EEC 412, Fall 2012

Quiz #4

This quiz consists of 7 pages. Please check that you have a complete copy. You may use both sides of each sheet if needed.

Your Family name: ________________

Your Given name: ________________

Your student ID: ________________

Name of your left neighbor: ________________

Name of your right neighbor: ________________

#	Points	Out of
1		3
2		3
3		6
4		8
5		2
TOTAL		22

ATTENTION: When necessary, make reasonable assumptions and state them clearly in your solutions.

Notes:

1. Make sure your handwriting is legible. If the teaching staff does not understand what your wrote, they mark your answer as if the unreadable text is missing.

2. Aim to be precise and to the point. The experience of teaching this course since 2004 suggests that excessively long answers tend to correlate with lower marks.

3. As in real world, stated questions and/or accompanied descriptions in this quiz are often open-ended and one has to make assumptions in order to answer them. If you do make assumptions, state them clearly and explicitly.

4. The mark for this quiz will be pro-rated. That is, the best answer receiving 100% and the marks for all other answers being pro-rated accordingly. So, don't panic if you feel like you are severely short on time. Everybody is. ☺

1. **(3 points) Explain what makes software security hard today. Provide examples to support your arguments.**

Complexity - Increase in the complexity of modern software. For example, early system had thousands of lines of codes (LOC), modern software often is measured in millions of LOCs.

Connectivity – Too often software is connected with other pieces of software, libraries of web services. All these pieces often are done by different developers, teams or even companies. Misunderstanding between them leads to bugs, which could be exploitable.

Extensibility – modern systems are highly extensible and can be modified on fly. For instance, Reflection functionality in .NET can help a program to consume another program even without knowing the API beforehand. This increases the effective software base, and thus can cause more bugs, which can be exploited.

2. **(3 points) What is the difference between *Implementation* and** *Design Flaws?* **Provide examples to support your argument. Explain what kind of flaws are hard to detect and why.**

   Implementation flaw is a bug which is usually an implementation that is not done according to the system requirements. Basically, such a flaw is a mistake of the developer who wrote this code. Other examples, buffer overflow, race conditions, unsafe system calls, etc.

   Design flaw is a more serious and fundamental. In particular, it is a flaw in the design specification, where the design decision is incorrect. For example, a design flaw can be a design of a protocol which does not protect against replay attack. Other examples, misuse of cryptography, compartmentalization problems, privilege protection failure, type safety confusion error, insecure auditing, etc.

3. On 28 May 2012 a new worm was identified and reported by MAHER Center of Iranian National Computer Emergency Response Team (CERT), Kaspersky Lab and CrySyS Lab of the Budapest University of Technology and Economics
In the appendix of this quiz, you can find a fragment from Wikipedia page "Flame (malware)." It has been speculated that the malware originated either from Israel or the US and targeted mostly Iranian computers.

Explain how the worm did the following functions:

1. **Reconnaissance**
   Through LAN and usb flash drives. Also uses Bluetooth to search for close enough devices that contains contact details.

2. **Attack**
   The worm contains of multiple modules and uses zero day vulnerabilities and root kits to attack operating system. In particular, it protects its own memory pages from being accessed by other applications on the same host, such as AV. It also modifies its behavior when AV software is detected on the host.

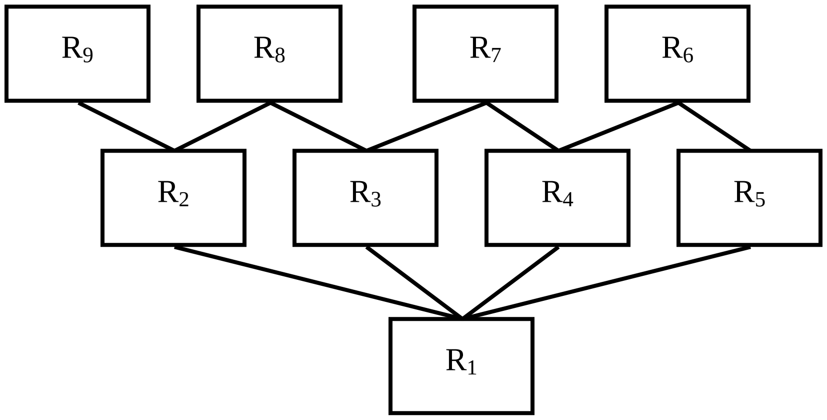
3. **Communication**
   This worm uses the Internet as primary channel for communication. It collects information on the host and waits for the commands from command center. It also uses Bluetooth to communicate to other devices and tries to read contact details from them.

