EC-Council

Building A Culture Of Security

C|ND

Certified Network Defender

Certified Network Defender v3

MODULE 11

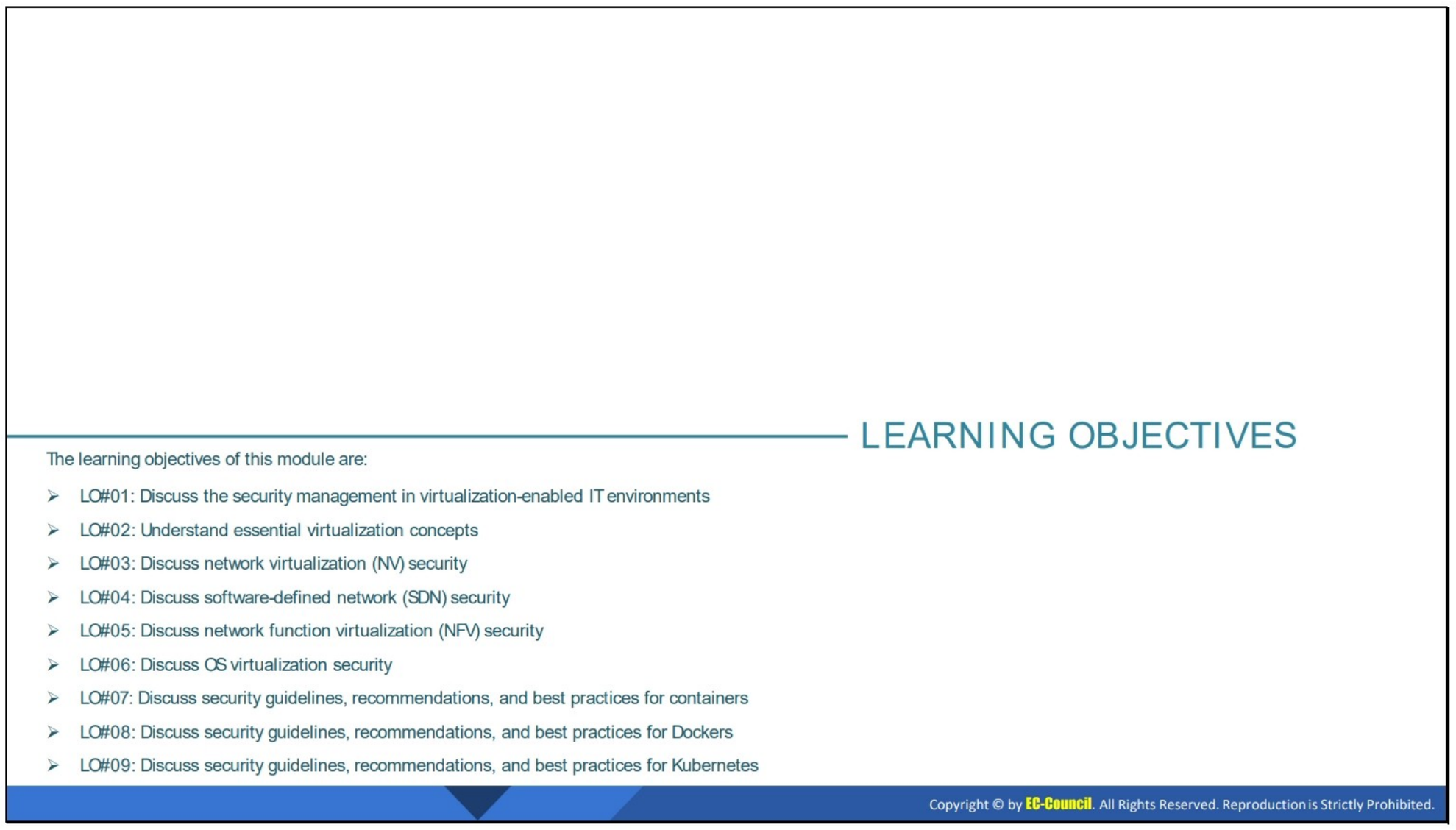
ENTERPRISE VIRTUAL NETWORK SECURITY

EC-Council Official Curricula

https://t.me/learningnets

This page is intentionally left blank.

https://t.me/learningnets



Learning Objectives

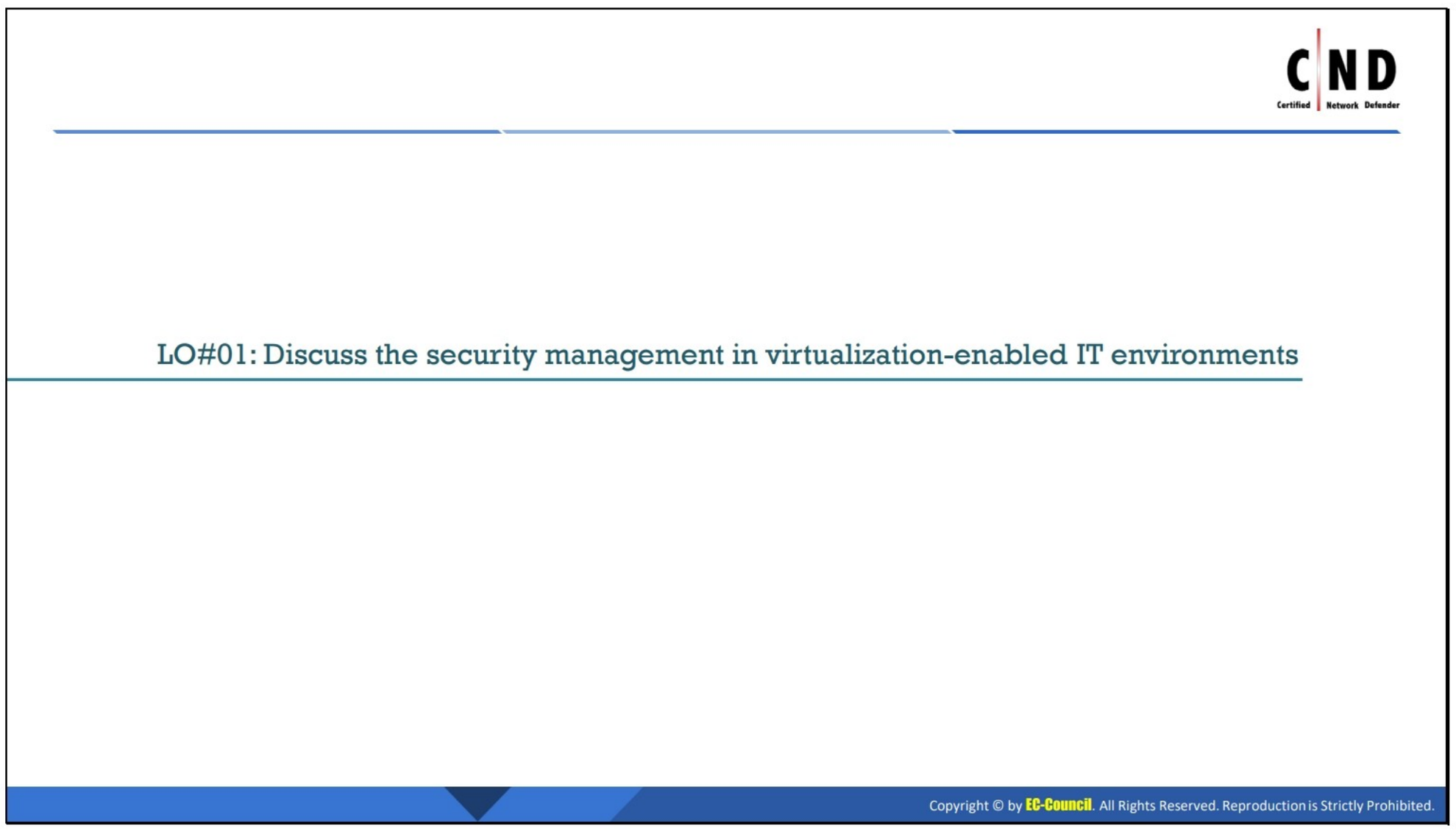
Modern IT environments use server virtualization, network virtualization, storage virtualization, and desktop virtualization for faster provisioning of network environments and to keep pace with modern technologies. Virtualization has been changing security concepts in modern IT environments, as various security challenges associated with virtualization are unique and distinct from those of traditional environments. This module discusses virtualization concepts and technologies such as network virtualization, software-defined network, and network function virtualization, and their security.

The following are the learning objectives of this module:

- Evolution of network and security management concepts in modern virtualized IT environments.
- Essential concepts in virtualization.
- Network virtualization (NV) security.
- Software-defined network (SDN) security.
- OS virtualization security.
- Security guidelines, recommendations and best practices for containers.
- Security guidelines, recommendations and best Practices for Dockers.
- Security guidelines, recommendations and best practices for Kubernetes.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

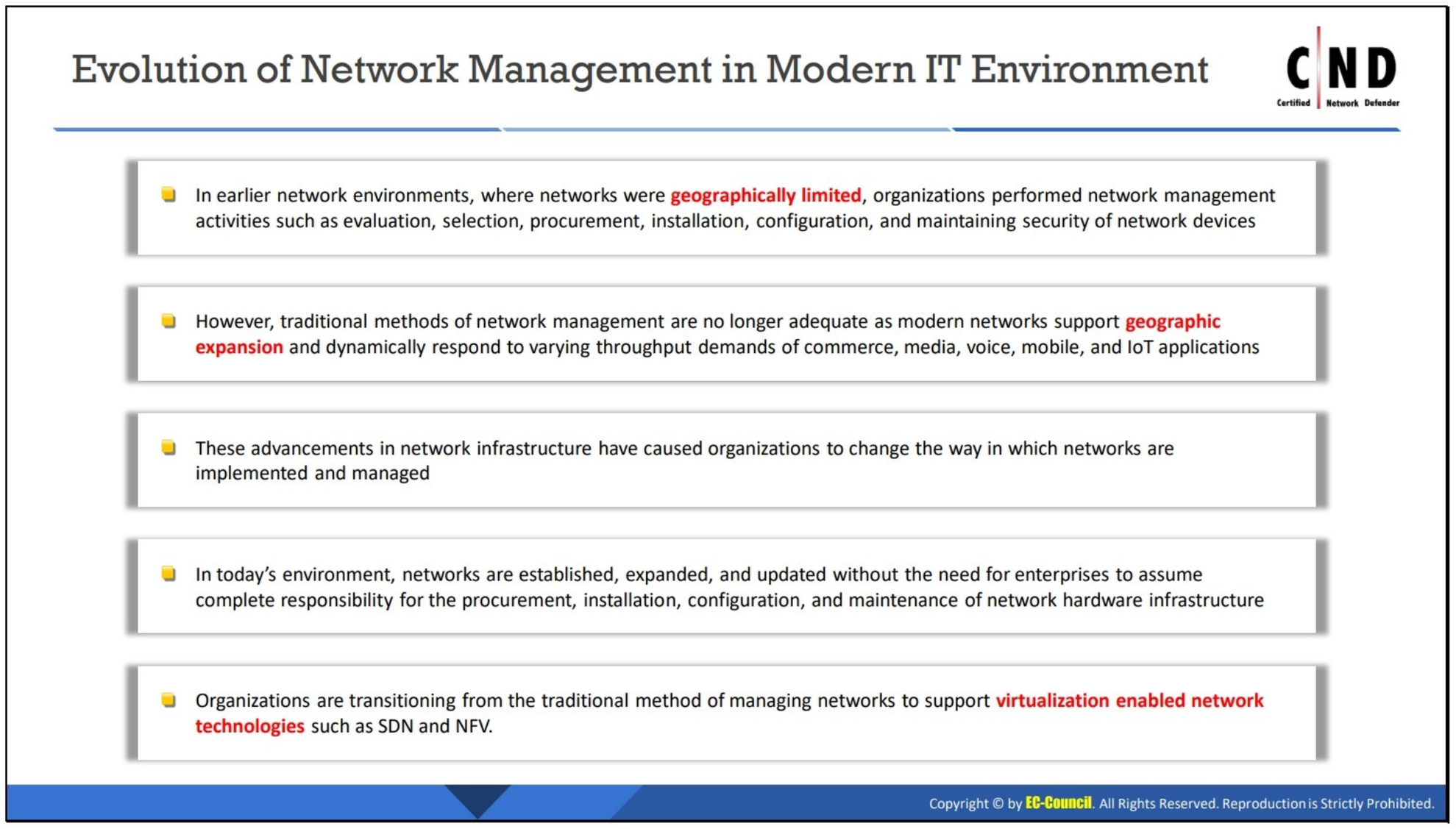


LO#01: Security Management in Virtualization-Enabled IT Environments

The objective of this section is to explain the evolution of network management in modern virtualized IT environments, and the various security requirements and challenges in these environments.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Evolution of Network Management in Modern IT Environment

In earlier network environments, where networks were geographically limited, organizations performed network management activities such as evaluation, selection, procurement, installation, and configuration of network devices, in addition to managing their security.

The traditional network environment relied on purpose-built hardware functionality, which did not provide adequate flexibility to an organization. Due to protracted procurement cycles, the installation of a device in a remote location was difficult. It was difficult to control the organization's resources and enforce policies, and the nature of the network was increasingly disaggregated. The traditional network paradigm limited the fulfillment of dynamic requirements such as provisioning and de-provisioning of infrastructure such as servers, security policy modification, and performance monitoring.

Due to tremendous growth in businesses, there is a need for faster deployment of network services, rendering the traditional method of network management to be no longer adequate. This has caused organizations to fundamentally change the manner in which networks are implemented and managed, in order to achieve geographic expansion and dynamically respond to the varying throughput demands of commerce, media, voice, mobile, and IoT applications.

In today's environment, networks are established, expanded, and updated without the need for enterprises to assume complete responsibility for the procurement, installation, configuration, and maintenance of network hardware infrastructure.

Therefore, organizations are transitioning from the traditional method of managing networks to virtualization-enabled network technologies such as software-defined networking (SDN) and network function virtualization (NFV). The transformation and evolution of network management from the traditional network environment to virtualization-enabled technologies

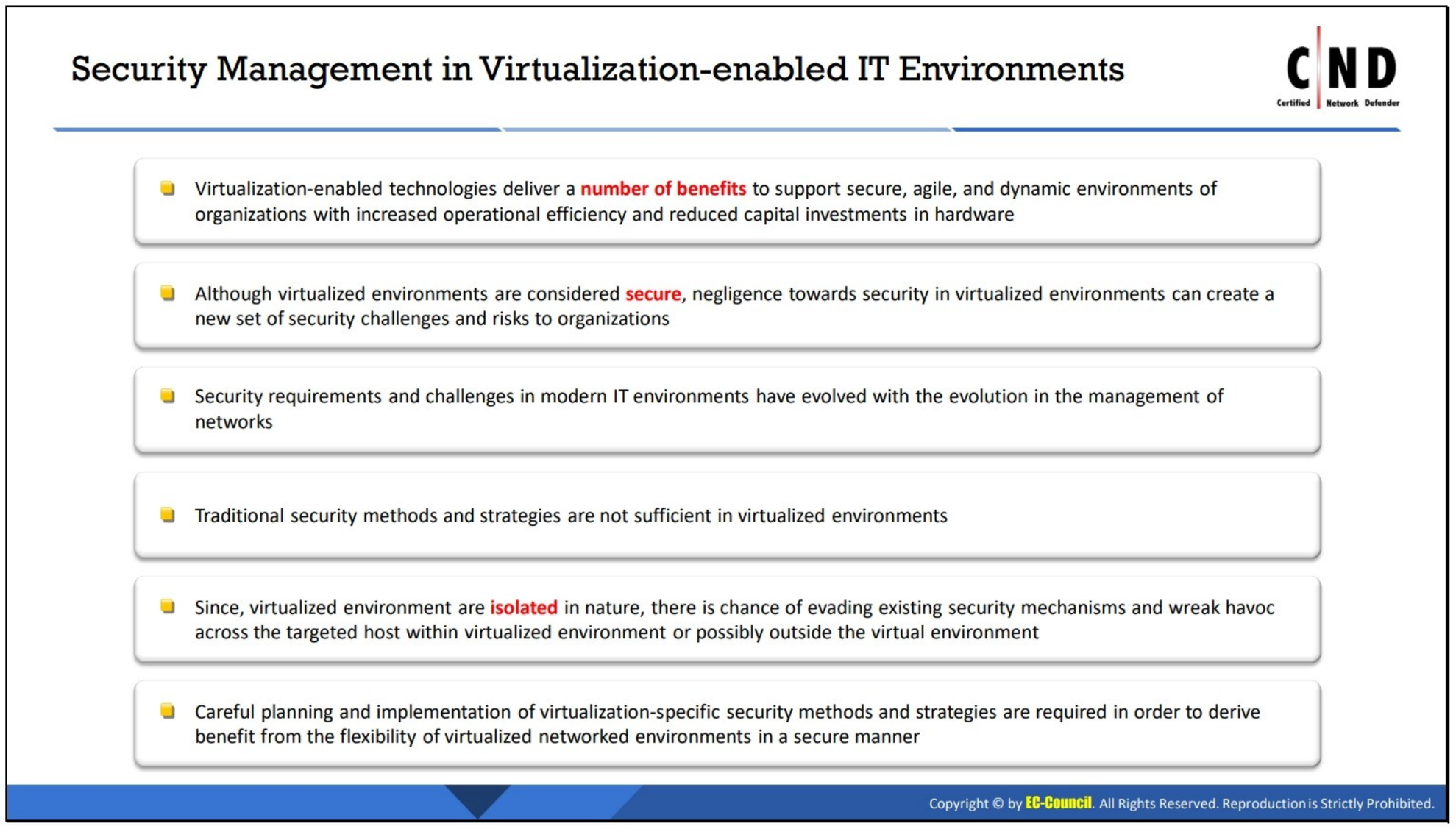
Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

completely replaces the physical with the digital. These technologies provide greater flexibility to an organization, centralized control of organizational resources, fine granularity in policies enforcement, protection of data and applications, security of resources, improved management of processing demands with network automation, and enhanced network efficiency.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Security Management in Virtualization-enabled IT Environments

Virtualization-enabled technologies deliver a number of benefits to support secure, agile, and dynamic environments in organizations, with increased operational efficiency and reduced capital investments in hardware. Nevertheless, with the evolution of network management in modern IT environments, the security needs and challenges of these environments have also evolved. Traditional security methods and strategies are not sufficient in virtualized environments. Since multiple virtualized environments maybe be physically collocated within a single host, and the isolation of each virtualized environment is software based, a breach in security mechanisms can wreak havoc not just within the virtualized environment, but also potentially across the entire targeted host.

Although virtualized environments are considered secure, utilizing virtualized environments with negligence towards security can create new security challenges and risks to organizations. Hence, virtualization-specific security methods and strategies should be carefully planned and implemented to reap the benefits of the flexibility and security of virtualized networked environments.

Examples of risks associated with virtual environments.

The following are some examples of risks associated with virtual environments:

- **Virtual Machines (VM) sprawling**
  - As VMs can be easily created in virtual environments, it is possible that the number of VMs increases to a point that the administrator is no longer able to effectively manage them. This phenomenon is referred to as VM sprawling, and it can potentially increase the number of unpatched VMs in the network environment.

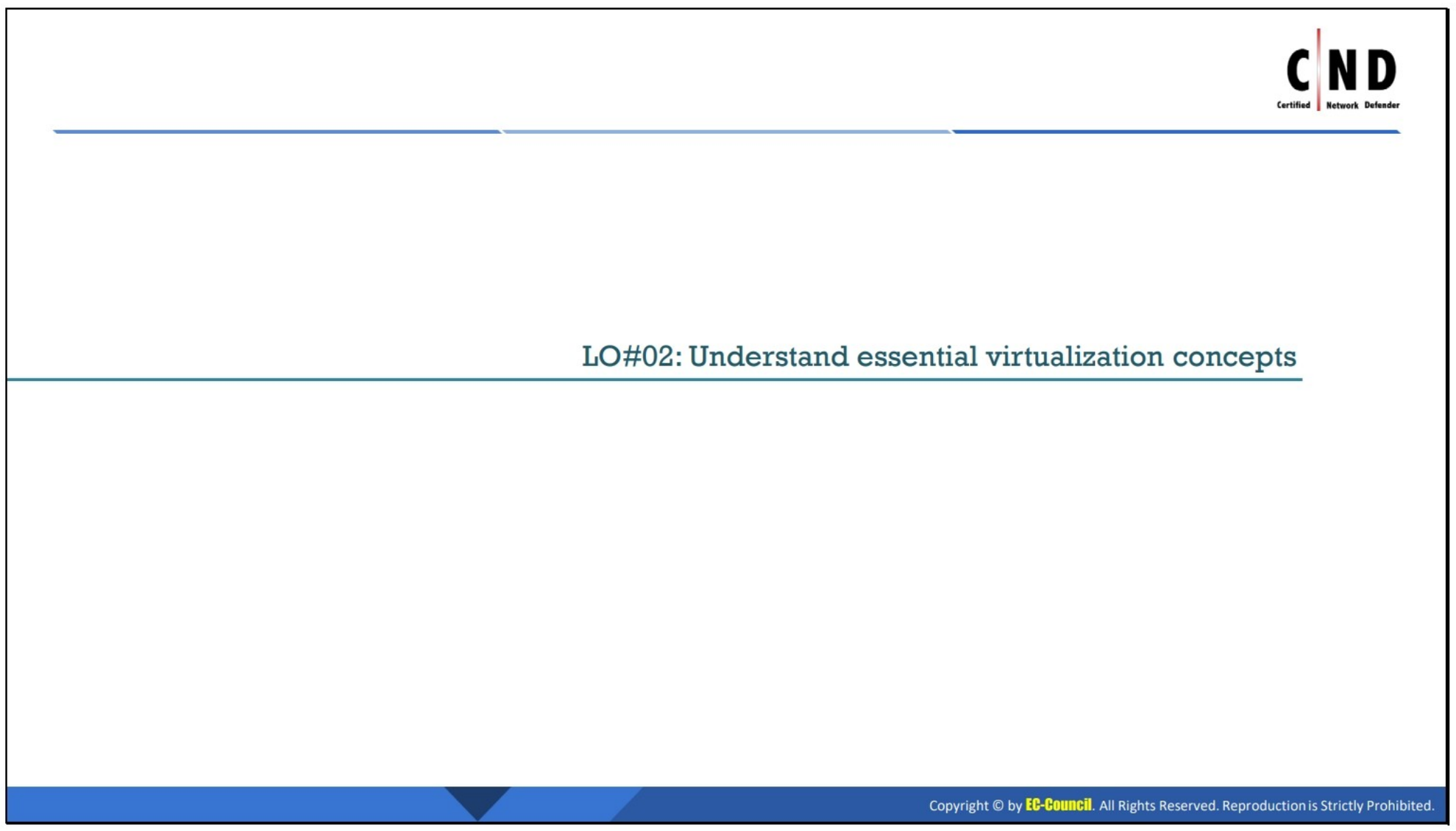
Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

- **Sensitive data within a VM**
  - With the ability to move VMs across the network easily, sensitive data in a VM can be at risk of being compromised.
- **Security of offline and dormant VMs**
  - Offline and dormant VMs can endanger the network environment, as these VMs may lag behind the baseline security of the environment, and if started, may serve as potential entry points for breaches.
- **Security of pre-configured/active VMs**
  - An attacker can tweak the security settings of pre-configured or active VMs by gaining unauthorized access to the platform in which the VMs are hosted, unless appropriate security is in place.
- **Lack of visibility and control over virtual networks**
  - Traditional security protection devices lack visibility of and control over traffic in virtual networks
- **Resource exhaustion**
  - Over-allocation of virtual environments can lead to significant performance degradation and exhaustion of resources.
- **Hypervisor security**
  - Unauthorized access to the hypervisor can pose a significant security risk to virtual environments, because the security of a device or server can be changed on the hypervisor. Therefore, the hypervisor is potentially a single point of failure for VMs on the host.
- **Account or service hijacking**
  - The virtual environment and hypervisor are often accessed through a self-service portal. The compromise of an account on such a portal can have significant consequences for security.
- **Workloads of different trust levels located on the same server**
  - VMs or applications with sensitive data in workloads can be at risk when workloads of different trust levels are located on the same server.
- **Cloud service provider APIs**
  - Using the cloud service provider APIs for communication between environments can pose additional risk.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

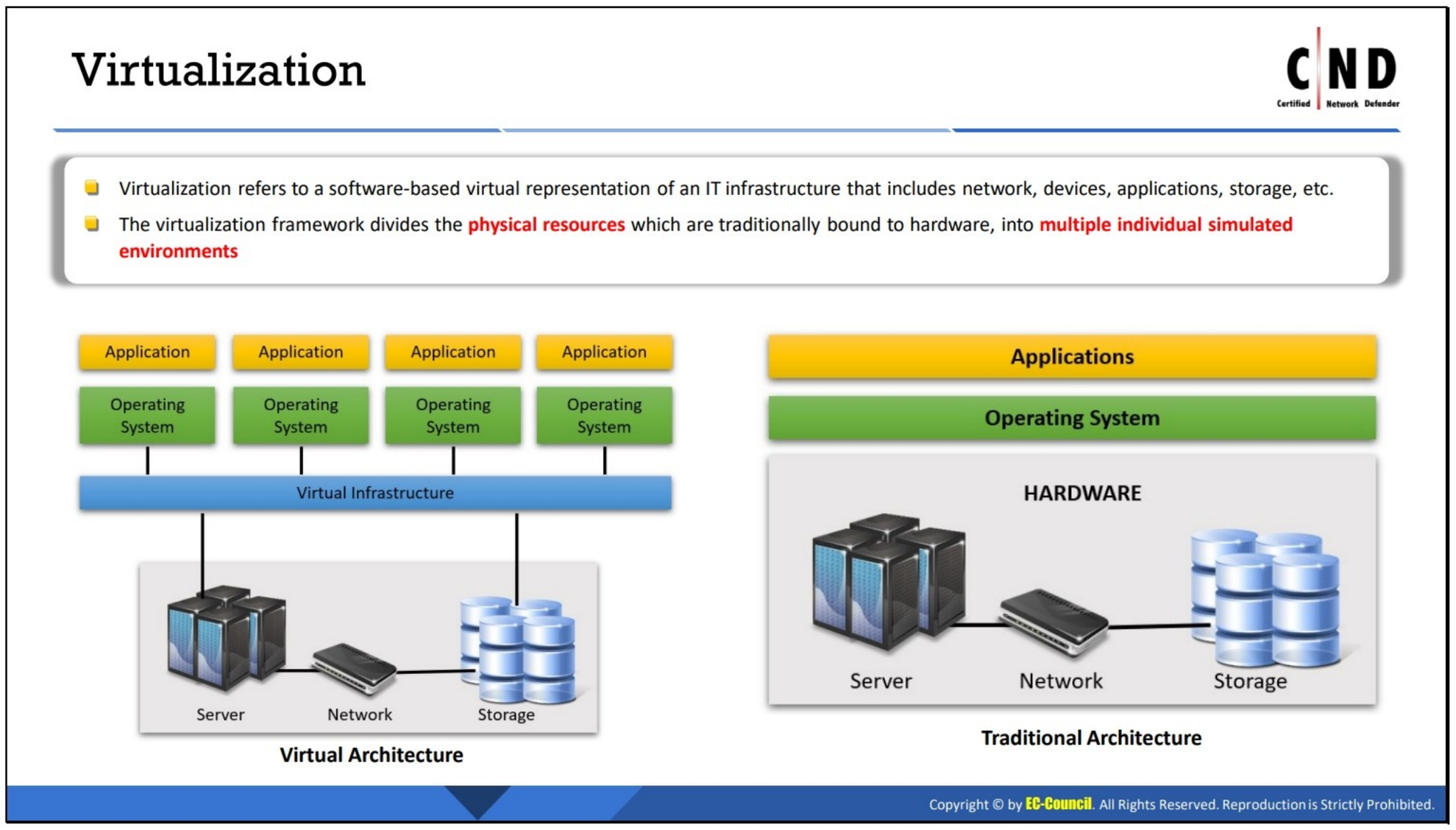


LO#02: Understand Essential Virtualization Concepts

The objective of this section is to explain virtualization concepts, the types of virtualization, the various components of virtualization, and the various enablers of virtualization technology.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Virtualization

Virtualization refers to the creation of virtual, as opposed to actual or physical, versions of computer resources in which software simulates the functionality of hardware. In a virtual environment, a number of virtual resources may be created from one physical resource, or one virtual resource may be created from one or more physical resources. Virtualization enables companies to operate multiple operating systems and execute numerous applications on a single server. Thus, virtualization enhances efficiency and the scale of the economy of an organization.

The virtualization architecture may be best illustrated by contrasting it with the traditional architecture. In the traditional architecture, the hardware infrastructure (host machine) runs a single operating system in which all applications are executed.

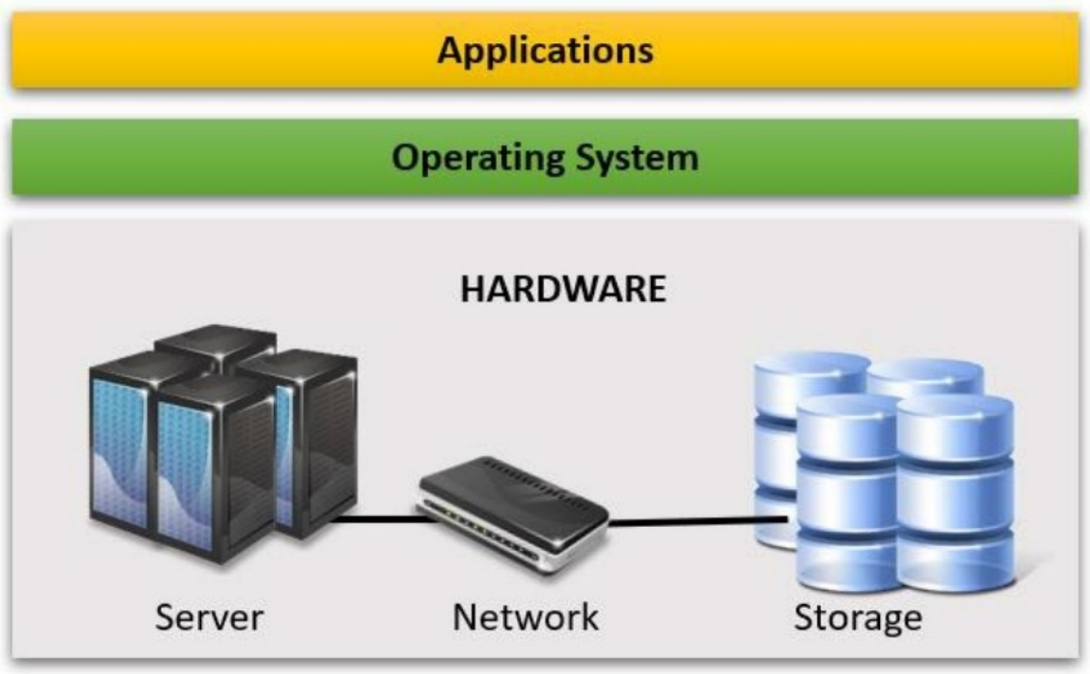


Figure 11.1: Traditional Architecture

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

The above figure illustrates of the traditional architecture. In the figure, a single instance of an operating system, with a set of applications, completely utilizes the available 32-bit hardware infrastructure. The host OS directly interacts with the hardware to request system resources.

By contrast, in the virtualization architecture, the hardware platform (host machine) is used to run multiple sets of virtual operating systems (guest OSes) and their applications.

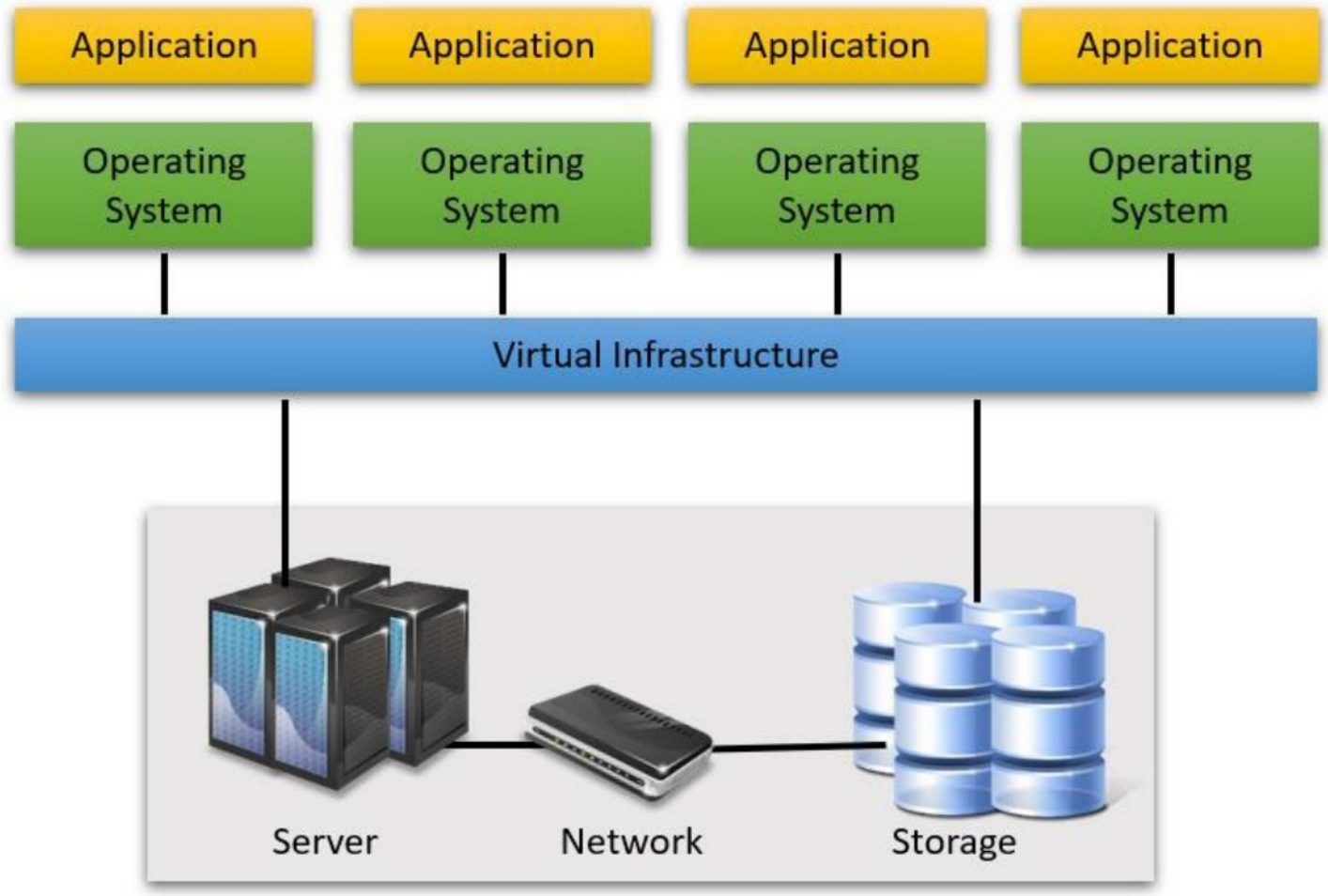


Figure 11.2: Virtualization Architecture

The above figure illustrates the virtualization architecture. As shown in the figure, the virtualization layer acts as middleware between the operating systems and the computer hardware. It logically partitions the hardware resources based on the requests received from the host and the guest operating systems. The host OS directly interacts with the computer hardware, but the guest OSes interact through the virtualization layer.

