**Handouts of Lecture 21 Professional Practices (IT)**

**Lecture Title: Computer and Network Security**

**Hackers, Past and Present**

Data mining is put to a wide variety of uses in modern society. Here are a few examples.

In its original meaning, a hacker was an explorer, a risk taker, someone who was trying to make a system do something it had never done before. Hackers in this sense of the word abounded at MIT’s Tech Model Railroad Club in the 1950s and 1960s. The club constructed and continuously improved an enormous HO-scale model train layout. Members of the Signals and Power Subcommittee built an elaborate electronic switching system to control the movement of the trains. Wearing chino pants, short-sleeved shirts, and pocket protectors, the most dedicated members would drink vast quantities of Coca-Cola and stay up all night to improve the system. To them, a “hack” was a newly constructed piece of equipment that not only served a useful purpose but also demonstrated its creator’s technical virtuosity. Calling someone a hacker was a sign of respect; hackers wore the label with pride.

In 1959, after taking a newly created course in computer programming, some of the hackers shifted their attention from model trains to electronic computers. The term “hacker” came to mean a “person who delights in having an intimate understanding of the internal workings of a system, computers and networks in particular”. In the 1983 movie WarGames, a teenager breaks into a military computer and nearly causes a nuclear Armageddon. After seeing the movie, a lot of teenagers were excited at the thought that they could prowl cyberspace with a home computer and a modem. A few of them became highly proficient at breaking into government and corporate computer networks. These actions helped change the everyday meaning of the word “hacker.” Today hackers are people who gain unauthorized access to computers and computer networks. An example of this use of the word is a story in Computerworld describing how hackers broke into the Web site of USA Today on July 11, 2002, and inserted fabricated news stories.

Typically, you need a login name and password to access a computer system. Sometimes a hacker can guess a valid login name/password combination, particularly when system administrators allow users to choose short passwords or passwords that appear in a dictionary.

Three other low-tech techniques for obtaining login names and passwords are ***eavesdropping, dumpster diving, and social engineering***.

**Eavesdropping**, such as simply looking over the shoulder of a legitimate computer user to learn his login name and password, is a common way that hackers gain access to computers.

**Dumpster diving** means looking through garbage for interesting bits of information. Companies typically do not put a fence around their dumpsters. In midnight rummaging sessions, hackers have found user manuals, phone numbers, login names, and passwords.

**Social engineering** refers to the manipulation of a person inside the organization to gain access to confidential information. Social engineering is easier in large organizations where people do not know each other very well. For example, a hacker may identify a system administrator and call that person, pretending to be the supervisor of his supervisor and demanding to know why he can’t access a particular machine. In this situation, a cowed system administrator, eager to please his boss’s boss, may be talked into revealing or resetting a password.

**List of password dos and don’ts from security experts**.

Do not use short passwords. Modern computers can quickly crack short passwords. As a general rule, the longer a password is, the less likely it is to be guessed. .

Do not use a word from the dictionary. Again, such a password is too easy to crack.

Do not rely on substituting numbers for letters (e.g., replacing “E” with “3”). Password cracking programs know these tricks.

Do not reuse passwords. If accounts share passwords, as soon as one account is compromised, the other ones are, too. If you must write down your passwords on a piece of paper in order to remember them, that is safer than reusing passwords in today’s environment where an online attack is a greater danger than someone rummaging through your desk. .

Give ridiculous answers to security questions. That way they serve as a secondary password. Example: What is your pet’s name? Ford Fiesta.

Enable two-factor authentication if available. When you log in from an unfamiliar computer, the system will send you a text message with a confirmation code.

Have password recoveries sent to a secure email address. You don’t want hackers to know where your password reset messages are sent. Have these messages sent to an account you never use to send email.

**Computer Fraud and Abuse Act**

Under US law, the maximum penalties for hacking are severe. The Computer Fraud and Abuse Act criminalizes a wide variety of hacker-related activities, including

Transmitting code (such as a virus or worm) that causes damage to a computer system.

Accessing without authorization any computer connected to the Internet, even if no files are examined, changed, or copied.

Transmitting classified government information.

Trafficking in computer passwords.

Computer fraud.

Computer extortion

The maximum penalty imposed for violating the Computer Fraud and Abuse Act is 20 years in prison and a $250,000 fine.

