Pointer:-

17 August 2023 11:50

101986598209

In rse_engine , we do the policy check for traffic based on different parameters.

Ctags indexes source code to make it easy to jump to functions, variables, classes, and other identifiers in Vim.

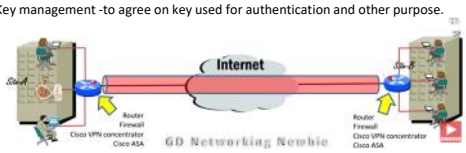
Commands for bridge:-

ip addr show dev <bridge-name>

Ipsec (Internet protocol security)

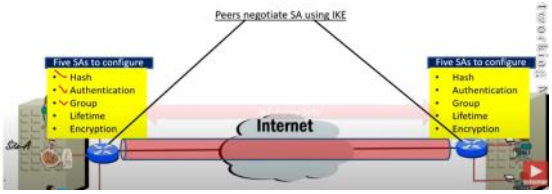
Ipsec is a protocol suite to authenticate and encrypt the packets being exchanged between two points.
VPN is a private connection over a public network - layer 2 or layer 3.
Ipsec is a standard by IETF to create a VPN tunnel at layer 3(network layer).
Ipsec provides :

1. Integrity -it indicates that the received msg is same msg that was sent[ex- MD5,SHA].
2. Authentication -refers to verifying identity of a network entity like user/device.
3. Confidentiality -it is used to hide information
4. Key management -to agree on key used for authentication and other purpose.



- To achieve the goal of creating a secure tunnel, two peers (site-A and site-B) needs to negotiate all the requires parameters.
- Ipsec uses following protocols :
  - Authentication header(AH) -provides authentication and integrity
  - Encapsulation security protocol(ESP) -provides authentication, integrity and confidentiality
  - Internet key exchange(IKE) -key management protocol, used to negotiate security associations(SA)
    [SA are security policies for communication between peers]

- IKE performs its job using ISAKMP framework using two phases:
  - Phase-1(IKEV1) : used to negotiate ISAKMP policy by exchanging 5 parameters referred to as HAGLE
    - In this phase, peers authenticate each other and calculate a shared secret key
    - Phase 1 gives a secure tunnel to be used in second IKE phase.



- This is what happens in phase 1:
  1. Authenticate and protect the identities of the IPsec peers.
  2. Negotiate a matching IKE policy between IPsec peers to protect the IKE exchange.
  3. Perform an authenticated Diffie-Hellman exchange to have matching shared secret keys.
  4. Setup a secure tunnel for IKE phase 2.

  - Phase-2 (IKEV2) :Once the sender and receiver established the ISAKMP tunnel in phase-1 they move to phase-2. phase-2 always operates in Quick mode. Phase -2 used to negotiate Ipsec security parameters[negotiate protocols and algorithms]. Here the security associations and services between the two devices are negotiated. The devices will choose which protocol(Authentication Header or Encapsulation Security Protocol) and which algorithm to use inorder to send/receive data.
    IKEV2 sets up a security association (SA) that negotiates security keys used by both the VPN client and the VPN server.Once IKEv2 validates the security association, a secure tunnel is set, which prompts encrypted communication between the two peers. The VPN protocol also usesthe famous Diffie-Hellman Key Exchange algorithm that allows for the secure exchange of private keys.

    In this phase the two firewalls will negotiate about the IPsec security parameters that will be used to protect the traffic w ithin the tunnel. In short, this is what happens in phase 2:
    1. Negotiate IPsec security parameters through the secure tunnel from phase 1.
    2. Establish IPsec security associations.
    3. Periodically renegotiates IPsec security associations for security.

Ipsec helps to protect IP traffic on network layer.
It can be uses on devices such as, routers, firewalls, hosts and server.
Ex: b/w 2 routers to create a site-to-site VPN
Ipsec protect IP packets by building the ipsec tunnel b/w it's peers.
Two phases to build ipsec tunnel is :- [IKE : internet key exchange]
IKEv1
IKEv2
Ipsec-tools is used for configuring, managing and monitoring ipsec implementations. Some of the tools are :
Racoon
Strongswan

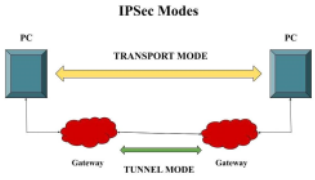
Racoon :-
IKE daemon responsible for negotiating and establishing ipsec VPN .
Uses ikev1

GRE provide the tunnel only but ipsec provide the security also
ANAP ---------(IKEv2)-------POP----------(tsec protocol)---------POP

Ipsec is a set of communication protocols or rules for setting up secure connections over a network.
Ipsec SA(security associations) -> specify the protocols, authentication and encryption on host machine inorder to establish a secure connection between hosts.
Ipsec SADB(security associations database) -> it is a database where all the Sas are stored, monitores and processed by ipsec.
To use IPsec security services, we create SAs between hosts. An SA is a simplex connection that allows two hosts to communicate with each other securely by means of IPsec. There are two types of SAs: manual and dynamic.