Virtualization Approaches

Various approaches can be adopted to achieve virtualization, as described below:

- **Full Virtualization:** In this type of virtualization, the guest OS is not aware that it is running in a virtualized environment. It sends commands to the virtual machine manager (VMM) to interact with the computer hardware. The VMM then translates the commands to binary instructions and forwards them to the host OS. The resources are allocated to the guest OS through the VMM.
- **OS assisted Virtualization or Para Virtualization:** In this type of virtualization, the guest OS is aware of the virtual environment in which it is running and communicates with the host machine to request for resources. The commands are translated into binary code by the guest OS for the computer hardware. The VMM is not involved in the request and response operations.
- **Hardware assisted Virtualization:** Modern microprocessor architectures have special instructions to aid the virtualization of hardware. These instructions enable the guest OS

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

to execute privileged instructions directly on the processer. The operating system treats the system calls as user programs.

- **Hybrid Virtualization:** In this type of virtualization, the guest OS adopts the functionality of para virtualization and uses the VMM for binary translation to different types of hardware resources.

Further, the design of a virtual environment may incorporate several levels of virtualization.

The following are some levels of virtualization that a virtual environment may leverage.

- **Storage Device Virtualization:** This is the virtualization of storage devices using techniques such as data striping and data mirroring. RAID is an example of storage virtualization, in which multiple storage devices are combined into a single logical unit.
- **File System Virtualization:** This refers to the virtualization of data at the level of the file system. It facilitates convenience of sharing and protection of data within the software. Virtualized data pools manipulate files and data based on user demand.
- **Server Virtualization:** Server level virtualization enables the partition (or virtualization) of the server's operating system environment. This involves the logical partitioning of the server's hard drive.
- **Fabric Virtualization:** This level of virtualization makes the virtual devices independent of the physical computer hardware. It creates a massive pool of storage areas for different virtual machines running on the hardware. Storage area network (SAN) technology is used to achieve fabric level virtualization.

Types of virtualization:

- **Operating System Virtualization:** This type of virtualization enables the hardware to execute multiple operating systems simultaneously, thus enabling the user to run applications requiring different operating systems on a single system. This is done directly in the kernel of the operating system, which not only reduces hardware costs, but also saves time spent on updating software on multiple machines.
- **Network Virtualization**: While actual hardware resources were visible and allocated to software in traditional networks, network virtualization creates an abstraction of these network resources. In network virtualization, multiple physical networks are combined into a single software-based virtual network, or a single physical network is divided and exists as multiple independent virtual networks.
- **Server Virtualization**: This is the virtualization of server resources such as physical servers, processors, and operating systems. This process enables the creation and abstraction of multiple virtual machines on a single server. Each virtual machine works independently and runs its own operating system.
- **Desktop Virtualization**: In this virtualization technology, the operating system instance, representing the user's desktop, is located within a central server on the cloud. This enables the user to control the desktop on the cloud, and use any device to access it. The data and files are not stored on the system with which the user

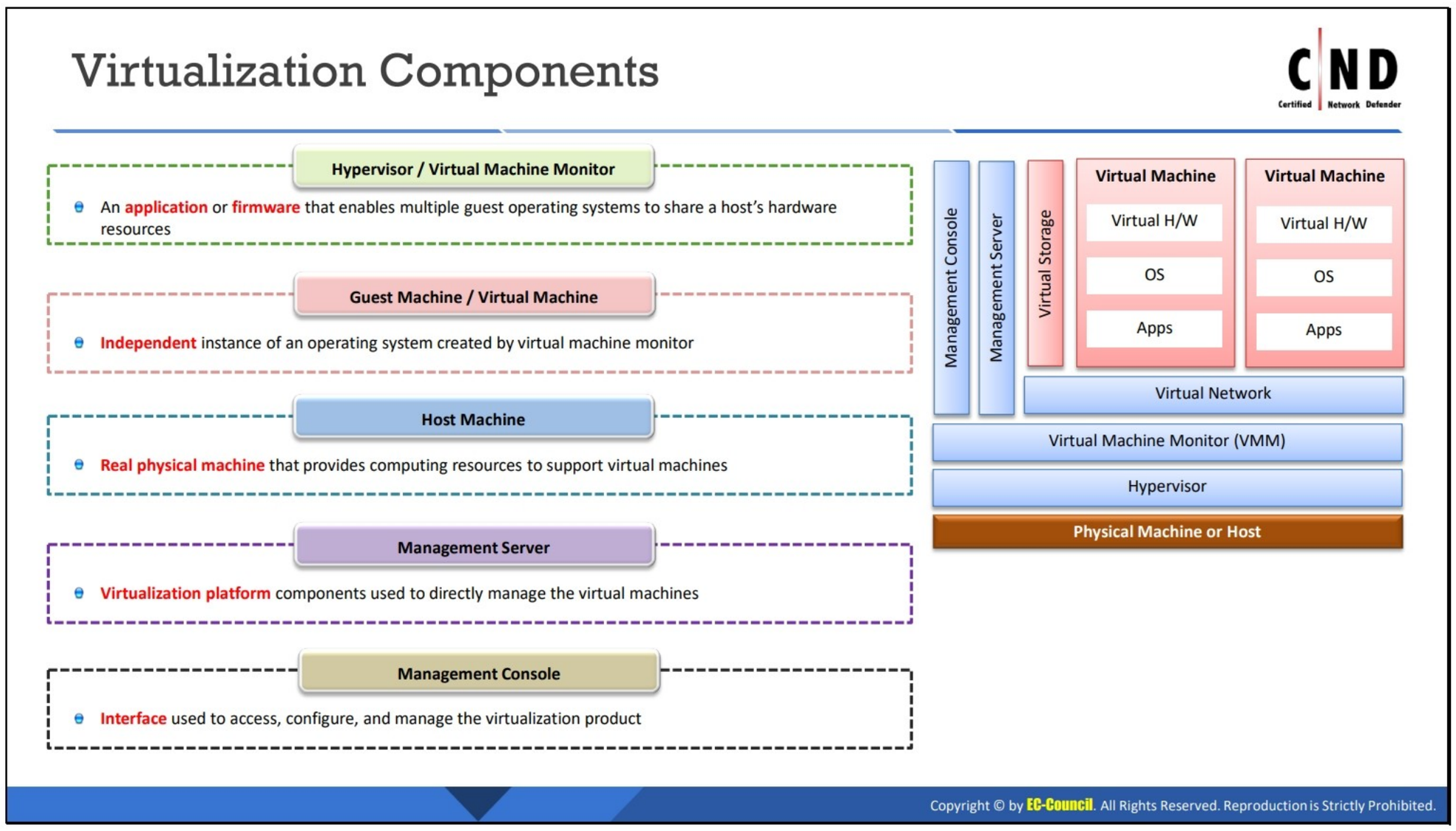
Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

accesses the desktop, but are instead stored in the cloud. Virtualized desktops can be accessed through a server and are hosted on a remote central server, which could be a cluster of computers, thus enabling the user to maintain a desktop on a single central server or cloud. Desktop virtualization reduces the cost of ownership and downtime, enables centralized management and may enhance security.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Virtualization Components

The basic components of virtualization include,

- **Hypervisor / Virtual Machine Monitor:** An application or firmware that enables multiple guest operating systems to share a host's hardware resources.
- **Guest machines:** Independent instances of operating systems created by a virtual machine monitor (VMM). With the resources provided, the guest machine works as if it is an actual physical machine.
- **Host/physical machine:** Real physical machine which provides computing resources to support guest machines. It is the server component of the virtual machine that supports the guest machine.
- **Management Server:** Virtualization platform components used to directly manage the virtual machines and to simplify the administration of resources.
- **Management Console:** Component used to access, configure, and use the management interface of the virtualization product
- **Network Components:** Components for creating a virtual network to support virtual machines. Firewalls, load balancers, storage, switches, network interface cards, etc. are examples of the components of a virtual network.
- **Virtual Storage:** Components for abstracting physical storage in a single storage device. This enables the multiple systems present in the host machine to use the available storage among themselves.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

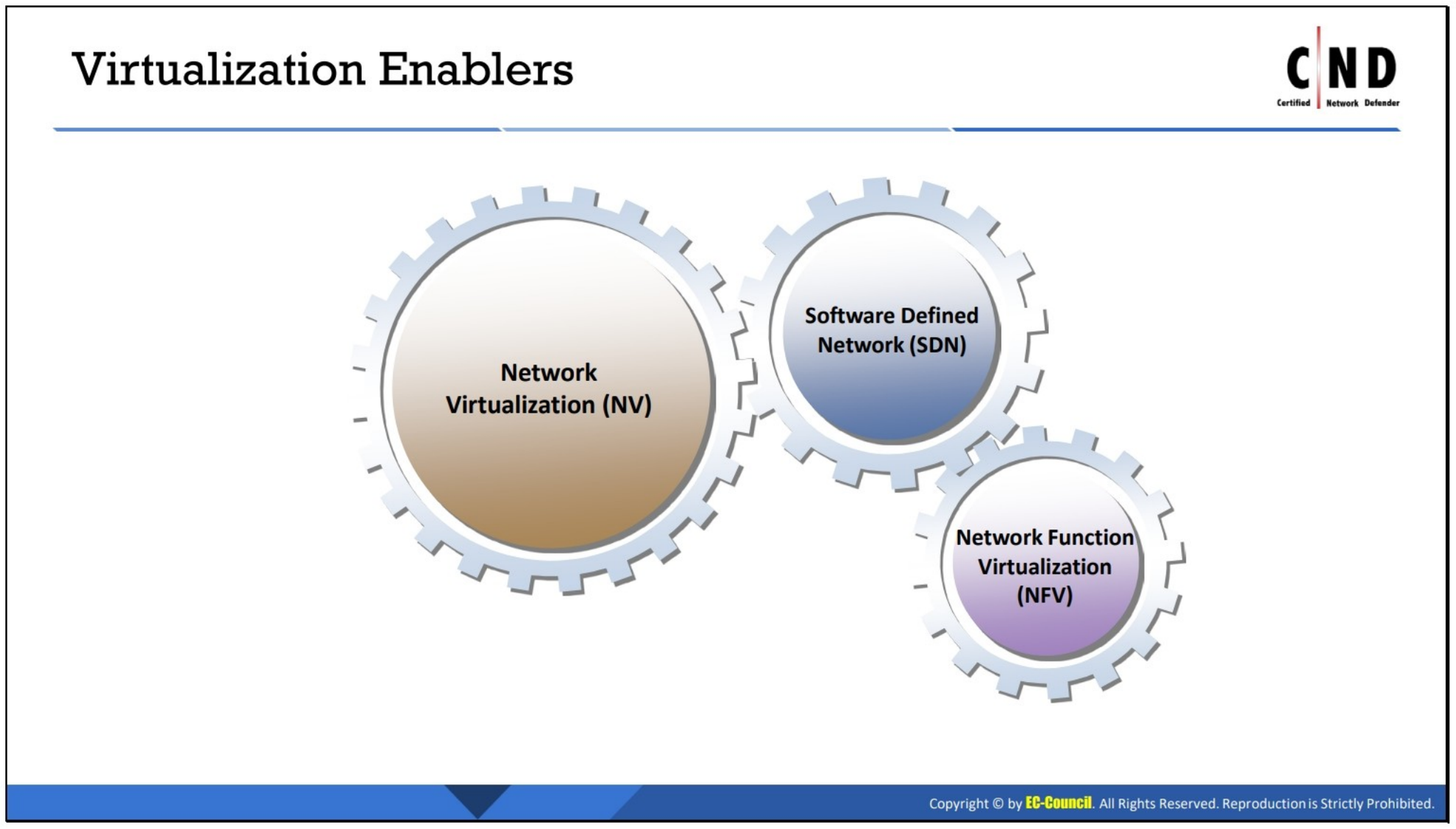
Virtualization has moved beyond just server and storage capacities and now encompasses the network as well.

https://www.mitre.org

Copyright © by EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

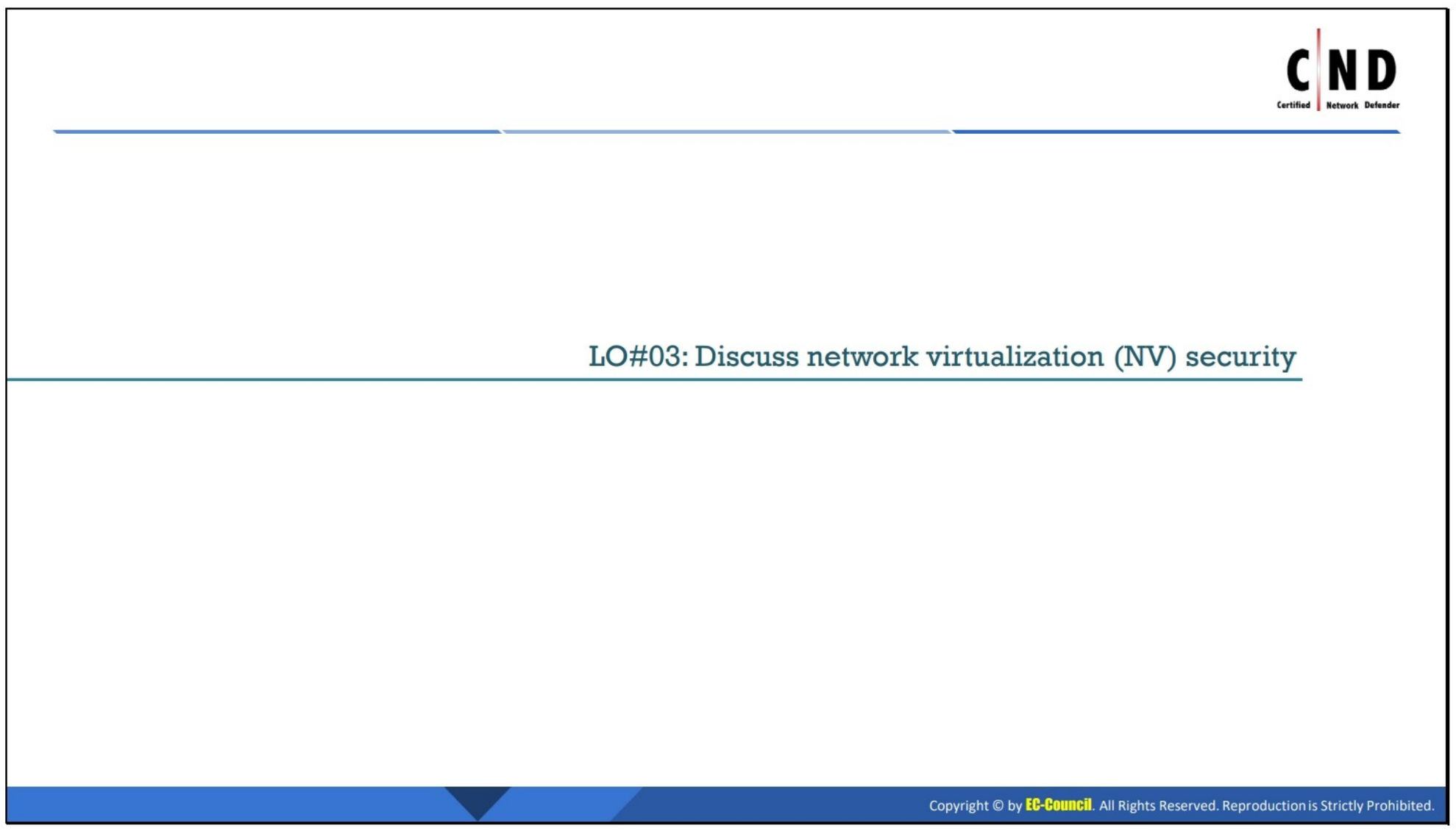


Virtualization Enablers

Some examples of technologies by which virtualization can be realized are network virtualization (NV), software-defined network (SDN), and network function virtualization (NFV). These technologies are key enablers responsible for creating virtual environments. They help in creating logical and virtual networks that are decoupled from the underlying network hardware, and these virtual networks can be integrated with virtual environments. The virtual networks can run independently over a physical network in a hypervisor. Software-defined network (SDN) and network functions virtualization (NFV) are responsible for decoupling control and forwarding planes. These technologies combine hardware and software to create a completely software-defined network that enables simpler provisioning and management of network resources and play key role in virtualization.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



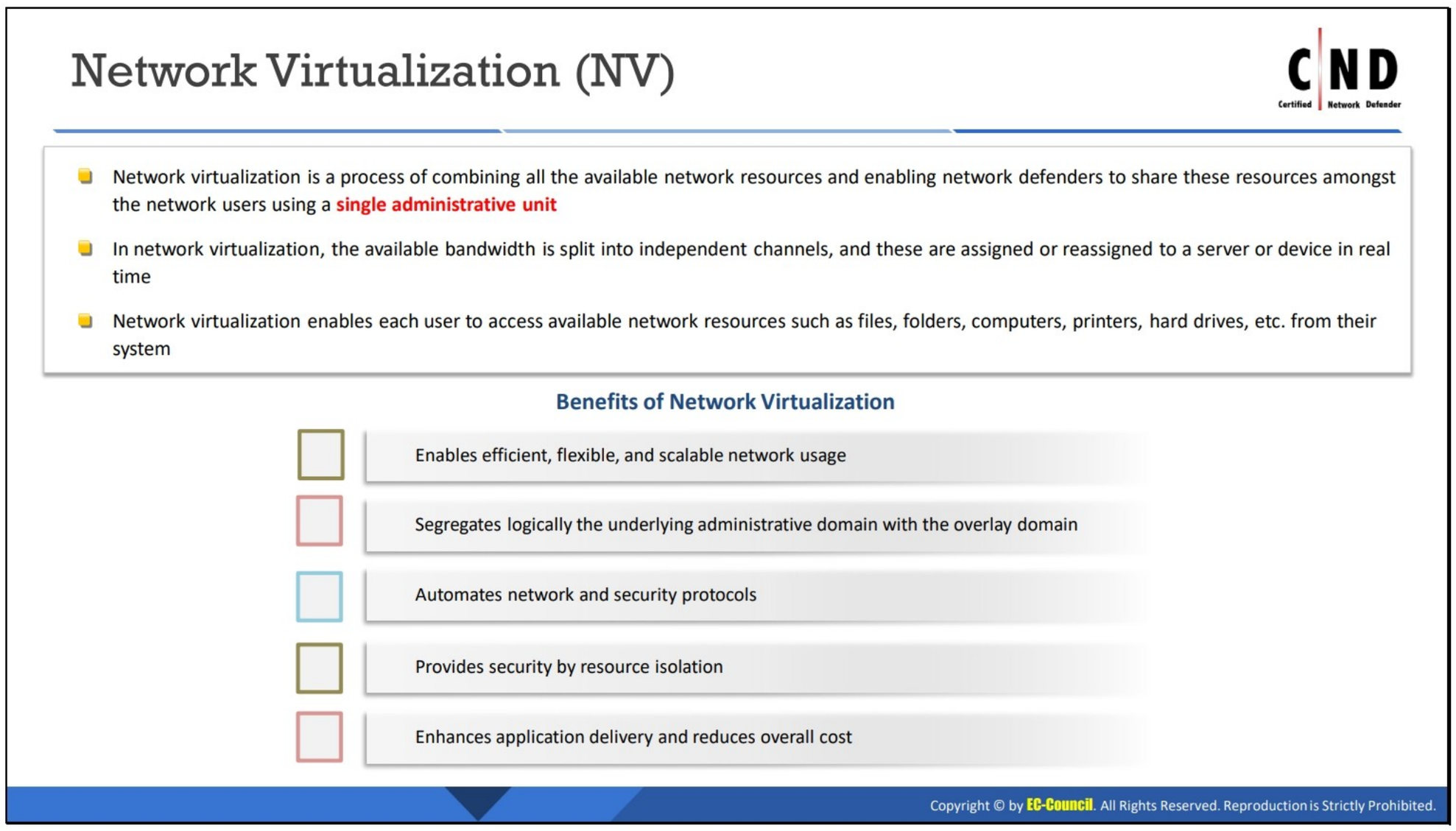
LO#03: Network Virtualization (NV) Security

The objective of this section is to introduce network virtualization technologies, the vulnerabilities and attacks associated with them, and the security measures that can be adopted to protect them.

First, the section first introduces the concept of network virtualization. Virtual networks, which are the end result of network virtualization, and the two kinds of virtual networks, are discussed. Second, with this foundation, the vulnerabilities of hypervisors and virtual networks, and potential attacks on virtual LANs (VLANs) are explained. Finally, the security of hypervisors, virtual networks, and VLANs is discussed, with numerous examples and best practices.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Network Virtualization (NV)

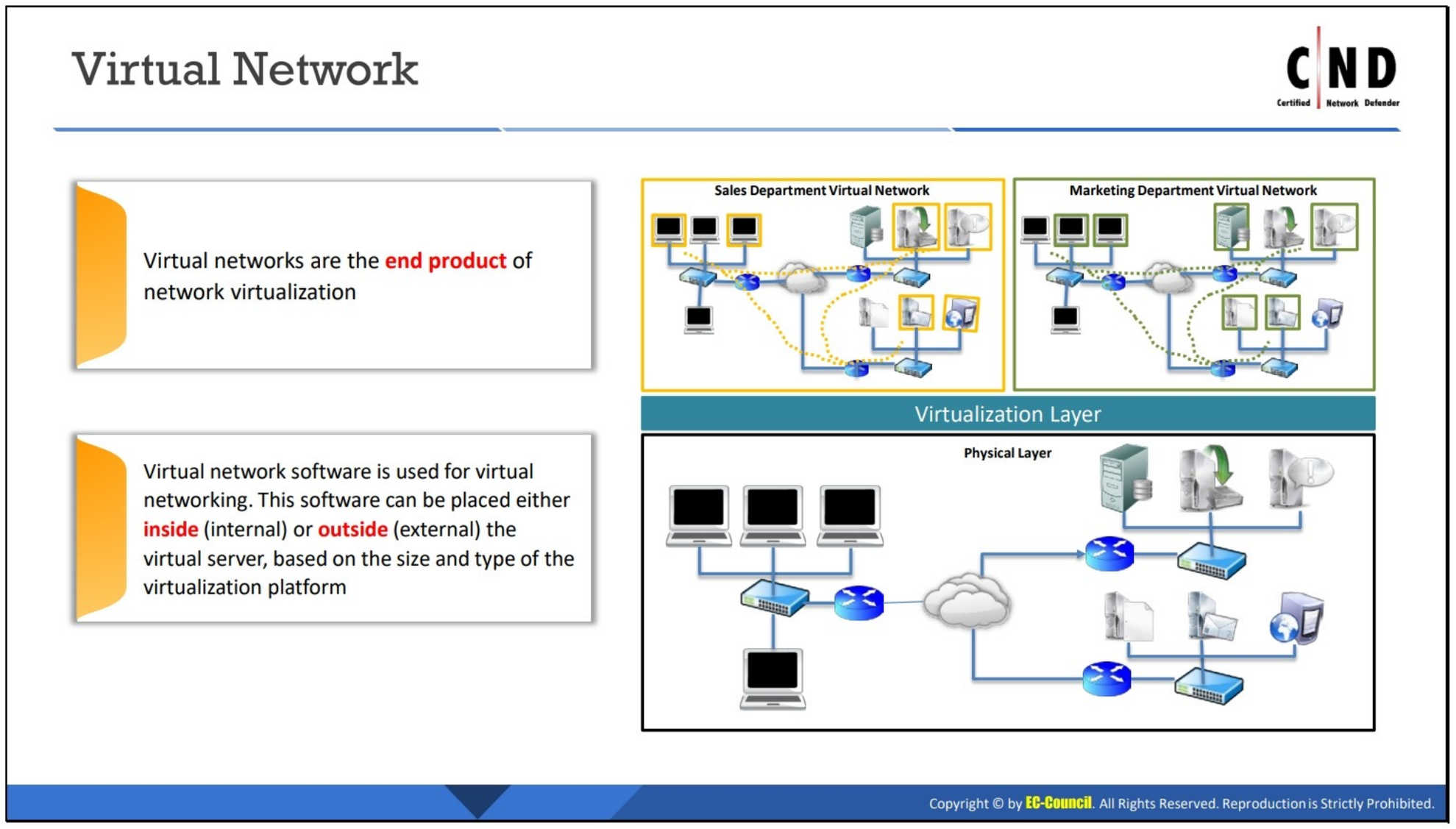
Network Virtualization (NV) is a process of combining all available network resources and enabling network defenders to share these resources amongst the network users using a single administrative unit. It abstracts network resources traditionally allocated as actual hardware to software. NV can combine multiple physical networks into one virtual, software-based network, or divide one physical network into separate, independent virtual networks. NV provides systems and users with efficient, controlled, and secured sharing of network resources (files, folders, computers, printers, hard drives, etc.). NV splits the available bandwidth into independent channels, which can be assigned or reassigned to a particular server or device in real-time. For example, a virtual LAN (VLAN) can unite network devices into one unit irrespective of their physical location, thereby enabling the creation of a subsection of the local area network (LAN).

The following are the key advantages of network virtualization:

- It enables efficient, flexible, and scalable usage of the network.
- It logically segregates the underlay administrative domain with the overlay domain.
- It accommodates the dynamic nature of server virtualization.
- It provides security and isolation of traffic and network details from one user to another.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Virtual Network

A virtual network is a software-based network that enables the consolidation of virtualized network resources into a single administrative unit. It is the end product of network virtualization. It enables domains to use a single network interface to access a physical network and eases interactions with remote systems. It is usually configured through a virtual switch to which several virtual network devices are connected. Each virtual network in a network virtualization environment (NVE) is a collection of virtual nodes and virtual links. The virtual networks are subsets of underlying physical network resources. The main purpose of a virtual network is to provide a well-organized networking structure for the applications hosted by the data center or service provider network. It also offers a controlled and secure sharing of the networking resources between systems and users. Examples of virtual networks include virtual LAN (VLAN), virtual service network (VSN), virtual private network (VPN), active and programmable networks, and overlay networks.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

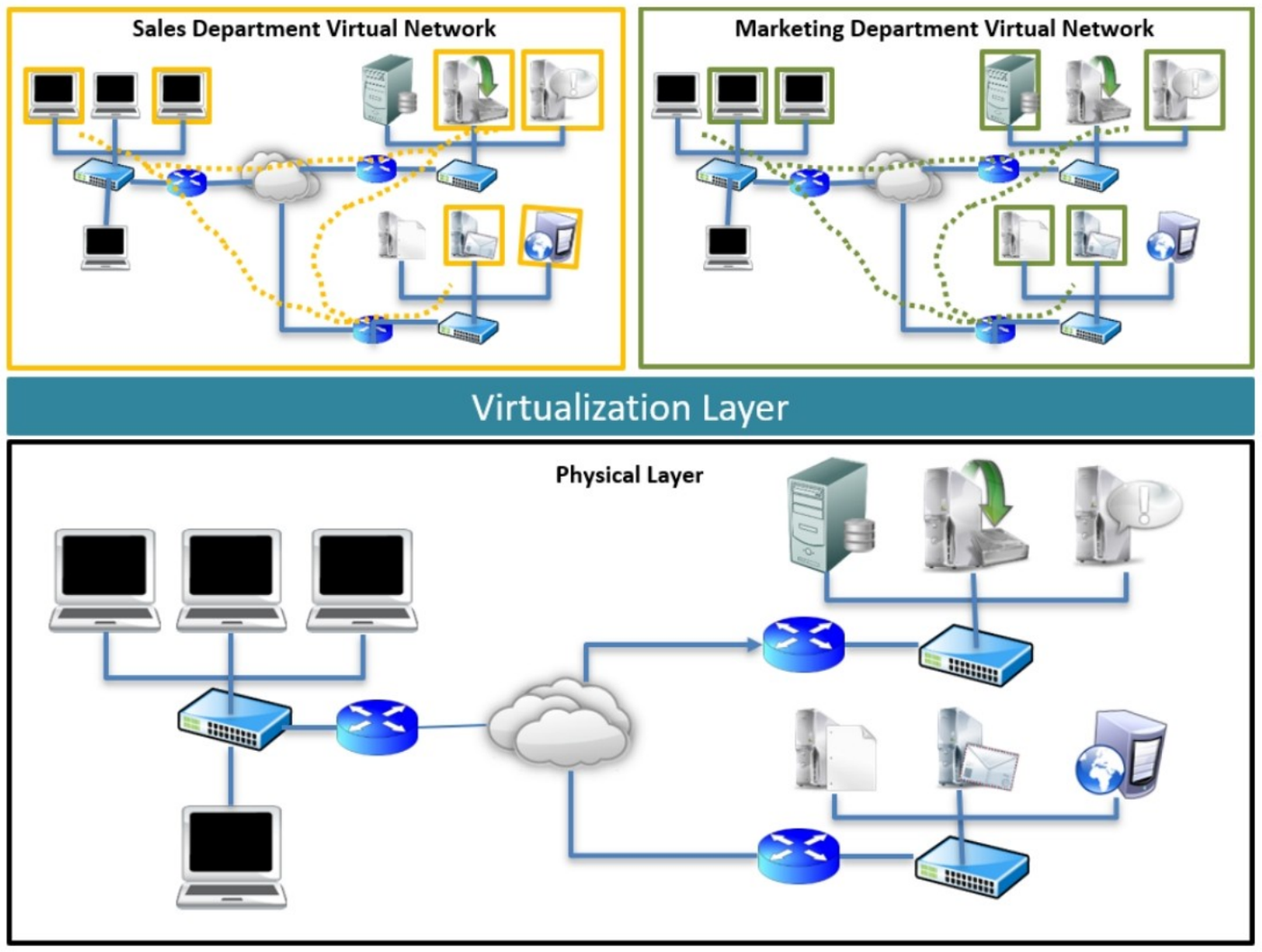


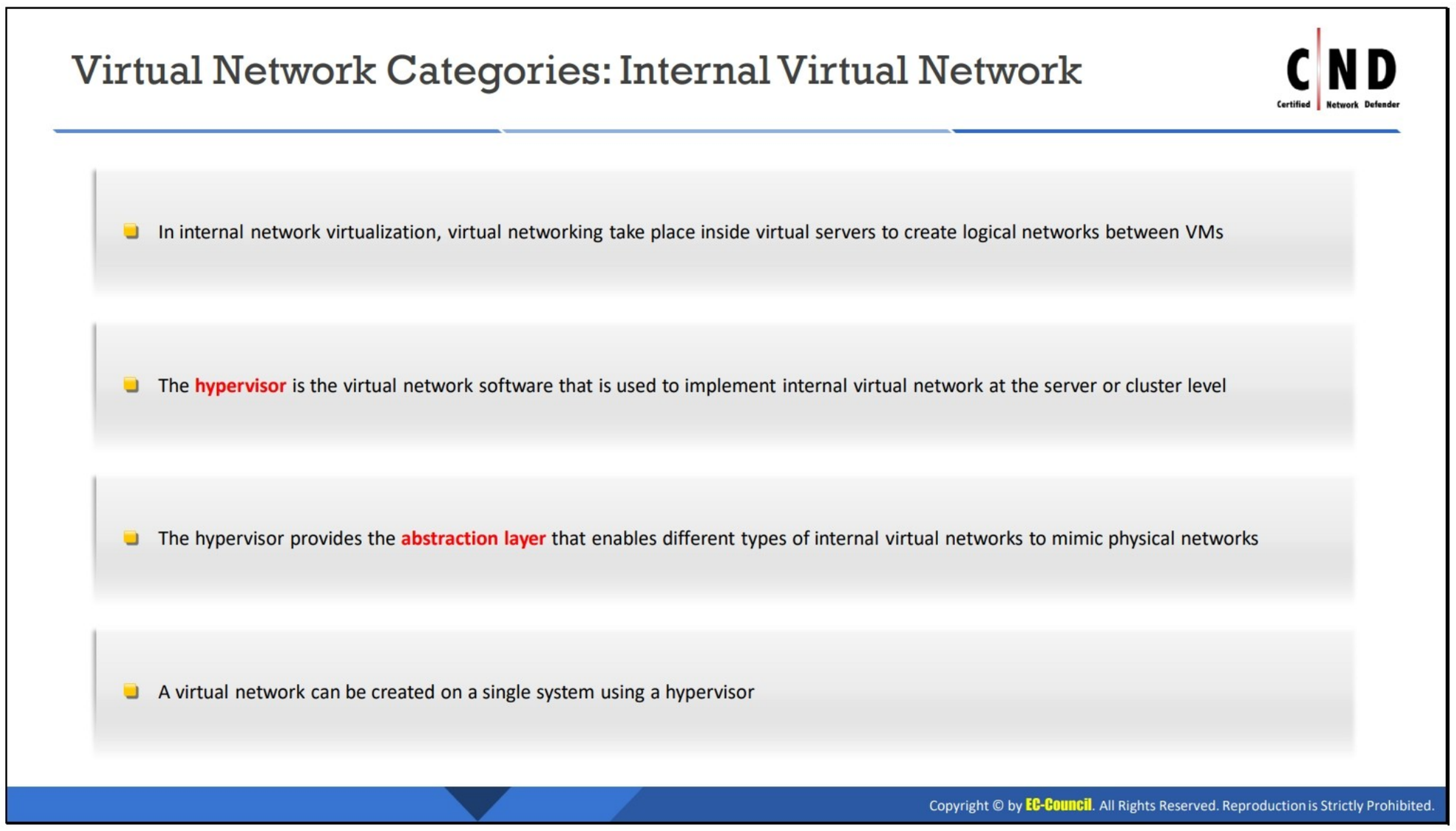
Figure 11.3: Virtual Network

Depending on the size and type of the virtualization platform, the Virtual Network software can be placed in any of the below given two ways or Virtual Networks are categorized into,

- **Outside a virtual server or External (External Virtual Networks)**
- **Inside a virtual server or Internal (Internal Virtual Networks)**

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



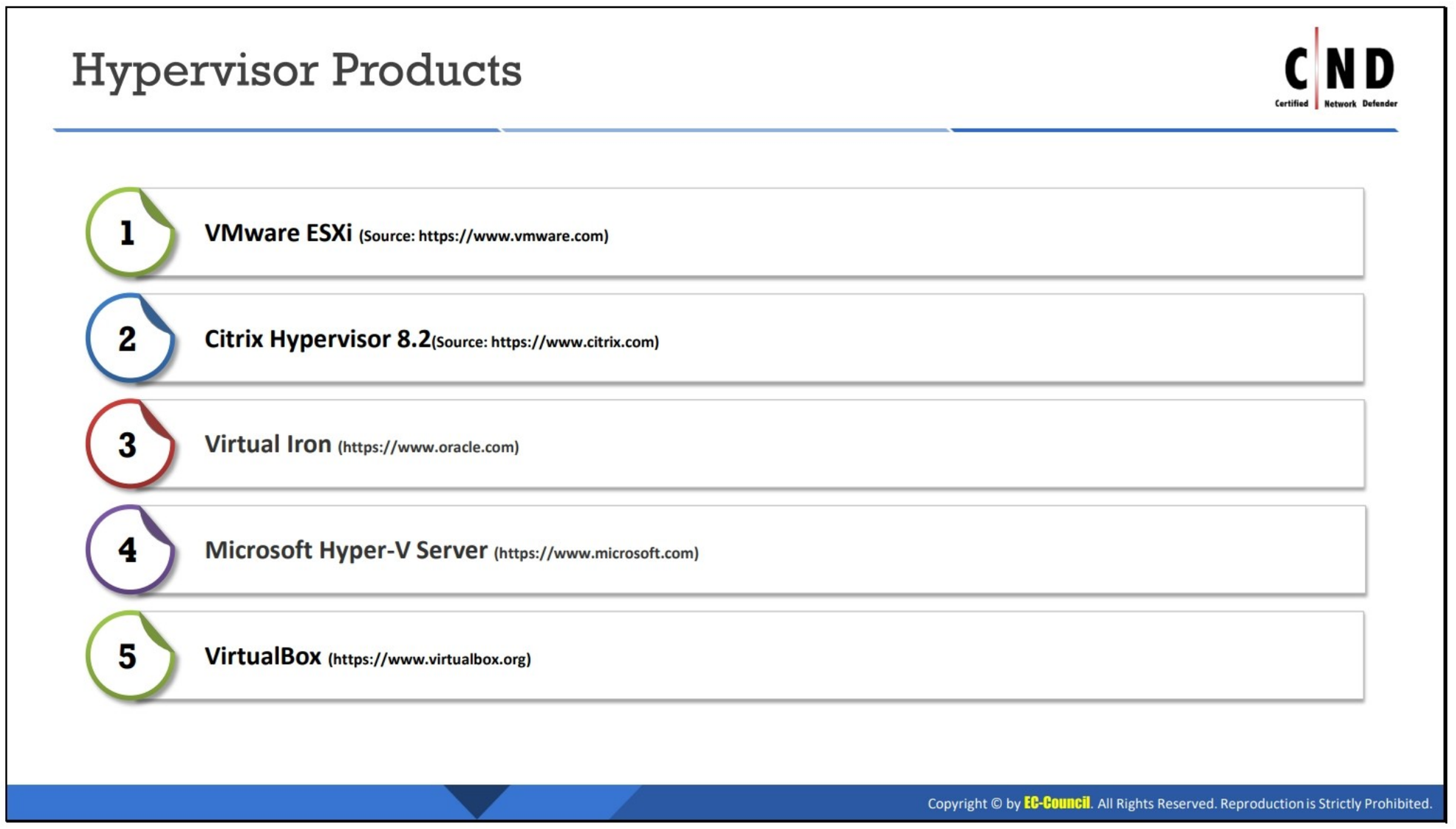
Virtual Network Categories: Internal Virtual Network

In internal network virtualization, networking functionality is provided for communication between VMs that operate on the same system. This creates a logical network, based purely on software, between the VMs in the system. Specifically, in internal network virtualization, a single system is configured with containers (for example, domains) clubbed with hypervisor control programs or pseudo-interfaces (for example, virtual network interface cards (VNICs)) to create a network in a box.