**Sidejacking**

It is the hijacking of an open Web session by the capturing of a user’s cookie, giving the attacker the same privileges as the user on that Web site. Ecommerce Web sites typically use encryption to protect the username and password people provide when logging in, but they do not encrypt the cookie that the Web browser sends to the user to continue the session. Sidejacking is possible on unencrypted wireless networks because another device on the wireless network can “hear” the cookie being transmitted from the Web site back to the user’s computer. Even though the Internet security community had known and complained about the sidejacking vulnerability for years, ecommerce Web sites did not change their practices.

**Case Study: Firesheep**

On October 24, 2010, Eric Butler released an extension to the Firefox browser called Firesheep. Firesheep makes it easy for a Firefox user to sidejack open Web sessions. The user starts the Firefox browser, connects to an open Wi-Fi network, and clicks on a button called “Start Capturing.” When someone using the network visits an insecure Web site that Firesheep knows about, the user’s name and photo are displayed in a sidebar, along with the name of the Web site he is connected to, such as Amazon, Facebook, or Twitter. By double-clicking on the photo, the attacker becomes logged in as that user on that Web site and is able to do the same things that the legitimate user is able to do, such as post status messages and purchase products.

Butler released Firesheep as free, open-source software for Mac OS X and Windows. He wrote: “Websites have a responsibility to protect the people who depend on their services. They’ve been ignoring this responsibility for too long, and it’s time for everyone to demand a more secure web. My hope is that Firesheep will help the users win”. The Firesheep extension was downloaded more than 500,000 times in its first week of availability, and it attracted a great deal of media attention [12]. The typical story warned social network users about the dangers of using unencrypted wireless public networks and criticized the social network companies for not providing more security.

**Act utilitarian analysis**

The release of Firesheep led the media to focus on the risks associated with the use of certain Web sites from unsecured wireless networks, and a few months later Facebook and Twitter made their Web sites more secure. There continues to be strong pressure for other Web services to follow suit. These are tremendous benefits for everyone who accesses the Web at a public Internet access point without encryption. Butler was right when he predicted that Firesheep would not turn people into criminals. Even though half a million people downloaded Firesheep in the first week, there was no evidence of a big increase in identity theft or even malicious pranks. The harms caused by Firesheep appeared to be minimal. Because the release of Firesheep caused great benefits and negligible harm, we conclude it was a good action from a utilitarian point of view.

**Virtue ethics analysis**

Butler demonstrated civic responsibility by using his technical skills to develop Firesheep, a piece of software that dramatically illustrated, even to nontechnical people, the lack of security when unencrypted HTTP messages are sent over an unencrypted Wi-Fi network. Butler’s stated purpose for releasing Firesheep was “to demonstrate just how serious this problem is”. Later he wrote, “It goes without saying that harassing or attacking people is a terrible thing to do. To suggest Firesheep was created for this purpose is completely false; Firesheep was created to raise awareness about an existing and frequently ignored problem”. All of these statements are characteristic of someone truly interested in protecting the privacy of visitors to popular Web sites. Butler demonstrated courage by taking personal responsibility for creating Firesheep, and he demonstrated benevolence by making it freely available. Therefore, from the perspect of virtue ethics, Butler’s actions and statements were characteristic of someone interested in promoting the common good. He seemed to sincerely believe that something significant needed to be done to get the companies to change their privacy policies.

**Kantian analysis** To begin with, accessing someone else’s user account is an invasion of that person’s privacy and is wrong. Butler clearly agrees with this perspective because he refers to people who sidejack accounts as “evil.” Butler’s goal was to pressure Facebook, Twitter, Amazon, and other Web sites to adopt proper security measures to protect their users. He saw the best way to achieve this end was to release a tool that would bring to light a well-known security problem that had not gotten sufficient attention. Criminals already knew how to sidejack Web sessions before Butler created Firesheep. What Firesheep did was make sidejacking so simple that even ordinary computer users could do it. More than half a million copies of Firesheep were downloaded in the first week, and undoubtedly some of these people actually used the software to sidejack Web sessions, which is wrong. It is disingenuous for Butler to “reject the notion that something like Firesheep turns otherwise innocent people evil.” He provided a tool that made it much simpler for people to do something that is wrong, and therefore he has some moral accountability for the misdeeds of the people who downloaded Firesheep. Ultimately, Butler was willing to tolerate a short-term increase in privacy violations in the hope that users would pressure Facebook, Twitter, and other sites to improve their security, which would result in fewer privacy violations in the long term. In other words, he was willing to use the victims of Firesheep as a means to his end. From a Kantian perspective, it was wrong for Butler to release Firesheep to the public. There are other ways Butler could have achieved his goal without using other people. For example, he could have gone on a popular television show and hacked into the host’s Facebook page, generating a great amount of publicity without having to release the software.

