absolute location of every character and item multiple times a second, vectors are exchanged instead. These vectors have a speed and direction for every object. This allows the client to predict the movements of objects if packets are lost. This way, the server only needs to transmit data when it knows something in the client world is incorrect. For example, when a monster runs towards the player, the server transmits the current location, direction, and speed of the monster in vector form. If packets are lost and an updated vector does not make it to the client, the monster keeps moving along the previously received vector and the client does not notice the interruption in communication. This greatly reduces the latency and bandwidth requirements of the game.

For any online game, there are high standards for latency, bandwidth, and connection reliability. Latency is key from a players perspective. As they play the game and react to situations, the commands they give need to be transmitted to the server and back to the client with as little lag as possible. The latency should be undetectable to the player. The vector based unit movement communication between client and server allows latency to be tolerable to a degree. Bandwidth is important for the servers, as thousands of players may be connected simultaneously. This issue is tackled with the district system for towns that is described above. Maximum bandwidth requirements come when game updates occur. When a patch is released, thousands of players will have to download the changes simultaneously. To help deal with this, patches are usually released around 4 AM when the amount of users online is lowest. The connection quality and reliability is very important, as the competitive nature of the game creates a demand for perfect synchronization between the server and all connected clients. Times when a large number of users are connected and lots of action is going on, like in a large organized guild vs guild battle, require the lowest latency and “perfect” connections. Latency and connection issues may be caused by the server, but mostly are due to the unreliable

connections from the clients. The server must be able to accommodate these dropped packets and not let the problems from a single client affect the gameplay experience of others.

2) Explain in a paragraph or two in your own words the analogy "Middleware is to socket programming what high-level language programming is to assembler programming".

Answer:

Middleware provides a layer of abstraction over the operating system so that the application software can be implemented easier. Although it is software, it is not part of the application software. Instead, it provides services beyond the services that the operating system can provide.

Such services that middleware provides are communication services. The operating system allows a programmer to use sockets in order to communication with different hosts. Directly working with the operating system sockets is doable, but is much more tedious and error prone. Middleware also masks many of the differences found in distributed systems. It can provide a single interface to multiple operating systems or programming languages.

Because middleware sufficiently eases the process of programming with a distributed system, it can be likened to a high level programming language; the primitive operations that middleware builds on top of, such as sockets, can be likened to assembler language. While a distributed system can use only primitives, and software can be developed in assembly, it is much easier to do so using middleware and high-level languages, respectively.

A high-level language is much more readable than assembly. This makes it more efficient and effective to program in as it is readily understandable. But, the compiler will turn high-level language into some type of assembly code. Paralleling this is the process that middleware uses: the software makes use of middleware which in turn uses the operating system services. Both high-level languages and middleware allow programmers to have an easier time developing software while still using the services or language that are not as simple for a programmer to develop with.

3. Which of the following is false for a NASPInet-like critical infrastructure data delivery service for a power grid?

A. The scale is such that tracking per-flow state at a router (or similar forwarding device) is feasible.
B. The admission control perimeter can be complete.
C. Post-error recovery is sufficient.
D. The predictability of the delivery latency, even in the presence of a small (bounded) number of failures, is very good or better.

4. Which of the following is more burdensome for the application programmer?

A. Remote procedure call
B. Publish-subscribe
C. Remote method invocation
D. Request-reply protocols

5. Which of the following is space-uncoupled?

A. Remote procedure call
B. Publish-subscribe
C. Remote method invocation
D. Request-reply protocols

6. Which is not either a motivating observation or a property provided by P2P systems (i.e., what is false):

A. Compared to exploiting in-network logic, the same level of multicast efficiency and very low delivery latency is achievable (even in the presence of a few faults).
B. CPU/GPU and storage resources are often more abundant "at the edges" of a distributed system.
C. A much larger address space can be handled compared to IPv4 or even IPv6.
D. Using only edge resources means that deployment is much simpler than if in-network logic (e.g., GridStat forwarding engines) were required.

7. Which if the following challenge for the bulk power grid cannot be mitigated with much better wide-area communications (and sensors feeding it):

A. The different physics of some kinds of renewable power (e.g., no reactive power) and how that changes the dynamics of a grid at large.
B. Negligible storage of power in the grid.
C. Large numbers of retiring operators who are leaving with a lot of institutional knowledge—intuitive "seat of the pants" understanding—of how their grid operates in many contingencies or unusual situations (and combinations thereof).
D. Increasing stress on grid due to insufficient new transmission capacity compared to increases in both generation and load.