Here, the hypervisor can act as a virtual network, implement internal network virtualization at the server or cluster lever, and enable the creation of a virtual network on a single system. The hypervisor provides the abstraction layer that enables different types of internal virtual networks to mimic physical networks. An internal virtual network can enhance a single system's efficiency by isolating applications to separate containers or pseudo-interfaces. Virtual network software can emulate network connectivity within the server and enable VMs hosted on that server to exchange data. Internal network virtualization is offered by various software vendors.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Hypervisor Products

A hypervisor is software that runs and manages virtual machines. Many popular hypervisors, provided by various vendors, exist. They include,

- **VMware ESXi**

  Source: www.vmware.com

  VMware ESXi effectively partitions hardware to consolidate applications and cut costs with direct access to and control of underlying resources. It installs directly onto a physical server. VMware ESXi enables,

  - Consolidation of hardware for higher capacity utilization.
  - Enhanced performance for a competitive edge.
  - Streamlining of IT administration through centralized management.
  - Reduced capital expenditure (CapEx) and operating expense (OpEx).
  - Minimization of hardware resources needed to run the hypervisor, resulting in greater efficiency.

- **Citrix Hypervisor 8.2 (formerly XenServer)**

  Source: www.citrix.com

  The Citrix Hypervisor virtualization management platform is optimized for application, desktop, and server virtualization infrastructures. It enables organizations of any vertical or size to transform their business IT computational infrastructure.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

- **Virtual Iron**

  Source: www.virtualiron.fr

  Virtual Iron provides enterprise-class software for server virtualization and virtual infrastructure management. With advanced capabilities provided by Virtual Iron, users can:

  - Virtualize enterprise-class workloads running on unmodified Windows and Linux operating systems.
  - Improve the utilization of current systems and reduce power, space, and cooling issues through server consolidation.
  - Quickly set up development, test, and production environments.
  - Recover from failures quickly, reliably, and cost-efficiently.
  - Match resource capacity to workload demands automatically.
  - Reduce human labor and errors via policy-based automation.

- **Microsoft Hyper-V Server**

  Source: www.microsoft.com

  Hyper-V in Windows Server enables the creation of a virtualized computing environment to create and manage virtual machines. Multiple operating systems can be run on one physical computer and the operating systems can be isolated from each other.

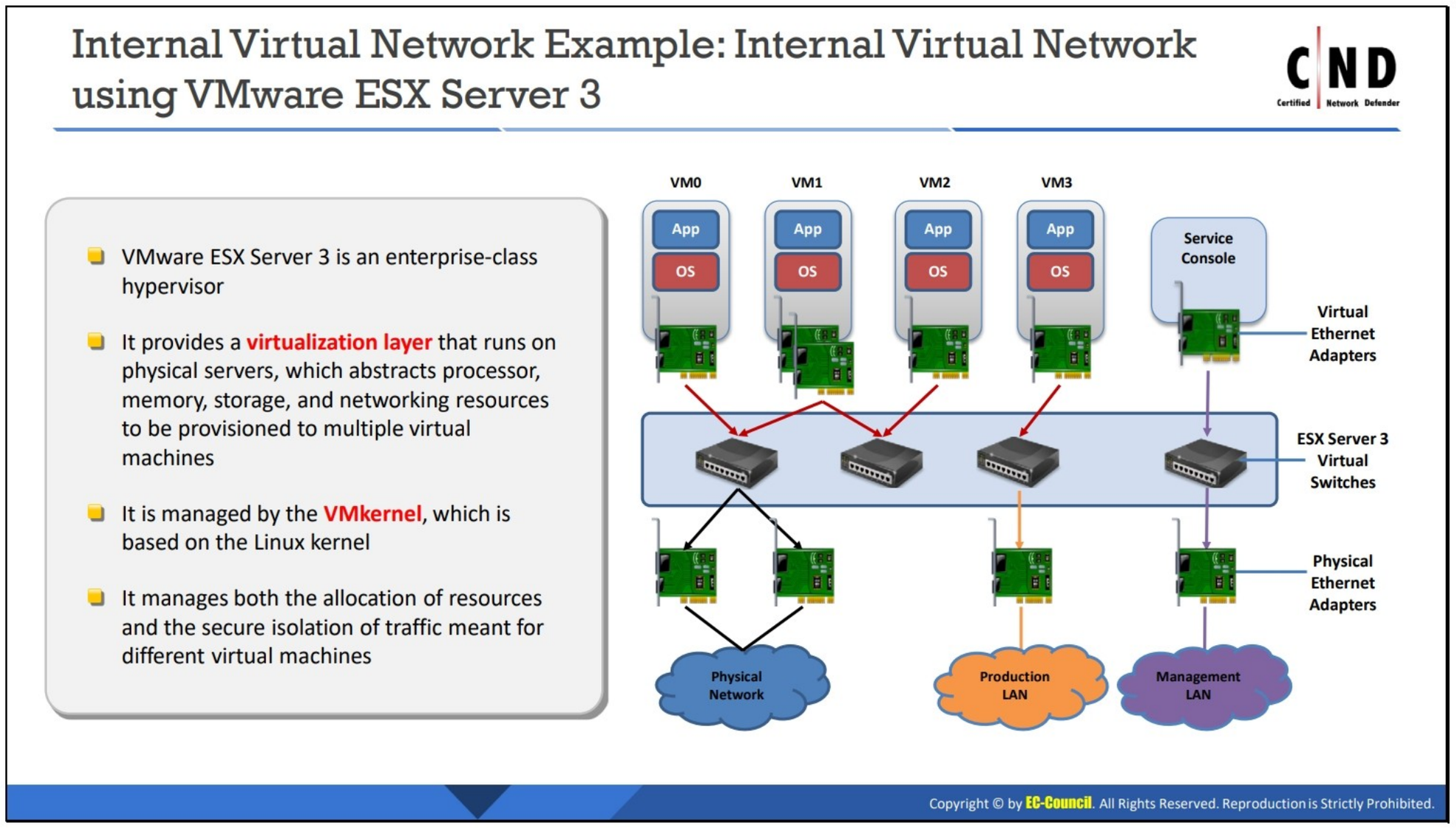
- **VirtualBox**

  Source: https://www.virtualbox.org/

  VirtualBox is a powerful x86 and AMD64/Intel64 virtualization product for enterprise as well as home use. VirtualBox is a feature-rich, high-performance product for enterprise customers, and is the only professional solution that is freely available as Open Source Software under the terms of the GNU General Public License (GPL) version 2.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Internal Virtual Network Example: Internal Virtual Network using VMware ESX Server 3

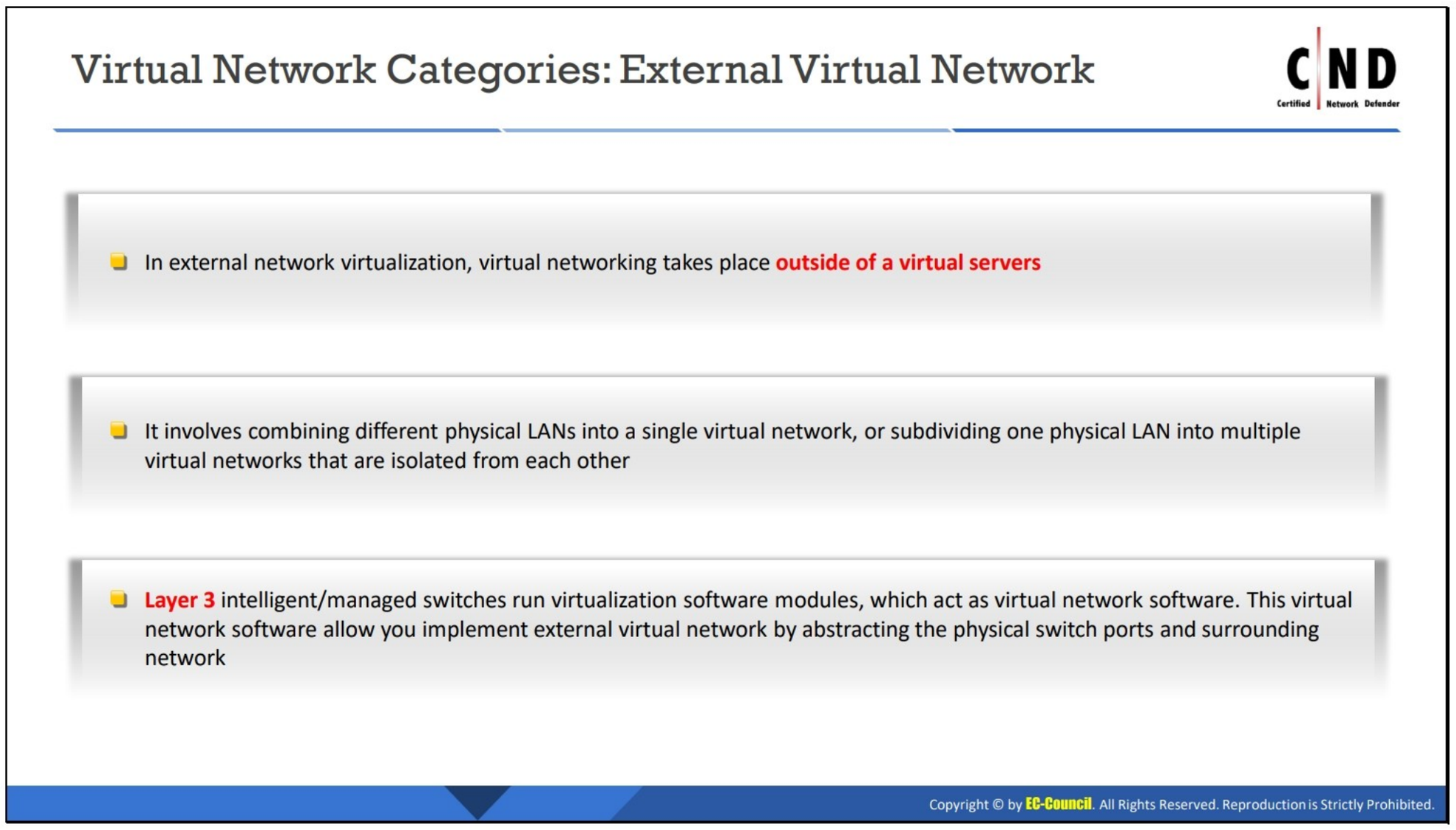
VMware Elastic Sky X (ESX), (currently ESX Integrated (ESXi)) is a production-proven, enterprise-class type-1 hypervisor, which is used for deploying and serving virtual computers. ESX is the key component in the VMware Infrastructure software suite, which replaces Service Console with a rudimentary OS. As it is type-1 hypervisor, ESX includes and integrates key OS components such as a Linux kernel (vmkernel/virtualization layer), and runs on bare metal (i.e., directly on a physical machine). VMware ESX Server abstracts processor, memory, storage, and networking resources to be provisioned to multiple virtual machines.

Initially, the vmkernel is started, and then it is used to load virtualization components including ESX. The service console invokes the Linux kernel – the primary virtual machine. During normal run-times, the vmkernel runs on the bare computer and the service console runs as the first virtual machine.

The below diagram illustrates internal network virtualization using VMware ESX as an example. In this diagram, the physical network cards connect to a layer of ESX server 3, which hosts the virtual networking split between various virtual machines. Virtual switches in ESX Server 3 connect virtual machines and the service console to each other and to external networks.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Virtual Network Categories: External Virtual Network

External network virtualization involves combining multiple physical LANs into a single virtual network or subdividing a physical LAN into multiple virtual networks that are isolated from each other, to increase the efficiency of a corporate network or data center. External network virtualization requires external virtual network software and involves network resources. For example, using VLAN and switch technology, systems that are physically attached to the same local network can be configured into the different virtual networks. VLAN also enables the combining of systems on separate local networks into a single VLAN spanning the segments of a corporate network.

The realization of external network virtualization requires that the routers and switches support virtualization while connecting multiple systems. Here, the managed/intelligent switches (layer 3 switches) help to run the virtualization software modules that abstract the physical switch ports and the surrounding network. This relationship between hardware and software enables the realization of external network virtualization. For example, a layer 3 intelligent switch runs virtualization software, while hardware manufacturers work on embedding their technology into hypervisor network topologies.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Layer 3 Intelligent/Managed Switches Vendors

The following are examples of layer 3 intelligent/managed switches that can run virtualization software.

CISCO

Source: https://www.cisco.com/

CISCO managed switches provide powerful performance at affordable prices. They are easy to install and use, provide an ideal combination of affordability and capabilities for small businesses (SMBs), and help create an efficient, better-connected workforce.

MOXA

Source: https://www.moxa.com

Moxa's layer 3 managed switches feature industrial-grade reliability, multicast availability, and security enhancements based on the IEC 62443 standard. Moxa offers toughened industry-specific products with multiple industry certifications such as parts of the EN 50155 standard for rail applications, IEC 61850-3 for power automation systems, and NEMA TS2 for intelligent transportation systems.

NETGEAR

Source: https://www.netgear.com

NETGEAR Fully Managed Switches connect end-users, critical services, servers, and storage across flexible core, distribution and access layers. NETGEAR’s Managed Infrastructure combines the latest advances in hardware and software engineering for higher flexibility, lower complexity

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

and stronger investment protection. NMS300, a single-pane-of-glass management platform, increases overall operational efficiency.

LINKSYS

Source: https://www.linksys.com

LINKSYS layer 3 switches come with features like advanced network management capabilities, advanced traffic-handling intelligence, advanced network security features, and fiber-optic network expansion support.

tp-link

Source: https://www.tp-link.com

TP-Link's L3 managed switches provide common features like L3 routing, physical stacking, and optional redundant external power unit. They are a reliable and cost-effective solution for enterprises and ISP networks.

Buffalo

Source: https://www.buffalotech.com

Buffalo's smart switches offer an easy and cost-effective way to extend a business network at gigabit speeds with advanced management features. Their common features include plug and play installation, no additional software or configuration, and auto-sensing gigabit ports.

HP

Source: https://www.hpe.com

HP layer3 switches deliver resiliency, security, and reliability at the enterprise edge with support for advanced layer 3 and multicast routing. The advanced layer 3 fast ethernet switch series has a robust feature set supporting IRF stacking, static rip routing, OSPF, BGP, IS-IS, PoE+, ACLs, and IPv6. Rhe HP 3600 Ei Switch Series can be managed with the optional HP intelligent management center (IMC) software platform for a single view of the entire network.

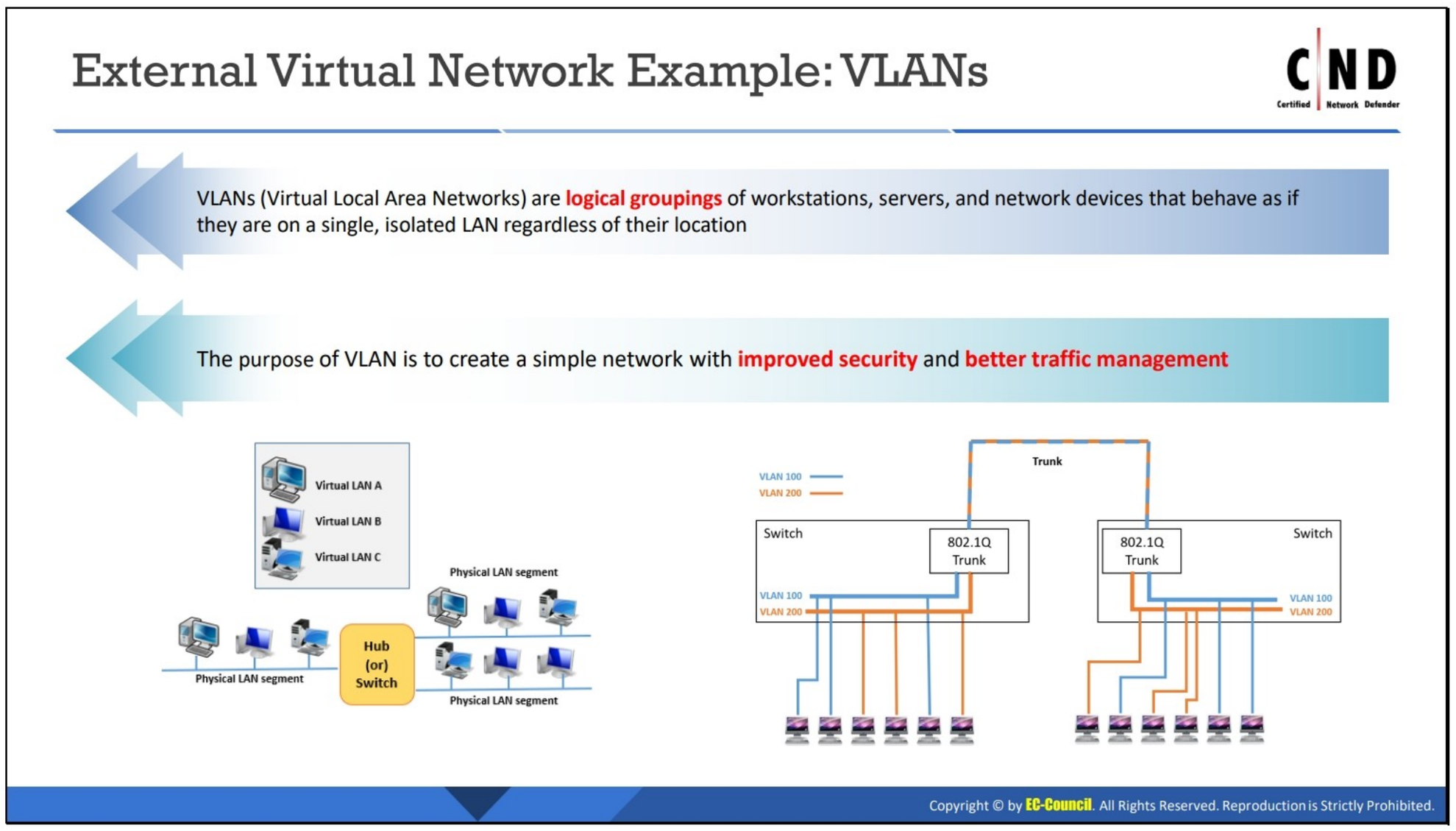
D-Link

Source: http://www.dlink.com

Dlink layer 3 switches are an ideal solution for large IP routed networks and may be applied at the network core. They support dynamic routing functions, advanced quality of service, security, are often stackable, and may have 10 gigabit uplinks.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



External Virtual Network Example: VLANs

Virtual LANs (VLANs) are logical groupings of workstations, servers, and network devices that behave as if they were on a single, isolated LAN. The purpose of VLAN is to create a simple network with improved security and better traffic management. VLANs are implemented in port switching hubs and LAN switches, and function at layer 2. As a VLAN isolates traffic within itself, a router is needed to bridge VLANs. The router works at the higher layer 3 network protocol, which requires identifying and coordinating network layer segments with the VLANs. This is a nontrivial task and VLANs tend to break down as the networks expand and the number of routers encountered increases. Solutions for "virtual routing" enable corporate network managers to view the entire network as a single routed entity.

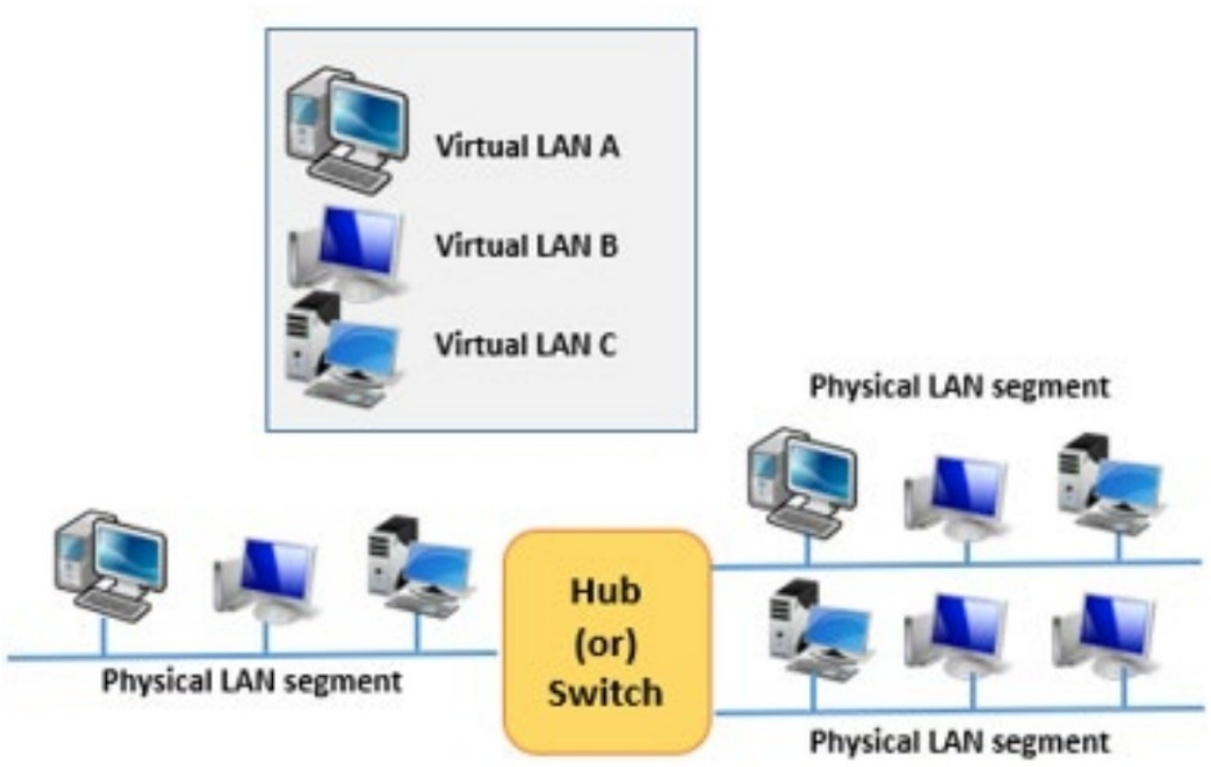


Figure 11.4: VLAN

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

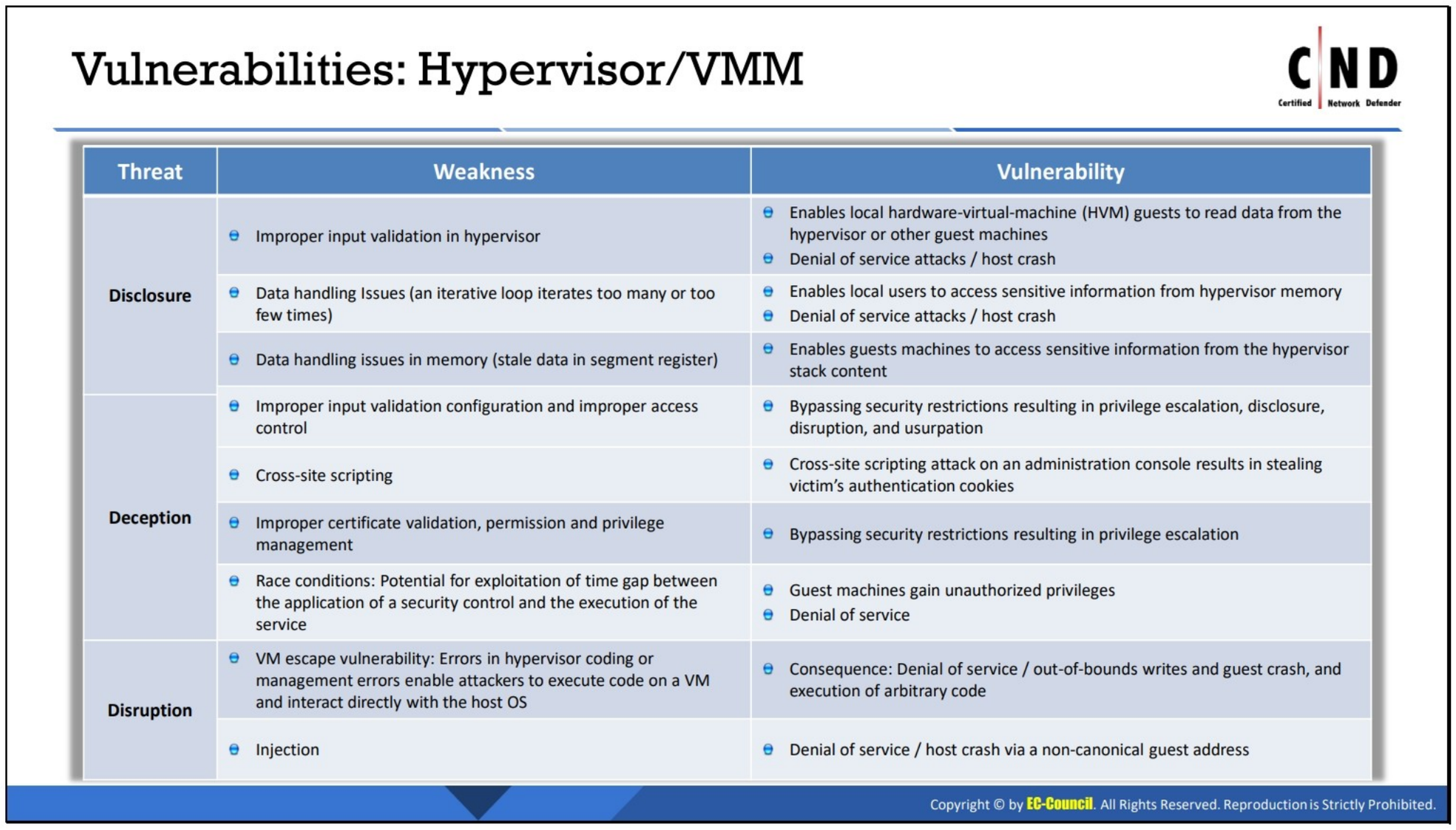
https://t.me/learningnets

A VLAN operates like a physical LAN, but can group together hosts that are not connected to the same switch. Besides this, there are several other advantages associated with using VLANs in the network:

- The size of a VLAN can be decreased by increasing the number of broadcast domains.
- VLAN improves security by restricting switching of frames broadcasted onto the network.
- Hosts that store sensitive information can be placed on a separate VLAN to enhance security.
- VLAN facilitates the creation of flexible network designs.
- Network changes can be easily effected by locating a port into the appropriate VLAN.
- VLAN minimizes latency and traffic load on the network and maximizes performance.
- VLAN enables segmentation and supports network management and scalability.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Vulnerabilities: Hypervisor/VMM

As the hypervisor is a fundamental component of virtualized systems, it is frequently targeted in attacks. Consequently, there are a number of vulnerabilities associated with the security features of hypervisors such as VM isolation and internal software-based channels for communication with VMs. There are common weaknesses and vulnerabilities that affect hypervisors, VMMs, and their management tools. Below are vulnerabilities of hypervisors classified according to the potential threat and the weakness involved with the vulnerability.

- **Disclosure**
  - Improper input validation in hypervisor

    This enables local hardware virtual machine (HVM) guests to read data from the hypervisor or other guest machines, and can also lead to a denial of service (DoS) or the crashing of the host. For example, in the Xen hypervisor, improper input validation can result when the intercept function in a software library uses an improper range.
  - Data handling issues due to off-by-one error (an iterative loop iterates too many or too few times) in a software function

    This enables local users to obtain sensitive information from hypervisor memory, and can also lead to a DoS or crashing of the host.
  - Data handling in memory (data-handling issues due to stale data in a segment register)

    This enables local guests to obtain critical data from the hypervisor stack content. For example, the guest OS users can change the address used in memory mappings and read from or write to arbitrary memory.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

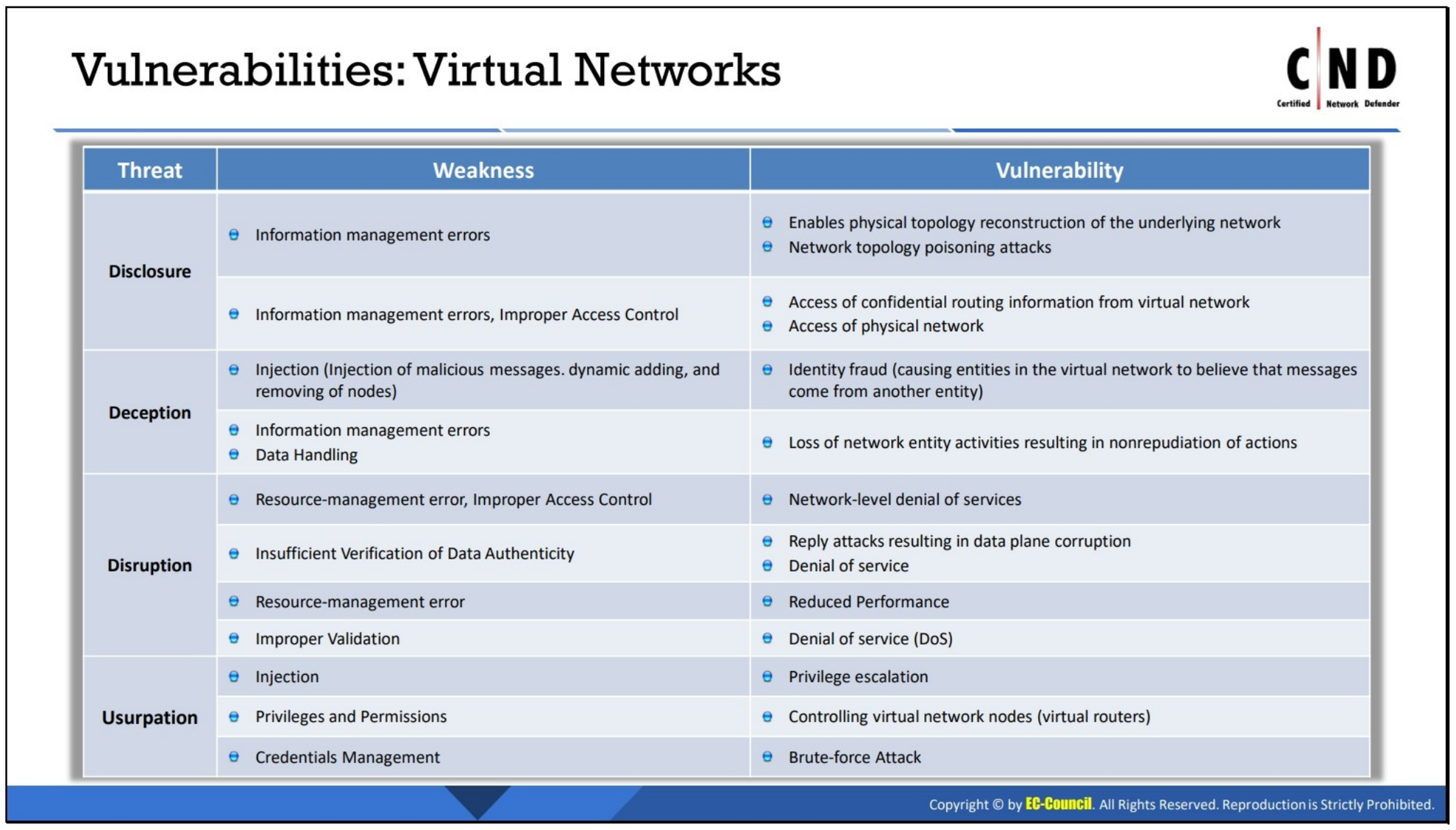
- **Deception**
  - Improper input validation, configuration, and access control.
  - Cross site scripting
  - Improper certificate validation, permission, and privilege management
  - Race conditions exploit time gap between the application of a security control and the execution of the service
- **Disruption**
  - Improper input validation, resource management errors

    This causes VM escape, in which attackers can run code on a VM to directly communicate with the hypervisor, by exploiting hypervisor coding or management errors. VM escape can enable the attackers to exploit a guest OS to cause DoS, out-of-bounds writes, crash the guest, and run arbitrary code.
  - Injection

    Injection in hypervisor software libraries enables local guest users to cause DoS and crash the host via a non-canonical guest address.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Vulnerabilities: Virtual Networks

The following are some of the threats, weaknesses, and vulnerabilities found in virtual networks:

- **Disclosure**
  - Information-management errors

    The uncontrolled handling of sequential requests for virtual networks enables reconstructing the physical topology of the underlying network and network topology poisoning attacks.
  - Information-management errors, improper access control

    The inspection of virtual network traffic or accessing a virtual router can enable attackers to obtain critical routing information from the virtual network.
- **Deception**
  - Injection

    Malicious messages may be injected while causing other entities in the virtual network to believe that the messages originate from a legitimate entity in the network. This can result in identity fraud.
  - Information management errors, data handling

    The rollback of networking activity logs stored in a VM can result in the loss of network entity activities, and subsequently impact the non-repudiation of actions.
- **Disruption**
  - Resource-management error, improper access control

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

Software-controlled latency over virtualized networks may result in DoS at the network-level.

