# NIT2102
# Cyber Security Essentials
# Session 3: Application and Networking-Based Attacks

Acknowledgment: Cengage's Instructor Materials

**Prepared By: Dr. Khandakar Ahmed**

CELEBRATING A CENTENARY OF OPPORTUNITY 2016

VICTORIA UNIVERSITY

VICTORIA UNIVERSITY
MELBOURNE AUSTRALIA

# Copyright

# Copyright Acknowledgement

**Acknowledgement of Copyright**

The following slides are subject to copyright and are provided with the permission of the Publisher.

The *Copyright Act 1968* (The Act) applies.

# Objectives

- List and explain the different types of server-side web applications attacks
- Define client-side attacks
- Explain how overflow attacks work
- List different types of networking-based attacks

**Instructor will spend 2 – 2.30 hours for this workshop slides leaving 1.30 ~ 2.00 hours for lab including submission

# Application Attacks

- Attacks on the applications in a networked computer system can be directed toward the server, the client, or both

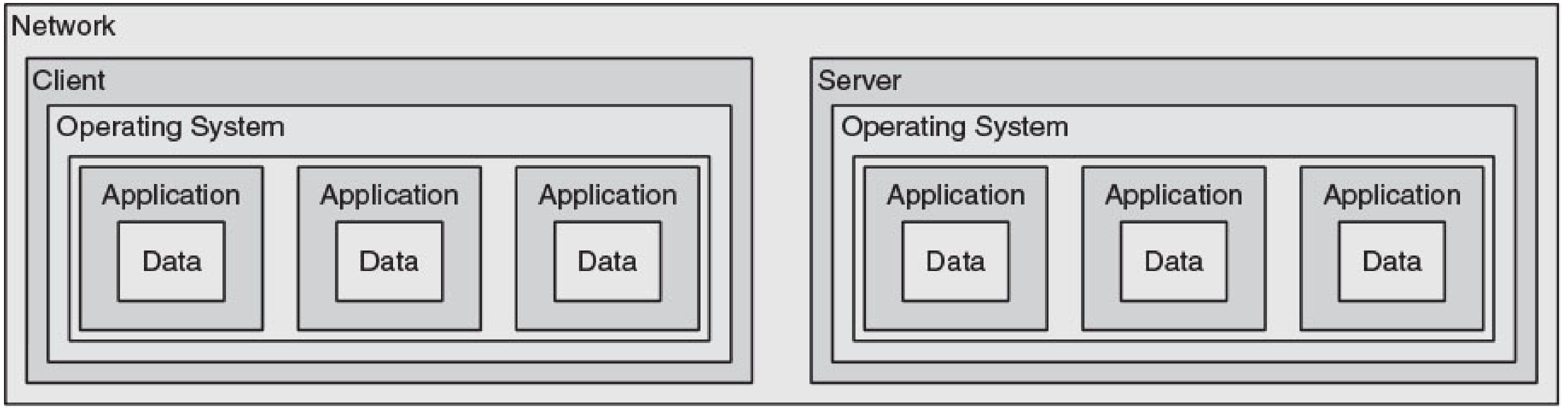


**Figure 3-1** Conceptual networked computer system

# Server-Side Web Application Attacks

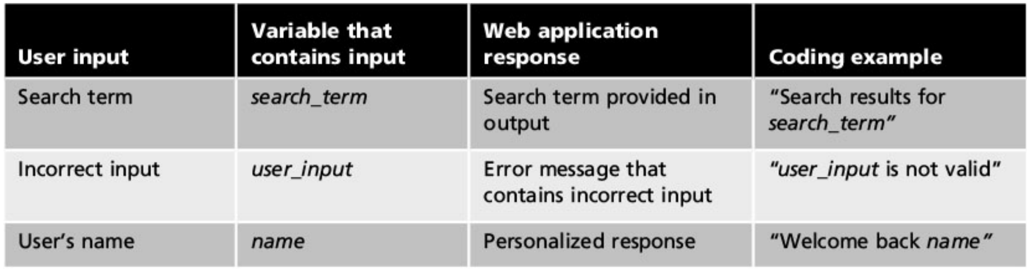
- Securing server-side web applications often considered more difficult than protecting other systems
- Traditional network security devices can block traditional network attacks, but cannot always block web application attacks
  - Many network security devices ignore the content of HTTP traffic
- **Zero-day attack** - an attack that exploits previously unknown vulnerabilities, victims have not time to prepare for or defend against the attack

# Server-Side Web Application Attacks

- Many server-side web application attacks target the input that the applications accept from users
- Such common web application attacks are:
  - Cross-site scripting
  - SQL injection
  - XML injection
  - Command injection/directory traversal [Out of the scope of this session, students can study on their own]

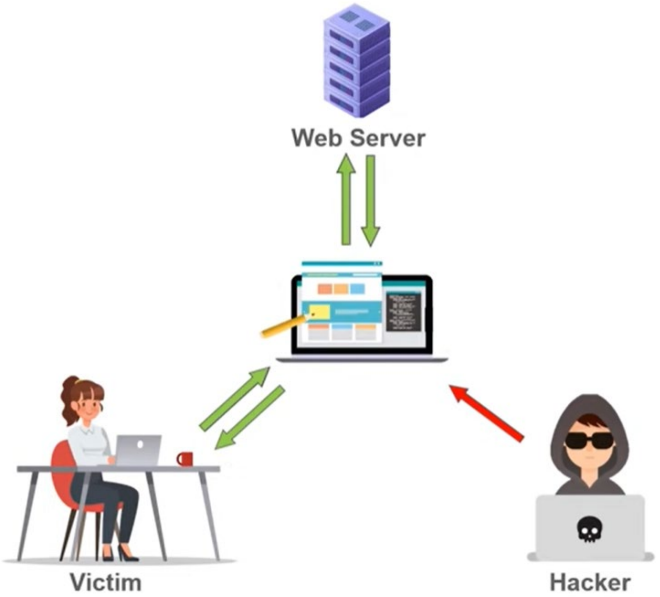
# Cross-Site Scripting (XSS)

- Injecting scripts into a Web application server to direct attacks at unsuspecting clients



| User input | Variable that contains input | Web application response | Coding example |
|---|---|---|---|
| Search term | *search_term* | Search term provided in output | "Search results for *search_term*" |
| Incorrect input | *user_input* | Error message that contains incorrect input | "*user_input* is not valid" |
| User's name | *name* | Personalized response | "Welcome back *name*" |

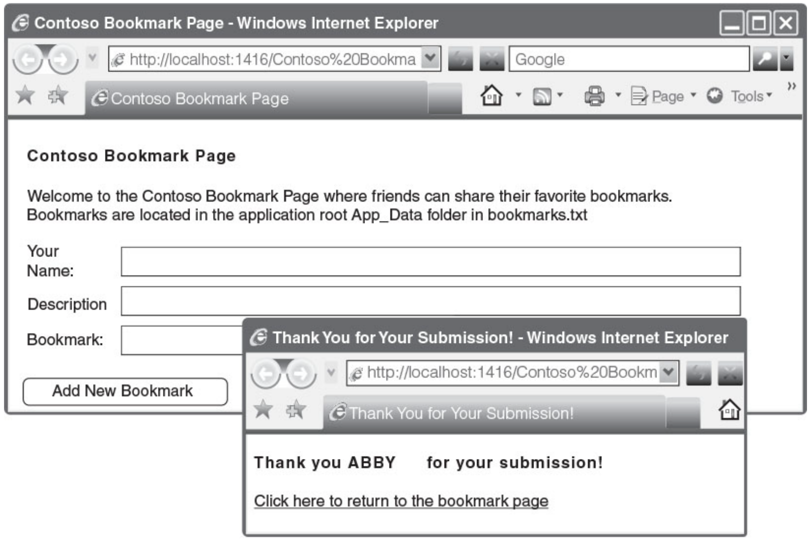
**Table 3-1** **Customized responses**



# Cross-Site Scripting (XSS)

- When victim visits injected Web site:
  - Malicious instructions are sent to victim’s browser
- Some XSS attacks are designed to steal information:
  - Retained by the browser when visiting specific sites
- An XSS attack requires a website meets two criteria:
  - Accepts user input without validating it
  - Uses input in a response

# Cross-Site Scripting (XSS)



Contoso Bookmark Page - Windows Internet Explorer

http://localhost:1416/Contoso%20Bookma

Contoso Bookmark Page

Welcome to the Contoso Bookmark Page where friends can share their favorite bookmarks.
Bookmarks are located in the application root App_Data folder in bookmarks.txt

Your Name:

Description

Bookmark:

Thank You for Your Submission! - Windows Internet Explorer

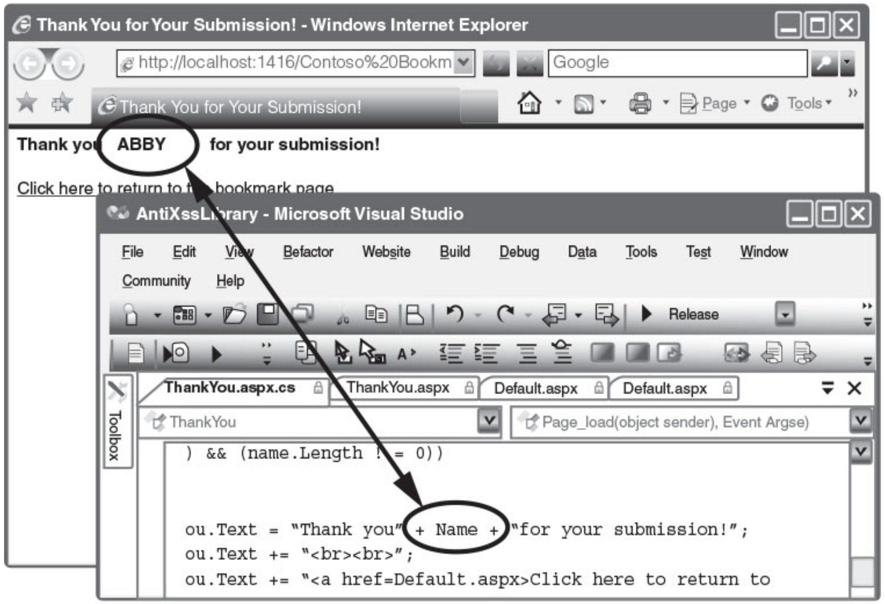
http://localhost:1416/Contoso%20Bookm

Thank you ABBY for your submission!

Click here to return to the bookmark page

**Figure 3-3** Bookmark page that accepts user input

*Source: Microsoft Inc.*



Thank You for Your Submission! - Windows Internet Explorer

http://localhost:1416/Contoso%20Bookm

Thank you ABBY for your submission!

Click here to return to the bookmark page

AntiXssLibrary - Microsoft Visual Studio

ThankYou.aspx.cs | ThankYou.aspx | Default.aspx | Default.aspx

ThankYou | Page_load(object sender), Event Argse)

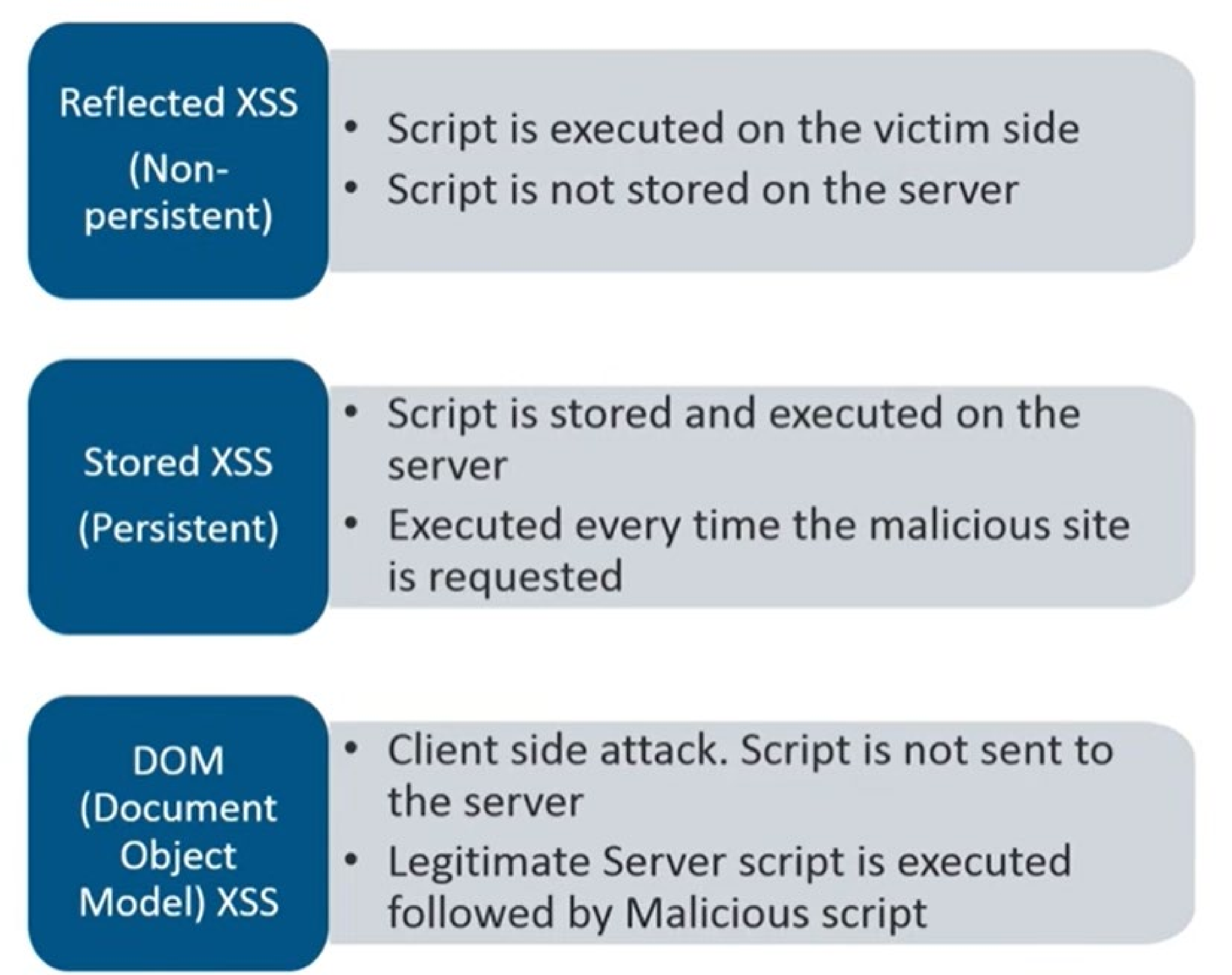
```
) && (name.Length != 0))


ou.Text = "Thank you" + Name + "for your submission!";
ou.Text += "<br><br>";
ou.Text += "<a href=Default.aspx>Click here to return to
```

**Figure 3-4** Input used in response

*Source: Microsoft Inc.*

# Type of Cross-Site Scripting (XSS)



**Reflected XSS (Non-persistent)**

- Script is executed on the victim side
- Script is not stored on the server

**Stored XSS (Persistent)**

- Script is stored and executed on the server
- Executed every time the malicious site is requested

**DOM (Document Object Model) XSS**

- Client side attack. Script is not sent to the server
- Legitimate Server script is executed followed by Malicious script

# SQL Injection

- Targets SQL servers by injecting malicious commands into them
- SQL (Structured Query Language)
  - Used to manipulate data stored in relational database
- Forgotten password example:
  - Attacker enters incorrectly formatted e-mail address
  - Response lets attacker know whether input is being validated

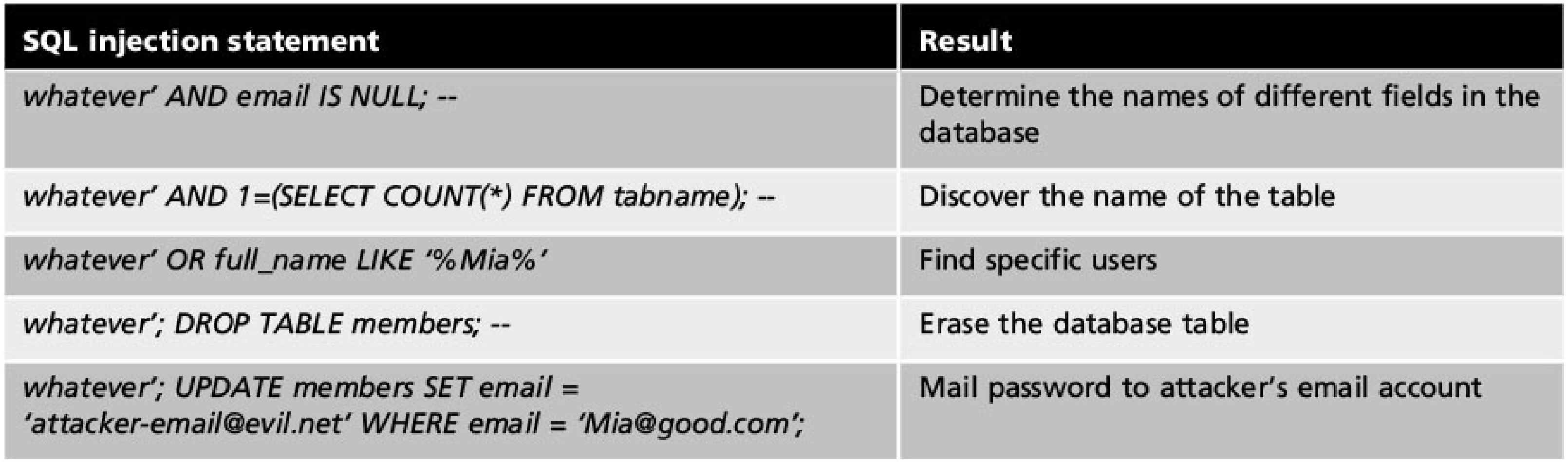
# SQL Injection

- Forgotten password example (cont’d.):
  - Attacker enters email field in SQL statement
  - Statement is processed by the database
  - Example statement:

```
SELECT fieldlist FROM table WHERE
field = ‘whatever’ or ‘a’=‘a’
```