			Manual :- statically defines the security parameter index(SPI) values, algo, keys, matching configurations on both ends of the tunnel.
			Dynamic:- here,we configure the IKE first and then SAs. The IKE configuration defines the algorithms and keys used to establish the secure IKE connection with the peer security gateway. This connection is then used to dynamically agree upon keys and other data use d by the dynamic IPsec SA. The IKE SA is negotiated first and then used to protect the negotiations that determine the dynamic IPsec S As.

IKE(internet key exchange) is a key management protocol that create dynamic SAs. It occurred in two phases :-



IKEV2(internet key exchange version 2) :-

- An **IKE_SA** contains crypto information related to a connection with a peer. It contains multiple IPsec CHILD_SA, for which it is responsible. All traffic is handled by an IKE_SA, using the task manager and its tasks.
- An ike_cfg_t defines the rules to set up an IKE_SA.
- First, asn_tunnel is created then vti is established/updated
  - *VTI, or Virtual Tunnel Interface*, is a networking concept that simplifies the setup and management of VPN (Virtual Private Network) connections. It allows network traffic to be sent over a secured tunnel, facilitating encrypted connections between distant networks over the public internet. VTI is particularly used in IPsec VPN scenarios.
  - How VTI Works:-
    i. Tunnel Creation: A virtual interface is created on the device (e.g., router or firewall) for the VPN tunnel.
    ii. Configuration: The VTI is configured with IPsec or another tunneling protocol to secure the traffic. IP addresses are assigne d to the interface, much like any physical interface.
    iii. Routing: Routes are configured to direct traffic destined for the remote network through the VTI. Dynamic routing protocols c an be enabled on the interface to exchange routes automatically.
    iv. Traffic Flow: Traffic to and from the networks at either end of the VPN is routed through the VTI. The data packets are encap sulated and encrypted as they enter the tunnel and are decrypted when they exit the tunnel at the other end.

Difference b/w ~ and /

[root@server11 ~] : suggests that you are in the root user's home directory.
[root@server11 /] : suggests that you are at the root of the filesystem, which is the top-level directory in Linux.
The use of ~ in the first prompt signifies that you are in the root user's home directory, while the / in the second prompt indicates that you are at the root of the entire filesystem hierarchy.

- **.h**: Commonly used for both C and C++ header files, traditionally associated with C but widely used in C++ projects.
- **.hh and .hpp**: Specifically indicate C++ header files, with .hpp being more common. They help differentiate C++ headers from C headers and can make the codebase easier to navigate and understand

Makefile

- 'Makefile' is a special file used by the 'make' build automation tool to compile and build programs.
- It defines how to compile and link the program, specifying dependencies b/w files, and provide rules on how to build and update each files.
  $@ is the name of the file to be made.
  $? is the names of the changed dependents
- name of the file must either be “Makefile” or “makefile”.
- In Make variables can only be strings, they can’t be integer, float. Booleans, etc, just plain strings. You typically use, “:=”
- Types of variable being assigned :
  = (recursive) – It only looks for the variables when the command is used
  := (simply expanded) – only those defined so far get expanded

- **Building** is done when preparing an application for release, which includes compiling, packaging, testing, etc. Building is the sequence composed of compiling and linking, with possibly other tasks such as installer creation.
- **Compiling** is done at any time the compiler is involved in translating programming language code to machine code. Compiling is the act of turning source code into object code.
-

htonl() stands for "Host TO Network Long." It is a function used in network programming to ensure that integer values are represented in a consistent manner across different computer architectures when communicating over a network. Specifically, it converts unsigned integer values from host byte order to network byte order.

- **Host byte order** refers to the ordering of bytes used by the memory architecture of the host computer.
- **Network byte order** is the standard order in which bytes are transmitted over the network.

ngstats:- record flow and security related logs.