- Insufficient verification of data authenticity

  Misbehaving virtual routers may send old control messages repeatedly (reply attacks), resulting in corruption of the data plane and DoS.

- Resource management error
  This causes degradation of performance due to uncontrolled allocation of resources of different virtual networks on the same substrate as physical network.

- Improper Validation

  This causes DoS due to the incorrect throwing of exceptions when handling malformed, truncated, or maliciously crafted packets.

- **Usurpation**

  - Injection

    The injection of malicious messages from a fake source with high privileges can result in privilege escalation.

  - Privileges and permissions

    The improper handling of identities and associated privileges enable the controlling of virtual network nodes like virtual routers.

  - Credentials management

    The access of network management console using brute-force password guessing enables attacks on the network.

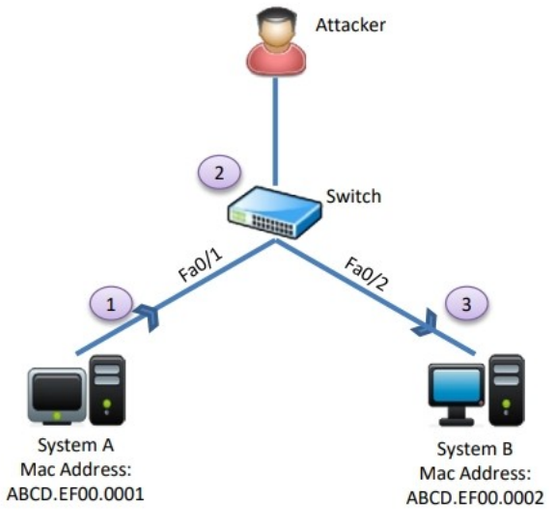
Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

VLAN Attacks

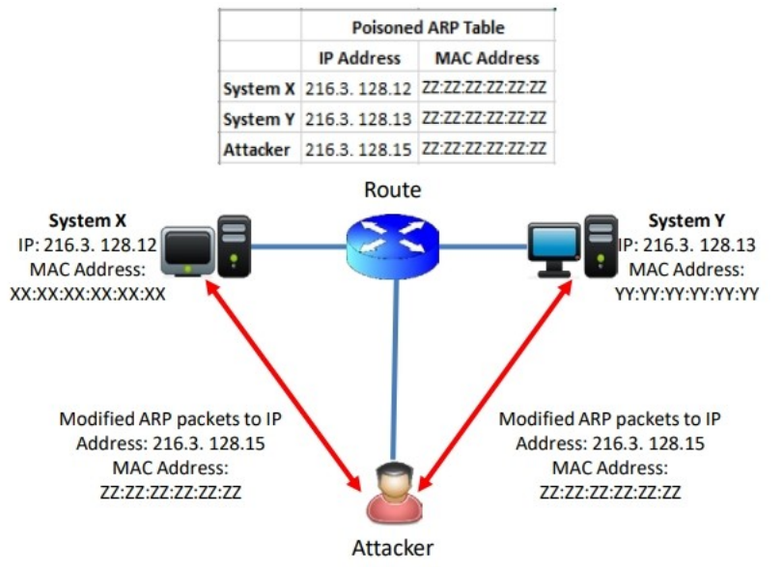
MAC Flooding Attack

- Large number of **fake MAC addresses** are sent by the attacker to overflow the **CAM table**. Once the table runs out of space, traffic without MAC entries floods out to all the ports of VLAN. When the traffic floods out, attacker can more easily view and retrieve the generated traffic



ARP Attack

- To associate the attacker's MAC address with the IP address of a legitimate computer or server on the network, the attacker sends **fake ARP** messages over LAN

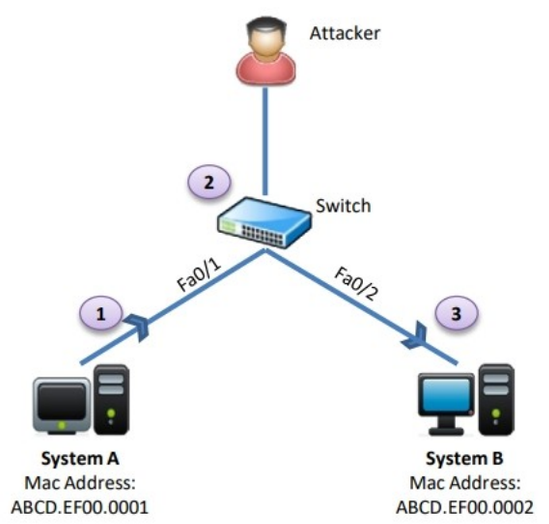


Copyright © by EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited.

VLAN Attacks (Cont'd)

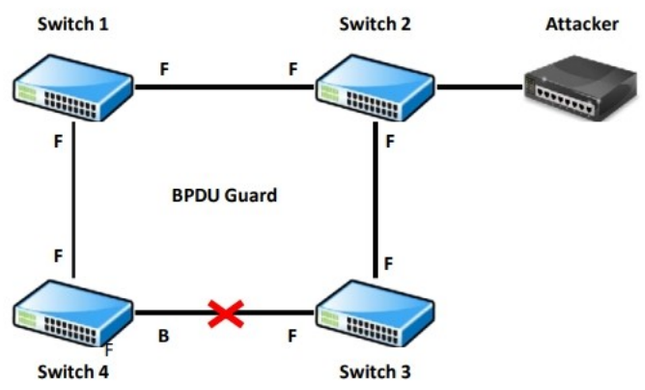
VLAN Hopping Attack

- The attacker transmits traffic to other VLANs or ports that are not normally accessible from a given end system. This is mainly conducted in the **Dynamic Trunking Protocol**



STP Attack

- Installing a new **STP** device in the network and sending **junk data traffic** through it causes flooding of data packets, which leads to the shutdown of services for a short period of time

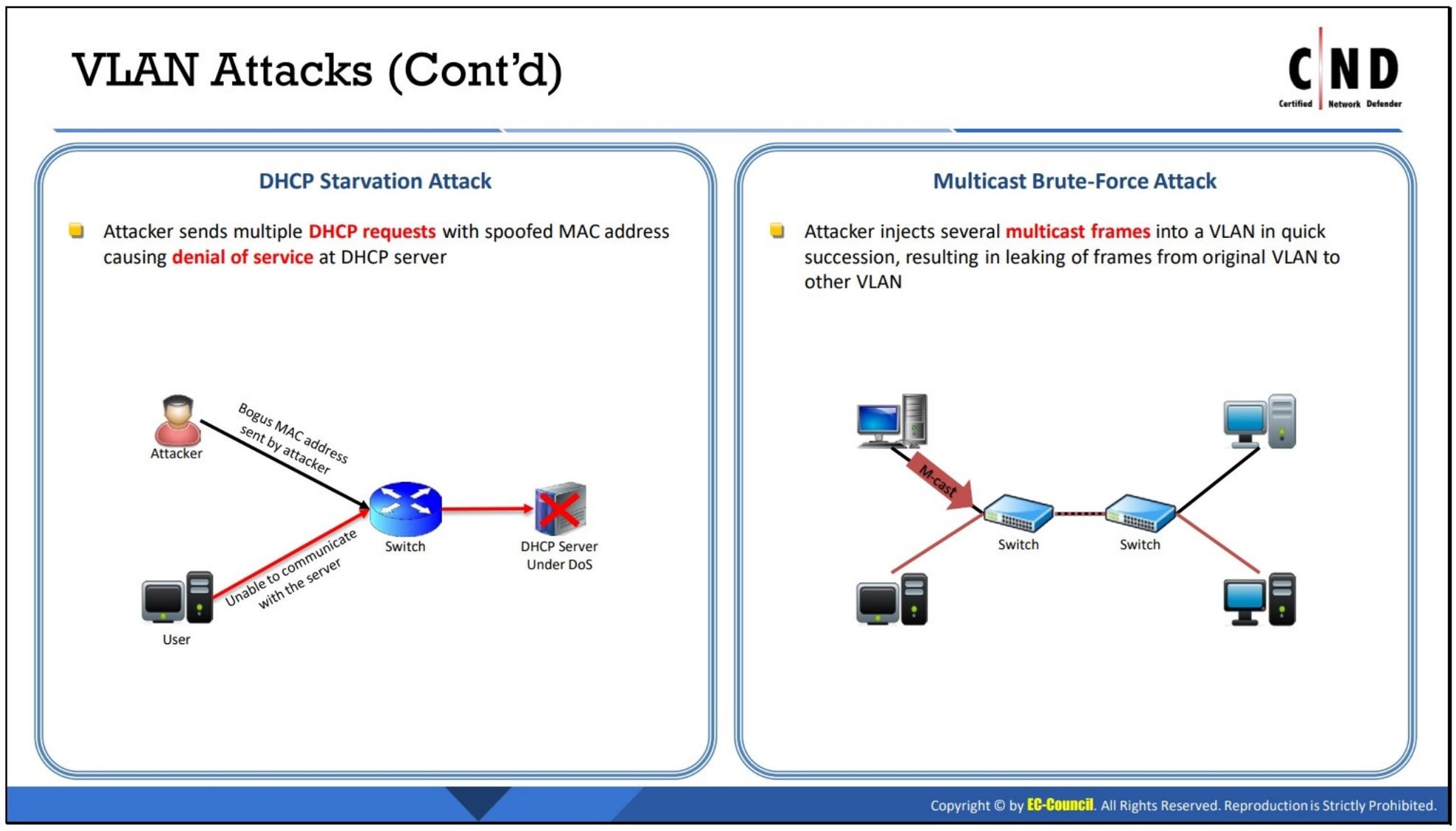


Copyright © by EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24



VLAN Attacks

VLAN switches are not equipped with a mechanism to detect an attack. Therefore, most layer 2 attacks target the incapability of the switch to track the attacker. Common VLAN attacks are as follows:

MAC Flooding Attack: A large number of fake MAC addresses are sent by the attacker to overflow the content addressable memory (CAM) table. When the table runs out of space, traffic without MAC entries flood all the ports of the VLAN. This makes it easy for the attacker to view and retrieve the generated traffic.

ARP Attack: To associate the attacker's MAC address with the IP address of a legitimate computer or server on the network, the attacker sends fake address resolution protocol (ARP) messages over the LAN. ARP attacks are executed to trick a switch so that it forwards packets with forged identities to another device in a different VLAN. In the same VLAN, an ARP attack can trick the end nodes of the network such as routers and workstations.

VLAN Hopping Attack: The attacker transmits traffic to other VLANs or ports that are not normally accessible from a given end system. There are two types of VLAN hopping attacks.

- **Double Tags**: In this type of VLAN hopping attacks, two 802.1Q tags are sent by the attacker. The inner VLAN tag is the tag of the VLAN that the user wants to reach, and the outer VLAN tag is the tag of the native VLAN. After receiving the 2 tags, the switch removes the native VLAN and forwards the second frame to the trunk interface(s). At this point in time, the attacker jumps from native VLAN to the user VLAN. The attacker can use this strategy to conduct a DoS attack.
- **Switch Spoofing**: In this type of VLAN hopping attack, the attacker exploits the default "dynamic auto" or "dynamic desirable" switch port mode. The attacker exploits an

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

incorrectly configured trunk port to spoof itself to be a switch, and then emulates 802.1Q and DTP messages.

Spanning-Tree Attack: In this type of attack, after obtaining the port ID information, the attacker sends spanning tree protocol (STP) configuration/topology change acknowledgement bridge protocol data units (BPDUs) indicating that the attacker is the new root bridge with lower priority, and finally gains access to the network traffic. By installing and transmitting junk data through a new STP device in the network, the attacker can cause a flooding of data packets, which leads to the shutdown of services for a short period of time.

DHCP Starvation Attack: The attacker sends multiple DHCP requests with spoofed MAC addresses causing DoS at the DHCP server.

Multicast Brute-Force Attack: The attacker injects several multicast frames into a VLAN in quick succession, resulting in leaking of frames from the original VLAN to other VLANs.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

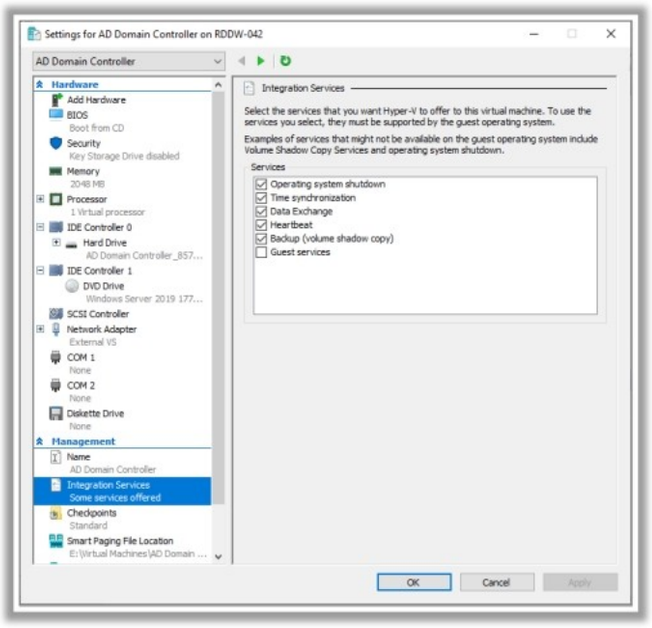
Technet24

Hypervisor Security: Hyper-V Security

C|ND Certified Network Defender

Time Synchronization

- VMs lose track of time if their date and time are not in **synchronization** with the host machine
- Saved virtual machines will have incorrect time when they are restored if their time is not **synchronized**



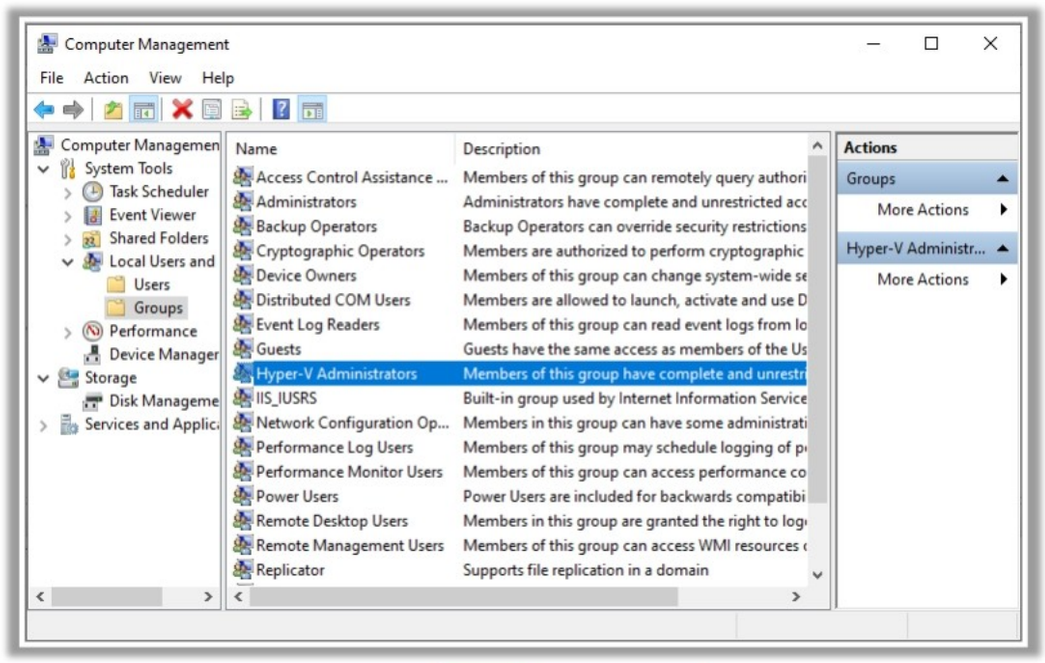
Copyright © by EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited.

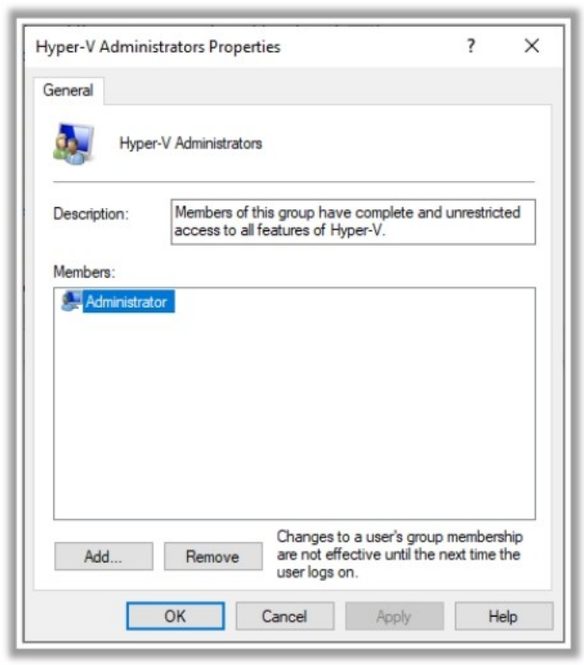
Hypervisor Security: Hyper-V Security (Cont'd)

C|ND Certified Network Defender

Set Access Privileges for Users

- By default, Hyper-V provides a **group of admins** having all administrative rights for the Virtual Machines
- Non-admin users who need access, and admins who do not need access to the hypervisor can be added or removed as per requirement





Copyright © by EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited.

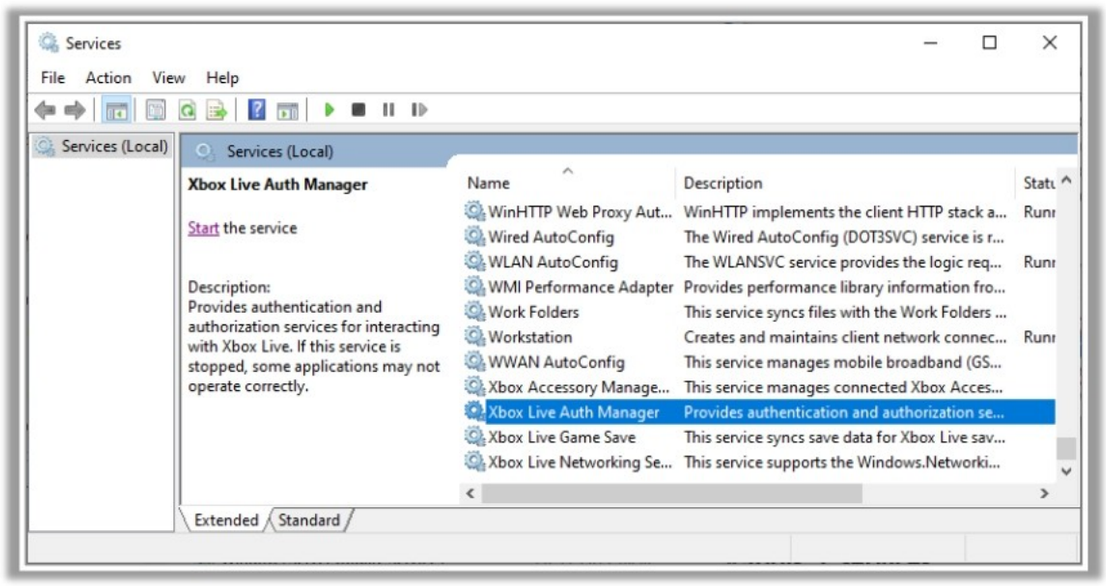
Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

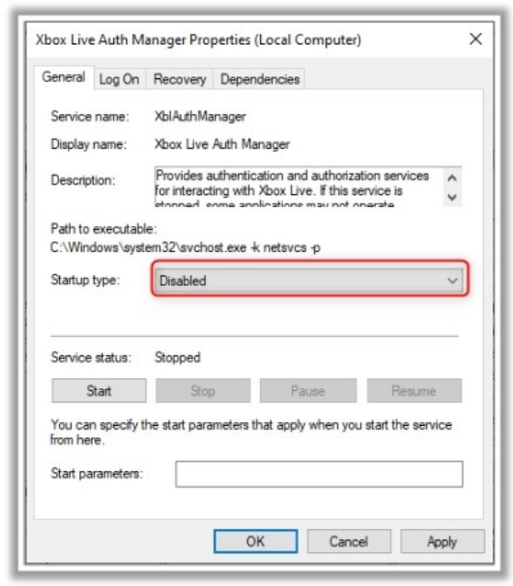
https://t.me/learningnets

Hypervisor Security: Hyper-V Security (Cont'd)

Disable Unnecessary Services

- By **default** Windows runs unnecessary services that can affect performance if not disabled
- It is recommended that the following services and their respective scheduled tasks be **disabled** on Windows Server 2016
  - Xbox Live Auth Manager
  - Xbox Live Game Save



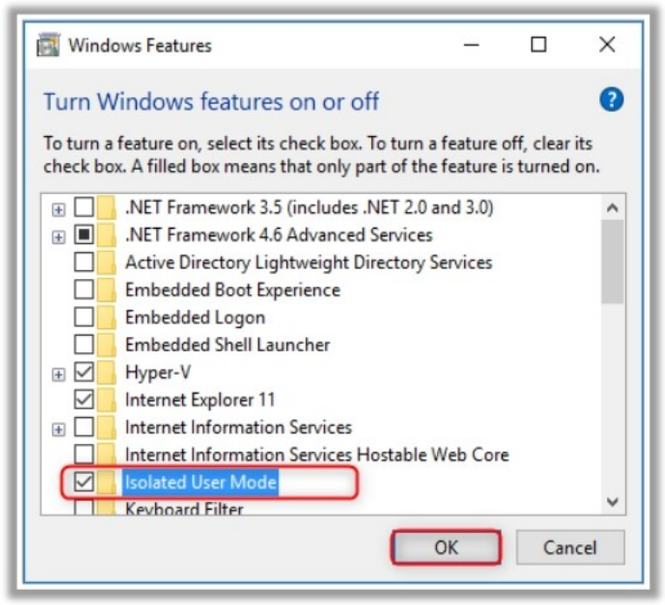


Copyright © by EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited.

Hypervisor Security: Hyper-V Security (Cont'd)

Isolated User Mode (IUM)

- Microsoft introduced a virtualization-based security feature called **Isolated User Mode (IUM)** to separate business data and the underlying processes
- This feature is introduced particularly for Windows 10 Enterprise version and Windows Server 2016

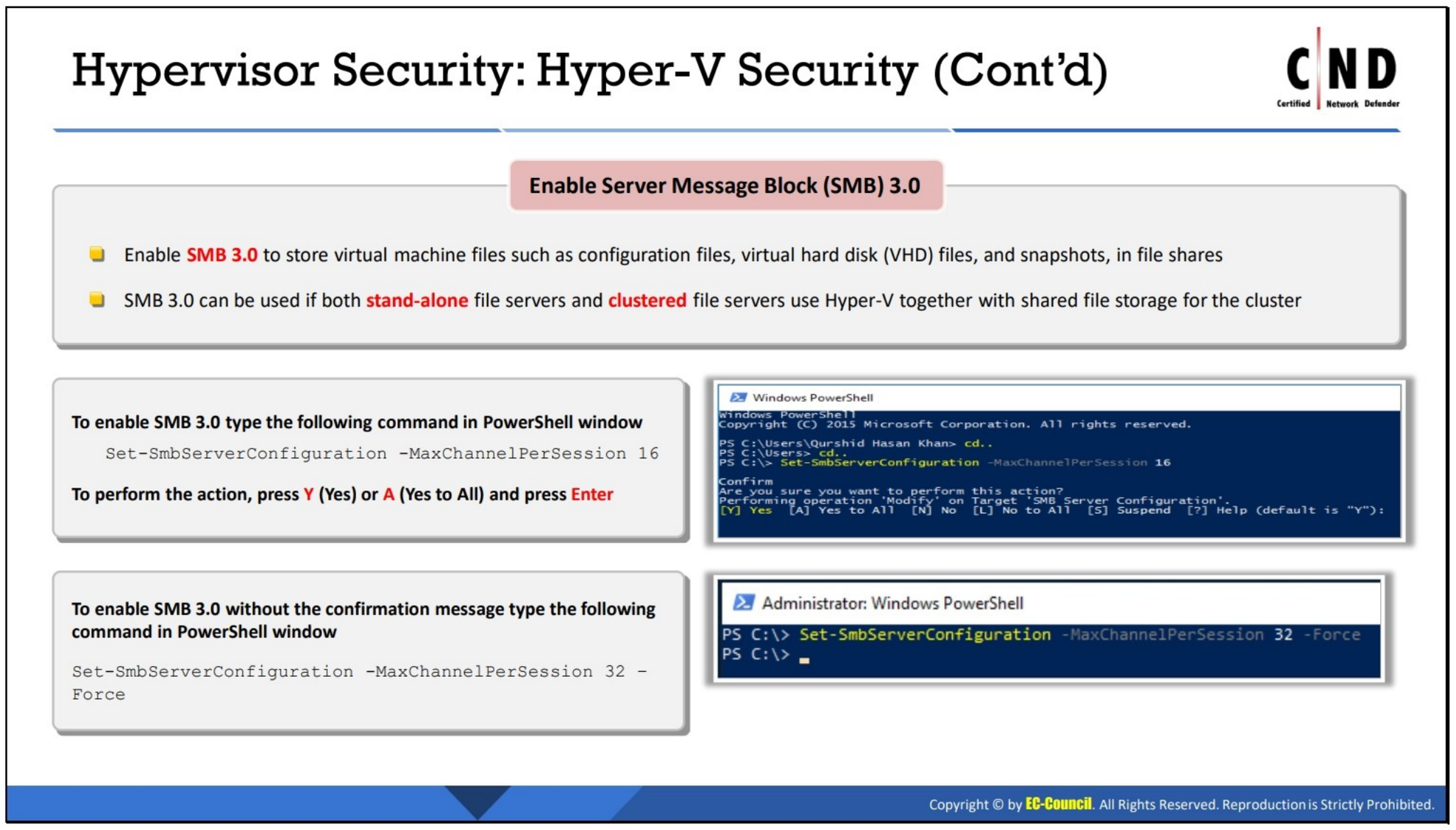


Copyright © by EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24



Hypervisor Security: Hyper-V Security

Time synchronization

Time synchronization provides security and event correlation. VMs lose track of time if the date and time settings are not in synchronization between the host machine and the VM. Saved VMs have wrong time settings when restored if the time is not synchronized. When multiple interconnected VMs utilize time to track transactions, the time should be consistent across all the VMs. Inconsistent time settings between VMs can cause authentication failures, and affect security protocols such as Kerberos, certificate-dependent technologies that rely on time synchronization, billing processes, etc.

To enable time synchronization in Hyper-V

- From the Windows start menu, open **Hyper-V Manager** on the host machine.

Figure 11.5: Search for Hyper-V Manager

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

- Select the virtual machine, right-click on it, and then click **Settings**.

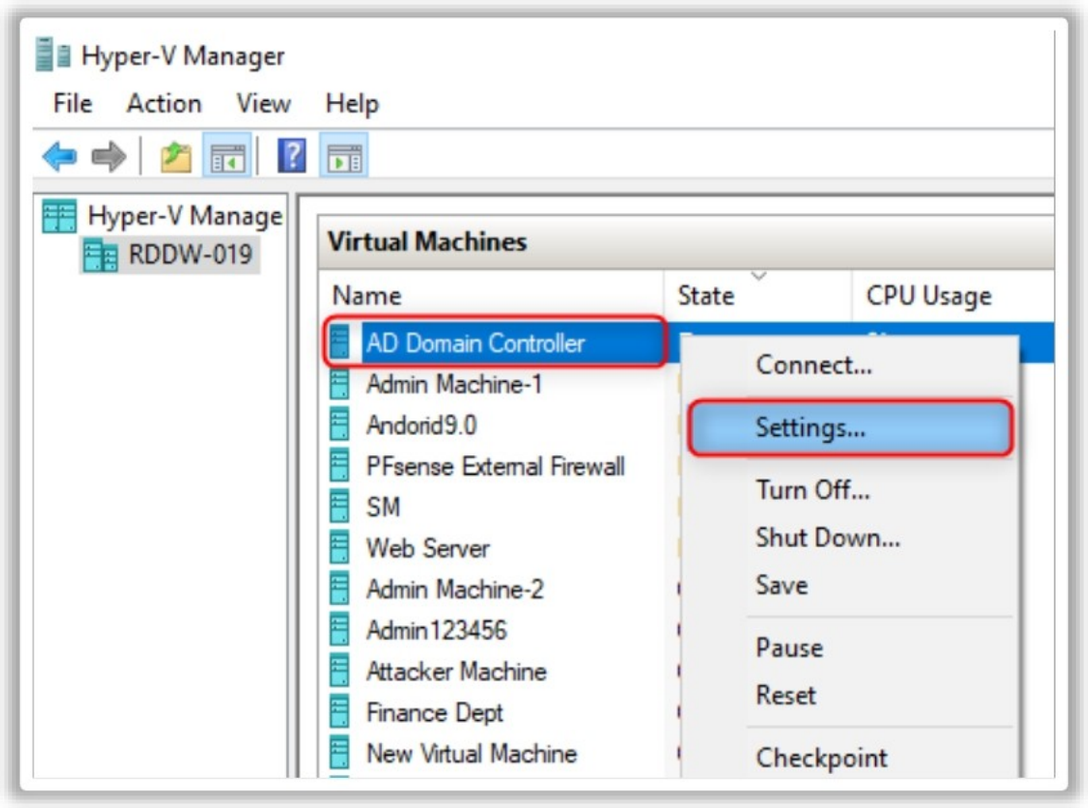


Figure 11.6: Click Settings for VM

- In the **Management** Section, select **Integration Services.**

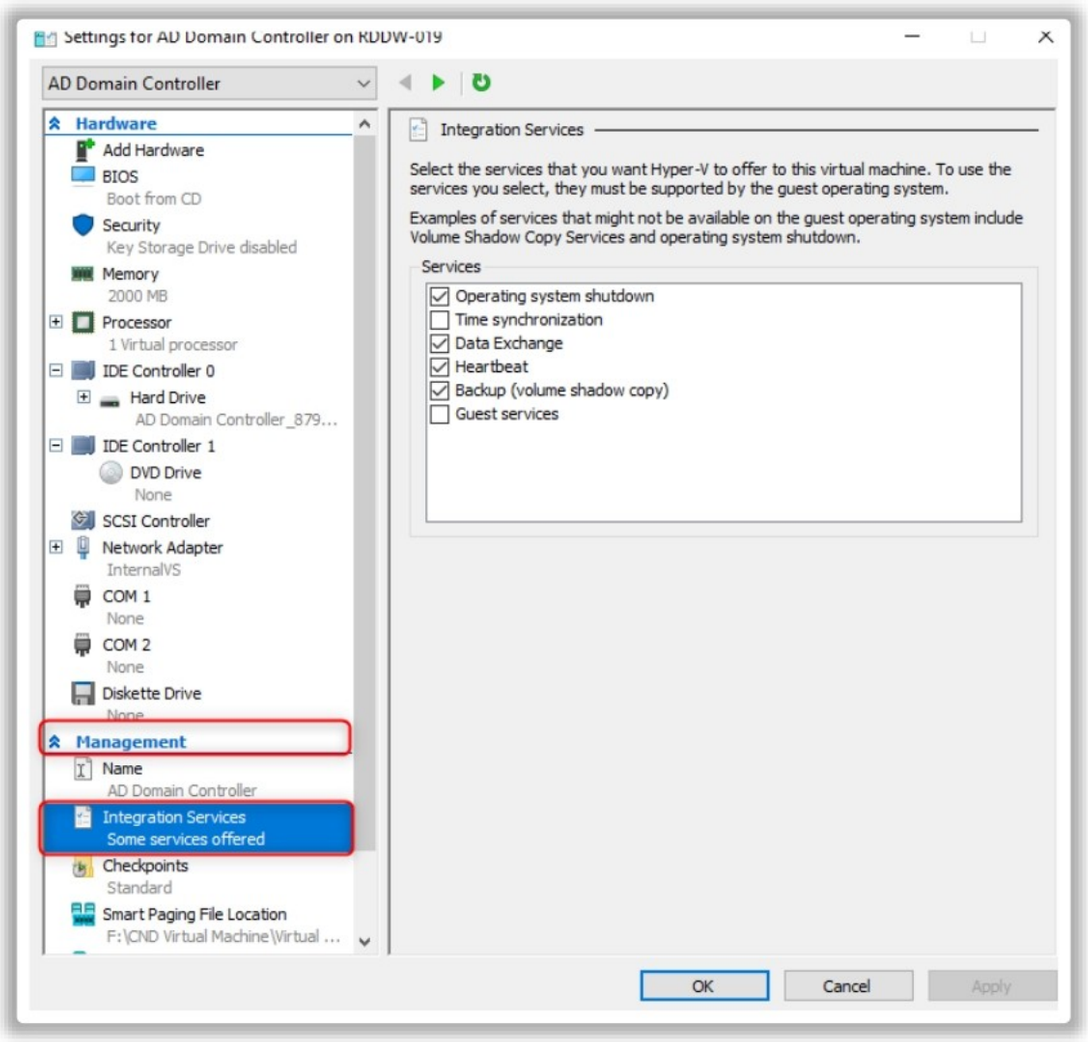


Figure 11.7: Select Integration Services

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

- In **Integration Services**, check the box **Time synchronization**, click **Apply**, and then click **OK**.