  - Result: All user email addresses will be displayed

# SQL Injection



| SQL injection statement | Result |
|---|---|
| *whatever’ AND email IS NULL; --* | Determine the names of different fields in the database |
| *whatever’ AND 1=(SELECT COUNT(*) FROM tabname); --* | Discover the name of the table |
| *whatever’ OR full_name LIKE ‘%Mia%’* | Find specific users |
| *whatever’; DROP TABLE members; --* | Erase the database table |
| *whatever’; UPDATE members SET email = ‘attacker-email@evil.net’ WHERE email = ‘Mia@good.com’;* | Mail password to attacker’s email account |

**Table 3-2** **SQL injection statements**

# XML Injection

- Markup language
    - Method for adding annotations to text
- HTML
    - Uses tags surrounded by brackets
    - Instructs browser to display text in specific format
- XML
    - Carries data instead of indicating how to display it
    - No predefined set of tags
        - Users define their own tags

# XML Injection

- **XML injection** attack
  - Similar to SQL injection attack
  - Attacker discovers a Web site that does not filter user data
  - Injects XML tags and data into the database
- XPath injection
  - Specific type of XML injection attack
  - Attempts to exploit XML Path Language queries that are built from user input

# In Class Group Activity 1 [10 minutes]

- **Group Activity** –
  - Instructor will divide student into 3-4 groups and will send them to breakout room.
  - List all top 10 Web Application attacks **[5 minutes]**
  - One student presents the group key discussion points to the class **[1 minute/Group]**
  - Instructor feedback

# Client-Side Application Attacks

- Web application attacks are server-side attacks
- Client-side attacks target vulnerabilities in client applications that interact with a compromised server or process malicious data
- The client initiates connection with the server, which could result in an attack

# Client-Side Attacks

- *Drive-by download*
  - Client computer is compromised simply by viewing a Web page
  - Attackers inject content into vulnerable Web server
    - Gain access to server’s operating system
  - Attackers craft a zero pixel Iframe (short for *inline frame*) to avoid visual detection
  - Embed an HTML document inside main document
  - Client’s browser downloads malicious script
  - Instructs computer to download malware

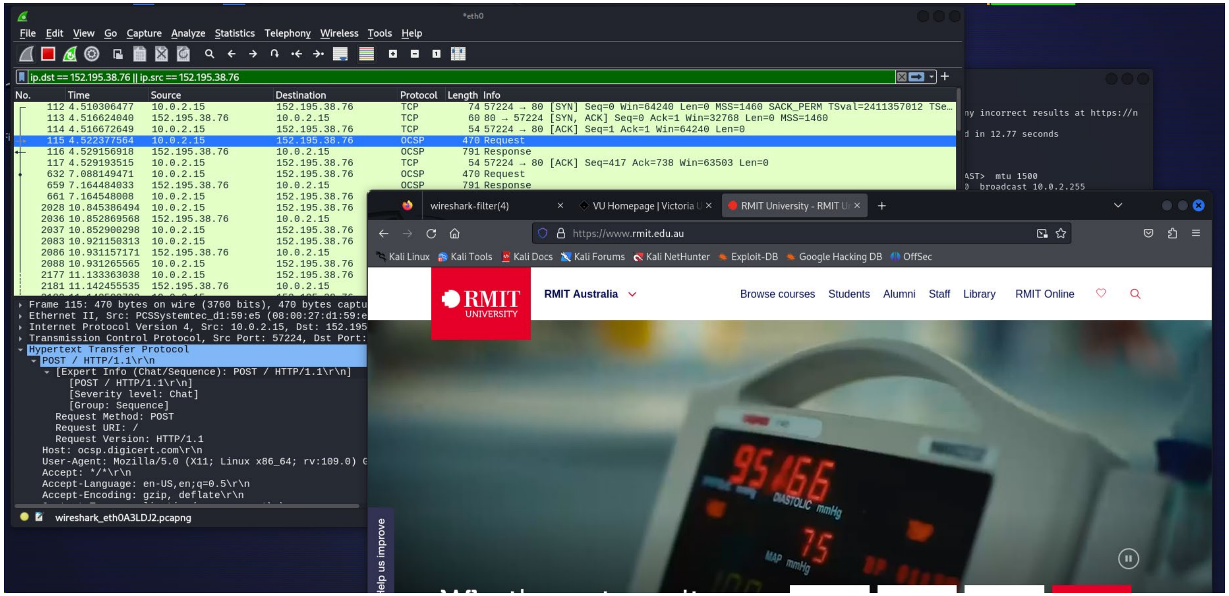
# Client-Side Attacks

- Header manipulation
  - **HTTP header** contains fields that characterize data being transmitted
  - Headers can originate from a Web browser
    - Browsers do not normally allow this
    - Attacker’s short program can allow modification
- Examples of **HTTP header manipulation**
  - Referrer
  - Accept-language
  - Response splitting

# Client-Side Attacks

- *Referer* field indicates the site that generated the Web page
  - Attacker can modify this field to hide the fact it came from another site
- *Accept-language* field contents may be passed directly to an SQL database
  - Attacker could inject SQL command by modifying this header
- *Response splitting* is one of the most common HTTP header manipulation attacks

# Client-Side Attacks



```
ip.dst == 152.195.38.76 || ip.src == 152.195.38.76

No.   Time          Source          Destination     Protocol  Length Info
  112 4.510306477   10.0.2.15       152.195.38.76   TCP         74 57224 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=2411357012 TSe…
  113 4.516624040   152.195.38.76   10.0.2.15       TCP         60 80 → 57224 [SYN, ACK] Seq=0 Ack=1 Win=32768 Len=0 MSS=1460
  114 4.516672649   10.0.2.15       152.195.38.76   TCP         54 57224 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
  115 4.522377564   10.0.2.15       152.195.38.76   OCSP       470 Request
  116 4.529156918   152.195.38.76   10.0.2.15       OCSP       791 Response
  117 4.529193515   10.0.2.15       152.195.38.76   TCP         54 57224 → 80 [ACK] Seq=417 Ack=738 Win=63503 Len=0
  632 7.088149471   10.0.2.15       152.195.38.76   OCSP       470 Request
  659 7.164484033   152.195.38.76   10.0.2.15       OCSP       791 Response
  661 7.164548008   10.0.2.15       152.195.38.76
 2028 10.845386494  10.0.2.15       152.195.38.76
 2036 10.852869568  152.195.38.76   10.0.2.15
 2037 10.852900298  10.0.2.15       152.195.38.76
 2083 10.921150313  10.0.2.15       152.195.38.76
 2086 10.931157171  152.195.38.76   10.0.2.15
 2088 10.931265565  10.0.2.15       152.195.38.76
 2177 11.133363038  10.0.2.15       152.195.38.76
 2181 11.142455535  152.195.38.76   10.0.2.15

› Frame 115: 470 bytes on wire (3760 bits), 470 bytes captu
› Ethernet II, Src: PCSSystemtec_d1:59:e5 (08:00:27:d1:59:e
› Internet Protocol Version 4, Src: 10.0.2.15, Dst: 152.195
› Transmission Control Protocol, Src Port: 57224, Dst Port:
⌄ Hypertext Transfer Protocol
  ⌄ POST / HTTP/1.1\r\n
    ⌄ [Expert Info (Chat/Sequence): POST / HTTP/1.1\r\n]
        [POST / HTTP/1.1\r\n]
        [Severity level: Chat]
        [Group: Sequence]
      Request Method: POST
      Request URI: /
      Request Version: HTTP/1.1
    Host: ocsp.digicert.com\r\n
    User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) G
    Accept: */*\r\n
    Accept-Language: en-US,en;q=0.5\r\n
    Accept-Encoding: gzip, deflate\r\n
```

```
ny incorrect results at https://n
d in 12.77 seconds
AST>  mtu 1500
0  broadcast 10.0.2.255
```

- Instructor will run Wireshark network protocol analyser and browse any website
- Filter the packet using source and destination IP filter and expand the HTTP request and HTTP response packet to investigate the HTTP header file.