**Malware**

The Firesheep extension to the Firefox browser highlights a significant security weakness of unencrypted Wi-Fi networks. Computers have security weaknesses, too, and there are a variety of ways in which malicious software, or malware, can become active on your computer. If you are lucky, these programs will do nothing other than consume a little CPU time and some disk space. If you are not so lucky, they may destroy valuable data stored in your computer’s file system. An invading program may even allow outsiders to seize control of your computer. Once this happens, they may use your computer as a depository for stolen credit card information, a Web server dishing out pornographic images, or a launch pad for spam or a denial-of-service attack on a corporate or government server.

**Viruses**

Viruses represent one way in which malicious code can get into a computer. A virus is a piece of self-replicating code embedded within another program called the host.



When a user executes a host program infected with a virus, the virus code executes first. The virus finds another executable program stored in the computer’s file system and replaces the program with a virus-infected program. After doing this, the virus allows the host program to execute, which is what the user expected to happen. If the virus does its work quickly enough, the user may be unaware of the presence of the virus. Because a virus is attached to a host program, you may find viruses anywhere you can find program files: hard disks, thumb drives, CD-ROMs, email attachments, and so on. Viruses can be spread from machine to machine via thumb drives or CDs. They may also be passed when a person downloads a file from the Internet. Sometimes viruses are attached to free computer games that people download and install on their computers. Today many viruses are spread via email attachments. We are all familiar with ordinary attachments such as photos, but attachments may also be executable programs or word processing documents or spreadsheets containing macros, which are small pieces of executable code. If the user opens an attachment containing a virus, the virus takes control of the computer, reads the user’s email address book, and uses these addresses to send virus-contaminated emails to others

**Antivirus Software Packages**

Commercial antivirus software packages allow computer users to detect and destroy viruses lurking on their computers. To be most effective, users must keep them upto-date by downloading patterns corresponding to the latest viruses from the vendor’s Web site. Unfortunately, many people are negligent about keeping their virus protection software up-to-date. According to the statistics office of the European Union, a survey of Internet users revealed that 31 percent of them had experienced a computer virus in the previous 12 months that had resulted in a loss of information or time, even though 84 percent of them said that their computer was running antivirus software. That means they were not keeping their virus protection current.

**Worm**

A worm is a self-contained program that spreads through a computer network by exploiting security holes in the computers connected to the network. The technical term “worm” comes from The Shockwave Rider, a 1975 science fiction novel written by John Brunner. The most famous worm of all time was also the first one to get the attention of the mainstream media, which is why it is popularly known as the Internet worm, even though many other worms have been created that propagate through the Internet.

A worm spreads to other computers by exploiting security holes in computer networks.



**Designing the worm**

Morris entered the graduate program in computer science at Cornell University in the fall of 1988. He became intrigued with the idea of creating a computer worm that would exploit bugs he had found in three Unix applications: ftp, sendmail, and fingerd. His “wish list” for the worm had about two dozen goals, including the following:

Infect three machines per local area network

Only consume CPU cycles if the machines are idle

Avoid slow machines

Break passwords in order to spread to other computers

The goal of the worm was to infect as many computers as possible. It would not destroy or corrupt data files on the machines it infected.

**Launching the worm**

On November 2, 1988, Morris learned that a fix for the ftp bug had been posted to the Internet, meaning his worm program could no longer take advantage of that security hole. However, nobody had posted fixes to the other two bugs Morris knew about. After making some last-minute changes to the worm program, he logged in to a computer at the MIT Artificial Intelligence Lab and launched the worm at about 7:30 p.m. The worm quickly spread to thousands of computers at military installations, medical research facilities, and universities. Unfortunately, due to several bugs in the worm’s programming, computers became infected with hundreds of copies of the worm, causing them to crash every few minutes or become practically unresponsive to the programs of legitimate users.

Morris contacted friends at Harvard to discuss what ought to be done next. They agreed that Andy Sudduth would anonymously post a message to the Internet. Sudduth’s message is shown here.1 Harvard’s computers were not affected (the security holes had already been patched), and you can tell from the last sentence that Sudduth was having a hard time believing Morris’s story:

A Possible virus report:

There may be a virus loose on the internet.