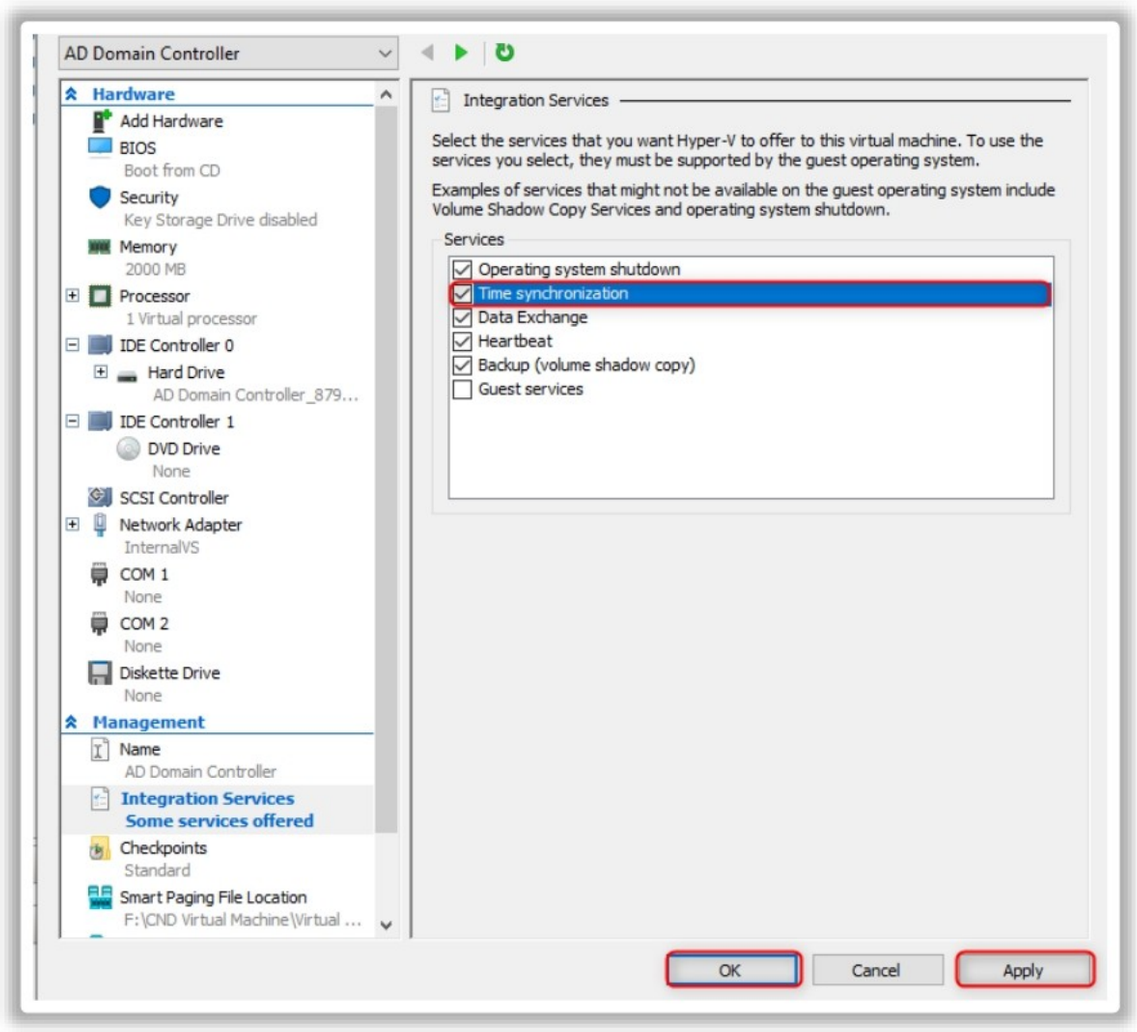


Figure 11.8: Check "Time synchronization" Service

- Navigate to the VM and open it. The time of the VM is now synchronized with that of the host system.

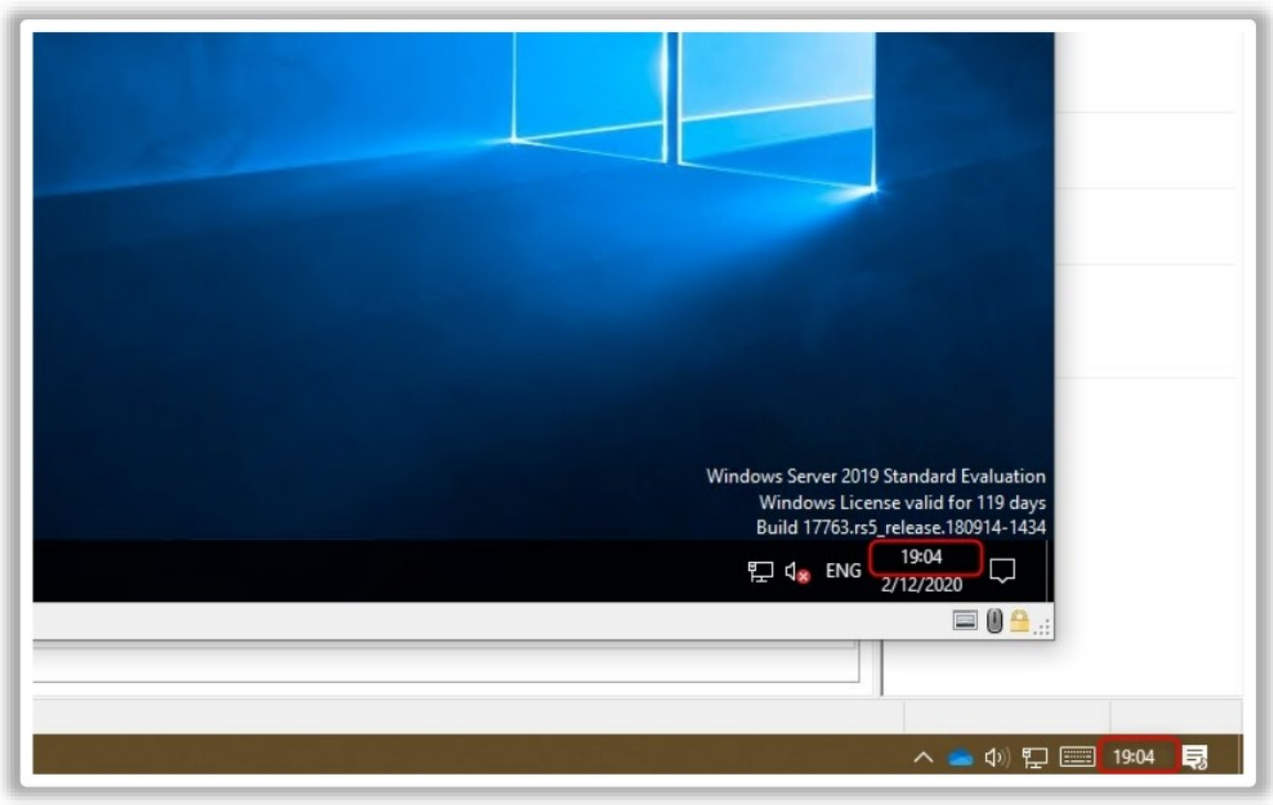


Figure 11.9: See the Synchronized Time of VM with Host

Setting Access privileges

Setting access privileges for users not only controls their access but also reduces the attack surface area, preventing damage from external and internal attacks. By default, Hyper-V comes with a group of admins having all administrative rights for the virtual machines.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

To set access privileges for users

- Navigate to Windows **Control Panel**, and click **System and Security**.

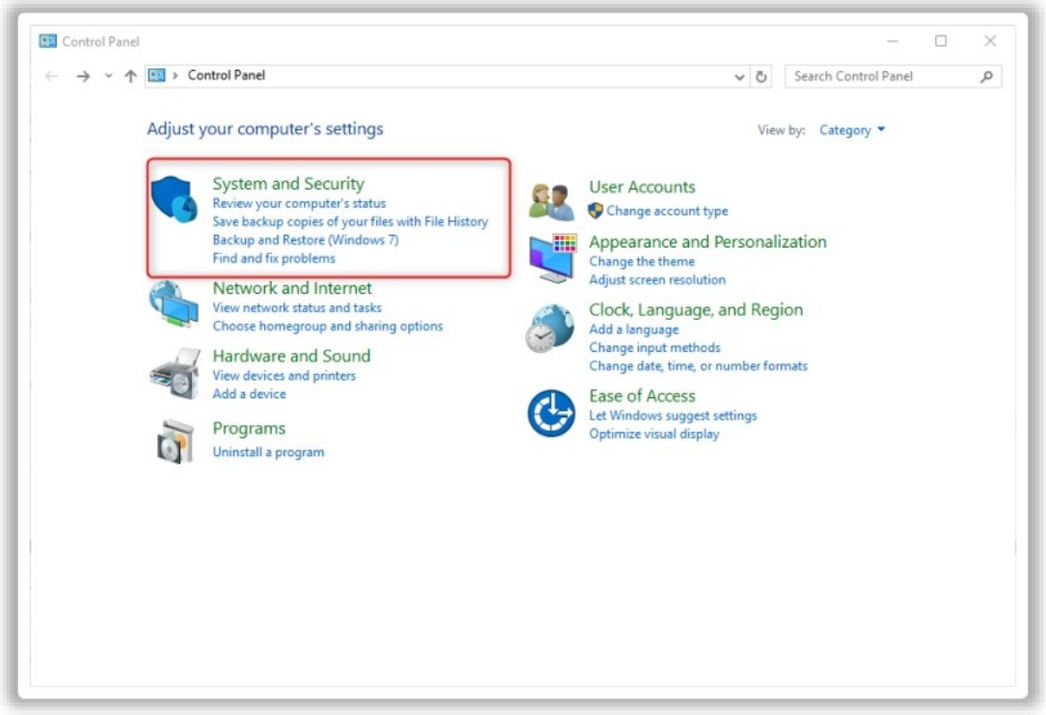


Figure 11.10: Select "System and Security" Setting

- Under **System and Security**, click **Administrative Tools**.

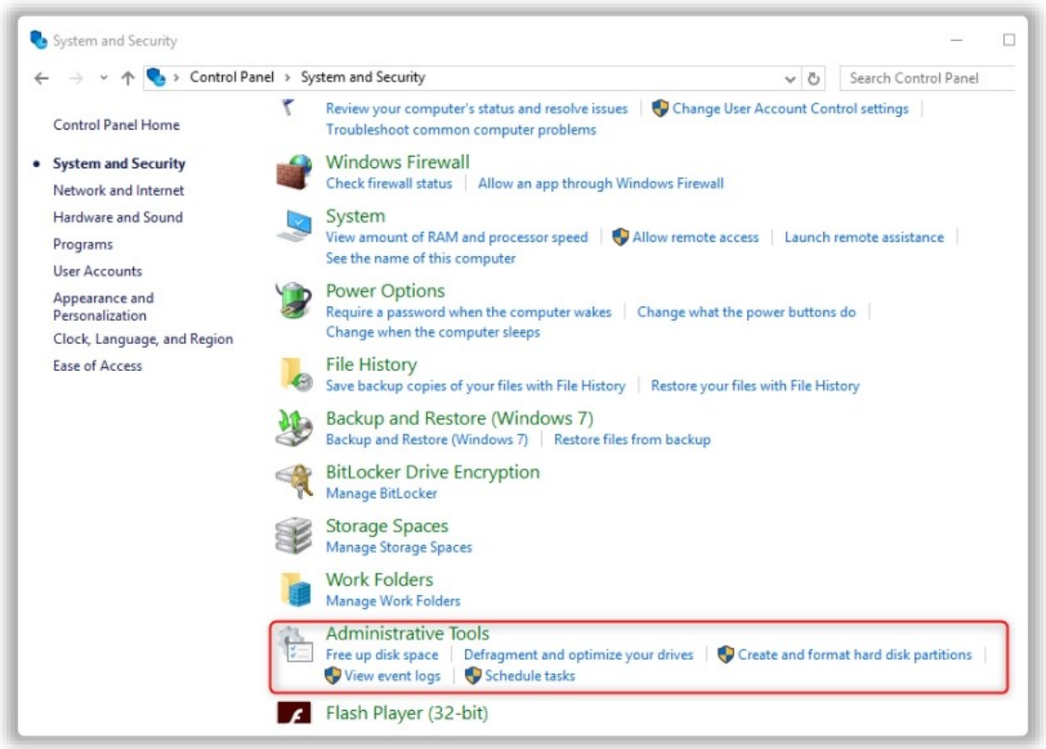


Figure 11.11: Select "Administrative Tools" under "System and Security"

- In the **Administrative Tools** section, click **Computer Management**.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

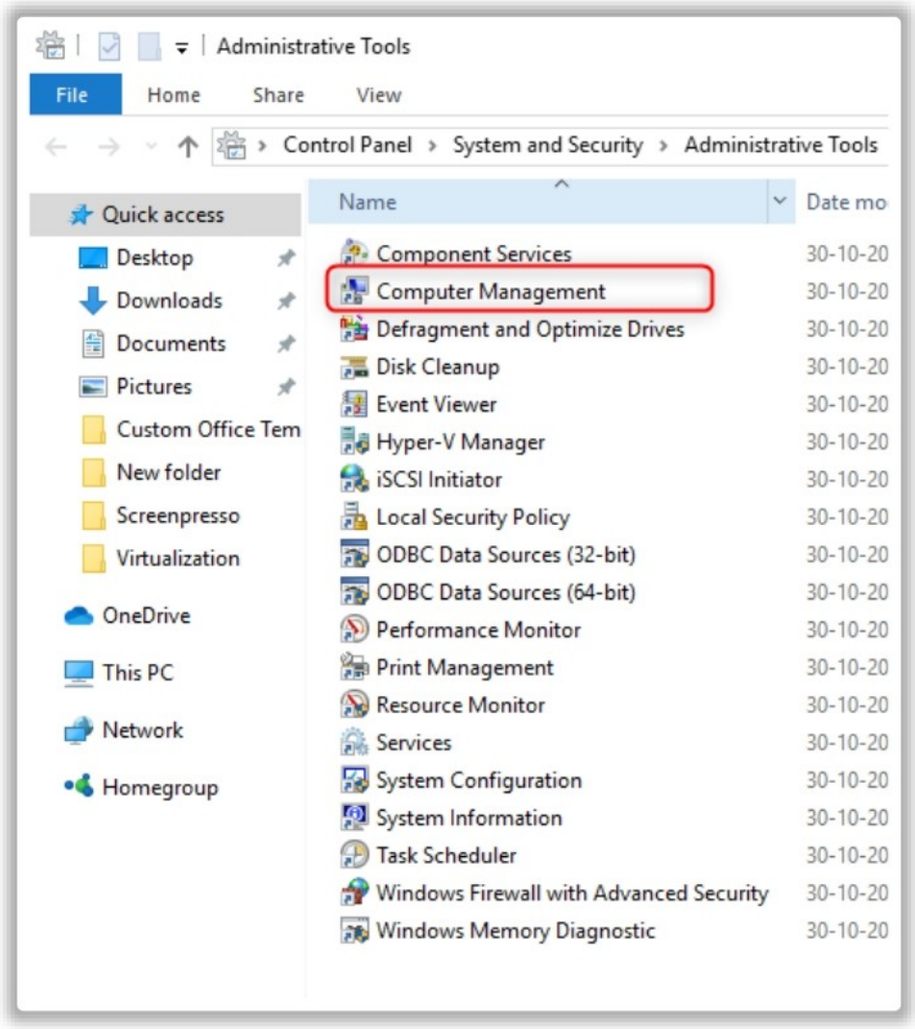


Figure 11.12: Click "Computer Management"

- Under **Computer Management**, click **System Tools**, and then click **Local Users and Groups**.

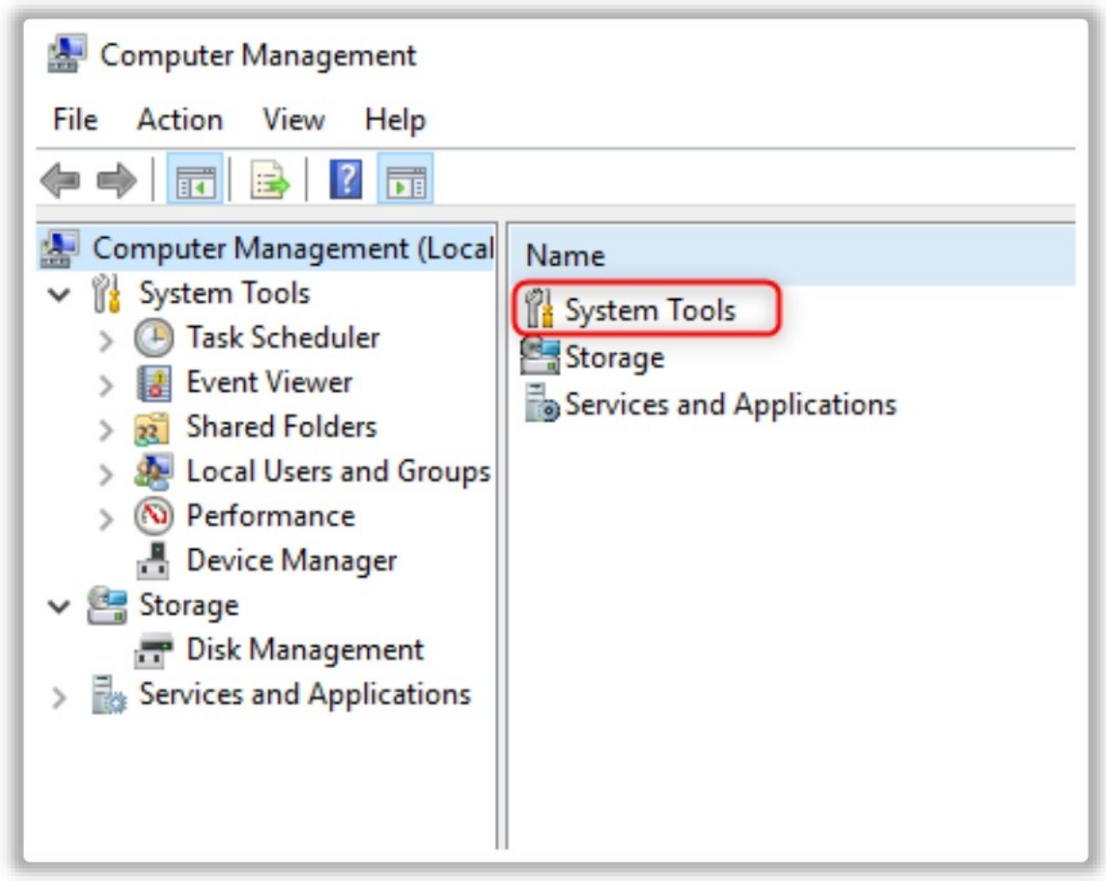


Figure 11.13: Click "System Tools"

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

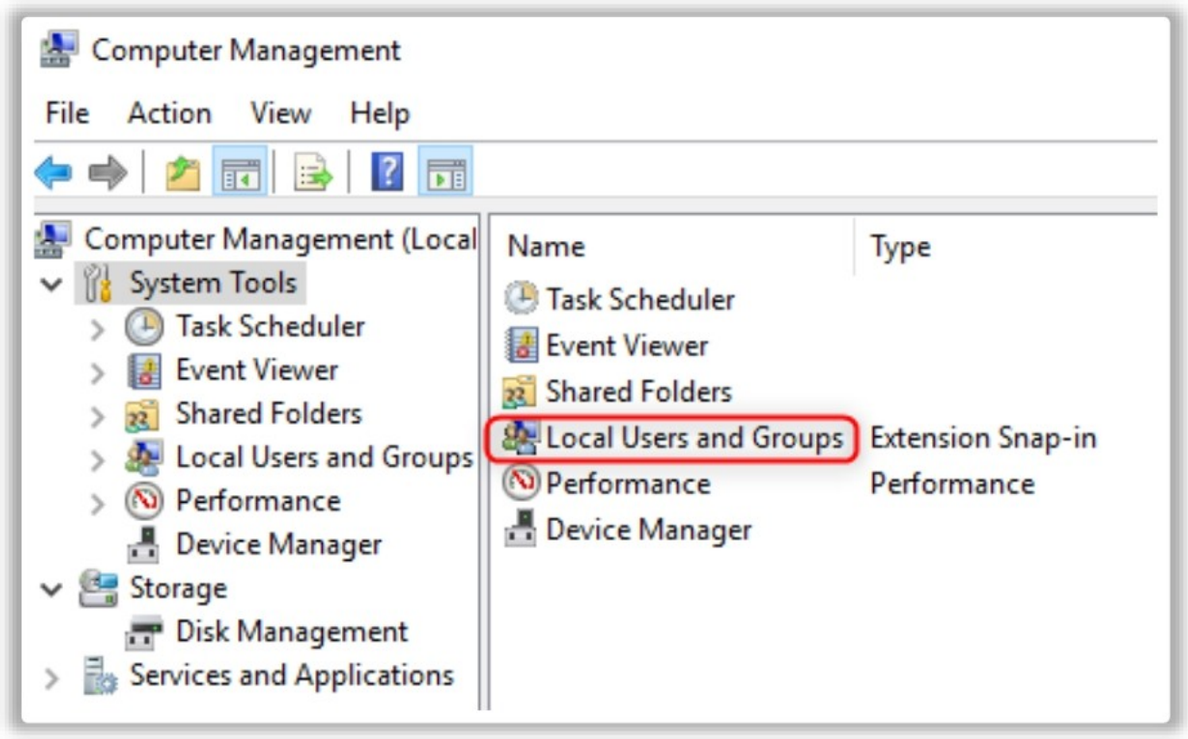


Figure 11.14: Select "Local Users and Groups" under "System Tools"

- Under **Local Users and Groups**, click **Groups.**

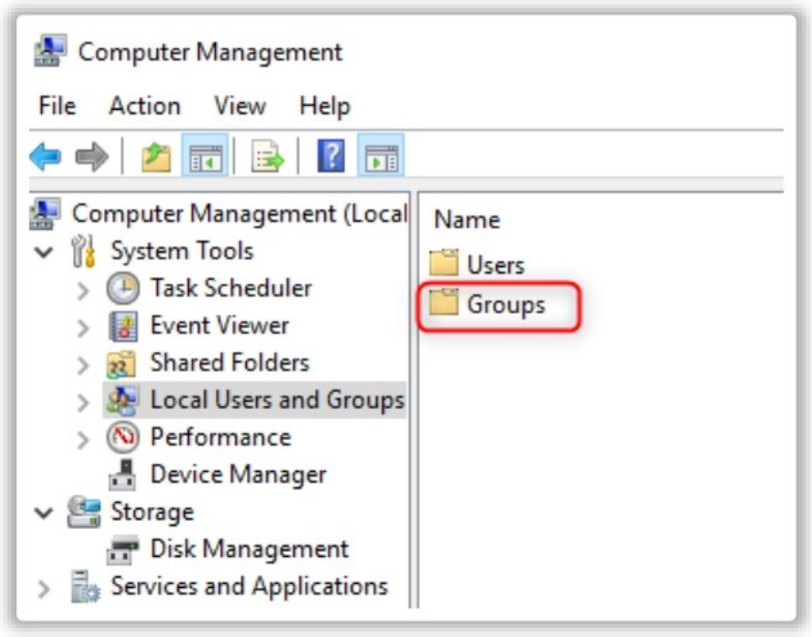


Figure 11.15: Select "Groups" under "Local Users and Groups"

- Navigate to **Hyper-V Administrators** and double click on it, and then click **Add** button.

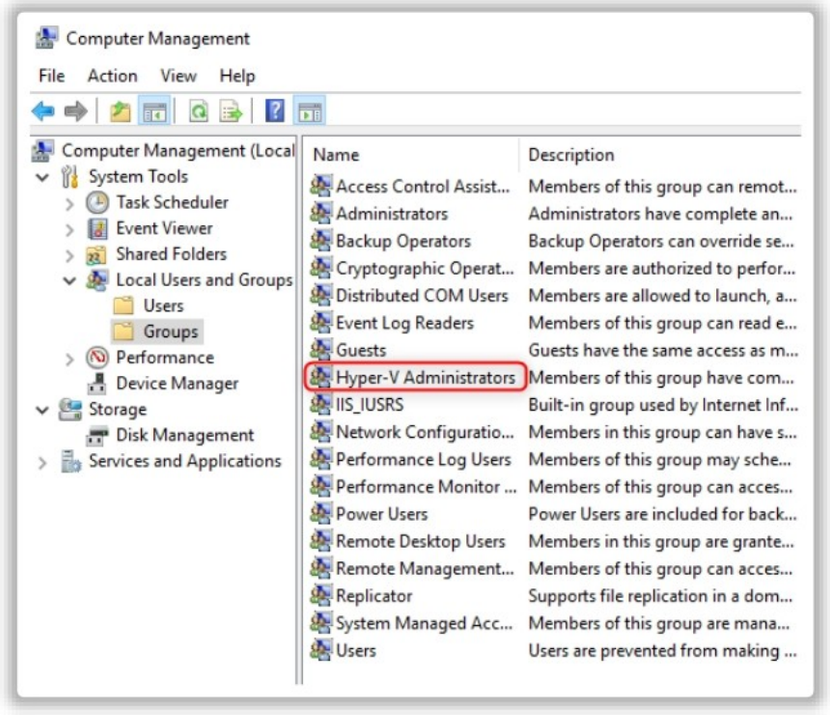


Figure 11.16: Navigate to "Hyper-V Administrators" Group

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

Figure 11.17: Add Members to Hyper-V Administrators

- In **Select Users** window, under "**Enter the object names to select"**, enter the name of the user for setting privilege access and then click **Check Names**.

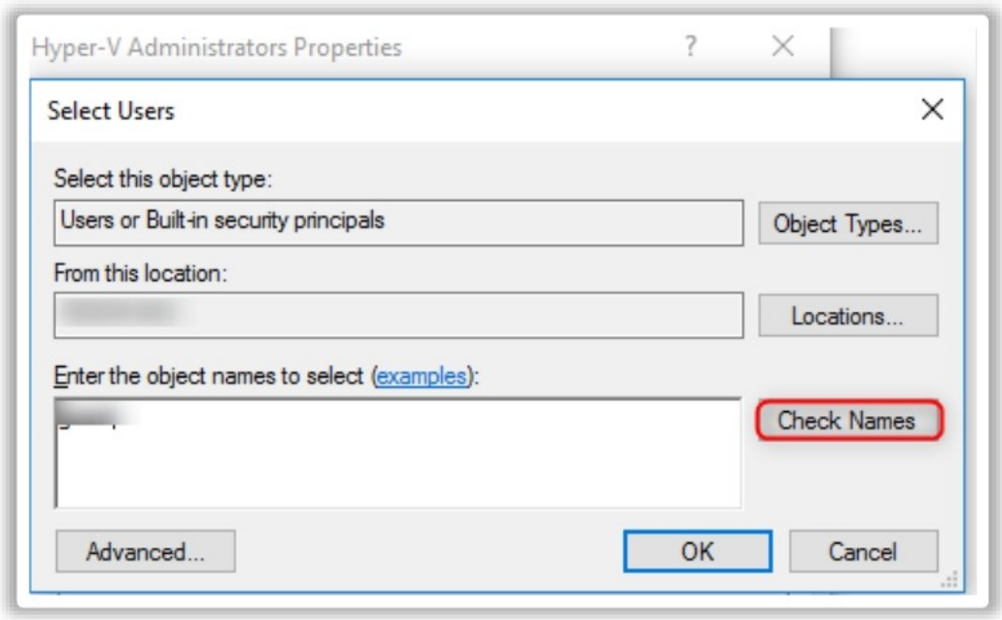


Figure 11.18: Click "Check Names" to Check the Entered Object Name

- Select the user and click **OK** to set access privilege.

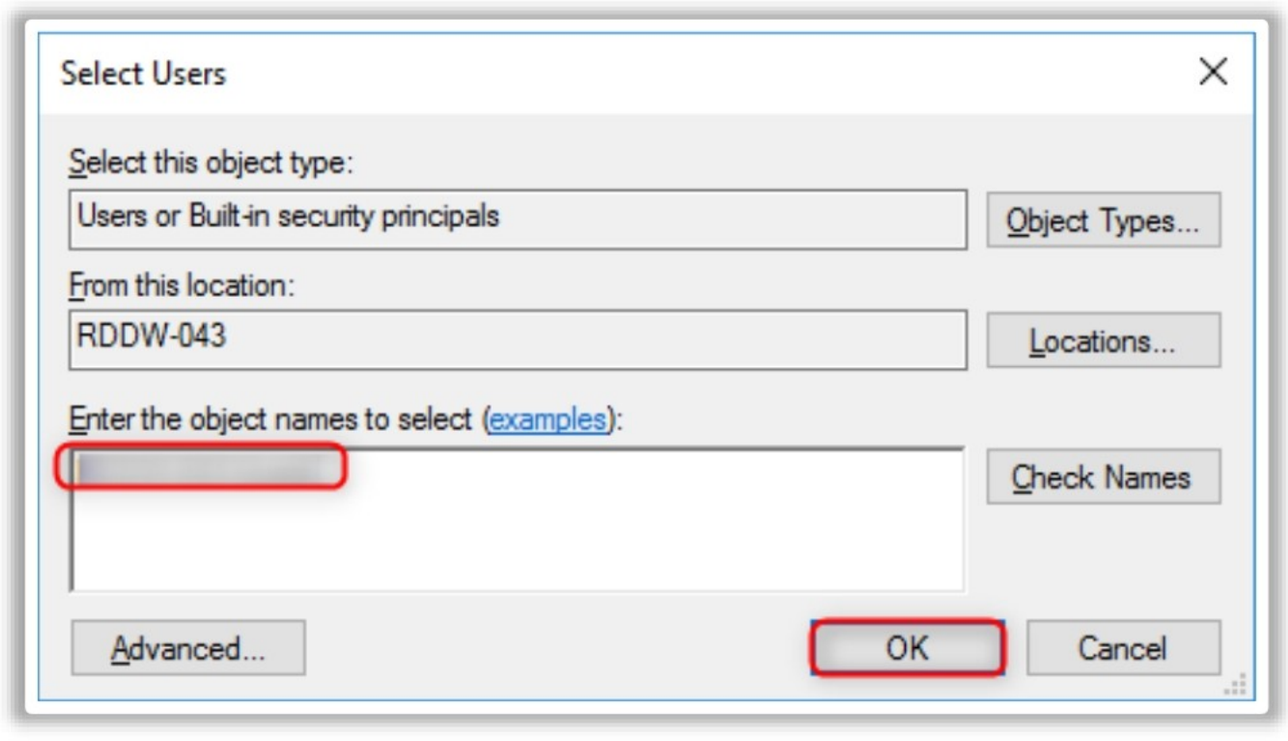


Figure 11.19: Apply OK for the Entered Object

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Disabling unnecessary services

- The user must disable unnecessary services in Windows, so that computer resources are not wasted and the system runs smoothly.
- By default, Windows runs unnecessary services, which can affect performance if they are not disabled.
- Microsoft recommends customers to disable the following services and their respective scheduled tasks on Windows Server 2016**:**
  - Xbox Live Auth Manager.
  - Xbox Live Game Save.

To Disable Xbox Live Auth Manager/ Xbox Live Game Save in Windows Server 2016 Hyper-V

- Navigate to Windows **Control Panel**, and then click **System and Security**.

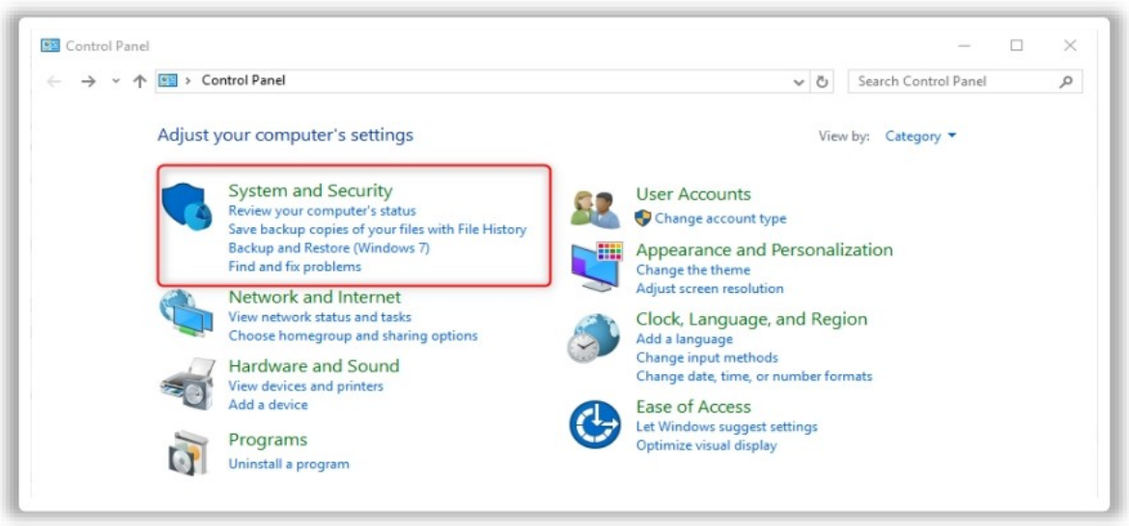


Figure 11.20: Navigate to Control Panel

- Under **System and Security**, click **Administrative Tools**.

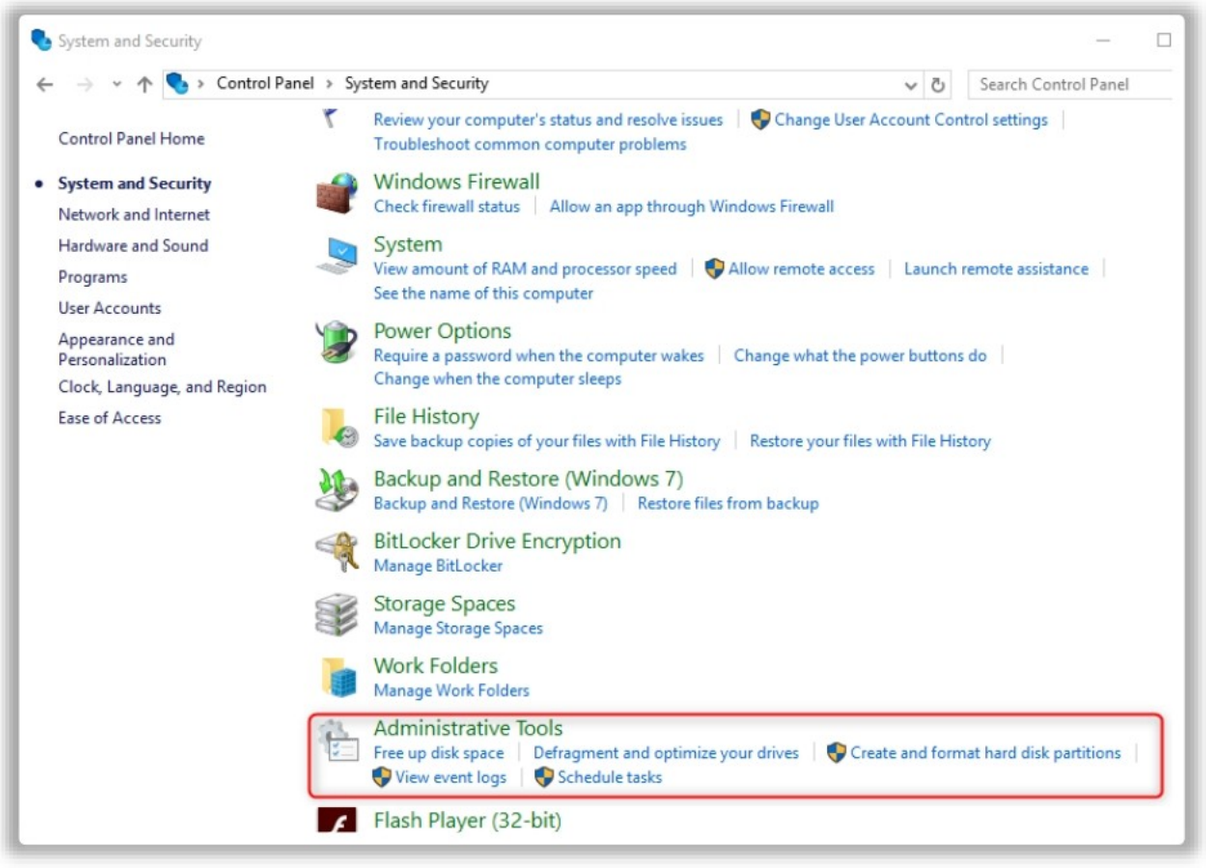


Figure 11.21: Click “Administrative Tools”

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

- In the **Administrative Tools** section, click **Services**.