# Client-Side Attacks

- Cookies
  - Cookies store user-specific information on user’s local computer
- Types of cookies:
  - **First-party cookie** - cookie created by Web site user is currently viewing
  - **Third-party cookie** - site advertisers place a cookie to record user preferences
  - **Session cookie** - stored in RAM and expires when browser is closed

# Client-Side Attacks

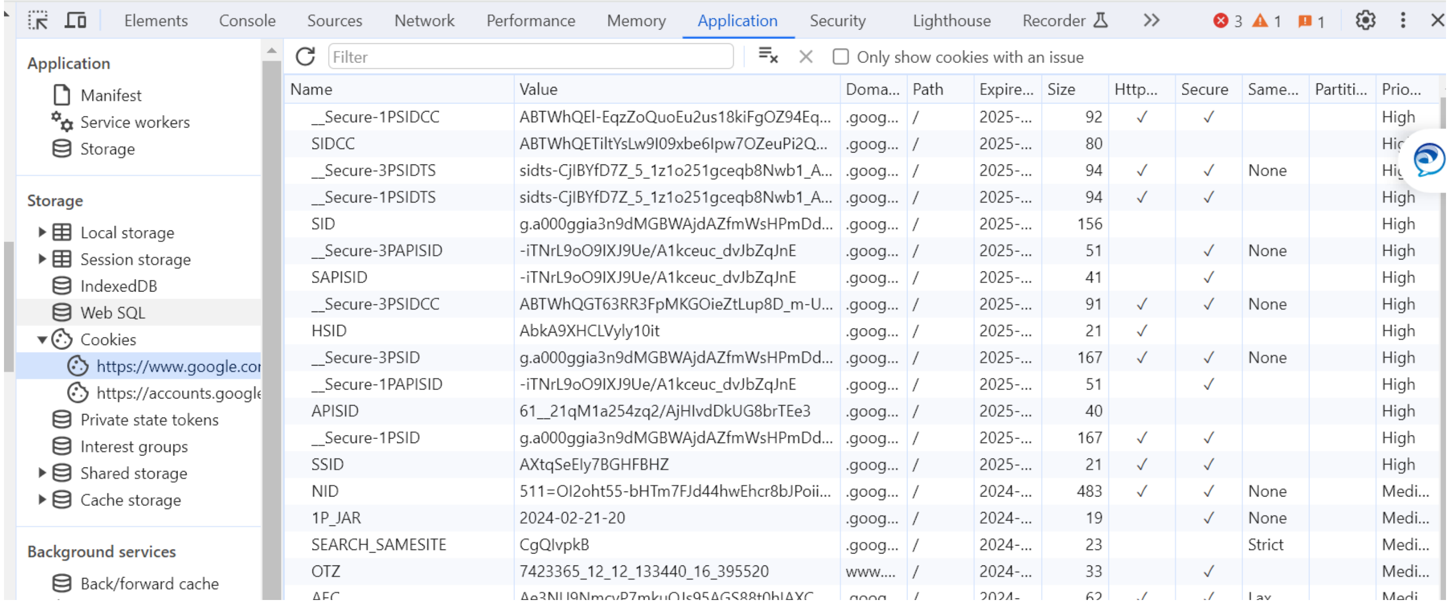
- Types of cookies (cont'd):
  - **Persistent cookie** - recorded on computer's hard drive and does not expire when the browser closes
    - Also called a tracking cookie
  - **Locally shared object (LSO)** - can store up to 100 KB of data form a website
    - More complex than the simple text found in a regular cookie
    - Also called a Flash cookie

CELEBRATING
A CENTENARY OF
OPPORTUNITY 2016
VICTORIA UNIVERSITY

# Client-Side Attacks

- Cookies pose security and privacy risks
  - First-party cookies may be stolen and used to impersonate the user
  - Used to tailor advertising
  - Can be exploited by attackers
- Attachments
  - Files that are coupled with email messages
  - Malicious attachments are commonly used to spread viruses, Trojans, and other malware

# Client-Side Attacks – Investigating Cookies



| Name | Value | Doma... | Path | Expire... | Size | Http... | Secure | Same... | Partiti... | Prio... |
|---|---|---|---|---|---|---|---|---|---|---|
| __Secure-1PSIDCC | ABTWhQEl-EqzZoQuoEu2us18kiFgOZ94Eq... | .goog... | / | 2025-... | 92 | ✓ | ✓ | | | High |
| SIDCC | ABTWhQETiltYsLw9I09xbe6Ipw7OZeuPi2Q... | .goog... | / | 2025-... | 80 | | | | | Hi |
| __Secure-3PSIDTS | sidts-CjIBYfD7Z_5_1z1o251gceqb8Nwb1_A... | .goog... | / | 2025-... | 94 | ✓ | ✓ | None | | Hi |
| __Secure-1PSIDTS | sidts-CjIBYfD7Z_5_1z1o251gceqb8Nwb1_A... | .goog... | / | 2025-... | 94 | ✓ | ✓ | | | High |
| SID | g.a000ggia3n9dMGBWAjdAZfmWsHPmDd... | .goog... | / | 2025-... | 156 | | | | | High |
| __Secure-3PAPISID | -iTNrL9oO9IXJ9Ue/A1kceuc_dvJbZqJnE | .goog... | / | 2025-... | 51 | | ✓ | None | | High |
| SAPISID | -iTNrL9oO9IXJ9Ue/A1kceuc_dvJbZqJnE | .goog... | / | 2025-... | 41 | | ✓ | | | High |
| __Secure-3PSIDCC | ABTWhQGT63RR3FpMKGOieZtLup8D_m-U... | .goog... | / | 2025-... | 91 | ✓ | ✓ | None | | High |
| HSID | AbkA9XHCLVyly10it | .goog... | / | 2025-... | 21 | ✓ | | | | High |
| __Secure-3PSID | g.a000ggia3n9dMGBWAjdAZfmWsHPmDd... | .goog... | / | 2025-... | 167 | ✓ | ✓ | None | | High |
| __Secure-1PAPISID | -iTNrL9oO9IXJ9Ue/A1kceuc_dvJbZqJnE | .goog... | / | 2025-... | 51 | | ✓ | | | High |
| APISID | 61__21qM1a254zq2/AjHlvdDkUG8brTEe3 | .goog... | / | 2025-... | 40 | | | | | High |
| __Secure-1PSID | g.a000ggia3n9dMGBWAjdAZfmWsHPmDd... | .goog... | / | 2025-... | 167 | ✓ | ✓ | | | High |
| SSID | AXtqSeEly7BGHFBHZ | .goog... | / | 2025-... | 21 | ✓ | ✓ | | | High |
| NID | 511=OI2oht55-bHTm7FJd44hwEhcr8bJPoii... | .goog... | / | 2024-... | 483 | ✓ | ✓ | None | | Medi... |
| 1P_JAR | 2024-02-21-20 | .goog... | / | 2024-... | 19 | | ✓ | None | | Medi... |
| SEARCH_SAMESITE | CgQIvpkB | .goog... | / | 2024-... | 23 | | | Strict | | Medi... |
| OTZ | 7423365_12_12_133440_16_395520 | www.... | / | 2024-... | 33 | | ✓ | | | Medi... |
| AEC | Ae3NU9NmcvP7mkuOIs95AGS88t0hIAXC | .goog... | / | 2024-... | 62 | ✓ | ✓ | Lax | | Medi... |

- Instructor will open any website
- Right-click and press inspect
- From the menu select ‘Application’
- From left under storage expand ‘Cookies’ and investigate different cookies that the site is storing

CELEBRATING A CENTENARY OF OPPORTUNITY 2016
VICTORIA UNIVERSITY

VICTORIA UNIVERSITY
MELBOURNE AUSTRALIA

# Client-Side Attacks

- Session Hijacking
  - Attacker attempts to impersonate user by stealing or guessing session token
  - Session token is a random string assigned to an interaction between user and web application
- An attacker can attempt to obtain the session token:
  - By using XSS or other attacks to steal the session token cookie from the victim’s computer
  - Eavesdropping on the transmission
  - Guessing the session token

