HOW JUMP BOX WORKS

What is a jump box?

Jump server or Jump box or Jump host is a system on a network used to access and manage devices in a separate security zone. It is a hardened and monitored device that spans two dissimilar zones and provides a controlled means of access between them.

EX: DMZ (Demilitarized Zone) – Purpose is to add an additional layer of security to an organization.

What is a network?

A Computer network is a digital telecommunication network for sharing resources between two different nodes, which are computing devices that uses a common telecommunications technology.

How it emerges?

In the 1990's when co-location facilities became more common there was a need to provide access between dissimilar security zones. The jump server concept emerged to meet this need. The jump server would span the two networks and typically be used in conjunction with a proxy service such as SOCKS to provide access from an administrative desktop to the managed device. As SSH tunneling became common, jump servers became the de facto method of access.

Implementation of a jump server:

Jump servers are typically placed between a secure zone and a DMZ to provide transparent management of devices on the DMZ once a management session has been established. The jump server acts as a single audit point for traffic and also a single place where user accounts can be managed. A prospective administrator must log into the jump server in order to gain access to the DMZ assets and all access can be logged for later audit.

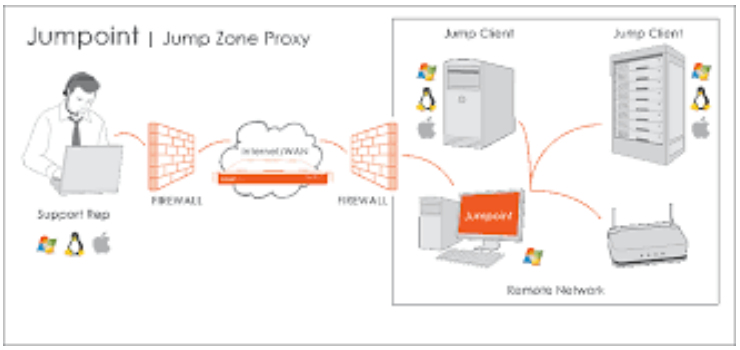

Fig 1.1 Image of Jump server

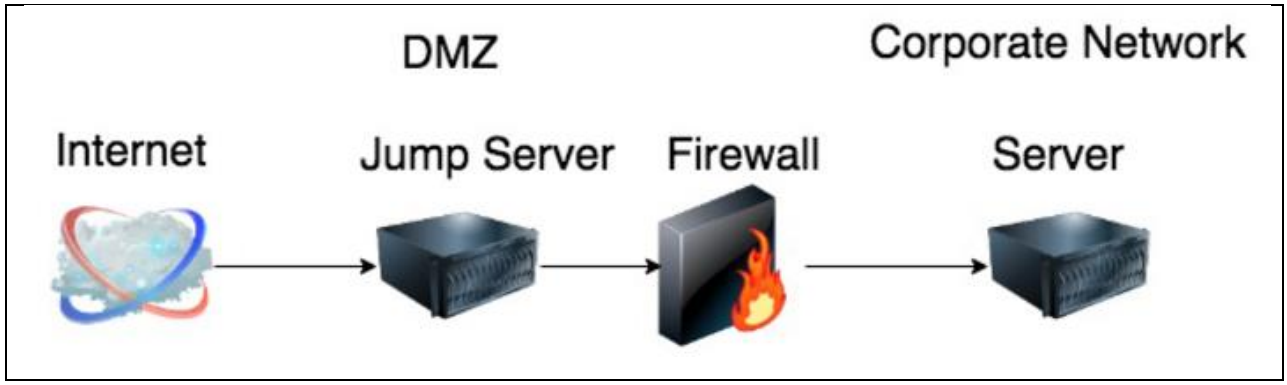

Fig 1.2 Image of Jump server

In UNIX:

A typical configuration is a hardened Unix (or Unix-like) machine configured with SSH and a local firewall.

In Windows:

A typical configuration is a Windows server running Remote Desktop Services that administrators connect to; this isolates the secure infrastructure from the configuration of the administrator's workstation.