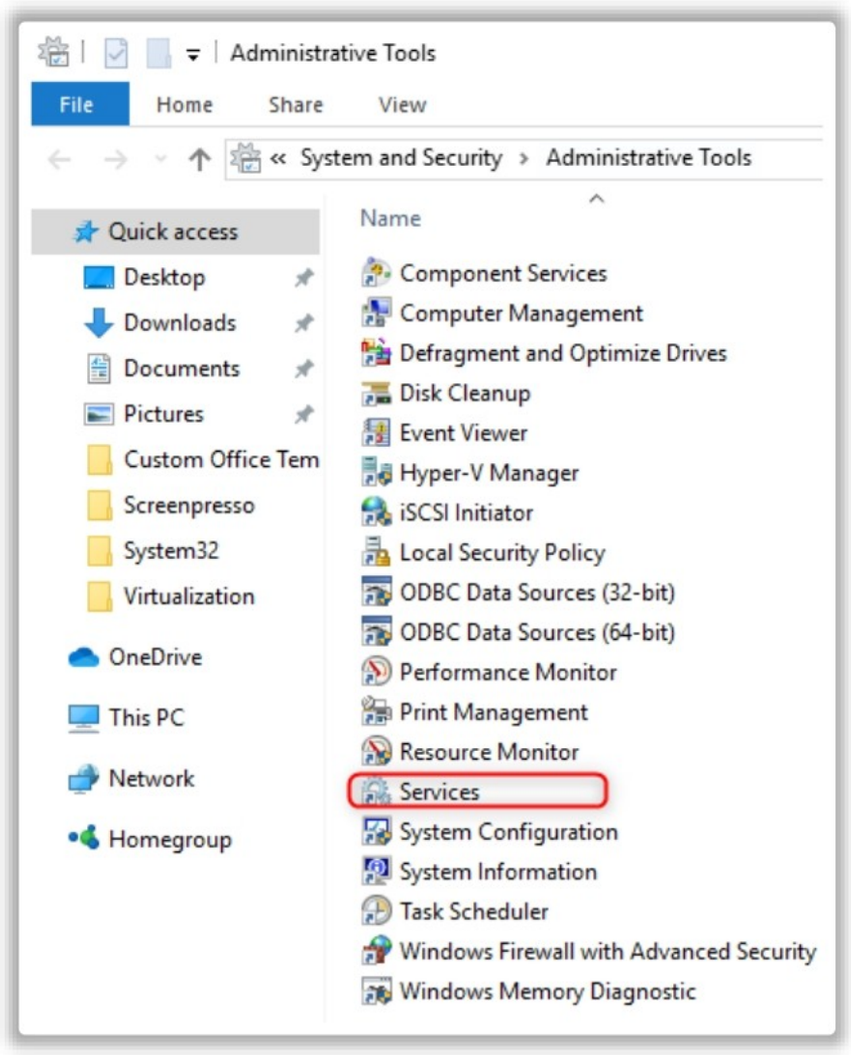


Figure 11.22: Select "Services" Administrative Tool

- Under **Services**, select either **Xbox Live Auth Manager** or **Xbox Live Game Save**, depending on which needs to be disabled, and then double click on it.

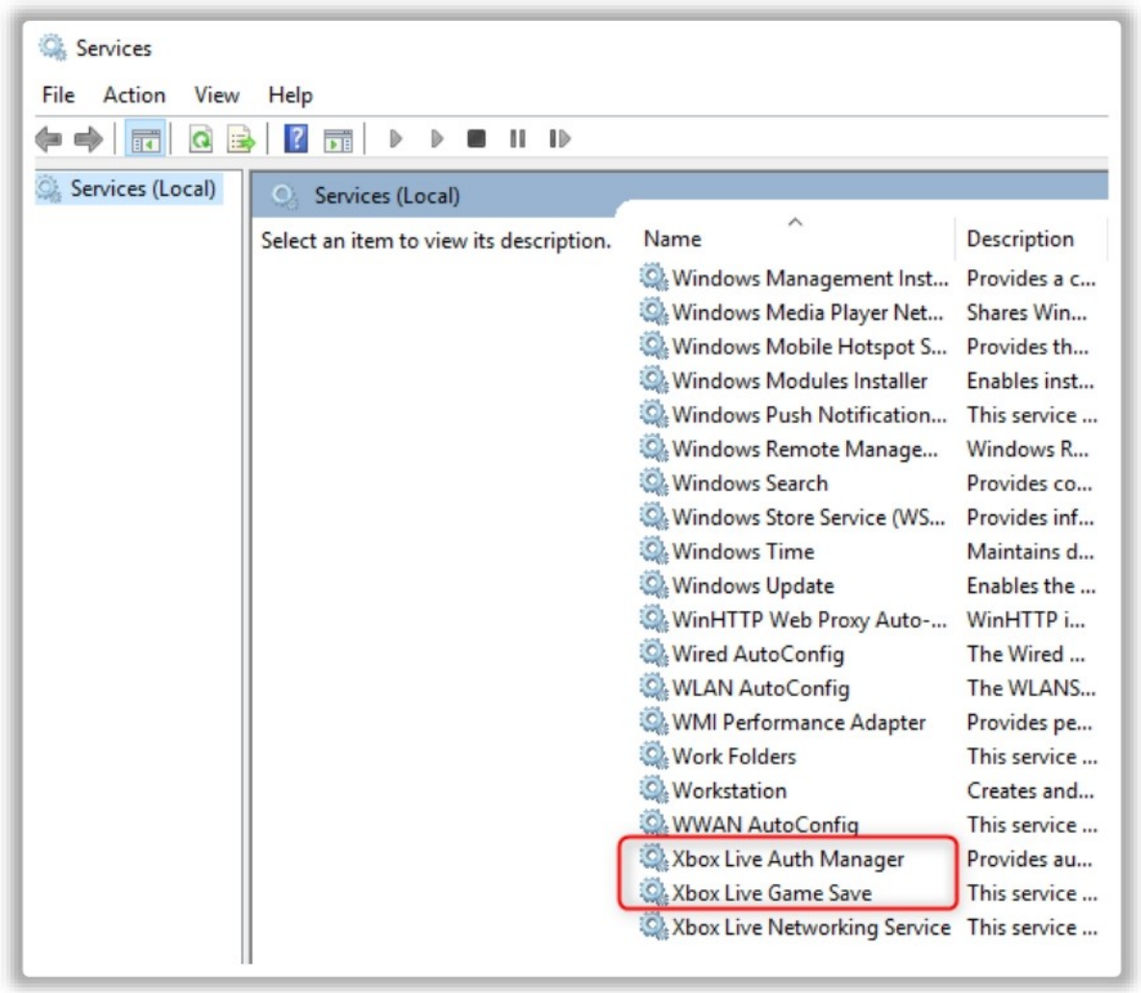


Figure 11.23: Select either "Xbox Live Auth Manager" or "Xbox Live Game Save"

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

- Based on the selected service, a new window opens. In that window, click the **General** tab and navigate to **Startup type**, then from the dropdown choose **Disabled**. Click **Apply** and then click **OK** to disable the service.

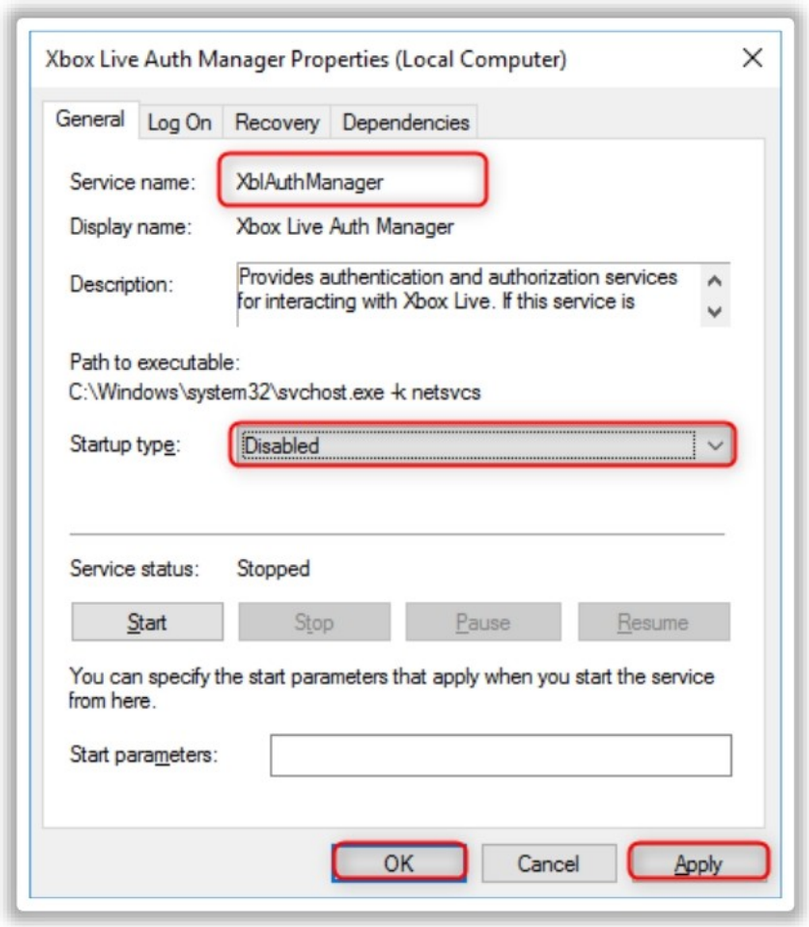


Figure 11.24: Set the Startup Type

Isolated User Mode

The Isolated User Mode (IUM) feature was introduced particularly for Windows 10 Enterprise version and Windows Server 2016. It is a virtualization-based security feature that utilizes the secure kernels, and separates business data and processes from the operating system. IUM, along with other virtualization-based security features of Windows such as Credential Guard and Device Guard, safeguards the system against pass-the-hash attacks and enables network defenders to change the application control policy. It provides protection against cyber criminals, who infiltrate and download malicious applications.

To Enable IUM

- Open Windows **Control Panel**, and click **Programs**.

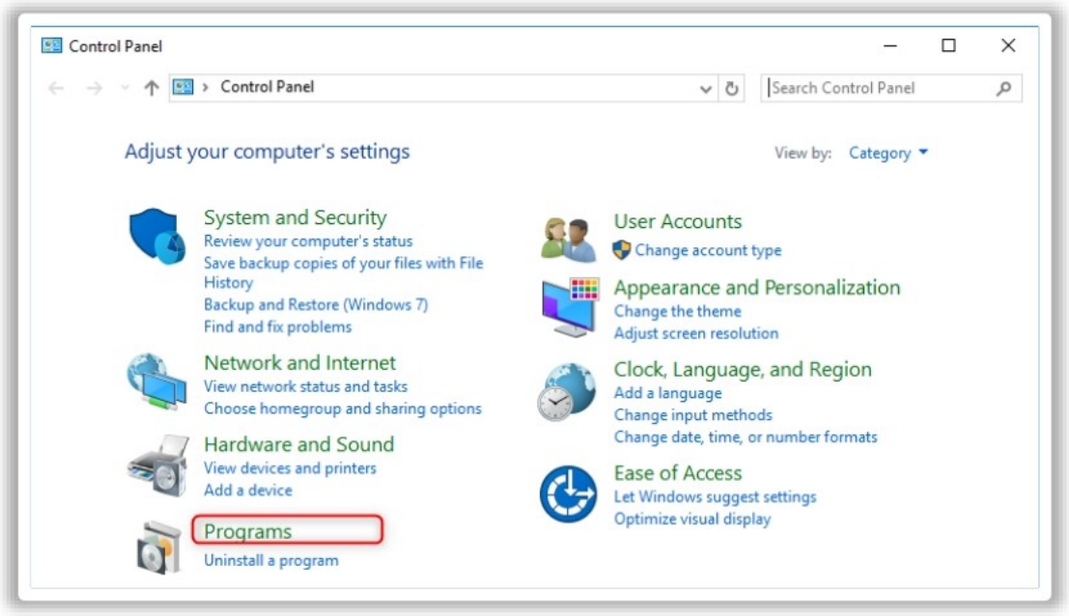


Figure 11.25: Navigate to Programs

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

- Under **Programs and Features**, click **Turn Windows features on or off**.

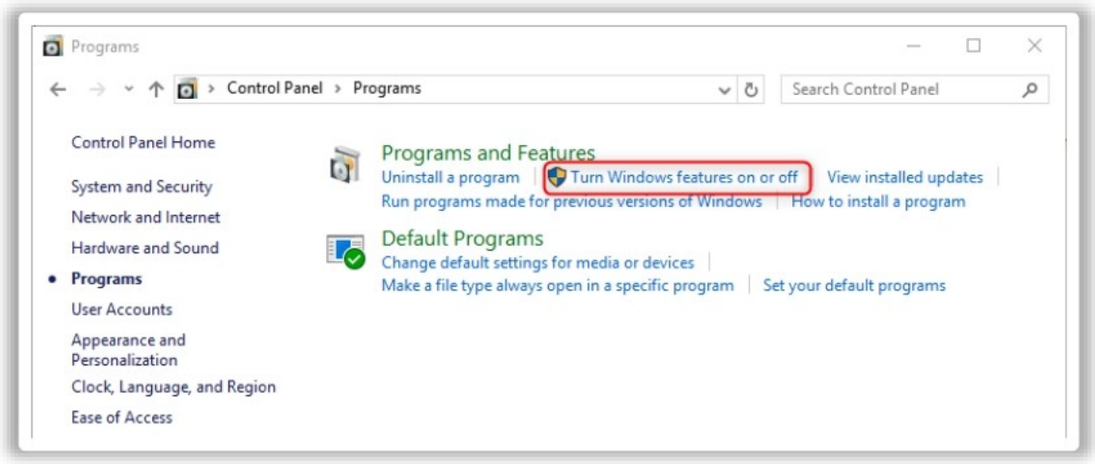


Figure 11.26: Select "Turn Windows features on or off"

- In **Turn Windows features on or off** window, scroll down to **Isolated User Mode**, check the box, and then click **OK**. Reboot the system to apply the changes.

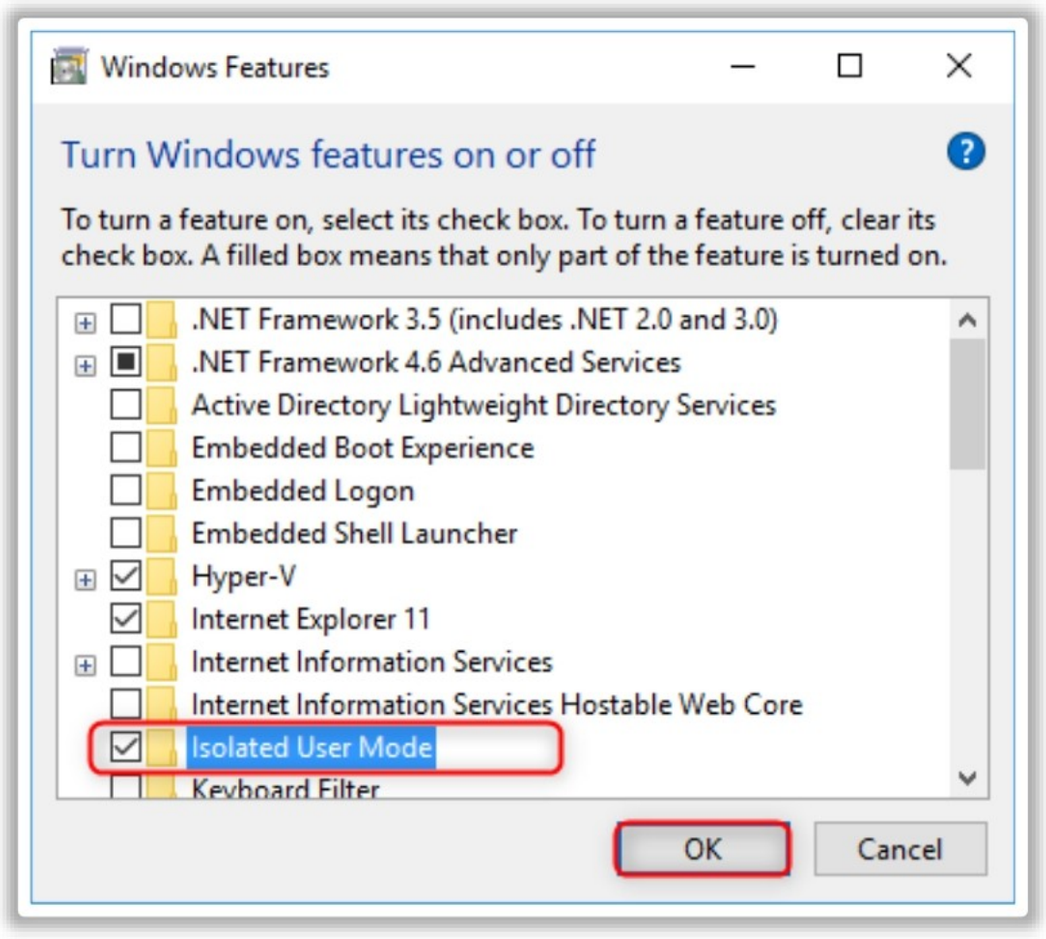


Figure 11.27: Check "Isolated User Mode" Windows Feature

SMB 3.0 file shares

SMB 3.0 file shares is a shared storage for Hyper-V (Windows server 2012 and 2012 R2). Therefore, all VM files stored on the Hyper-V such as configuration files, virtual hard disk (VHD) files, and snapshots can be stored in SMB 3.0 file shares. These files can be accessed via the SMB 3.0 protocol. SMB 3.0 can be utilized by standalone as well as clustered file servers that use Hyper-V and shared file storage.

Steps to Enable SMB 3.0

- Execute `Set-SmbServerConfiguration -MaxChannelPerSession 16` in the PowerShell to enable SMB.
- To perform the action, press Y (Yes) or A (Yes to All) and press Enter.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

- To enable SMB without the conformation message, use the command **`Set-SmbServerConfiguration -MaxChannelPerSession 32 -Force`** in PowerShell.

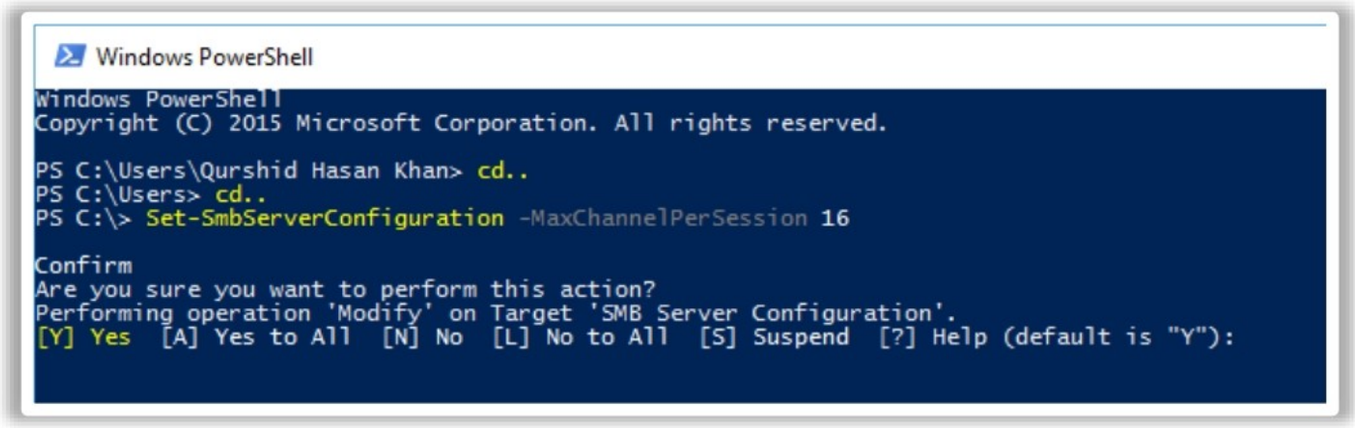


Figure 11.28: Enable SMB

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

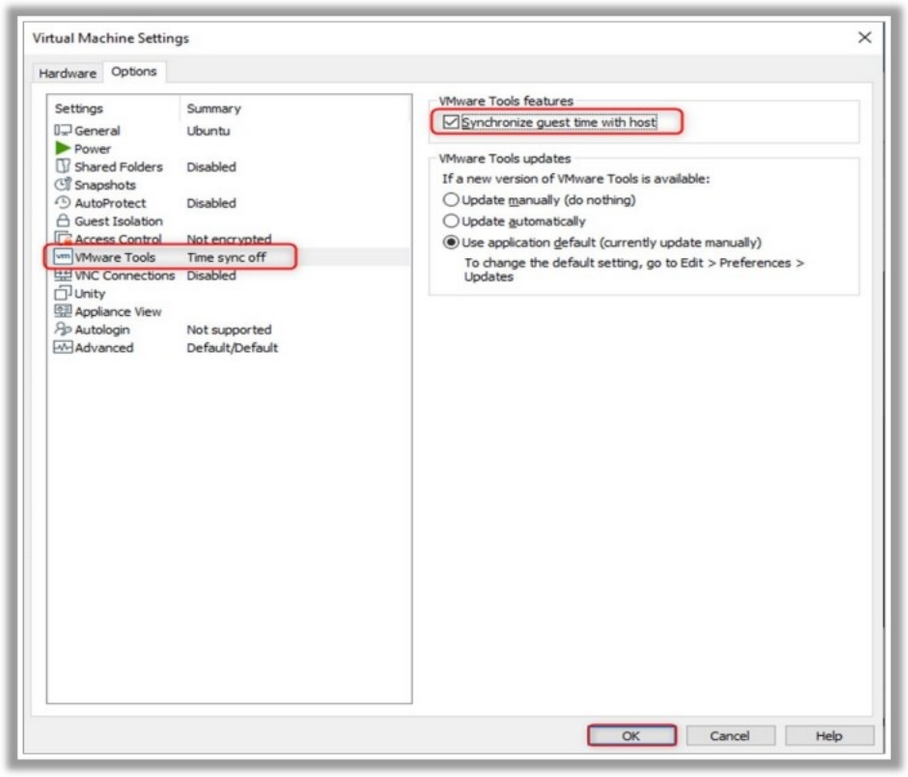
Technet24

Hypervisor Security: VM Ware Security

C|ND Certified Network Defender

Time Synchronization

- Virtual Machines may have a **time drift** in their clocks when compared to clocks in the hosting machine
- Due to the mismatch of the time in the clocks, necessary data may not be found at the time of forensic investigation
- VMware provides **time synchronization** option to rectify this problem



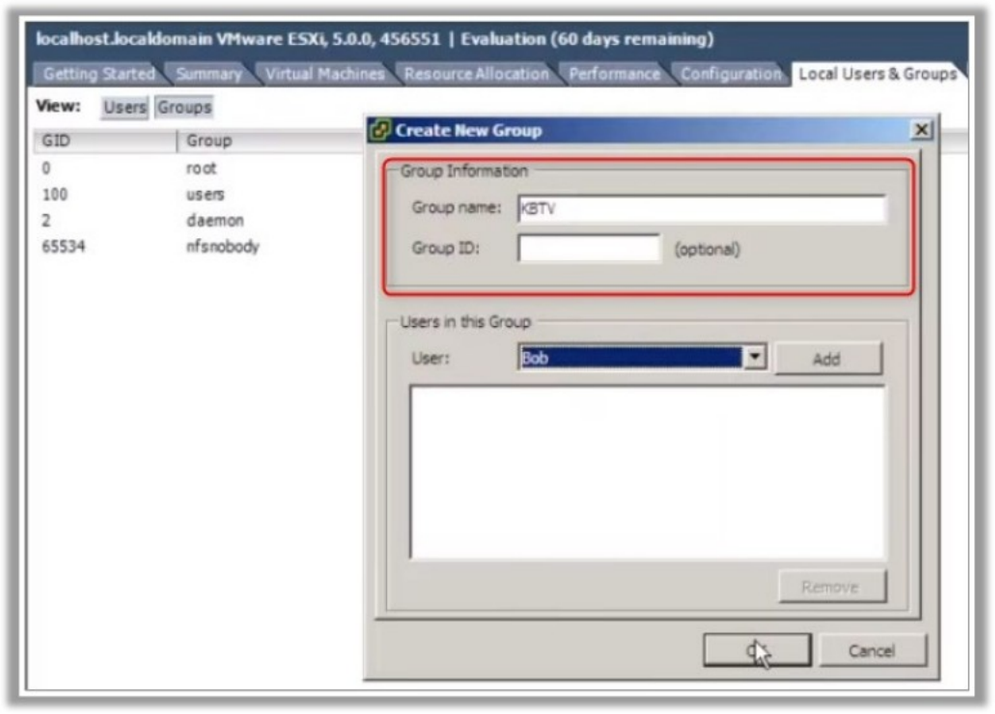
Copyright © by EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited.

Hypervisor Security: VM Ware Security (Cont'd)

C|ND Certified Network Defender

Restrict User Access

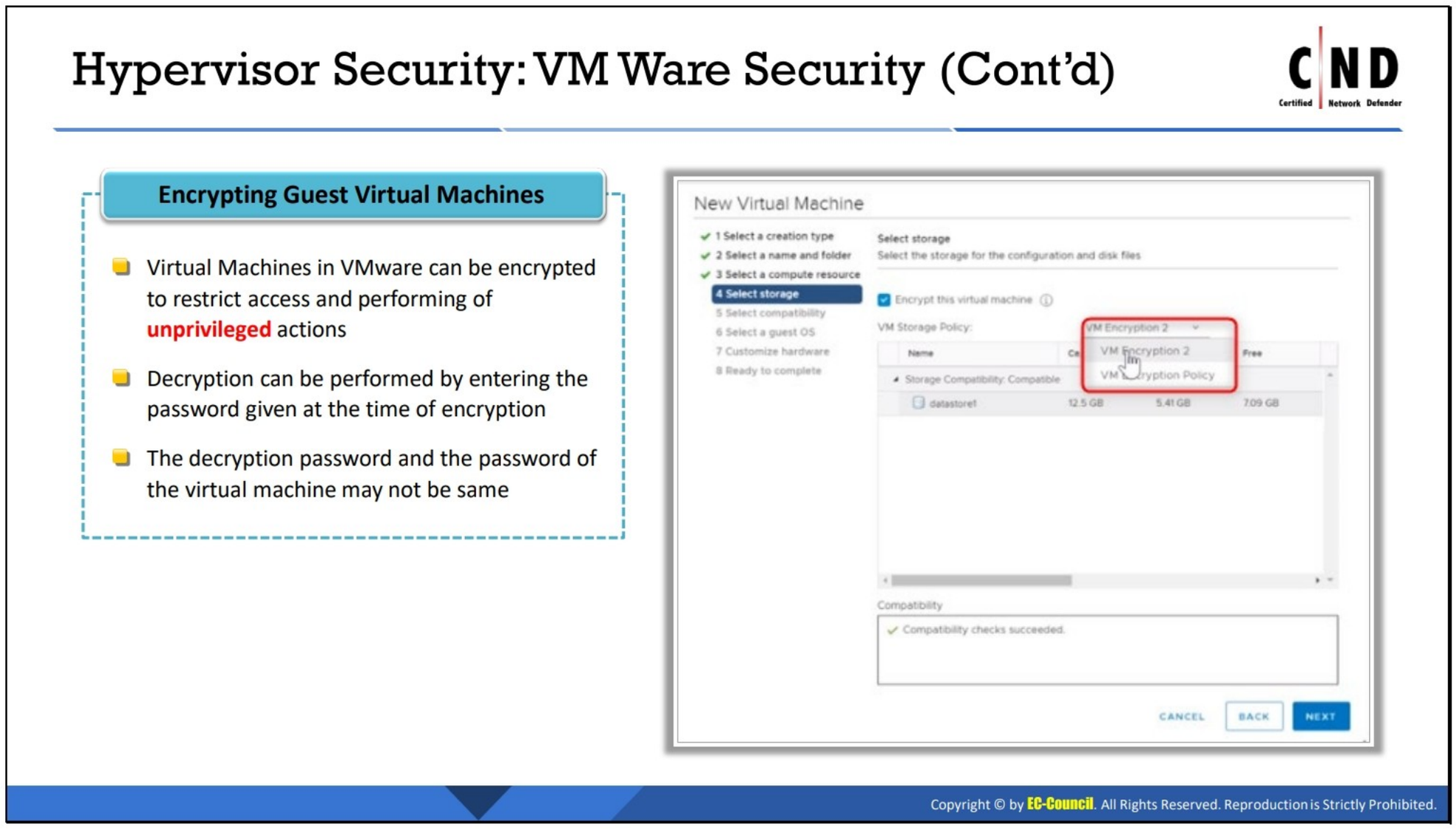
- A group is a set of users that share a common set of **rules** and **permissions**. Same permissions are assigned to all the members in a group
- Only groups with permissions can access the corresponding virtual machines



Copyright © by EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Hypervisor Security: VM Ware Security

The measures that can be adopted to enhance the security of VMware hypervisors are now discussed.

Time synchronization

Virtual Machines have a time lag in their clocks when compared to clocks in the host machine. This mismatch between the time in the clocks may cause challenges for forensic investigation as appropriate data may not be found. VMware provides the time synchronization option to address this issue.

To enable time synchronization in VMware

- Open the VMware virtual machine, right-click on **Virtual Machine** tab and Select **Virtual Machine Settings**.
- In the **Virtual Machine Settings** window, click **Options** tab
- Navigate and click **VMware Tools.**
- Browse to **VMware Tools Features** pane, check the box **Synchronize guest time with host**.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

- Click **OK**.

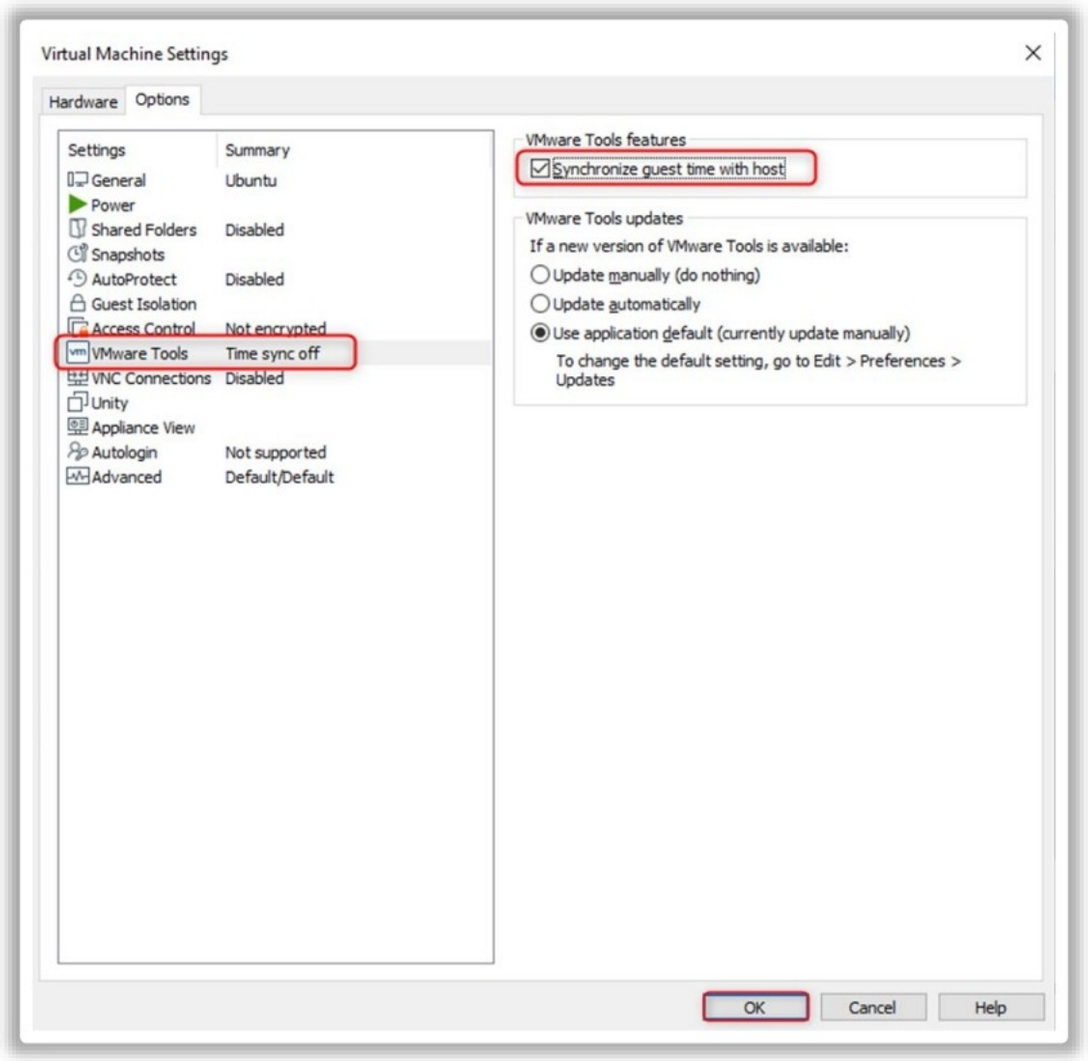


Figure 11.29: Check "Synchronize guest time with host" VMWare Tools feature

Restrict user access

To restrict user access in VM Ware, create a group (set of users that share a common set of rules and permissions). The same permissions are assigned all the members in a group. Only groups with permissions can access a virtual machine.