# Client-Side Attacks

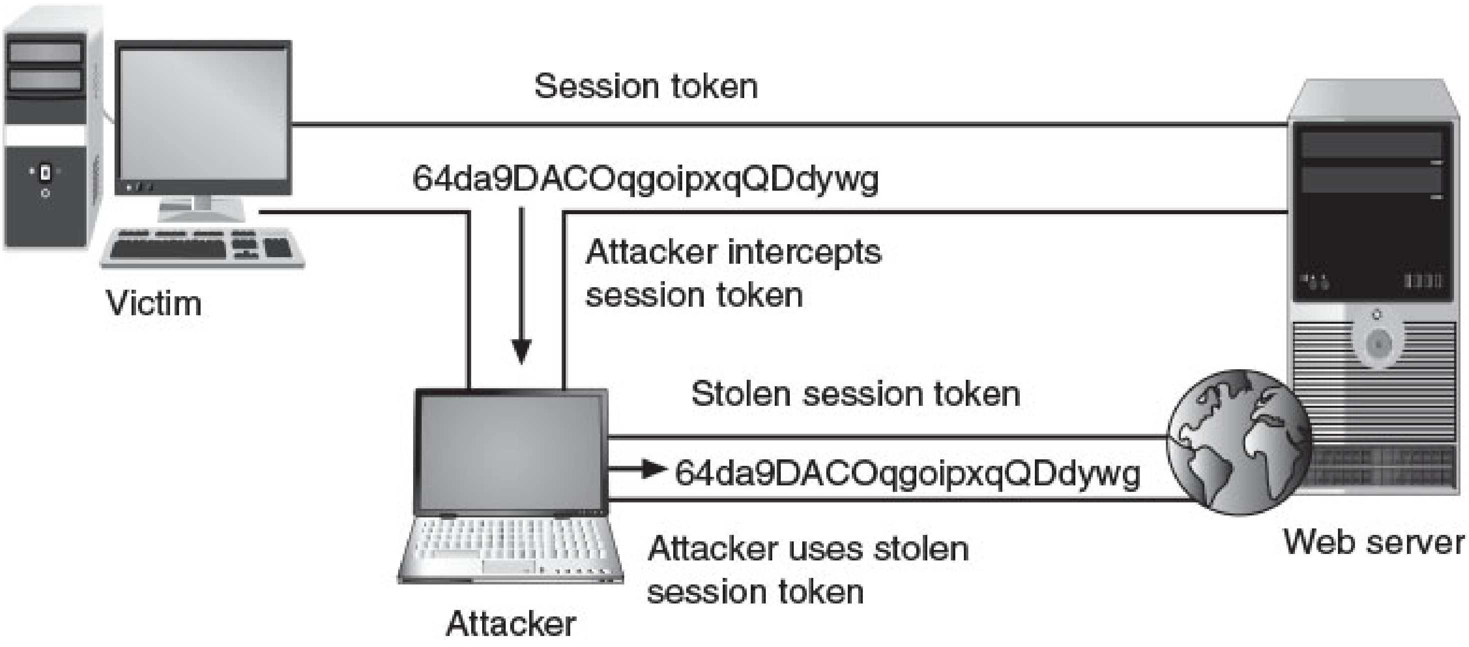


**Figure 3-7** Session hijacking attack

# Client-Side Attacks

- Malicious Add-ons
  - Plug-in - a third party library that attaches to a web browser and can be embedded inside a webpage
  - Add-ons or extensions - add functionality to the web browser
- Add-ons can do the following:
  - Create additional web browser toolbars
  - Change browser menus
  - Be aware of other tabs open in the same browser
  - Process the content of every webpage that is loaded

# Client-Side Attacks

- Security risks exist when using add-ons
  - Attackers can create malicious add-ons to launch attacks against the user's computer
- Malicious add-ons can be written by using Microsoft's **Active X**
  - ActiveX is a set of rules for how applications under the Microsoft Windows OS should share information
- Attackers can take advantage of vulnerabilities in ActiveX to perform malicious attacks on a computer

# Networking-Based Attacks

- Attackers place a high priority on targeting networks
  - Exploiting a single vulnerability may expose hundreds or thousands of devices to an attacker
- Types of networking-based attacks:
  - Denial of service
  - Interception
  - Poisoning
  - Attacks on access rights

# Denial of Service (DoS)

- Denial of service (DoS)
  - A deliberate attempt to prevent authorized users from accessing a system by overwhelming it with requests
- Most DoS attacks today are **distributed denial of service (DDoS)**
  - Using hundreds or thousands of zombie computers in a botnet to flood a device with requests

# Denial of Service (DoS)

- Ping flood attack
  - The ping utility is used to send large number of ICMP echo request messages
  - In a ping flood attack, multiple computers rapidly send a large number of ICMP echo requests to a server
    - Server will drop legitimate connections and refuse new connections

# Denial of Service (DoS)

- Smurf attack
  - Tricks devices into responding to false requests to an unsuspecting victim
  - An attacker broadcasts a ping request to all computers on the network but changes the address from which the request came from (called **spoofing**)
  - Appears as if victim's computer is asking for response from all computers on the network
  - All computers send a response to the victim's computer so that it is overwhelmed and crashes or becomes unavailable to legitimate users

# Denial of Service (DoS)

- SYN flood attack
  - Takes advantage of procedures for initiating a session
- In a SYN flood attack against a web server:
  - The attacker sends SYN segments in IP packets to the server
  - Attacker modifies the source address of each packet to computer addresses that do not exist or cannot be reached

# Denial of Service (DoS)

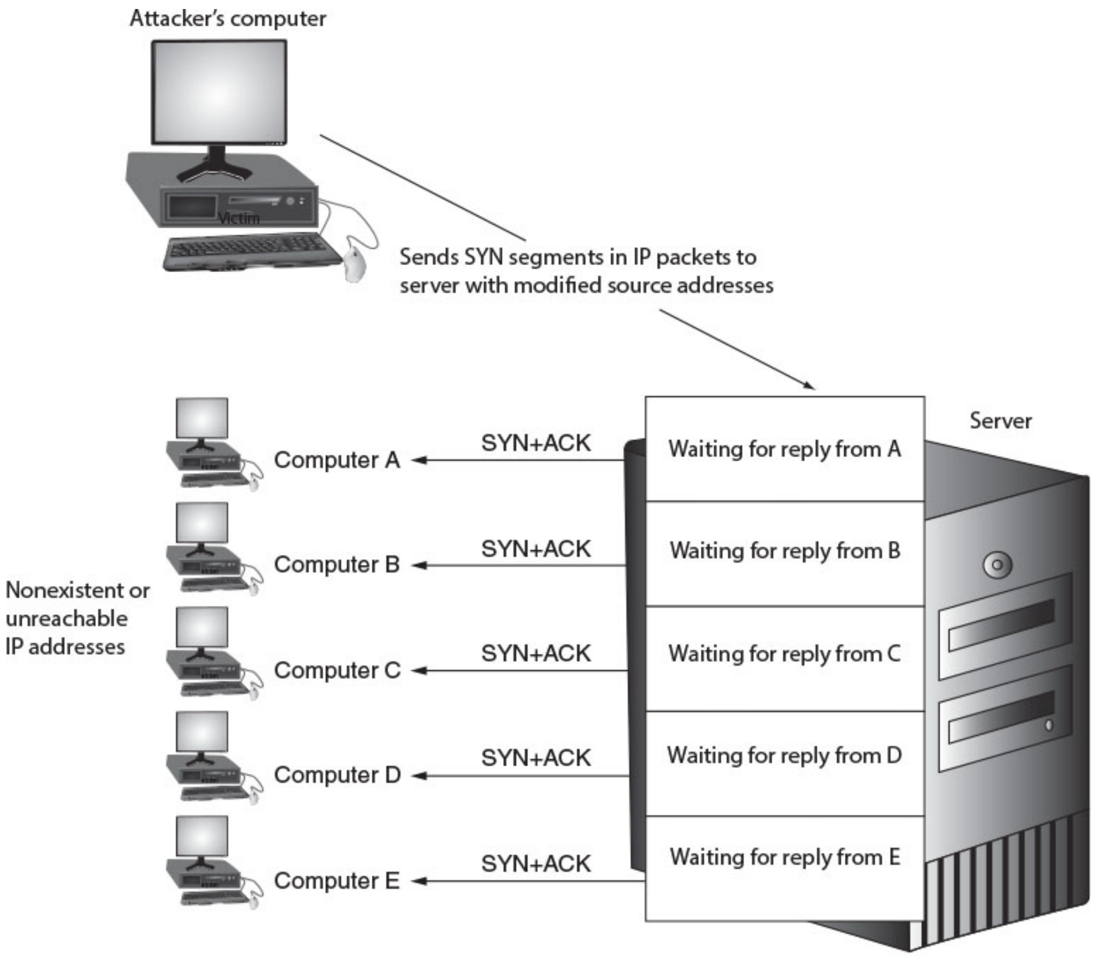
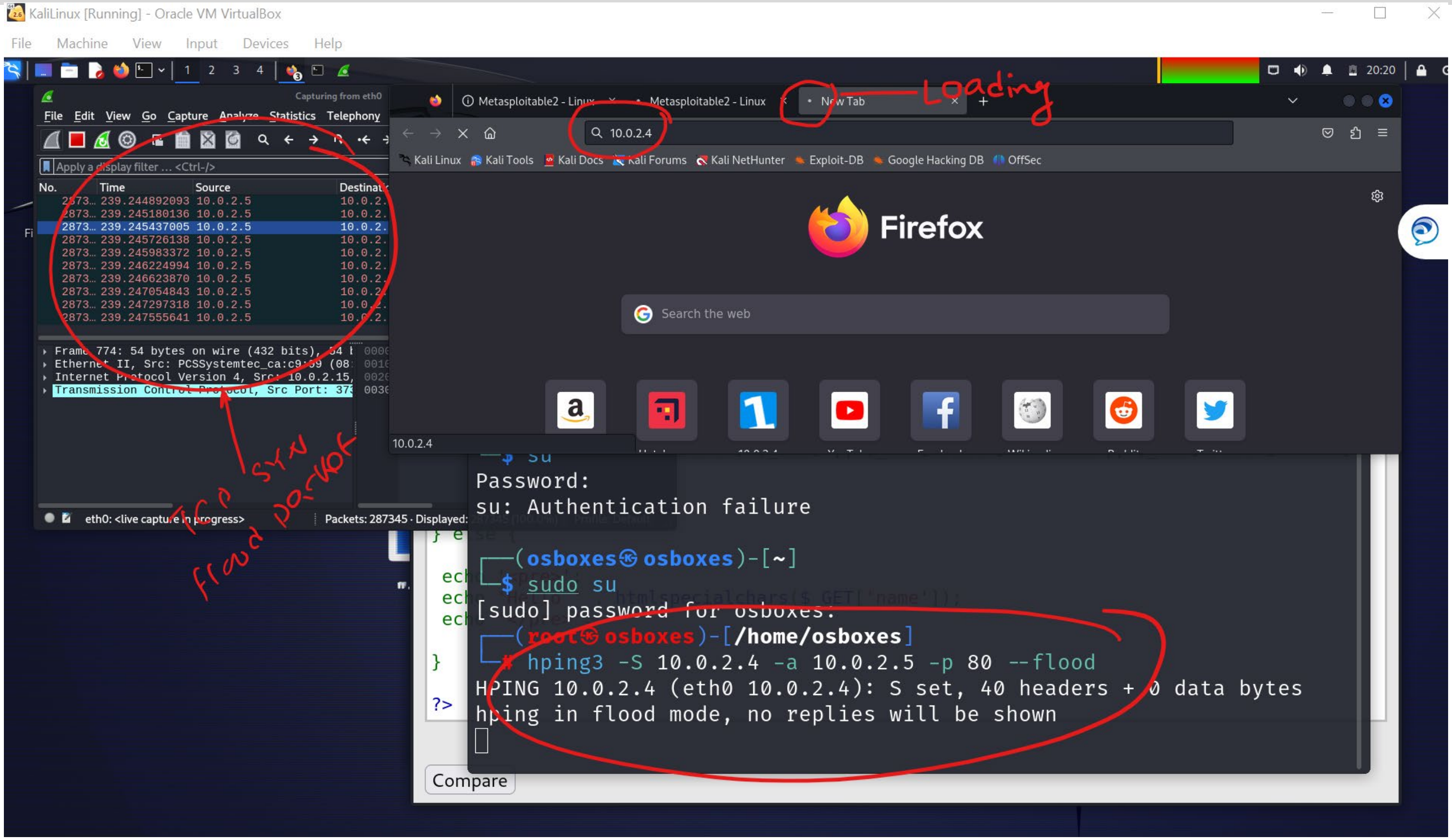