4. **Command**
   As been stated in communication, this malware receives commands from command and control servers over the internet. These C&C servers are scattered around the globe that makes it harder to detect and disable them.

5. **Intelligence**
   It is not stated clearly in what way Flame manages detection of infected host. Since it has been reported that there are similarities in the code base of Stuxnet and Flame it might be that it also uses P2P type of connection to manage infected machines without central location. On the other hand, it awaits commands from C&C server, and does not do anything with the collected information. That suggest that usual client-server type of communication is used as well.

4. **Access control. For the following two policies, determine whether they are equivalent (i.e., users have same permissions in both policies). Explain your answer.**
   a. Hierarchical RBAC policy

Role hierarchy (RH):



Permission-to-role assignment (PA):

permssn / role	$\mathbf{O_1}$	$\mathbf{O_2}$	$\mathbf{O_3}$	$\mathbf{O_4}$
$\mathbf{R_1}$	W			
$\mathbf{R_2}$		W		R
$\mathbf{R_3}$			w	
$\mathbf{R_4}$				W
$\mathbf{R_5}$			r	
$\mathbf{R_6}$		R		
$\mathbf{R_7}$	R			
$\mathbf{R_8}$				
$\mathbf{R_9}$				

User-to-role assignment (UA):

user / role	$\mathbf{U_1}$	$\mathbf{U_2}$	$\mathbf{U_3}$	$\mathbf{U_4}$
$\mathbf{R_1}$				
$\mathbf{R_2}$	X	X		
$\mathbf{R_3}$			X	
$\mathbf{R_4}$				
$\mathbf{R_5}$				X
$\mathbf{R_6}$			X	
$\mathbf{R_7}$				X
$\mathbf{R_8}$		X		
$\mathbf{R_9}$	X			

b. Policy expressed through the following access matrix:

object / user	O_1	O_2	O_3	O_4
U_1	w	w	-----	r,w
U_2	w	w	W	R
U_3	w	r	r,w	W
U_4	r,w	-----	r,w	W

The above two policies are equivalent: ____Yes, __X__No
Because if you encode RH, UA and PA in table it will not be the same as shown in table b:

object / user	O_1	O_2	O_3	O_4
U_1	W	W	---	R (no W)
U_2	W	W	W	R
U_3	W	R	R,W	W
U_4	R,W		R,W	W

5. Which assignment(s) can be removed from the UA in the previous without any user loosing any permission(s)?

U1 – R2
U2 – R2

Appendix (You are welcome to detach this appendix and take it home with you)

Flame (malware)

source: http://en.wikipedia.org/wiki/Flame_(malware)

Flame, also known as Flamer, sKyWIper,[b] and Skywiper is modular computer malware discovered in 2012 that attacks computers running the Microsoft Windows operating system. The program is being used for targeted cyber espionage in Middle Eastern countries. Its discovery was announced on 28 May 2012 by MAHER Center of Iranian National Computer Emergency Response Team (CERT), Kaspersky Lab and CrySyS Lab of the Budapest University of Technology and Economics. The last of these stated in its report that it "is certainly the most sophisticated malware we encountered during our practice; arguably, it is the most complex malware ever found."

Flame can spread to other systems over a local network (LAN) or via USB stick. It can record audio, screenshots, keyboard activity and network traffic. The program also records Skype conversations and can turn infected computers into Bluetooth beacons which attempt to download contact information from nearby Bluetooth-enabled devices. This data, along with locally stored documents, is sent on to one of several command and control servers that are scattered around the world. The program then awaits further instructions from these servers. According to estimates by Kaspersky in May 2012, Flame had initially infected approximately 1,000 machines, with victims including governmental organizations, educational institutions and private individuals. At that time 65% of the infections happened in Iran, Israel, Sudan, Syria, Lebanon, Saudi Arabia, and Egypt, with a "huge majority of targets" within Iran. Flame has also been reported in Europe and North America. Flame supports a "kill" command which wipes all traces of the malware from the computer. The initial infections of Flame stopped operating after its public exposure, and the "kill" command was sent.