The following are the steps to follow to add users individually to a group:

- Log in to ESXi using the vSphere client
- Click the **Local Users & Groups** tab and click Users.
- Right-click the user and click **Edit** to open the **Edit User** dialog box
- Enter the **Username** and in **Change password** section, add a new password
- To add the user to a group, select the group name from the **Group** drop-down menu and click **Add**
- Enter the Group name and select a user to add in that group and click **OK**

Encrypting Guest virtual machines

Virtual machines in VMware can be encrypted to restrict their access from users without privileges. Decryption can be performed by entering the password given at the time of encryption. The decryption password and the password of the virtual machine need not be same.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

The following are the steps to be followed to enable this option:

- Open VMware virtual machine, right-click on **Virtual Machine** tab and select **Virtual Machine Settings**.
- In the **Virtual Machine Settings** window, click **Options** tab
- Navigate and click **Access Control**.
- In the **Encryption** section, click the **Encrypt** button.
- The **Encrypt Virtual Machine** dialog box appears.
- Enter the **Password, Confirm** the password, and click **Encrypt**.
- Click **OK**.

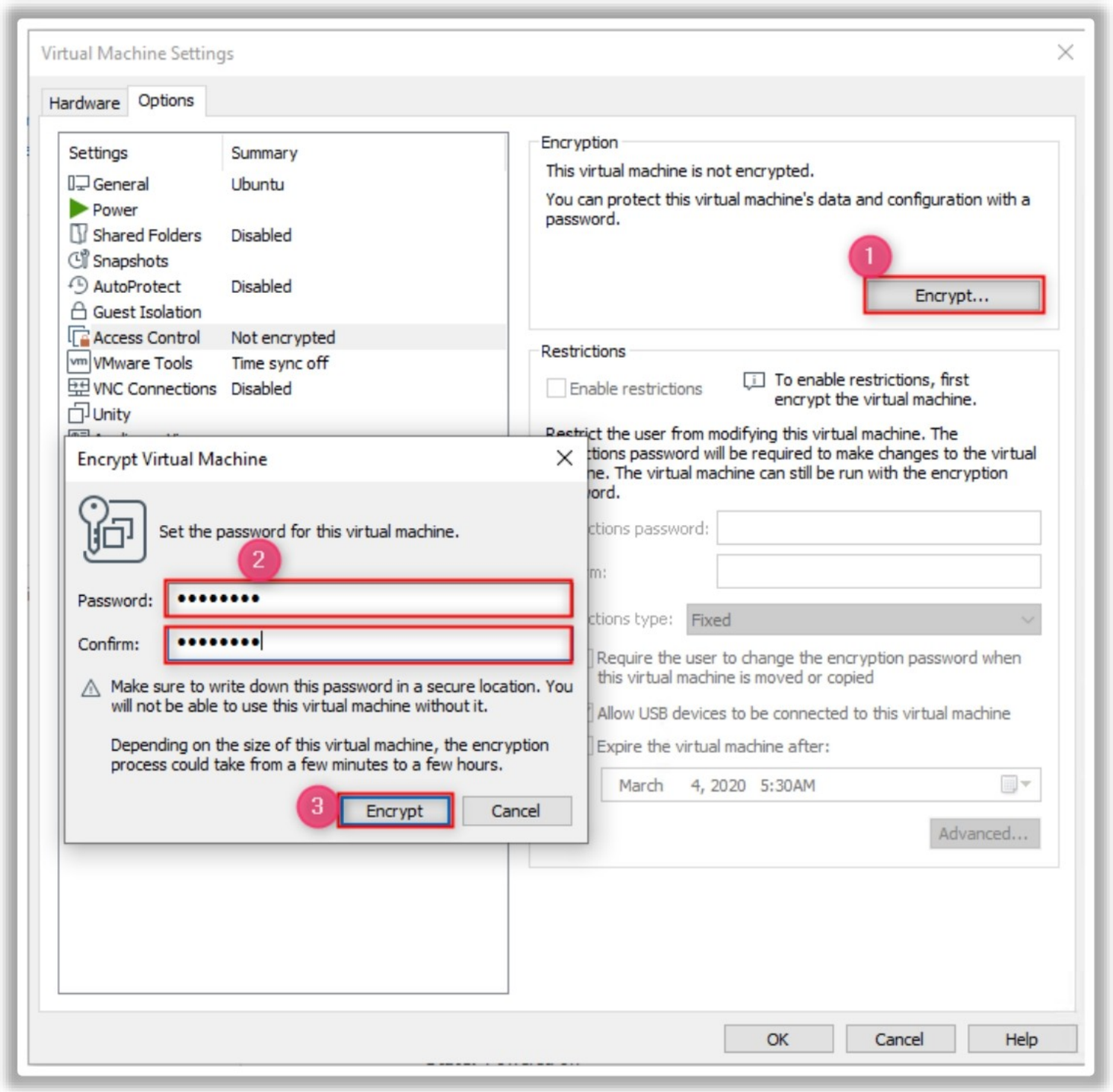


Figure 11.30: Set Password for VM

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

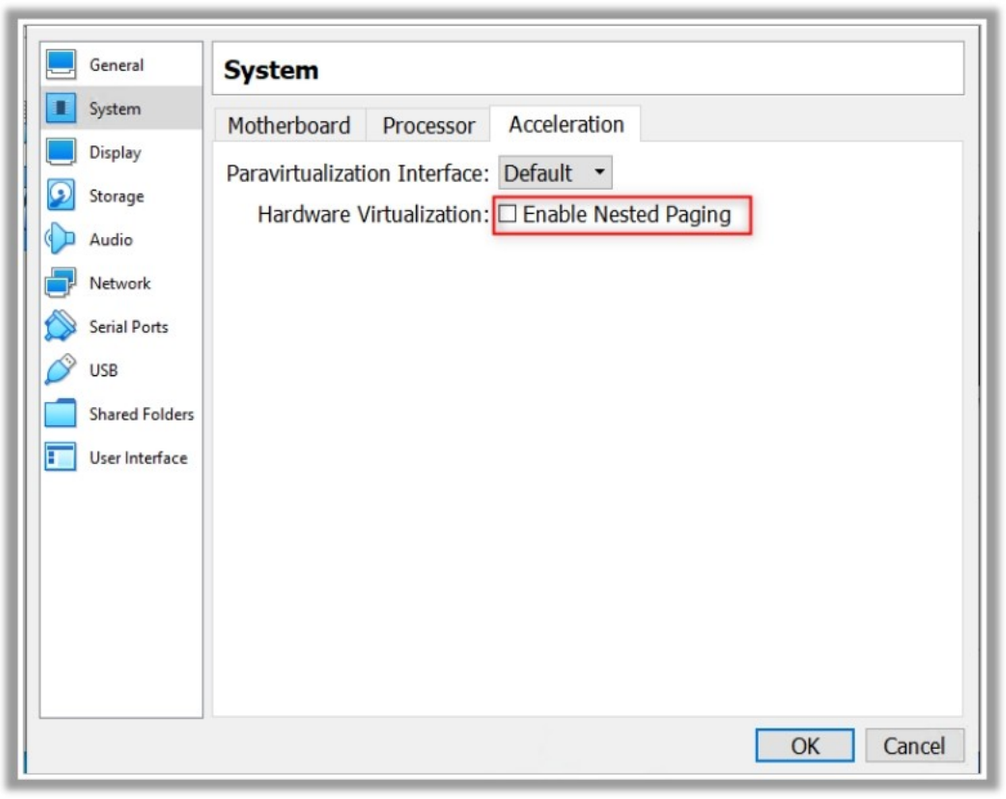
https://t.me/learningnets

Technet24

Hypervisor Security: Virtual Box Security

Disable Nested Paging

- Nested paging enhances performance by **enabling** guest machine to handle paging without intervention from the hypervisor
- **Disabling** nested paging ensures that VMM validates each entry before shadowing to prevent guest machine from adding inappropriate entries into the page tables



Copyright © by EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited.

Hypervisor Security: Virtual Box Security (Cont'd)

Disable Hyperthreading

- Removes potentially sensitive data from affected buffers, which leak information between the threads
- To **disable** hyper threading use the following Vbox Manage modify vm option

```
--mds-clear-on-vm-entry
```

Flush Level 1 Cache Data

- While running guest code, potentially sensitive data must be removed from level 1 cache.
- To **flush** the details use the following Vbox Manage modify vm option

```
--l1d-flush-on-vm-entry
```

Copyright © by EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited.

Hypervisor Security: Virtual Box Security

The measures that can be adopted to enhance the security of VirtualBox hypervisors are now discussed.

Disable nested paging

Nested paging enhances hardware virtualization by implementing memory management in hardware. The overheads associated with to VM exits and page table accesses are removed by

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

nested paging. Thus, it enhances performance by enabling the guest machine to handle paging without intervention from the hypervisor. However, this task is not meant to be performed by virtualization software.

Disabling nested paging ensures that the VMM validates each entry before shadowing to prevent the guest machine from adding inappropriate entries into the page tables. Disabling of nested paging makes several CPU features such as AVX, XSVAE, and POPCNT unavailable to guests. Consequently, guests might experience stability issues, especially during SMP configuration. However, by disabling nested paging, the guest is not be able to insert inappropriate entries in the page tables, thereby securing the VirtualBox environment.

Disable hyperthreading

Hosts that are affected by CVE-2018-12126 and CVE-2018-12127 should disable hyperthreading. Disabling hyperthreading removes potentially sensitive data from the affected buffers, which leak information between the threads. Traditionally, this was done by firmware setup, but certain OSes provide ways to disable hyperthreading, and in some cases, hyperthreading is disabled by default.

To disable hyperthreading use the following **VboxManage modifyvm** option

`-mds-clear-on-vm-entry`

The above command in VirtualBox clears the affected buffer on every VM entry. While this may reduce performance when compared to the default option, the actual impact depends on the workload.

`Flushing level 1 cache data`

While executing code from a guest, potentially sensitive data must be removed from the level 1 cache. The prerequisite for cache flushing mitigation is that the CPU microcode should be up-to-date. Traditionally, this was performed by the system firmware, but some OSes install it automatically.

To flush the details use the following **VboxManage modifyvm** option

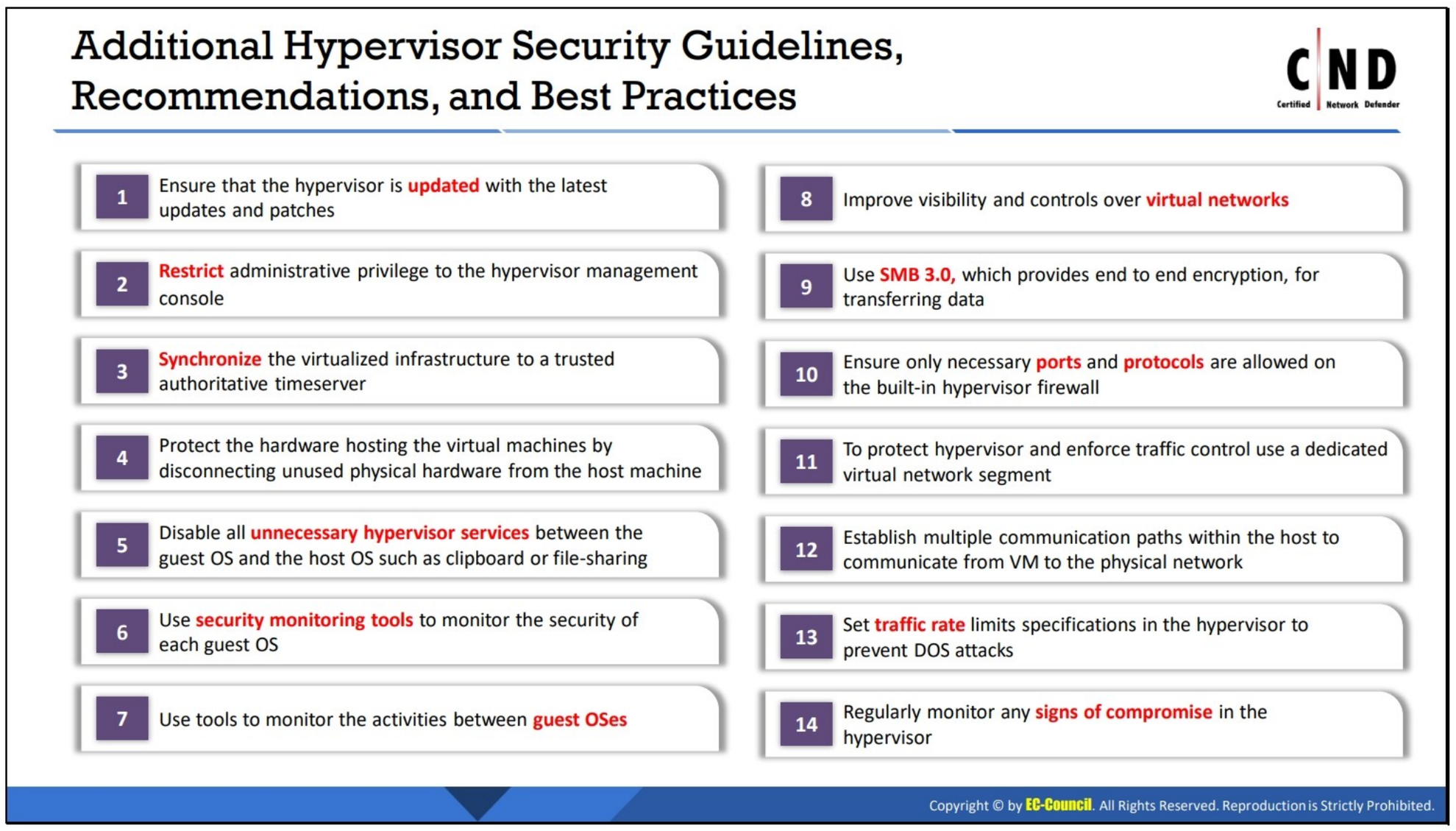
`--l1d-flush-on-vm-entry`

The above command flushes the level 1 data cache from every VM entry. While this may also reduce performance when compared to the default option, the actual impact depends on the workload.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24



Additional Hypervisor Security Guidelines, Recommendations, and Best Practices

The following are some additional hypervisor security guidelines, recommendations and best practices

- Install all updates to the hypervisor when the vendor releases them.
- Restrict administrative privileges to the hypervisor management console.
- Synchronize the virtualized infrastructure to a trusted authoritative time server.
- Protect the hardware hosting the virtual machines by disconnecting unused physical hardware from the host machine.
- Disable all unnecessary hypervisor services such as the clipboard or file-sharing, between the guest OS and the host OS.
- Monitor the security of each guest OS using tools.
- Monitor the security of activities between guest OSes using tools.
- Monitor the hypervisor for signs of compromise.
- Improve visibility and controls over virtual networks.
- Use server message block (SMB) for transferring of data, as it provides end-to-end encryption and protects the data from eavesdropping attacks.
- Configure the built-in hypervisor firewall to allow only necessary ports and protocols.
- To protect the hypervisor and enforce traffic control, use a dedicated virtual network segment.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

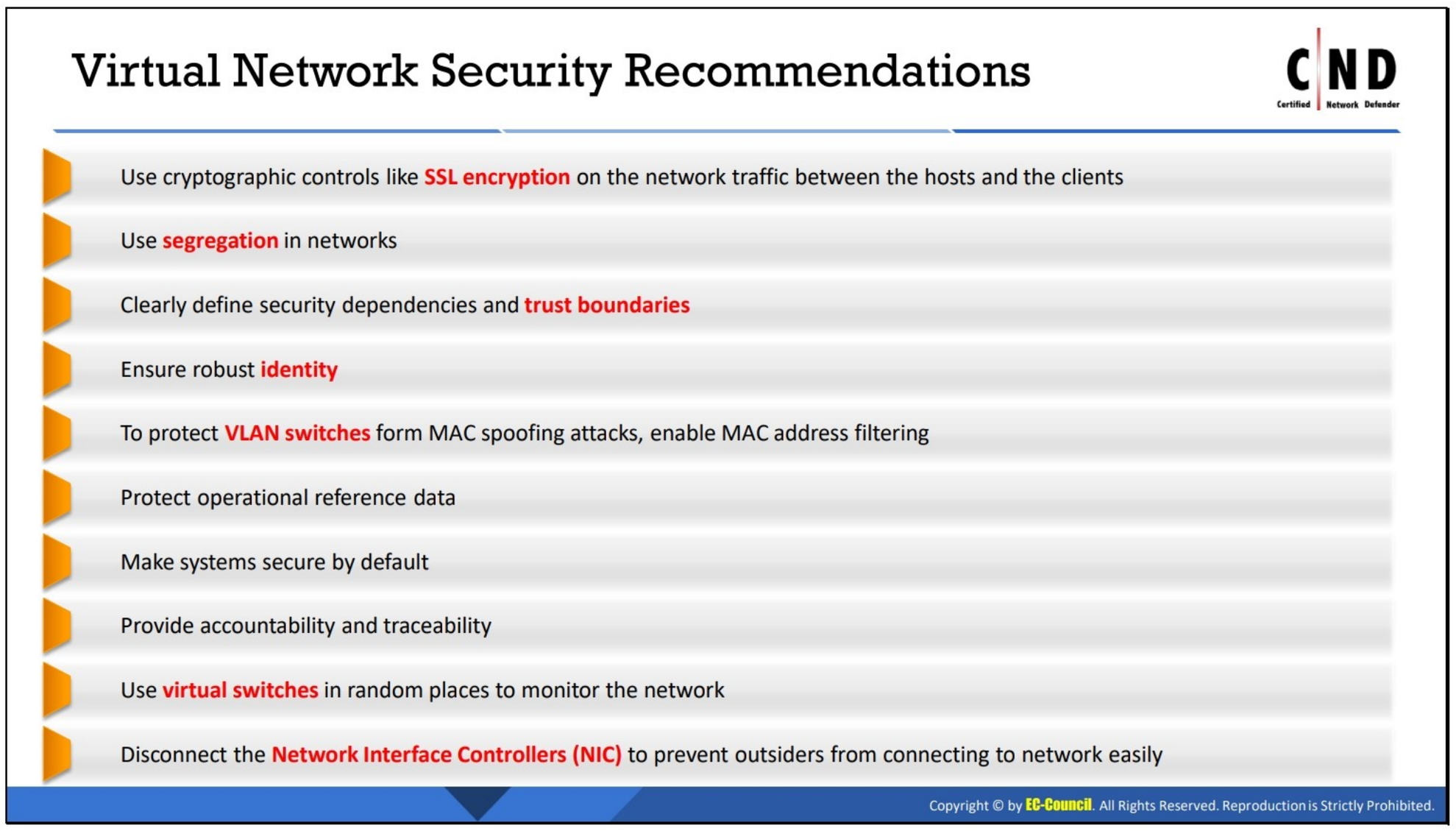
https://t.me/learningnets

- Establish multiple communication paths within the host to communicate from VMs to the physical network.
- Set traffic rate limit specifications in the hypervisor to prevent DoS attacks.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24



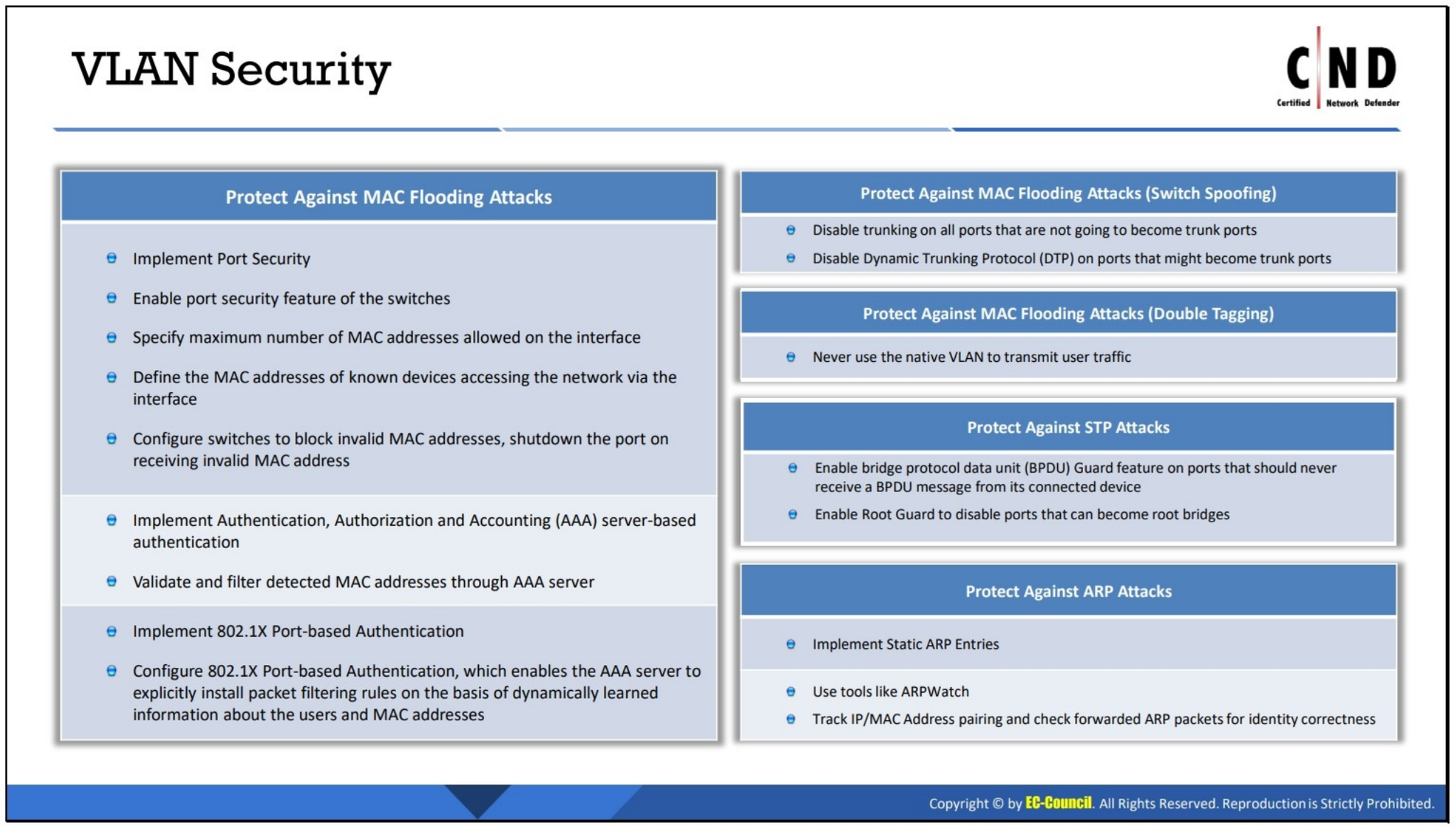
Virtual Network Security Recommendations

The following are some virtual network security recommendations:

- Use cryptographic controls like SSL encryption on the network traffic between the hosts and the clients.
- Use segregation in networks.
- Clearly define security dependencies and trust boundaries.
- Assure a robust identity.
- Ensure security on open standards.
- Protect operational reference data.
- Make systems secure by default.
- Provide accountability and traceability.
- Provide manageable security controls.
- Use virtual switches in random locations to monitor the network. To protect the switches form MAC spoofing attacks, enable MAC address filtering.
- Disconnect network interface controllers (NIC) to prevent outsiders from connecting to the network easily.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



VLAN Security

Protecting VLAN against MAC Flooding Attacks

- **Implement Port Security:** The aim of port security is to safeguard the switch interface from an attacker, who may send multiple ethernet frames with fake MAC addresses. This technique enables the network administrator to statically define MAC addresses for the port or to configure the switch to dynamically determine a limited number of MAC addresses. Implementing the port security feature not only safeguards the switch from MAC flooding attacks, but also from DHCP starvation attacks. However, the port security feature works only for the access port, and not for the trunk port.
- **Implement AAA server based Authentication:** Authentication, authorization, and accounting (AAA) security features validate the identity, grant access, and track the actions of the users. Servers with this feature possess central management capability. In this technique, detected MAC addresses are validated and filtered through authentication, authorization, and accounting functions by the server. To configure AAA authentication, the user must specify a list of authentication methods, and then implement that list across different interfaces.
- **Implement 802.1X Port-based Authentication:** This provides identity-based access control, visibility, and security at the network edge. Once the 802.1X port-based authentication is configured, the AAA server explicitly installs packet filtering rules on the basis of dynamically learned information about the users and MAC addresses.

Protecting VLAN against ARP Attacks

- **Implement Static ARP entries**: To minimize the risk of ARP spoofing, static ARP entries are created in the ARP cache. A static ARP entry is a constant entry in the ARP Cache. To

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets
Technet24

prevent ARP spoofing, the user can configure static ARP entries for ARP inspection. This prevents an adversary from sending an ARP response from their MAC address, provided the static ARP table consists of correct MAC addresses and their associated IP addresses.

Protecting VLAN against VLAN Hopping Attacks

The ports of some switches automatically turn into trunks when they receive DTP frames. This poses a significant security threat and can lead to VLAN hopping attacks. In order to protect a VLAN against VLAN hopping attacks, the user must

- **Prevent switch spoofing**: To prevent switch spoofing, the user must disable trunking on all ports that are not going to be trunks, and also disable DTP on the ports that are or might potentially be trunks.
- **Prevent double tagging**: The native VLAN should not be used to send user traffic. This helps in preventing VLAN hopping attacks that are conducted by double tagging.

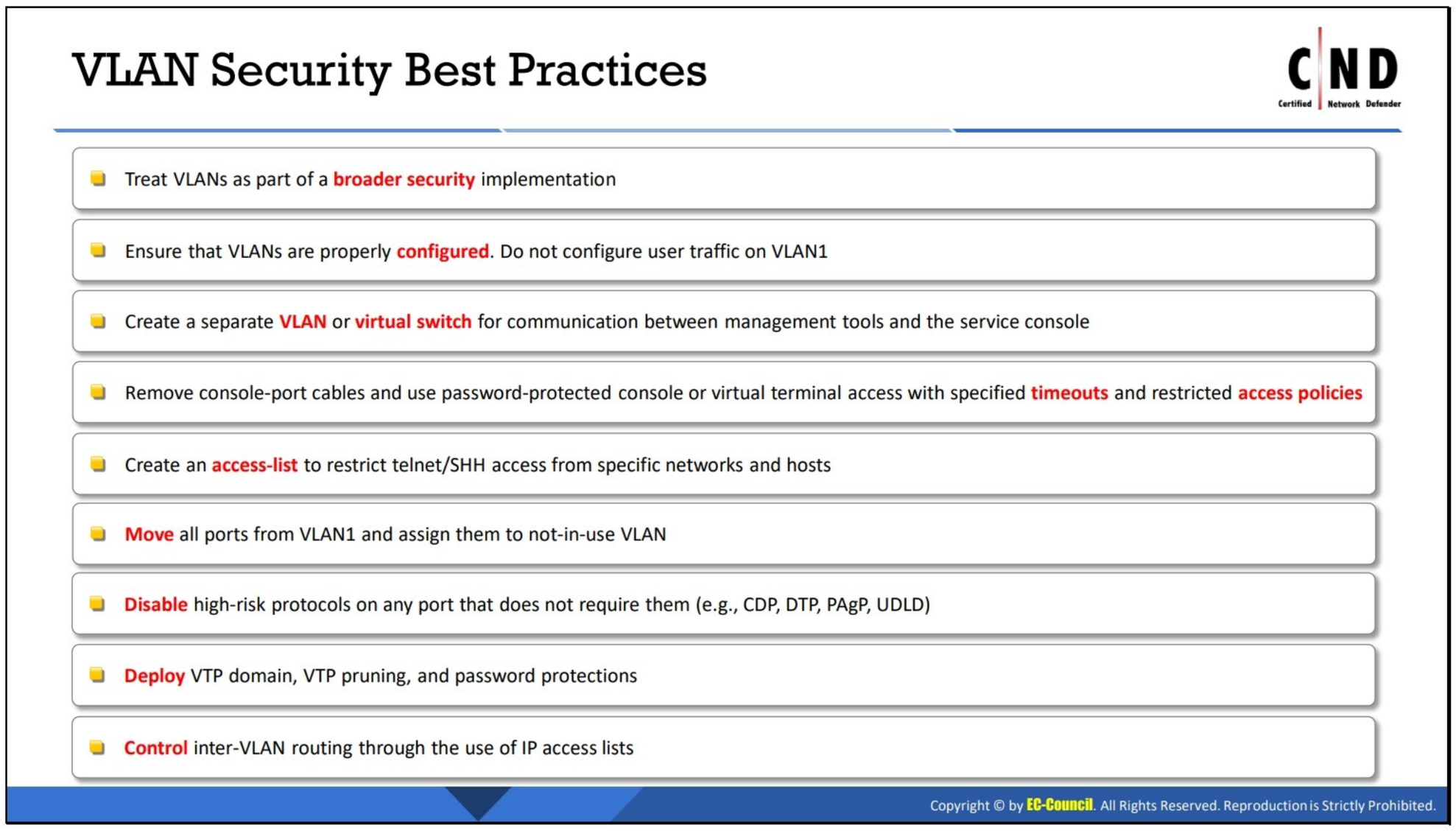
Protecting VLAN against STP Attacks

Implementing BPDU guard, BPDU Filter, root guard, loop guard, and UDLD features protect against STP attacks.

- **BPDU Guard**: This feature prevents the accidental connection of switch ports with those that are PortFast-enabled and prevents layer 2 loops or topology changes.
- **BPDU Filter**: This disables STP on selected ports by stopping the port from sending and receiving BPDUs.
- **Root Guard**: This prevents the switches that are configured as access ports from becoming the root switch.
- **Loop guard and UDLD**: These prevent bridging loops that can occur due to unidirectional links.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



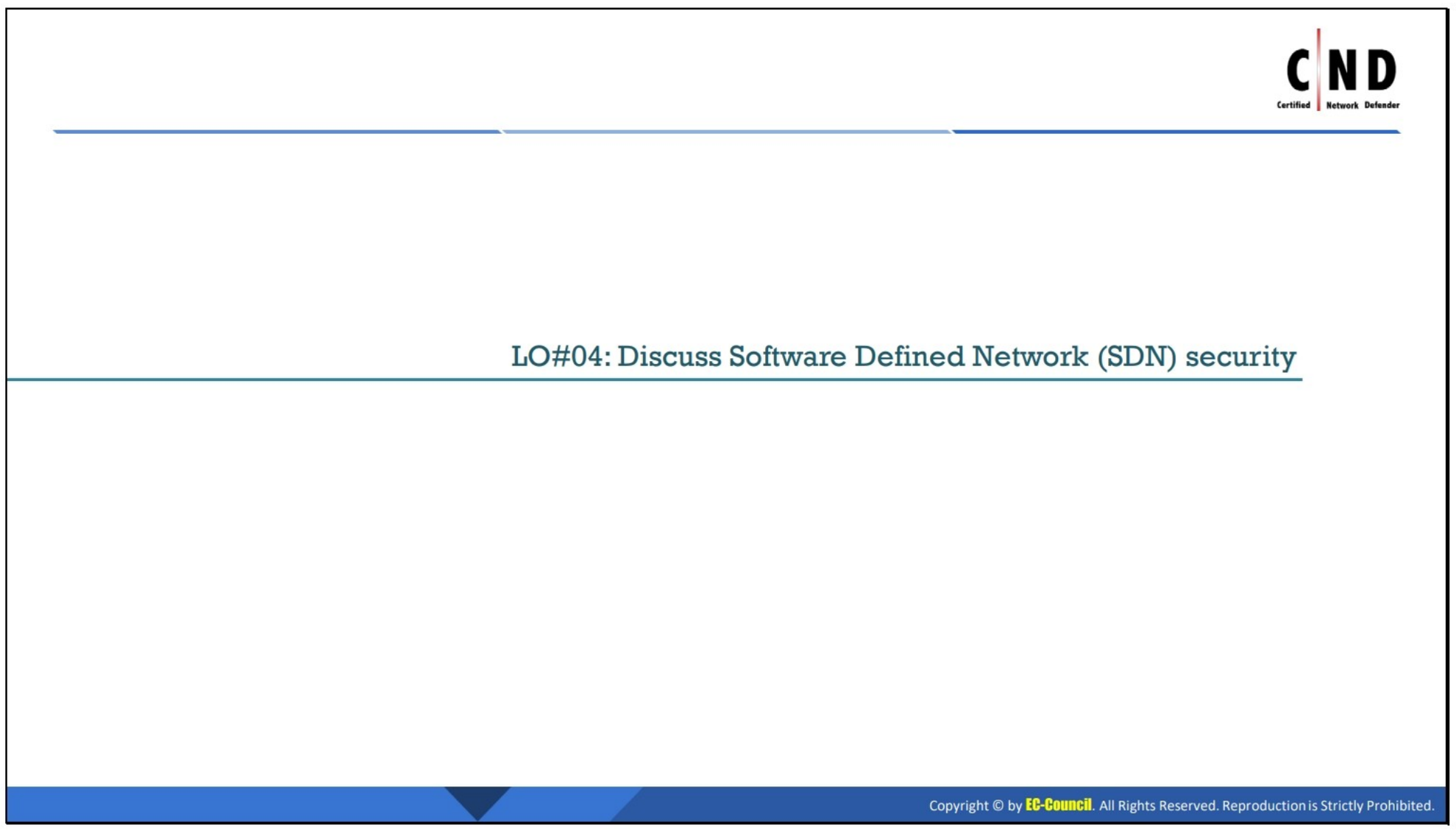
VLAN Security Best Practices

- Treat VLANs as part of broader security implementation.
- Ensure that VLANs are properly configured. Do not configure user traffic on VLAN1.
- Create a separate VLAN or virtual switch for communication between management tools and the service console.
- Remove console-port cables and use password protected console or virtual terminal access with specified timeouts and restricted access policies.
- Create an access-list to restrict telnet/SHH access from specific networks and hosts.
- Move all ports from VLAN1 and assign them to not-in-use VLAN.
- Disable high-risk protocols on any port that does not require them (e.g., CDP, DTP, PAgP, UDLD).
- Deploy VTP domain, VTP pruning, and password protections.
- Control inter-VLAN routing through the use of IP access lists.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

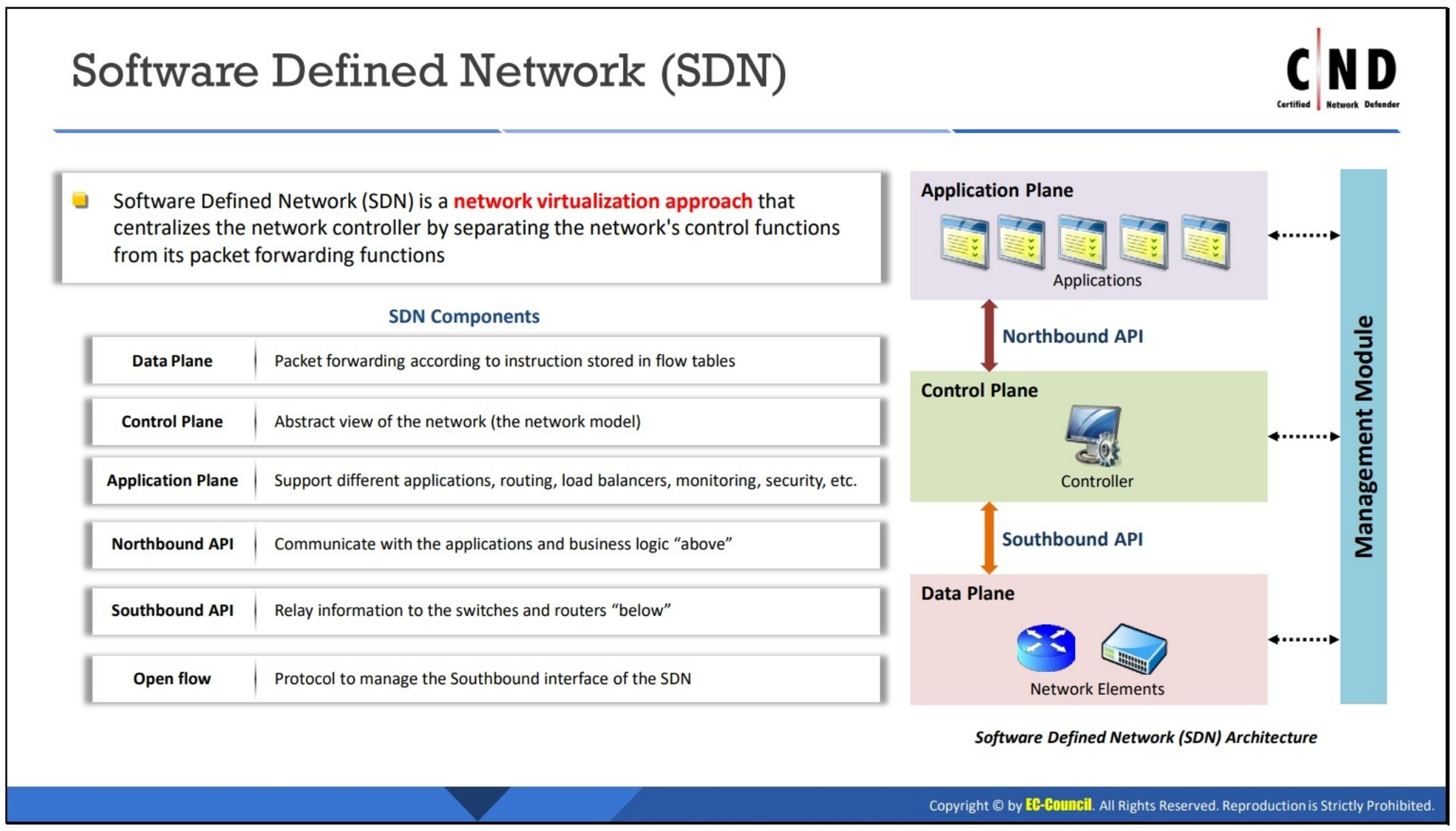


LO#04: Software Defined Network (SDN) Security

Many enterprises and service providers have been switching to software-defined networking (SDN) technology for consistency in the management of the network and devices across the entire network. SDN, therefore, is an approach to network management that differs from traditional network management. The objective of this section is to explain SDN, its benefits, and its limitations. The module also discusses SDN specific vulnerabilities and attacks, and security measures that can be adopted for securing SDN implementations.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Software Defined Network (SDN)

Software-defined networking (SDN) is a network framework, which centralizes the control of a network by separating the network's control functions from its data packet forwarding functions. Thus, it centralizes intelligence and reduces the complexity of the traditional network architecture for applications and services.