**Figure 3-9** SYN flood attack

# Denial of Service (DoS)


KaliLinux [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
1 2 3 4
20:20
Capturing from eth0
File Edit View Go Capture Analyze Statistics Telephony
Apply a display filter … <Ctrl-/>
Metasploitable2 - Linux
Metasploitable2 - Linux
New Tab
Loading
10.0.2.4
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec
Firefox
Search the web
TCP SYN flood packet
Compare

```
No.       Time           Source      Destinat
2873…  239.244892093  10.0.2.5    10.0.2.
2873…  239.245180136  10.0.2.5    10.0.2.
2873…  239.245437005  10.0.2.5    10.0.2.
2873…  239.245726138  10.0.2.5    10.0.2.
2873…  239.245983372  10.0.2.5    10.0.2.
2873…  239.246224994  10.0.2.5    10.0.2.
2873…  239.246623870  10.0.2.5    10.0.2.
2873…  239.247054843  10.0.2.5    10.0.2.
2873…  239.247297318  10.0.2.5    10.0.2.
2873…  239.247555641  10.0.2.5    10.0.2.

▸ Frame 774: 54 bytes on wire (432 bits), 54 b
▸ Ethernet II, Src: PCSSystemtec_ca:c9:09 (08
▸ Internet Protocol Version 4, Src: 10.0.2.15,
▸ Transmission Control Protocol, Src Port: 373

eth0: <live capture in progress>        Packets: 287345 · Displayed:
```

```
Password:
su: Authentication failure

┌──(osboxes㉿osboxes)-[~]
└─$ sudo su
[sudo] password for osboxes:
┌──(root㉿osboxes)-[/home/osboxes]
└─# hping3 -S 10.0.2.4 -a 10.0.2.5 -p 80 --flood
HPING 10.0.2.4 (eth0 10.0.2.4): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
```

- Demonstrate a DoS (SYN Flood) attack using hping3 in Kali and Metasploitable
- Instructor will open Wireshark in the backend and launch a DoS (SYN Flood) attack and will demonstrate how it consumes all resources of the Metasploitable machine and makes the machine unreachable. The machine won't be browsable and will take time to load.

# Interception

- Some attacks are designed to intercept network communications
- Man-in-the-Middle attacks
  - Interception of legitimate communication and forging a fictitious response to the sender
  - Two computers are sending and receiving data with a computer between them
  - In a passive attack, data is captured and recorded before sending it on to the original recipient
  - In an active attack contents of transmission are altered before they are sent to the recipient