8. Tripwire is a software tool intended to assure integrity of system files by detecting unexpected modifications (such modifications are often a sign of rootkit activity). One version of Tripwire reads the names of the directories to be protected from a configuration file. For each file in the specified directories, Tripwire computes its hash value and stores it in a database. What property must this hash function have?

A. preimage resistance
B. 2nd-preimage resistance
C. collision resistance
D. all of the above

9. SYN cookies (http://cr.yp.to/syncookies.html) are used to mitigate SYN flooding attacks. Which of the following is not used in computing a SYN cookie?

A. Timestamp
B. TCP flags
C. Maximum segment size
D. Port number

10. Suppose you have an intrusion detection system detecting a computer virus on the network with 90% accuracy. Precisely, the IDS detects a connection transferring a virus as an attack with 90% probability and a benign connection as an attack with 10% probability. When 1% of the connections contain a virus, what is the probability that a connection flagged by the IDS as an attack is actually benign?

A. 0–20%
B. 21–50%
C. 51–90%
D. 91–100%

11. The following statements describe Kerberos authentication. Find an incorrect statement.

A. The KDC is a single point of failure.
B. KDC and the ticket-granting server can run on a single machine.
C. Kerberos uses symmetric encryption.
D. Every message in Kerberos authentication is encrypted.

12. Which statement is false?

A. Stuxnet exploited multiple zero-day vulnerabilities.
B. Stuxnet attacked systems from certain vendors only.
C. A network physically separated from the Internet can be infiltrated by Stuxnet.
D. The author of Stuxnet created two fake certificates to sign drivers.

13) The article "W32.Duqu: The precursor to the next Stuxnet" describes Duqu, a threat similar to Stuxnet. Compare Stuxnet and Duqu from the following aspects.

(a) Initial infection

(b) Propagation

(c) Command and control

Answer:

W32.Duqu was developed by people with access to Stuxnet source codes. The initial infection methods were varied between the two.

Stuxnet's initial infection was done by exploiting multiple zero day vulnerabilities. Flash drives containing Stuxnet were used to initially introduce the worm to the system. Once plugged in, Stuxnet would autorun and exploit a zero day LNK vulnerability to gain administrative rights and load a temp file into memory. This would hook the Kernel32.dll file and hide the malicious files on the host machine. It would then hook Ntdll.dll and watch for special LoadLibrary calls from the worm with a specific name. That name points to an area where another dll file has been decrypted and stored previously. This method allows the worm to bypass IDS. Stuxnet was able to use real (however stolen) certificates to authenticate the drivers it used.

Stuxnet propagated in various ways and much more aggressively than Duqu. Stuxnet was able to use shared network drives to share itself with a delay of two minutes between infection and execution to avoid IDS. A vulnerability of a print spooler interfaced over RPC was used to futher propagate. Stuxnet used yet another vulnerability to propogate using remote code execution using a specially crafted RPC request that created a malformed path. WinCC machines were infected using a hardcoded database server password. Updates were propagated via peer to peer communication between infected computers.

Command and control was done by accessing foreign websites. First, internet connectivity was checked by accessing a common website like msn.com. Next, a connection was established with a C&C center at www.mypremierfutbol.com or www.todaysfutbol.com hosted in Malaysia and Denmark respectively. Stuxnet would send information about the infected computer to these sites, and the C&C center would respond with a command to execute and RCP routine, or encrypted binary code.

Duqu was installed in one case by utilizing a zero day exploit in Microsoft Word. The exploit uses a kernel mode shell code which first checks a registry value to see if the machine is already infected. If not, it decrypts a driver file and installer DLL from within the Word document. The shellcode allows the driver file to execute and inject code into services.exe. Finally, the installer DLL is executed and the shellcode is wiped from memory. The installation code has three

sections: the Duqu main DLL, the load point driver that starts Duqu after a reboot, and installer config file. The installer DLL decrypts these three files and is able to pass exectution to Duqu's main DLL by hooking ntdll.dll just like Stuxnet. From there is able to export and install itself. The main file and configuration file are encrypted and placed in %Windir%\inf\ folder. The driver file is able to decrypt this DLL when the computer starts up.