Control Plane and Data Plane

The control plane decides how data is managed, routed, and processed.
The data plane is responsible for the actual moving of data.
Example of CP:- controller, HA_manager, ipsec, ngipsec, agg-ni, pinger, ngstat
Example of DP:- tanni, panni

SSH (Secure Shell)

- https://zah.uni-heidelberg.de/it-guide/ssh-tutorial-linux
- Cryptographic network protocol
- used for secure communication and remote access over unsecured networks, such as the internet.
- provides a secure way to access and manage remote servers, devices, and systems.
- SSH features : encryption, authentication, data integrity, port forwarding, etc.
- SSH is used to connect to a remote host for a terminal session.
  ssh UserName@SSHserver.example.com

- SSH is a secure protocol used as the primary means of connecting to linux servers remotely. It provides a text-based interface by spawning a remote shell. After connecting, all commands you type in your local terminal are sent to the remote server and executed there.
- When you connect through SSH, you login using an account that exists on the remote server.

- SSH provides a safe and secure way of executing cmds, making changes and configuring services remotely.
- SSH working :
  - After login, shell session is opened and we can interact with remote server by entering cmds which are sent through an encrypted SSH tunnel and executed on the server.
  - SSH connection is implemented using a client-server model. For establishing SSH connection, remote machine has SSH daemon software running which listens for connections on a specific network port, authenticates connection requests and create the appropriate environment.
  - User's computer has a SSH client software, which knows how to communicate using SSH protocol and it is being provided the credentials that should be passed for authentication to remote server. Credentials includes info about remote host[ip], username to use, connection type.
- How SSH authenticates users ?
  - Client uses two ways to authenticate : using passwords or SSH keys
  - Passwords logins are encrypted but less secure and can be compromised by automated bots and malicious users.
  -

IP Address Masquerading

- How ping works
- How SSH establish connections
- network protocols such as TCP/IP,DNS,DHCP, BGP, OSPF, and MPLS
- knowledge of routing and switching protocols, subnetting, VLANs, and network segmentation.
- ICMP
- Stringstream in c++
- Friend class/function
- can I include .cpp header file in .c file ?

GIT:

- https://supersimpledev.github.io/references/git-github-reference.pdf
- Used for distributed version control and collaboration
- Version control : it is a practice of tracking and managing changes to software/source code over time. Track the changes in files/folders.
- Centralized version control involves getting the latest version of the code from a central repository

UNIX SOCKET

- Sockets are a direct connection between two processes.
- Sockets exists in communication domain, which determines which socket to use and what is it's range. Following domains are present inside modern os:

Domain	Communication	Communication	Address format	Address

GIT:

- https://supersimpledev.github.io/references/git-github-reference.pdf
- Used for distributed version control and collaboration
- Version control : it is a practice of tracking and managing changes to software/source code over time. Track the changes in files/folders.
- Centralized version control involves getting the latest version of the code from a central repository that will contain other ppl's code as well, making ur own changes in the code and then committing those changes into the central repository. Ex- subversion(SVN)
- In distributed version control, every developer has their own server and they will have a copy of the entire history/version of the code and all of its branches in their local machine. Therefore, can work offline as well. Ex- git
- Uses:
  - Simple to use
  - Fast
  - Branching
- Commands:
  - git status
  - git init : will create empty git repository with master branch.
  - git init -b main : will create the empty git **local** repository with main branch. [from October-2020 onward, the default branch of any GitHub repository is using "main" as the default branch name, instead of master]
  - Working directory is where we do the code changes, but inorder the git to handle this file we add this file into **staging area,** we can add the file to staging area by using cmd:
    >git add <filename>
  - git log : to get the list of all previous commits
  - git commit -m *"mention ticket num"* : it will send the file to local commit.
  - After doing the commit git provides a checksum value using SHA-1 , to maintain uniqueness for each commit.
  - After every changes in the file, we'll have to add it staging area and then commit it. Follow the same process after every modification
  - git commit -a -m *"add message"* : to directly commit after modification in file without staging.
  - git diff : to show the difference b/w previous commit and ur changes in working directory.
  - git diff --staged : to check the difference inside staging area
  - git add . : used to add all the modified file from working directory to staging area.
  - git rm --cached *filename* : use to remove file from git and after this we can remove it from our working directory. Then commit it to git after deletion ,so that latest update should be there in git.