# Interception

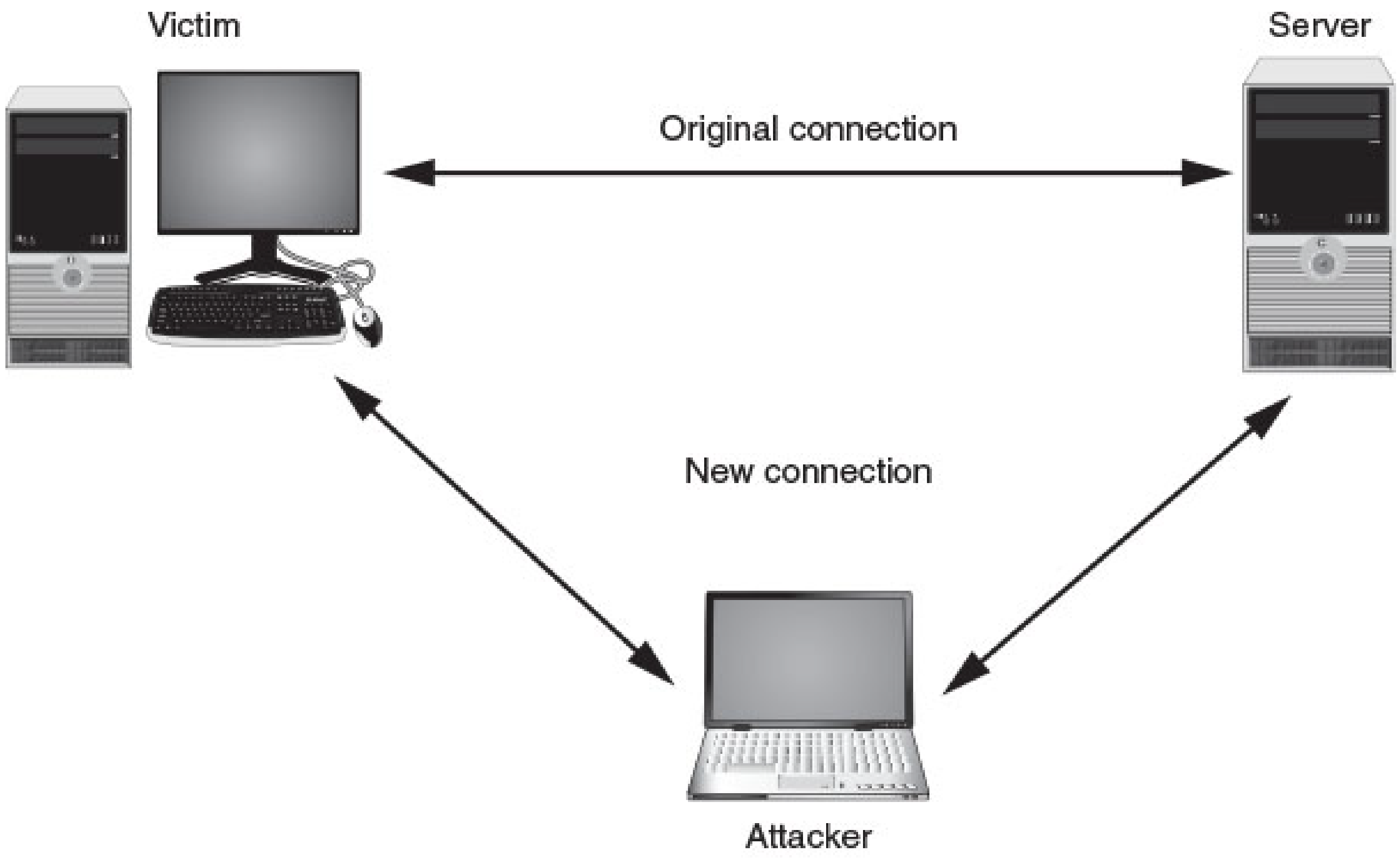


**Figure 3-10** Man-in-the-middle attack

# Interception

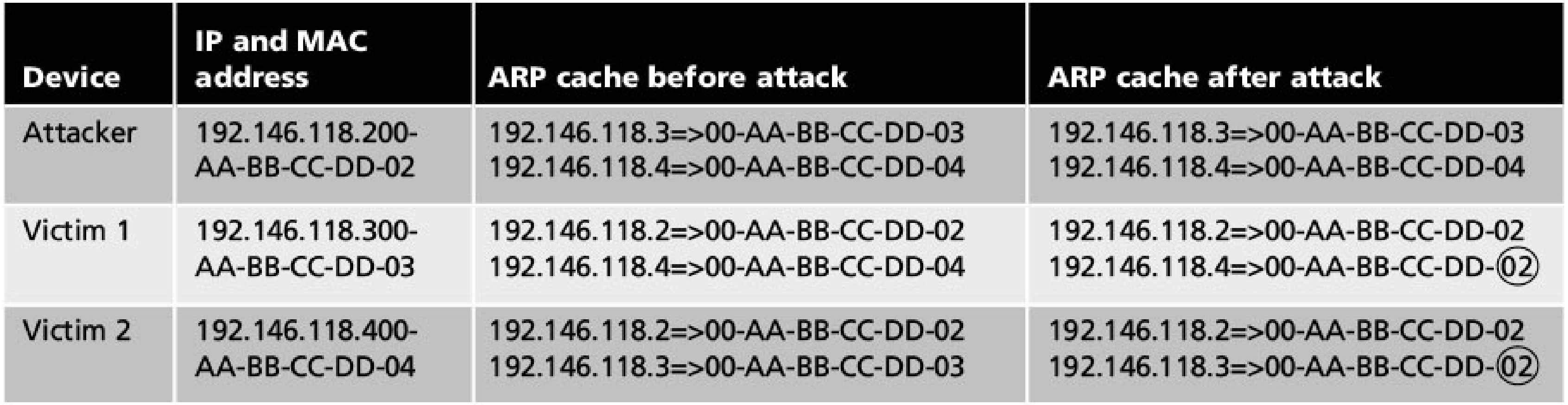
- Replay attacks
  - Attacker makes copy of transmission before sending it to the original recipient
    - Uses copy at a later time
  - Example: capturing logon credentials
- More sophisticated replay attacks
  - Attacker captures network device’s message to server and then later sends original, valid message to server
  - Establishes a trust relationship between attacker and server

# Poisoning

- Poisoning
    - The act of introducing a substance that harms or destroys
- Two types of attacks inject “poison” into a normal network process to facilitate an attack:
    - ARP poisoning
    - DNS poisoning

# Poisoning

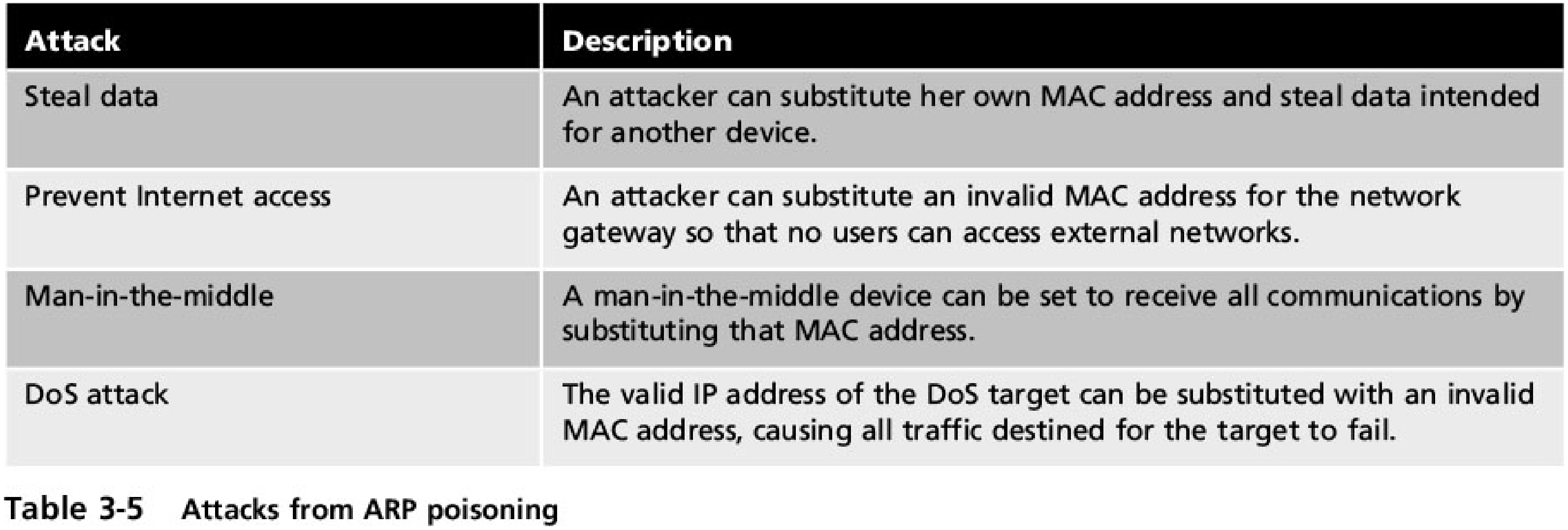
- ARP Poisoning
  - Attacker modifies MAC address in ARP cache to point to different computer



| Device | IP and MAC address | ARP cache before attack | ARP cache after attack |
|---|---|---|---|
| Attacker | 192.146.118.200-AA-BB-CC-DD-02 | 192.146.118.3=>00-AA-BB-CC-DD-03<br>192.146.118.4=>00-AA-BB-CC-DD-04 | 192.146.118.3=>00-AA-BB-CC-DD-03<br>192.146.118.4=>00-AA-BB-CC-DD-04 |
| Victim 1 | 192.146.118.300-AA-BB-CC-DD-03 | 192.146.118.2=>00-AA-BB-CC-DD-02<br>192.146.118.4=>00-AA-BB-CC-DD-04 | 192.146.118.2=>00-AA-BB-CC-DD-02<br>192.146.118.4=>00-AA-BB-CC-DD-02 |
| Victim 2 | 192.146.118.400-AA-BB-CC-DD-04 | 192.146.118.2=>00-AA-BB-CC-DD-02<br>192.146.118.3=>00-AA-BB-CC-DD-03 | 192.146.118.2=>00-AA-BB-CC-DD-02<br>192.146.118.3=>00-AA-BB-CC-DD-02 |

**Table 3-4** ARP poisoning attack

# Poisoning



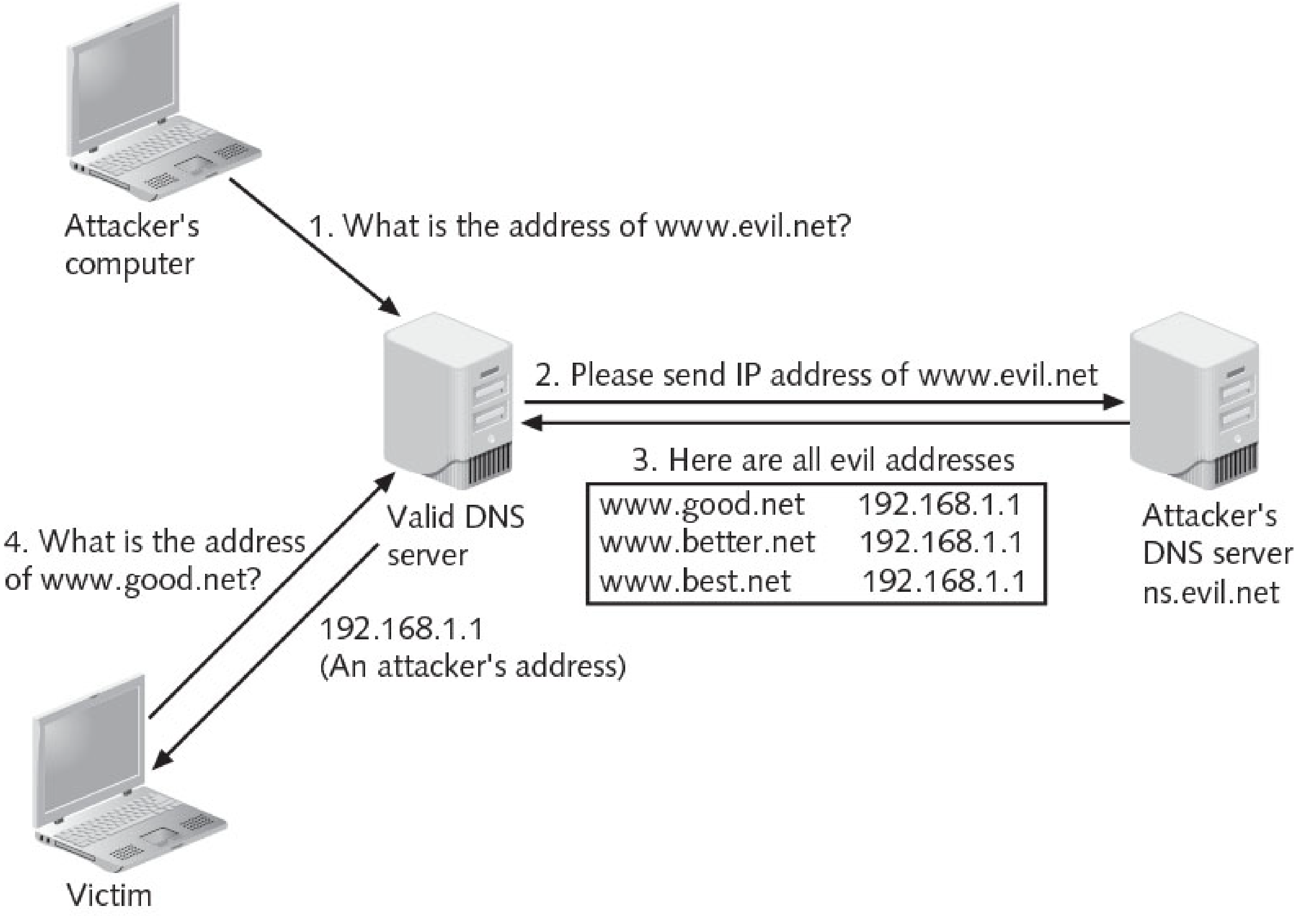
| Attack | Description |
|---|---|
| Steal data | An attacker can substitute her own MAC address and steal data intended for another device. |
| Prevent Internet access | An attacker can substitute an invalid MAC address for the network gateway so that no users can access external networks. |
| Man-in-the-middle | A man-in-the-middle device can be set to receive all communications by substituting that MAC address. |
| DoS attack | The valid IP address of the DoS target can be substituted with an invalid MAC address, causing all traffic destined for the target to fail. |