Replication and propogation of Duqu was done through lateral network propagation. Unlike Stuxnet, propagation was not a main part of the worm's code. Instead for Duqu, propagation was done actively via commands from C&C servers. The Duqu worm receives a keylogger from the C&C which it can use to acquire network passwords. The Duqu worm can then survey the network using code also received from the C&C servers. When Duqu finds computers of interest, it copies itself to the target computer via a shared folder and can authenticate itself with the acquired passwords from the keylogger. The newly infected computer does not contact the C&C server for further instructions. It instead connects to the infecting computer.

Command and control was done via proxy servers that rerouted to true C&C servers or other proxies. The servers were virtual machines running Linux, and when they were found most of the useful data had been securely deleted. The attackers forgot to securely delete the logrotate log files off the servers. These files gave evidence of an SSH connection that would repeatedly connect, drop and reconnect. These connections had been used to email a logfile to a root user. The content of these emails was found in the /var/spool/mail/root file. It was from this email log data that Symantec determined that some infected machines were communicating through other infected machines, as evidenced by logged port forwarding errors.

14) Read "21 Steps to Improve Cyber Security of SCADA Networks" (http://www.oe.netl.doe.gov/docs/prepare/21stepsbooklet.pdf) from DoE, and answer the following questions.

(a) If properly implemented, how would each step help preventing Stuxnet attacks?

b) Are those steps sufficient in preventing future attacks similar to Stuxnet or Duqu? If yes, justify it. If no, extend those steps to prevent such attacks.

Answer:

1. Identify all connections to SCADA networks.

Network mapping and identification is important to protect against general cyber security threats. Every connection into the SCADA network must be known and sufficiently protected. Stuxnet entered through flash drives due to good security from external connections. It had to use a physical connection rather than a network connection to enter the system.

2. Disconnect unnecessary connections to the SCADA network.

Decreasing the number of unnecessary connections increases the security of the network. Less network connections means more focus and effort can be placed on securing each one. Risk vs reward must be evaluated for each connection. Often a connection to an external network adds convenience, but at the risk of allowing attacks to propagate through that connection. Stuxnet used a printer spooling vulnerability to propagate. If these printers were disconnected this method would have been squelched.

3. Evaluate and strengthen the security of any remaining connections to the SCADA network.

Security must come in quantity as well as quality. A Stuxnet attack moves through the SCADA network by moving from computer to computer looking for paths to higher access levels such as the PLC devices. If the barriers to these access levels were strong, it wouldn't have taken place

4. SCADA networks by removing or disabling unnecessary services.

To decrease the complexity of the system it is necessary to interconnect SCADA network and other networks with minimum risks. Stuxnet spread via microsoft windows as SCADA control servers were built on commercial or open source operating system.

5. Do not rely on proprietary protocols to protect your system.

Proprietary protocols are more apt to have zero day vulnerabilities. In many cases vulnerabilities are found through extensive testing and use. Proprietary protocols have much less testing than public one. Additionally, vendors may have implemented backdoors or vendor interfaces that could be exploited. Stuxnet was able to exploit many zero day vulnerabilities such as the LNK vulnerability and the print spooler vulnerability.

6. *Implement the security features provided by device and system vendors.*

   Security features should be implemented throughout the SCADA system. Stuxnet was able to inject into a trusted process, which varied depending on the type of antivirus software. ETrust v5 to v6 was the only security project that Stuxnet could not bypass. If all security features were implemented like this security product, injection would have not occurred.

7. *Establish strong controls over any medium that is used as a backdoor into the SCADA network.*

   Backdoors or vendor communications create large vulnerabilities that attackers could exploit. Stuxnet did not specifically exploit any backdoors, but it did use stolen driver certificates in order to install kernel-mode rootkit drivers into Windows systems.

8. *Implement internal and external intrusion detection systems and establish 24-hour-a-day incident monitoring.*

   It is important to monitor and report suspicious activity. Stuxnet was able to bypass the host IDS by calling LoadLibrary with a non-existent file name. This activity called a failure of LoadLibrary, and could have been noted by the IDS as suspicious. Additionally, the command and control activities of Stuxnet could have be caught by an IDS, as communication sent to Denmark or Malaysia is not standard.