Here is the gist of a message I got:

I’m sorry.

Here are some steps to prevent further transmission:

1. don’t run finger, or fix it to not overrun its stack when reading arguments. 2) recompile sendmail w/o DEBUG defined 3) don’t run rexed Hope this helps, but more, I hope it is a hoax.

Sudduth’s email was supposed to get routed through a computer at Brown University. However, computers at Brown were already infected with the worm and did not have spare cycles to route the message. Also, the email did not have a subject line, which made it less likely to be read during a crisis. The result is that the message was read too late to be of any help to those fighting the worm. System administrators at various universities worked frantically to stop the spread of the worm.

Within a day they had examined the worm’s code, discovered the bugs in sendmail and fingerd, and published fixes to the Internet community. No one knows exactly how many computers were infected by the worm, but it did make a significant number of systems unusable for a day or two.

After some sleuthing by reporter John Markoff, the New York Times named Robert Tappan Morris Jr. as the author of the worm. Morris was suspended from Cornell University. A year later, he was the first person to receive a felony conviction under the US Computer Fraud and Abuse Act. He was sentenced to three years’ probation and 400 hours of community service, and was fined $10,000. His legal fees and fines exceeded $150,000.

**Ethical evaluation**

Was Robert Morris Jr. wrong to unleash the Internet worm?

A **Kantian evaluation** must focus on Morris’s will. Did Morris have good will? His stated goal was to see how many Internet computers he could infect with the worm. While Morris did not want to crash these computers or destroy any data stored on them, his motivation was fundamentally selfish: he wanted the thrill of seeing his creation running on thousands of computers. He used others because he gained access to their machines without their permission. There is also evidence Morris knew he was using others: he took measures designed to prevent people from discovering that he was the author of the worm. From a Kantian point of view, Morris’s action was wrong.

From a **social contract point of view**, Morris’s action was also wrong. He violated the property rights of the individuals and organizations whose computers were infected by the worm. They had the right to determine who would use their computers, and they attempted to enforce this right by requiring people to identify themselves by username and password. Morris took advantage of security holes in these computers to gain unauthorized access to them. When his worm caused these computers to become unresponsive or crash, he denied access to the legitimate users of these computers.

**A utilitarian evaluation** of the case focuses on the benefits and harms resulting from the spread of the worm. The principal benefit of the Internet worm was that organizations managing these Unix computers discovered there were two significant security holes in their systems. They received the instructions they needed to patch these holes before a truly malicious intruder took advantage of them to enter their systems and do a lot of damage to their data. Of course, Morris could have produced the same beneficial result simply by contacting the system administrators at UC Berkeley and informing them of the security holes he had found.

The Internet worm had numerous harmful consequences. A large amount of time was spent by system administrators as they defended their machines from further attacks, tracked down the problem, installed patches, and brought machines back on line. There was a disruption in email and file exchange traffic caused by computers being taken off the network. About 6,000 computers were unavailable for a day or two. During this time, many thousands of people were less productive than they could have been had the systems been up and running. Morris himself was harmed by his actions. He was suspended from Cornell and convicted of a felony, which resulted in a sentence of probation, community service, and a substantial fine.

Considering all of Morris’s options, it is clear that another course of action—simply alerting the Unix community to the bugs—would have produced all of the benefits with none of the harms. Therefore, from a utilitarian viewpoint, Morris was wrong to have released the Internet worm.

**From the perspective of virtue ethics,** Morris’s actions are not consistent with those of a virtuous person. He selfishly chose to use the Internet as an experimental laboratory, and he deceitfully released the worm from MIT rather than Cornell University. When the worm began spreading uncontrollably, he avoided taking responsibility for his actions by asking a trusted friend to post the message to the Internet explaining how to fight the worm.

In conclusion, Morris may not have been acting maliciously, but he was acting selfishly. If he had wanted to experiment with worms, he probably could have gotten permission to try out his creations on a local area network detached from the Internet, so that even if his worm multiplied out of control, there would have been no fallout to the rest of the computer community. Instead, he chose to use the entire Internet as his experimental laboratory, inconveniencing thousands of people.

***Reference***

***Lecture 21 slides: Computer and network security***

***Gao, Y. (2012). Ethics for the Information Age by Michael J. Quinn. World Libraries, 20(1).***