**Table 3-5** **Attacks from ARP poisoning**

# Poisoning

- DNS poisoning
  - Domain Name System is the current basis for name resolution to IP address
  - DNS poisoning substitutes DNS addresses to redirect a computer to another device
- Two locations for DNS poisoning
  - Local host table
  - External DNS server

# Poisoning



Attacker's computer

1. What is the address of www.evil.net?

Valid DNS server

2. Please send IP address of www.evil.net

Attacker's DNS server ns.evil.net

3. Here are all evil addresses

| | |
|---|---|
| www.good.net | 192.168.1.1 |
| www.better.net | 192.168.1.1 |
| www.best.net | 192.168.1.1 |

4. What is the address of www.good.net?

192.168.1.1 (An attacker's address)

Victim

**Figure 3-12** DNS poisoning

# In Class Group Activity 2 [10 minutes]

- **Group Activity** –
  - The same group now will discuss on one of the following topics - Denial of service, Interception, Poisoning & Attacks on access rights. **[5 minutes]**
  - One student will present the group's key discussion points **[1 minute/Group]**
  - Instructor feedback

**Instructor may alter the group activity and design a different one that is suitable to topics covered in this session

CELEBRATING
A CENTENARY OF
OPPORTUNITY 2016
VICTORIA UNIVERSITY

VICTORIA UNIVERSITY
MELBOURNE AUSTRALIA

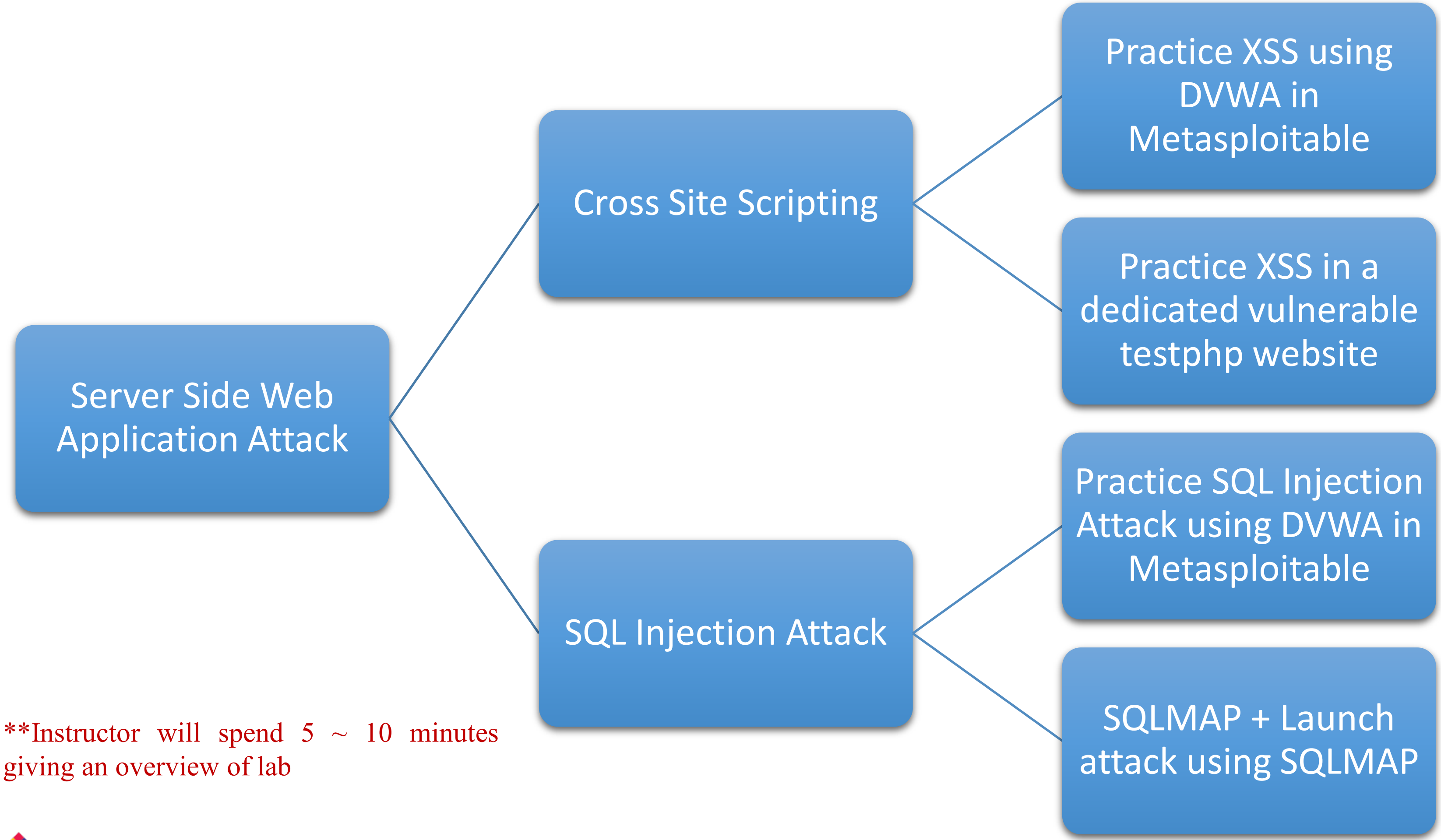
# Summary

- Web application flaws are exploited through normal communication channels, making web applications more difficult to protect
- An XSS attack uses Web sites that accept user input without validating it
  - Uses server to launch attacks on computers that access it
- Client-side attacks target vulnerabilities in client applications
  - Client interacts with a compromised server

# Summary

- Session hijacking is an attack in which an attacker steals a session token and impersonates user
- A buffer overflow attack attempts to compromise a computer by pushing data into inappropriate memory locations
- A Denial of Service attack attempts to overwhelm a system so that it cannot perform normal functions
- In ARP and DNS poisoning, valid addresses are replaced with fraudulent addresses
- Access rights and privileges may also be exploited

# Lab Overview



Server Side Web Application Attack

Cross Site Scripting

SQL Injection Attack

Practice XSS using DVWA in Metasploitable

Practice XSS in a dedicated vulnerable testphp website

Practice SQL Injection Attack using DVWA in Metasploitable

SQLMAP + Launch attack using SQLMAP

**Instructor will spend 5 ~ 10 minutes giving an overview of lab

# Exercise 3.1 - XSS at Kali Linux

This lab will demonstrate how to find a vulnerable website using a particular search pattern. However, ethically it is not recommended to practice launching an attack on real website. Therefore, first we will be using **Damn Vulnerable Web Application (DVWA)** in **Metasploitable** and access it from Kali Linux to practice and understand different type of XSS attacks. Second, we will practice further in a dedicated vulnerable live testphp website.

Please open "Lab 3 Working Procedure Step by Step Instructions.pdf" provided under laboratories module. The instruction will guide you in completing lab.

# Exercise 3.2 – SQL Injection Attack

In this exercise, first we will see how we can manipulate SQL query and launch SQL injection using DVWA in metasploitable. In the second part of this lab, we will use SQLMap to launch the SQL injection attack. You follow the video demonstration to complete this exercise.

Please open "Lab 3 Working Procedure Step by Step Instructions.pdf" provided under laboratories module. The instruction will guide you in completing lab.

You can also watch video demonstration available in pre-class activities of Session 3. Step by Step instruction will also point you to the relevant video to watch and complete lab.

# Working Procedures

**Submit your work through the '<u>Assessment 1 Practical Lab Work</u>' before you leave the class.

**The report should include screenshots and working procedures as an evidence of the completion of your lab task and is expected to be completed by lab hours.