Updaters Assignment

Saksham Dhull

April 2019

1 KVM

Kernel-based Virtual Machine (KVM) converts Linux into a bare-metal hypervisor. A hypervisor is a framework that is used to run Virtual Machines. A Bare-metal hypervisor is the one which runs directly on the host's hardware to control the hardware and to manage guest operating systems. All hypervisors need some operating system-level components such as memory, process scheduler, input/output, device drivers etc. to run VMs. KVM has all these components because it's part of the Linux kernel. Every VM is implemented as a regular Linux process, scheduled by the standard Linux scheduler, with dedicated virtual hardware like a network card, graphics adapter, CPU(s), memory, disks etc. KVM is often bound with QEMU which is a hypervisor which helps in hardware virtualization.

1.1 Storage

KVM is able to use any storage supported by Linux, including some local disks and network-attached storage (NAS). KVM also supports shared file systems so VM images may be shared by multiple hosts. Disk images support thin provisioning, allocating storage on demand rather than all up front.

1.2 Memory

KVM inherits the memory management features of Linux, including non-uniform memory access and kernel same-page merging. The memory of a VM can be swapped, backed by large volumes for better performance, and shared or backed by a disk file.

1.3 Migration

KVM supports live migration, which is the ability to move a running VM between physical hosts with no service interruption. The VM remains powered on, network connections remain active, and applications continue to run while the VM is relocated. KVM also saves a VM's current state so it can be stored and resumed later.

1.4 Performance

KVM inherits the performance of Linux, scaling to match demand load if the number of guest machines and requests increases. KVM allows the most demanding application workloads to be virtualized and is the basis for many enterprise virtualization setups, such as datacenters and private clouds

1.5 Resource Control

In the KVM model, a VM is a Linux process, scheduled and managed by the kernel. The Linux scheduler allows fine-grained control of the resources allocated to a Linux process and guarantees a quality of service for a particular process. In KVM, this includes the completely fair scheduler, control groups, network name spaces, and real-time extensions.

2 Kerberos

2.1 Working:

Kerberos is a network authentication protocol involving tickets for communication between any two nodes. All the authentications are also done using this ticketing system and this is usually done on port 88.
Steps involved:

- User tries to login with a certain username and password. Now a request(containing the username) is sent to the Key Distribution center which searches the username present in the Directory present in it. And upon hitting some matched username, it encrypts a message, that contains the further instructions to be followed, using the password stored along with the username in the files. Now this message is sent back to the user which tries to interpret it using the password entered by the user. The message compromises of two parts. First part contains a Ticket Granting service's secret key encrypted using the password of user as was present in the KDC and a TGT(ticket granting tickets which contains client/TGS session key along with other information like timeout, client's information etc.) which is encrypted using the TGS's secret key. If the first part of message is de-crypted correctly, the User will be able to get the key that will help in decoding the second part which the information needed for further processing. Upon failing, everything is set back to initial stage.
- All further communications are encrypted using this TGS secret key.
- Upon successfully de-crypting the TGT, it is stored on a KerbTray on the user's side. Whenever the user wants to access some services, It sends a message comprising of two parts, first is composed of TGT and the ID of the requested service encrypted using TGS secret key, second is composed of the client ID and timeout encrypted using client/TGS session key. This

message is sent to the Target Granting Server. TGS on accepting the request, de-crypts first part using it's secret key, which gives it with the request ID and the client/TGS session key which it uses to de-crypt the second part and compares client IDs in TGT and this message. If the ID's match the following messages are sent back, first contains Client-to-server ticket (which includes the client ID, client network address, validity period and Client/Server Session Key) encrypted using the TGS's secret key, second contains Client/Server Session Key encrypted with the Client/TGS Session Key.

- Upon receiving the messages back from TGS, the user de-crypts the first message using the TGS's secret key which provides it with the key to de-crypt the second message which contains all the information it will need to do server requests. The first message is then sent directly to the Service provider along with another message containing a new Authenticator, which includes the client ID, timestamp and is encrypted using Client/Server Session Key.

- The SP de-crypts the first ticket using its own secret key to retrieve the Client/Server Session Key. Using the sessions key, SP de-crypts the Authenticator and compares client ID from both the messages, if they match server sends the a message to the client, which contains the timestamp found in client's Authenticator encrypted using the Client/Server Session Key. to confirm its identity to serve the client. The client decrypts the confirmation using the Client/Server Session Key and checks whether the timestamp is correct. If so, then the client can trust the server and can start issuing service requests to the server.

- One important advantage of the system is the one time authentication needed until timeout happens. The above steps are only one time and are not repeated again until timeout happens. The user can now directly communicate with server and accept responses.

2.2 Limitations:

- The working of this whole model requires continuous availability of a central server otherwise new users can't log in.

- Also kerberos works on the basis of timestamps and thus the clocks of all the servers must be synchronised