Flame is an uncharacteristically large program for malware at 20 megabytes. It is written partly in the Lua scripting language with compiled C++ code linked in, and allows other attack modules to be loaded after initial infection. The malware uses five different encryption methods and an SQLite database to store structured information. The method used to inject code into various processes is stealthy, in that the malware modules do not appear in a listing of the modules loaded into a process and malware memory pages are protected with READ, WRITE and EXECUTE permissions that make them inaccessible by user-mode applications. The internal code has few similarities with other malware, but exploits two of the same security vulnerabilties used previously by Stuxnet to infect systems. The malware determines what antivirus software is installed, then customises its own behaviour (for example, by changing the filename extensions it uses) to reduce the probability of detection by that software. Additional indicators of compromise include mutex and registry activity, such as installation of a fake audio driver which the malware uses to maintain persistence on the compromised system.

Flame is not designed to deactivate automatically, but supports a "kill" function that makes it eliminate all traces of its files and operation from a system on receipt of a module from its controllers. Flame was signed with a fraudulent certificate purportedly from the Microsoft Enforced Licensing Intermediate PCA certificate authority. The malware authors identified a Microsoft Terminal Server Licensing Service certificate that inadvertently was enabled for code signing and that still used the weak MD5 hashing algorithm, then produced a counterfeit copy of the certificate that they used to sign some components of the malware to make them appear to have originated from Microsoft. A successful collision attack against a certificate was previously demonstrated in 2008, but Flame implemented a new variation of the chosen-prefix collision attack

Like the previously known cyber weapons Stuxnet and Duqu, it is employed in a targeted manner and can evade current security software through rootkit functionality. Once a system is infected, Flame can spread to other systems over a local network or via USB stick. It can record audio, screenshots, keyboard activity and network traffic. The program also records Skype conversations and can turn infected computers into Bluetooth beacons which attempt to download contact information from nearby Bluetooth enabled devices. This data, along with locally stored documents, is sent on to one of several command and control servers that are scattered around the world. The program then awaits further instructions from these servers.

Unlike Stuxnet, which was designed to sabotage an industrial process, Flame appears to have been written purely for espionage. It does not appear to target a particular industry, but rather is "a complete attack toolkit designed for general cyber-espionage purposes". Using a technique known as sinkholing, Kaspersky demonstrated that "a huge majority of targets" were within Iran, with the attackers particularly seeking AutoCAD drawings, PDFs, and text files. Computing experts said that the program appeared to be gathering technical diagrams for intelligence purposes. A network of 80 servers across Asia, Europe and North America has been used to access the infected machines remotely.

On June 19, 2012, The Washington Post published an article claiming that Flame was jointly developed by the U.S. National Security Agency, CIA and Israel’s military at least five years prior. The project was said to be part of a classified effort code-named Olympic Games, which was intended to collect intelligence in preparation for a cyber-sabotage campaign aimed at slowing Iranian nuclear efforts. According to Kaspersky's chief malware expert, "the geography of the targets and also the complexity of the threat leaves no doubt about it being a nation-state that sponsored the research that went into it." Kaspersky initially said that the malware bears no resemblance to Stuxnet, although it may have been a parallel project commissioned by the same attackers. After analysing the code further, Kaspersky later said that there is a strong relationship between Flame and Stuxnet; the early version of Stuxnet contained code to propagate via USB drives that is nearly identical to a Flame module that exploits the same zero-day vulnerability.

Iran's CERT described the malware's encryption as having "a special pattern which you only see coming from Israel". The Daily Telegraph reported that due to Flame's apparent targets—which included Iran, Syria, and the West Bank—Israel became "many commentators' prime suspect". Other commentators named China and the U.S. as possible perpetrators. Richard Silverstein, a commentator critical of Israeli policies, stated that he had confirmed with a "senior Israeli source" that the malware was created by Israeli computer experts. The Jerusalem Post wrote that Israel's Vice Prime Minister Moshe Ya'alon appeared to have hinted that his government was responsible, but an Israeli spokesperson later denied that this had been implied. Unnamed Israeli security officials suggested that the infected machines found in Israel may imply that the virus could be traced to the U.S. or other Western nations. The U.S. has officially denied responsibility.