9. *Perform technical audits of SCADA devices and networks, and any other connected networks, to identify security concerns.*

   Security concerns need to be identified and resolved for all SCADA devices and networks. Audits could find these potential security risks and allow them to be addressed. Stuxnet exploited multiple zero day vulnerabilities, antivirus evasion, and complex process injection and hooking code. If security audits were effective, some of these techniques could be mitigated.

10. *Conduct physical security surveys and assess all remote sites connected to the SCADA network to evaluate their security.*

It is recommended that the security at the site (connection to the SCADA network, mainly remote place) can prevent the unauthorized access. Stuxnet entered the system through flash drives. Physical security could have prevented this.

11. *Establish SCADA "Red Teams" to identify and evaluate possible attack scenarios.*

    Establishing relevant protection strategies with the help of Red Team, which helps in identifying the weakness of the system, could have identified some of the vulnerabilities that Stuxnet exploited.

12. *Clearly define cyber security roles, responsibilities, and authorities for managers, system administrators, and users.*

    There should be a proper establishment of cybersecurity organization which takes care of all mitigations related to risks.

    Knowledgeable personnel could have stopped Stuxnet infection. If no personnel let a foreign flash drive plug into their workstation, the worm could have been stopped.

13. *Document network architecture and identify systems that serve critical functions or contain sensitive information that require additional levels of protection.*

    Detailed understanding of the functions of network architecture is vital for assessing risks and making strategies for the protection of the network.

    The machines that ran Step7 and communicated with the PLCs have critical function and should have had additional security to avoid a Stuxnet-like attack.

14. *Establish a rigorous, ongoing risk management process.*

    Ongoing risk management could have increased the security around the SCADA system and prevented a Stuxnet attack. When Stuxnet first was discovered on July 15, 2010, a denial of service attack took place on the industrial security mailing list servers and prevented communication about the attack from spreading.

15. *Establish a network protection strategy based on the principle of defense-in-depth.*

    Stuxnet propagated through network shares, and print spooler vulnerabilities. Defense-in-depth could have added more security between internal systems and stopped these propagation methods.

16. *Clearly identify cyber security requirements.*

Creating cyber security policies and requirements could have helped prevent Stuxnet. This includes protection from outsiders as well as malicious insiders. If a security policy prohibited flash drives in the workplace, these kinds of attacks could have been prevented.

17. *Establish effective configuration management processes.*

    Any changes to the network, be it hardware or software have to be properly evaluated by the processor for the secure system.

    Attackers execute arbitrary code via .lnk or .pif shortcut files, which would have been prevented by proper management processes.

18. *Conduct routine self-assessments.*

    Getting a feedback on the efficient operation of the cyber security and the technical implementation is very crucial in knowing the loopholes of the network for vulnerabilities.

    Self-assessment could have identified some of the vulnerabilities Stuxnet exploited.

19. *Establish system backups and disaster recovery plans.*

    Quick recovery of the system from the attacks can be done by making changes to the recovery plans from the experience.

    These plans could have stopped the effect of Stuxnet as soon as it was discovered.

20. *Senior organizational leadership should establish expectations for cyber security performance and hold individuals accountable for their performance.*

    If cyber security practices were set out by senior leadership, Stuxnet could have been avoided. This could include cracking down on flash drive usage, and advertising the importance of cyber security.

21. *Establish policies and conduct training to minimize the likelihood that organizational personnel will inadvertently disclose sensitive information regarding SCADA system design, operations, or security controls.*

    The number of zero day vulnerabilities and system exploits used may have been realized thanks to insider knowledge.

b)

Due to the reasons of complexity of the network, cost some steps were not properly implemented. If these loopholes were recovered, the propagation of Stuxnet would have been mitigated.

Of course all of the security possible will never keep a system 100% safe from attacks. But following these 21 will steps will certainly place a system at a very high level of security; thus, preventing many but not all future attacks similar to Stuxnet.

These steps identify how to keep a system safe from attacks. Physical and network accesses must both be protected. By properly implementing these improvements to the cyber security a SCADA network would be safe from a majority of malicious attacks introduced through any type of access. The system would also be able to find and alert persons in charge if a virus had infected the system.

But no network can ever be completely secure. Even if all 21 steps are properly implemented and great minds are working to find and seal loopholes in the system, there will still be malicious people who are able to find different loopholes and exploit them. Implementing these 21 steps will however bring a system as close to secure as possible.