Working directory	Local repository(.git)	
	Staging area	Commit history (objects)
firstcode.txt	Firstcode.txt	first commit

- Remote repository :
- git clone *<addr of remote repo>* : clone the remote repo into ur local machine.
- Create remote repo using github/bitbucket
- Pushing a file from local repo to remote repo :
  - Create a folder "demo_fold", inside demo_fold :
  - Create local repo [> git init]
  - ls -a then add a file "file.txt"
  - git status
  - Add file to staging area
  - Commit the file [this commit is happening inside a local repo]
  - git branch -M main : will Create a main branch
  - How to connect the local repo to remote repo ?
    - Setup the ssh , > ssh-keygen -o
    - cd .ssh
    - ls -a
    - .pub file will contain the ssh key
    - Copy the key -> open github -> profile -> add/new ssh key -> paste ssh key
  - Link to the remote repo : >git remote add origin *<ssh value>*
  - git push -u origin main : pushing the code to remote repo
- git remote -v : with this cmd we can see the origin of our server
- Tagging : A tag is an object referencing a specific commit within the project history, similar to chapter markers in a book.
  - List all the tags given for a project: > git tag
  - git tag -a <version no. / v1.0> -m "write msg" : provide tagging to a particular version
  - git show v1.0[version no.] : gives description about this particular tag.
  - When we are pushing the code to remote repositories, it won't push the tags which we have added in our local repositories, that's why need to push the tags separately
  - git push origin <tag name/ v1.0> : push the tag to remote repo

- git log --pretty=oneline : shows the list of commits in a line
- Git Branch :
  - To create a new branch : > git checkout -b <name-of-branch>
  - Shows the List of local branch: > git branch
  - switch b/w branches[here we switched to main branch] : >git switch main
  - Shows the list of local + remote branches : >git branch --all
  - Moved back to the previous branch: >git switch -[hypen]
  - To dlt the branch : >git branch -d feature2<branch-name>
  - Push the new branch to remote repo:
    - git push origin <branch name>
  - How branching work ?
    - Chk online
  - Merging two branches
    - Move to main branch
    - >git merge feature1<other branch-name>
    - Push to remote repo: git push origin main
  - >git pull origin <branch-name>

- GITHUB :
  - Website to upload repositories online like bitbucket,etc
  - Provides backup
  - Provides visual interface
  - Make collaboration easier

TMUX

- It runs in terminal and allow other terminal programs to run inside it.
- Panes : contains a terminal and running program
- Tmux server can have multiple sessions
- Sessions consists of multiple windows.
- Windows groups one or more panes together
- Each window has one active pane
- Prefix+c = create a new window
- Prefix+w = shows the list of windows
- Prefix+, = rename the window
- Prefix+& = close the current window
- Prefix = ctrl+a

UNIX SOCKET

- Sockets are a direct connection between two processes.
- Sockets exists in communication domain, which determines which socket to use and what is it's range. Following domains are present inside modern os:

Domain	Communication performed	Communication between applications	Address format	Address structure
AF_UNIX	within kernel	on same host	pathname	*sockaddr_un*
AF_INET	via IPv4	on hosts connected via an IPv4 network	32-bit IPv4 address + 16-bit port number	*sockaddr_in*
AF_INET6	via IPv6	on hosts connected via an IPv6 network	128-bit IPv6 address + 16-bit port number	*sockaddr_in6*

- Socket types:-
  1. TCP socket(SOCK_STREAM)
  2. UDP socket(SOCK_DGRAM)

  inorder to use stream socket, both the end should have a sockets i.e. operate in connected pairs.
  Unlike a stream socket, a datagram socket doesn't need to be connected to another socket in order to be used.

- Socket system calls:
  1. socket()
  2. bind()
  3. listen()
  4. accept()
  5. connect()
- Socket creation: int sockfd = socket(int domain, int type, int protocol)

- **RESOURCES:**
  - **Book**: *Linux Kernel Networking: Implementation and Theory* by Rami Rosen
  - XFRM — The Linux Kernel documentation
  - Introduction to Netlink — The Linux Kernel documentation
  - Netlink Handbook — The Linux Kernel documentation
  - strongSwan Wiki - strongSwan
  - **Book**: *IPsec: The New Security Standard for the Internet, Intranets, and Virtual Private Networks* by Naganand Doraswamy and Dan Harkins
  - **Book**: *The Linux Programming Interface* by Michael Kerrisk
  - Port hunting

C Programming:-

- Lint : Lint was the name of a program that would go through your C code and identify problems before you compiled, linked, and ran it. It was a static checker, much like FindBugs today for Java. It's used for static checking ur source code.