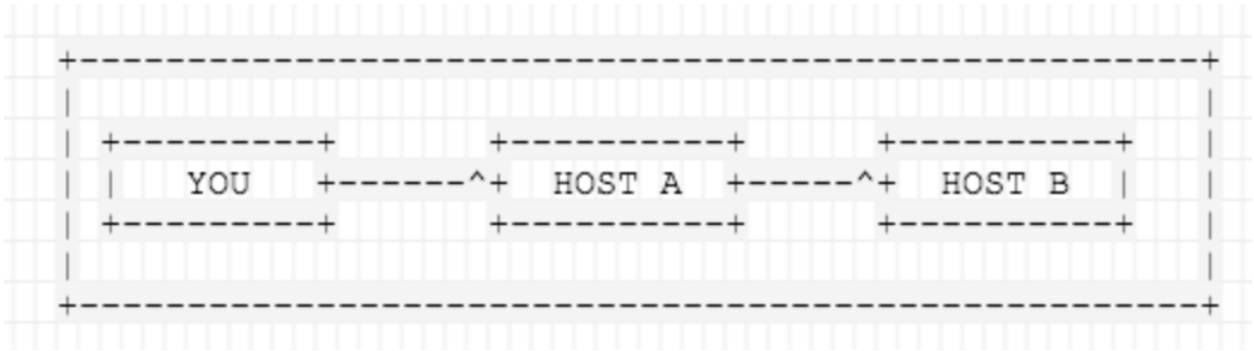

Fig 1.3 Image of Jump server

In above scenario, you want to connect to HOST 2, but you have to go through HOST 1, because of firewalling, routing and access privileges. There is a number of valid reasons why jump hosts are needed.

Dynamic Jump Host

The simplest way to connect to a target server via a jump host just is using a flag from a command line. This tells SSH to connection to the jump server.

```
$ ssh -J host1 host2
```

If usernames or ports on machines differ, specify them on the terminal as shown.

```
$ ssh -J username@host1:port username@host2:port
```

Multiple Jump host List

The same syntax can be used to make jumps over multiple servers.

```
$ ssh -J username@host1:port,username@host2:port
username@host3:port
```

Read a German (Deutsch) article on installation in Ubuntu 18.x https://www.datamate.org/ezeelogin-installation/

How to install Jump server in Ubuntu 18?

Enter the following command on your terminal to install ezeelogin dependency package on Ubuntu 18

```
root@jumpserver:~# apt update; apt-get install php mysql-server apache2 libapache2-mod-php7.2 php-mysql php-curl php7.2-xml php7.2-ldap nodejs

root@jumpserver:~#  apt install php-dev libmcrypt-dev php-pear ; apt-get -y install gcc make autoconf libc-dev pkg-config

root@jumpserver:~#  apt-get -y install php7.2-dev ; apt-get -y install libmcrypt-dev

root@jumpserver:~#  sudo pecl install mcrypt-1.0.1

root@jumpserver:~#   echo "extension=mcrypt.so" >> /etc/php/7.2/cli/php.ini
```

Set the root password with following command

```
root@jumpserver:~#  mysql_secure_installation
```

Hardware Requirements

- Minimum 512 MB Ram
- Minimum 1 GHz processing power
- Virtual Server or Dedicated server.

Software Requirements

- OS Architecture (64 bit Linux [Centos/RHEL/Ubuntu]).
- Web server (apache, lighttpd, nginx etc.)
- MySQL server (from version 5.5 to 5.7)/MariaDB (from version 5.1 to 10.1)
- PHP (from version 5.6.x and above, upto <= php 7.2)
- Ioncube loader version 10 and above for PHP
- MySQLi extension for PHP
- JSON extension for PHP
- Mcrypt extesion for PHP
- LDAP extension for PHP (for LDAP webpanel authentication)
- Nodejs

- OpenSSL

Security Risks

- Reducing the subnet size (increasing the number of subnets), and securing those VLANs using a firewall or router.
- Using higher security authentication, such as multi-factor authentication.
- Keeping the operating system and software on the jump server up to date.
- Using ACLs to restrict access to only the people that require it.
- Do not allow outbound access to the rest of the internet from the jump server.
- Restrict which programs can be run on the jump server.
- Enable strong logging.