Due to the rapid growth in multimedia content, cloud computing, mobile technology, etc., many enterprises, carriers, and service providers have been switching to SDN technology, which enables them to manage their entire network and devices in a consistent manner.

SDN Architecture: There are three layers in the SDN architecture, which are as follows:

- **SDN Application**: The SDN application layer consists of programs, which communicate with the SDN controller through SDN APIs. It consists of all the applications and services that run on the network.
- **SDN Controller**: The SDN controller is the centralized seat of control of the network, and is therefore metaphorically the brain of the network. It is a logical entity that transfers information from SDN applications to the networking components. The SDN controller also extracts and transfers information from hardware devices to SDN applications.
- SDN Networking Devices: SDN networking devices such as switches, routers, and supporting hardware regulate the forwarding and data processing capabilities.

SDN Components: The key conceptual components of SDN are as follows:

- **Data Plane:** This refers to packet forwarding according to instructions stored in flow tables.
- **Control Plane:** This is the abstract view of the network (the network model).

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

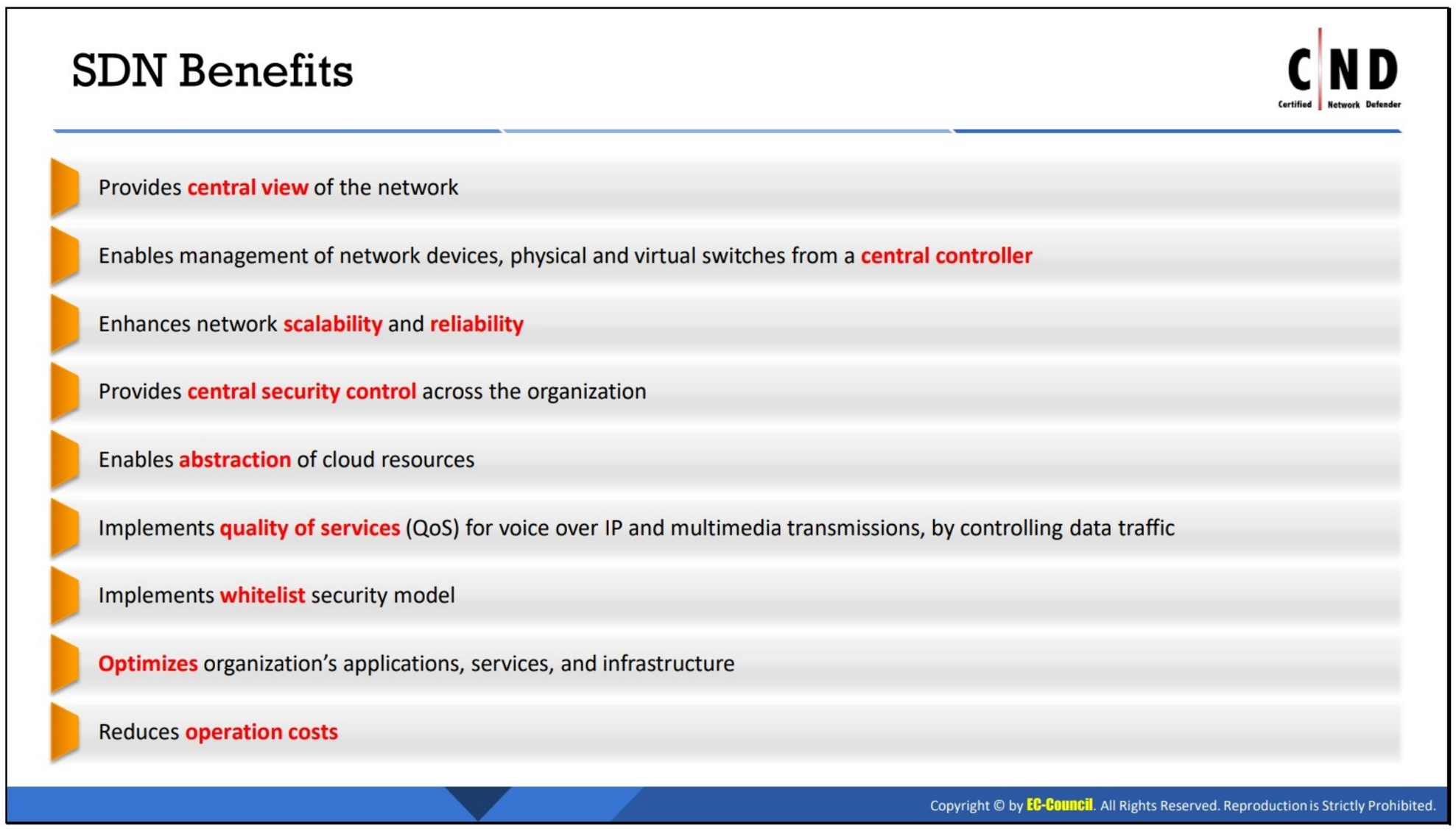
https://t.me/learningnets

Technet24

- **Application Plane:** This supports different applications such as routing, load balancers, monitoring, and security.
- **Northbound API:** This connects the SDN application layer and SDN controller, and enables communication between the network services and business applications.
- **Southbound API:** This connects the SDN controller and SDN networking devices, and relays information from network services to network devices such as switches and routers.
- **Open flow:** This is a protocol used to manage the southbound interface of the SDN.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



SDN Benefits

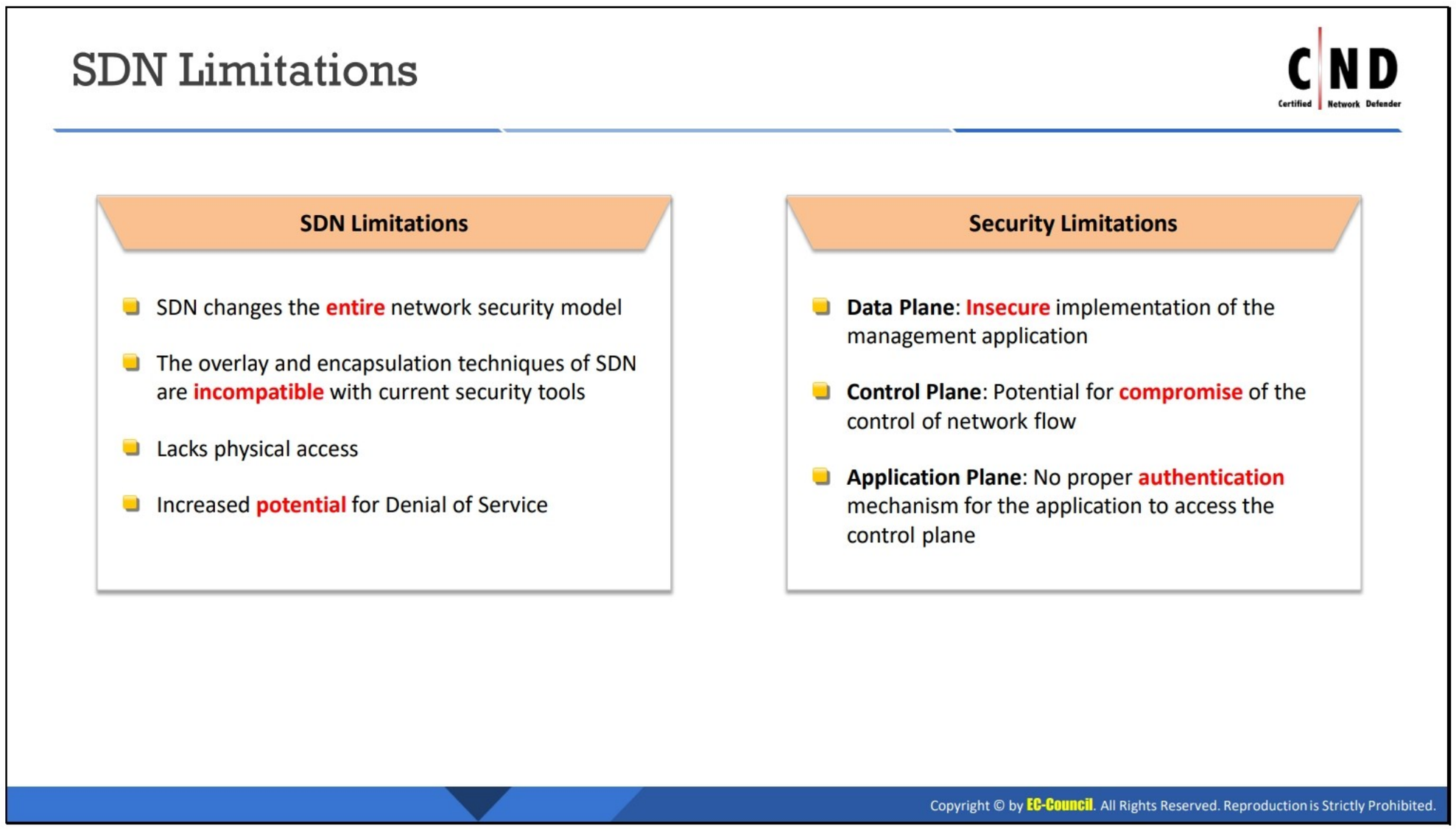
The following are benefits of SDN Technology

- **Network Programmability**: SDN enables the network operator to introduce new services and write programs by utilizing SDN APIs to control network behavior.
- **Logically centralize intelligence and control**: Traditional networks are based on a distributed control architecture that only permitted a low level of awareness about the state of the network. By contrast, SDN is based on logically centralized network topologies, which enables intelligent control and management of the network resources.
- **Abstraction of the network**: The services and applications in SDN are abstracted from the underlying technologies and communicate with the network via APIs.
- **Openness**: SDN provides a common software environment to run network services and applications. The open APIs of SDN support applications such as OSS/BSS, SaaS, cloud orchestration, and business-related applications.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24



SDN Limitations

The following are some limitations of SDN Technology

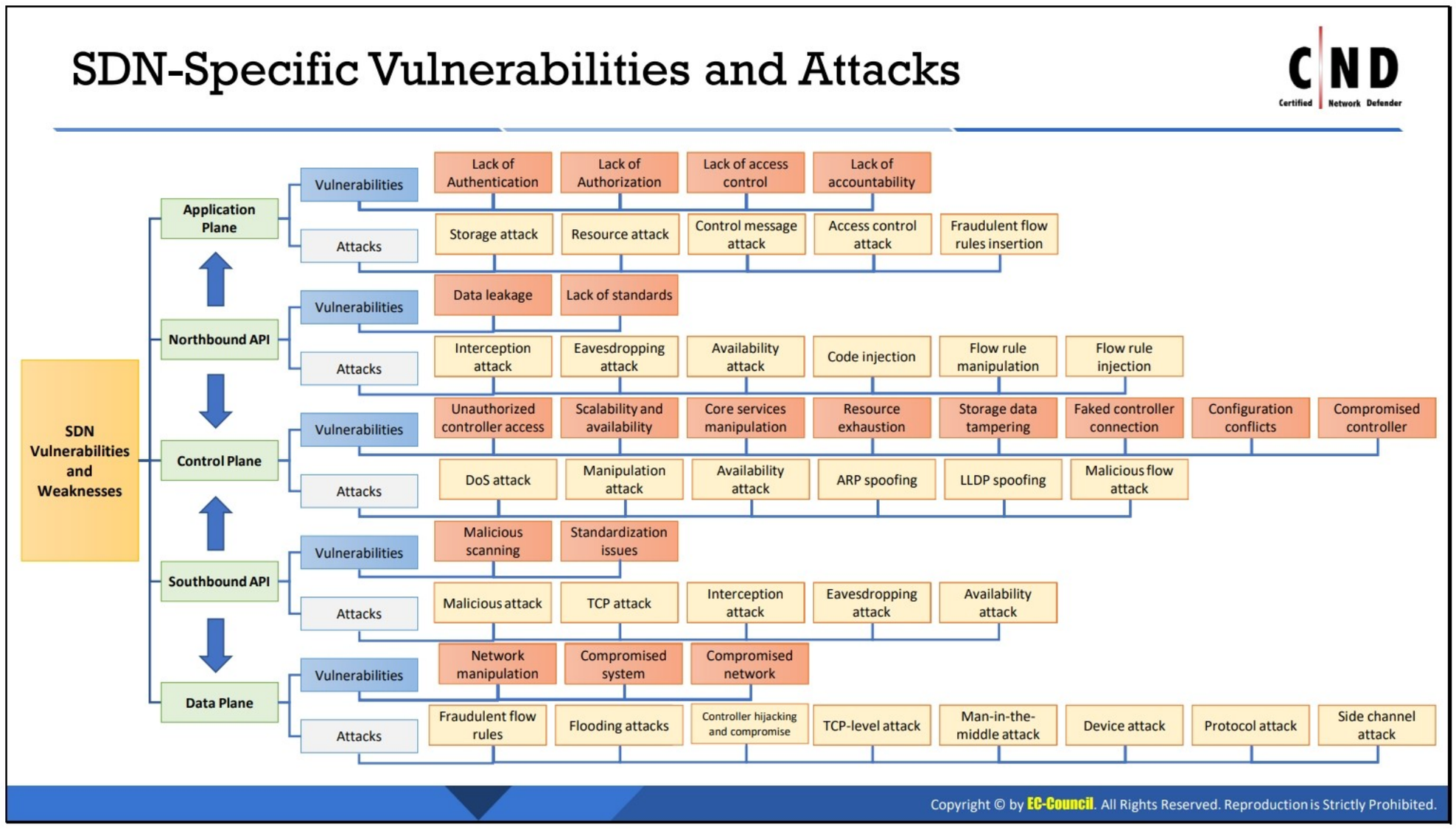
- Implementing the SDN protocol and SDN controller requires reconfiguration of the network infrastructure
- It requires the incorporation of new tools and trained employees to use them.
- SDN is virtual and lacks the security that the router, switches, and firewalls provide.
- The SDN security model is different from the traditional security model as it lacks physical access. SDN technologies are vulnerable to various attacks like DoS.
- SDN security risks are distributed across the control and data plane, and can possibly make the network vulnerable to various attacks.

The following are some security limitations of SDN.

- An insecure implementation of applications for managing the data plane can cause the data plane to be vulnerable to various attacks like DDoS and side-channel attacks.
- The controller must be fully secured, as a compromised controller can lead to various security loopholes.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



SDN-Specific Vulnerabilities and Attacks

The SDN architecture layers are the most common targets of various attacks. The potential vulnerabilities of the SDN architecture are summarized in the table below.

SDN Layer	Category of Attack	Type of Attack
Application Layer	Unauthorized Access	Unauthorized/Unauthenticated Application Attack
	Malicious/Compromised Application	Fraudulent Rule Insertion
	Configuration Issues	Lack of TLS Adoption
		Policy Enforcement
		Lack of Secure Provisioning
Application-Control Layer Interface	Unauthorized Access	Unauthorized/Unauthenticated Application Attack
	Malicious/Compromised Application	Fraudulent Rule Insertion
	Configuration Issues	Lack of TLS Adoption
		Policy Enforcement
		Lack of Secure Provisioning
Control Layer	Unauthorized Access	Unauthorized Controller Access/Controller Hijacking

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

		Unauthorized/Unauthenticated Application
	Data Leakage	Forwarding Policy Discovery (Packet Processing Timing Analysis)
	Data Modification	Flow Rule Modification to Modify Packets (Man-in-the-Middle attack)
	Malicious/Compromised Applications	Fraudulent Rule Insertion
	Denial of Service	Controller-Switch Communication Flood
	Configuration Issues	Lack of TLS Adoption
		Policy Enforcement
		Lack of Secure Provisioning
	System-Level SDN Security	Lack of Visibility of Network State
Control-Data Interface	Unauthorized Access	Unauthorized Controller Access/Controller Hijacking
	Data Leakage	Forwarding Policy Discovery (Packet Processing Timing Analysis)
	Data Modification	Flow Rule Modification to Modify Packets (Man-in-the-Middle attack)
	Denial of Service	Controller-Switch Communication Flood
	Configuration Issues	Lack of TLS Adoption
		Lack of Secure Provisioning
	System-Level SDN Security	Lack of Visibility of Network State
Data Layer	Unauthorized Access	Unauthorized Controller Access/Controller Hijacking
	Data Leakage	Flow Rule Discovery (Side-Channel Attack on Input Buffer)

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

		Credential Management (Keys, Certificates for each Logical Network)
		Forwarding Policy Discovery (Packet Processing Timing Analysis)
	Data Modification	Flow Rule Modification to Modify Packets (Man-in-the-Middle attack)
	Denial of Service	Controller-Switch Communication Flood
		Switch Flow Table Flooding.
	Configuration Issues	Lack of TLS Adoption.
		Lack of Secure Provisioning
	System-Level SDN Security	Lack of Visibility of Network State

Table 11.1: SDN-Specific Vulnerabilities and Attacks

Based on the components impacted, SDN attacks are broadly classified into the following types:

- **Data Plane**: The data plane is compromised through three major attacks.
  - **Device Attack**: Cybercriminals target the software or hardware vulnerabilities of the SDN switch. An adversary can target software bugs such as firmware attacks or hardware features such as the ternary content-addressable memory (TCAM) of the device.
  - **Protocol Attack**: The attacker targets network protocol vulnerabilities of the forwarding device.
  - **Side Channel Attack**: The adversary deduces the network forwarding policy based on the performance metrics of the forwarding device.
- **Southbound API**: The southbound API is compromised through three major types of attacks.
  - **Interception Attacks**: The adversary targets network behavior by modifying the exchanged messages.
  - **Eavesdropping Attacks**: The attacker targets the information exchanged between the control and data plane.
  - **Availability Attacks**: The cybercriminals send numerous requests, which lead to failure in the implementation of network policy.
- **Control Plane**: There are three major types of attacks on the control plane.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

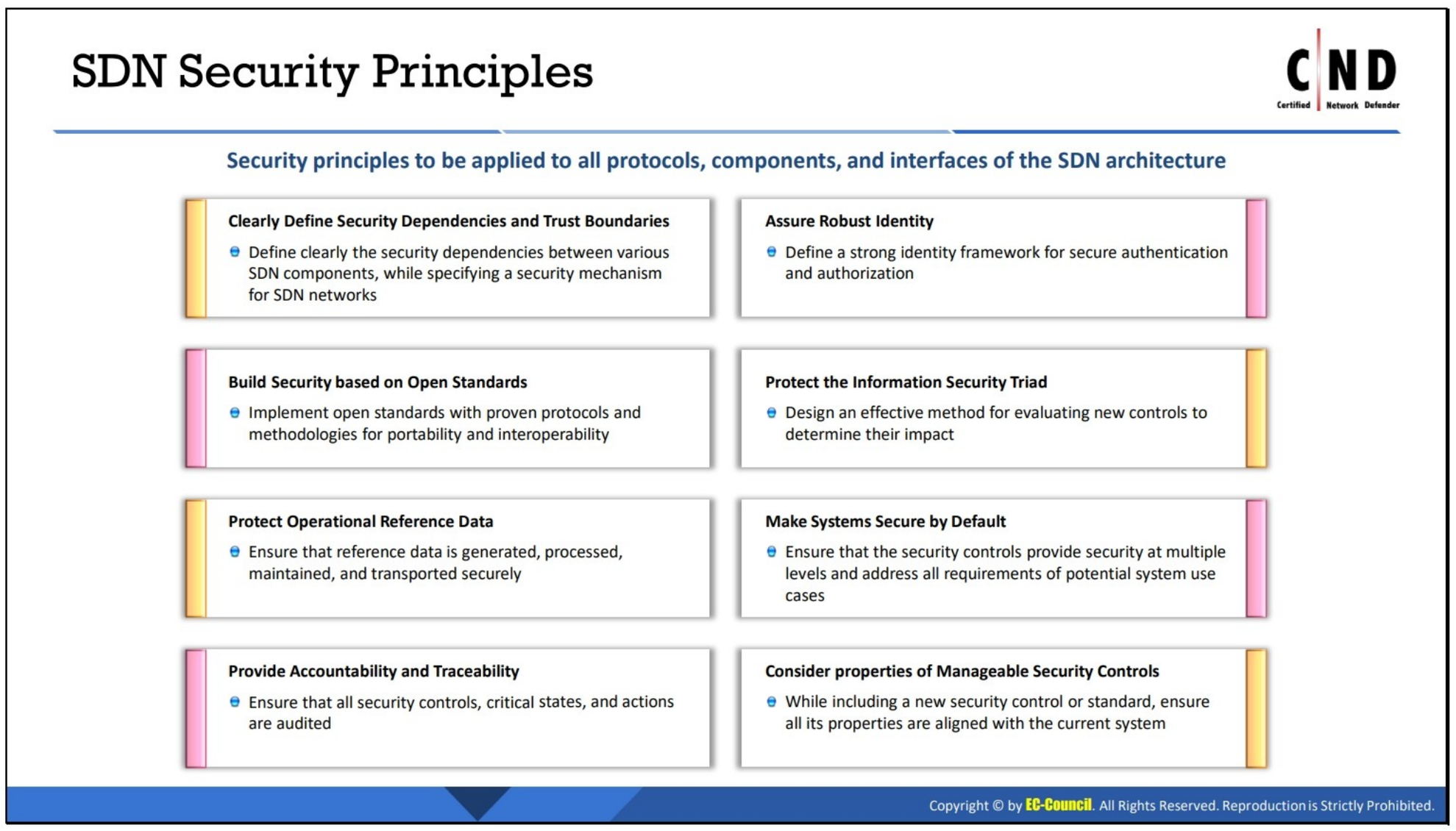
Technet24

- **Manipulation Attack**: In this type of attack, the adversary targets the controllers' understanding of the data plane leading to improper decision making.
- **Availability Attack**: In this type of attack, the controller is not available for a certain period of time for part or all of the network. This might be effected, for instance, by sending numerous unauthenticated packet-in messages to the controller.
- **Software Hack**: Since the SDN controller is hosted on a commodity server, it is vulnerable to software hacks. For instance, the alteration of a system variable like time may cause the controller to go offline.

- **Northbound API**: The northbound API is compromised through interception attacks, eavesdropping attacks, and availability attacks, whereas the southbound API is compromised through other kinds of malicious attacks and standardization issues. While, the nature the attacks on the two APIs are similar, there are certain differences:
  - The adversary targeting the northbound API requires high level of access and operates in the application plane of the system.
  - Since the information exchanged between the application plane and control plane affects the network policies, the impact of a compromised northbound layer is potentially higher compared to that of a compromised southbound API.
  - OpenFlow is the standard protocol for the southbound API, whereas the northbound API lacks any standards.
- **Application Plane**: The following are policy attacks targeting the application plane:
  - **Storage Attack**: The adversary targets the access privilege of SDN applications to access shared storage.
  - **Control Message Attack**: In this type of attack, a malicious application targets the network control by sending control messages.
  - **Resource Attack:** The adversary targets system resources such as memory and central processing unit, affecting the performance of the application and controller.
  - **Access Control Attacks**: In access control attacks, the adversary targets the controller when there is improper enforcement of control over authentication, authorization, and accountability.
- **Controller**: Policy attacks conducted by the adversary in order to compromise the controller can be executed through malicious actions and configuration conflicts.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



SDN Security Principles

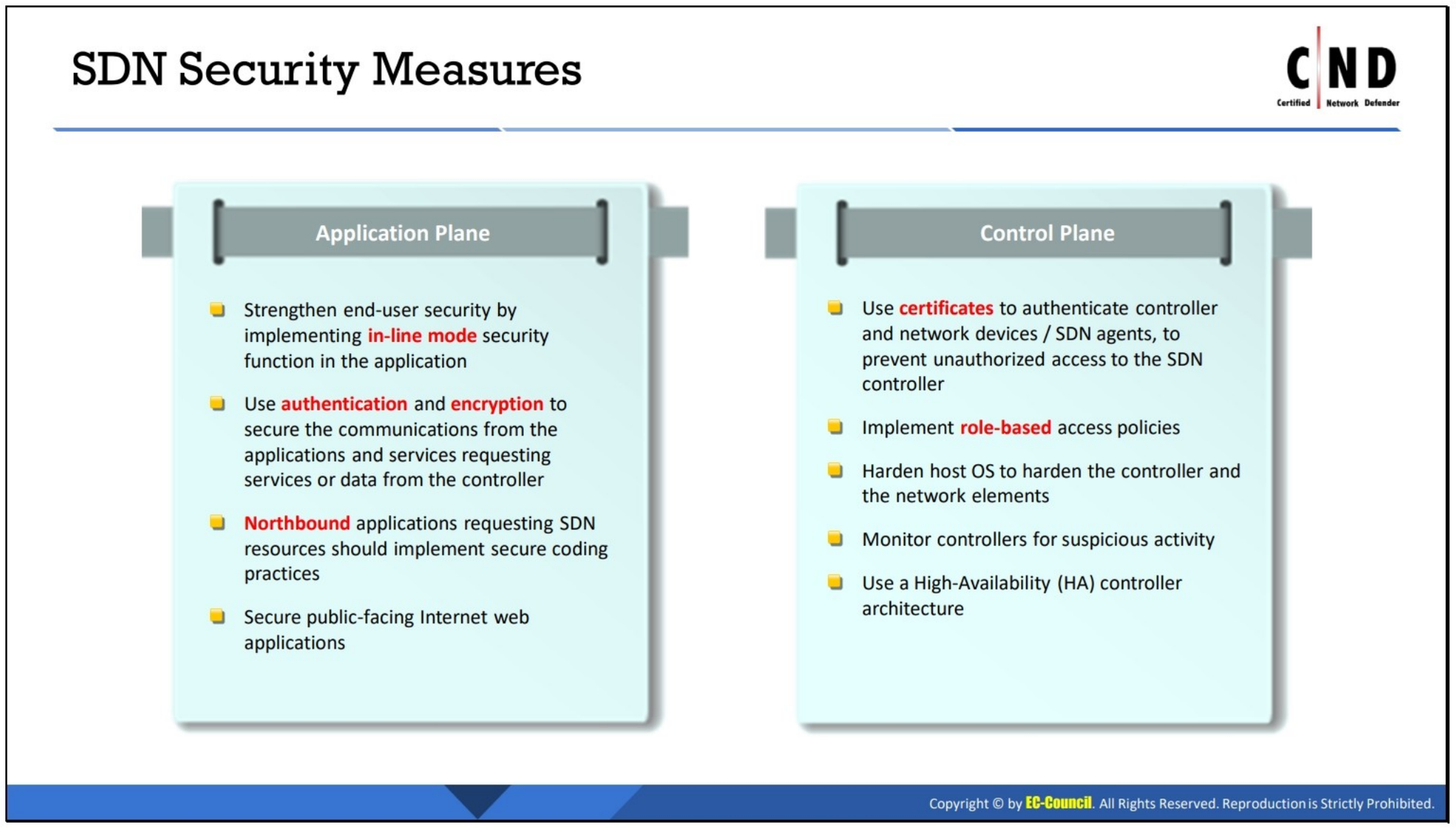
The following are security principles to be applied to all protocols, components, and interfaces of the SDN architecture.

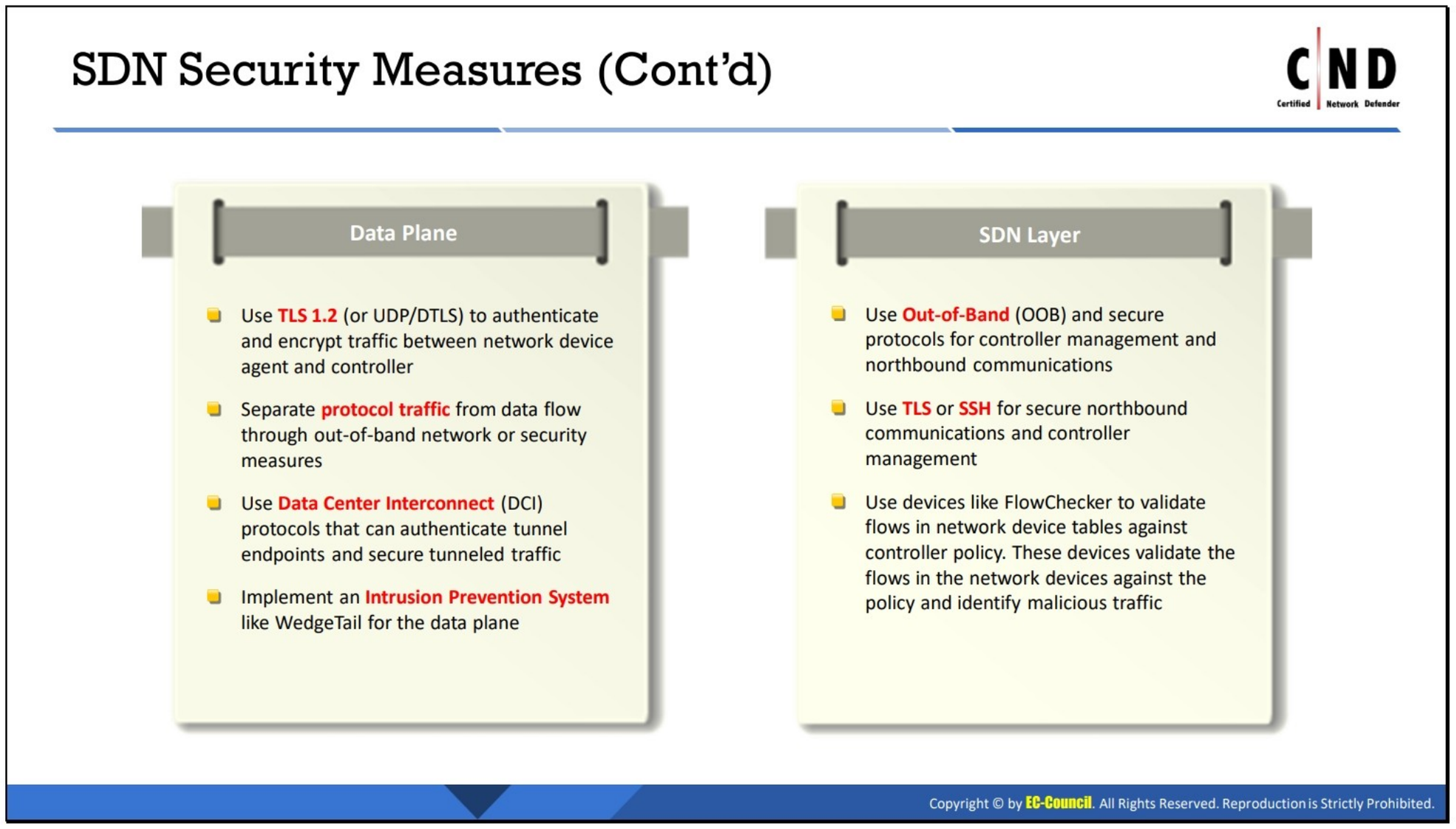
- **Clearly define security dependencies and trust boundaries:** Define clearly the security dependencies between various SDN components, while specifying a security mechanism for SDN networks.
- **Assure robust identity:** Define a strong identity framework for secure authentication and authorization.
- **Build security based on open standards:** Implement open standards with proven protocols and methodologies for portability and interoperability.
- **Protect the information security triad:** Design an effective method for evaluating new controls to determine their impact on the information security triad of confidentiality, integrity, and availability.
- **Protect operational reference data:** Ensure that reference data is generated, processed, maintained, and transported securely.
- **Make systems secure by default:** Ensure that the security controls provide security at multiple levels and address all requirements of potential system use cases.
- **Provide accountability and traceability:** Ensure that all security controls, critical states, and actions are audited.
- **Consider properties of manageable security controls:** While including a new security control or standard, verify that all its properties are aligned with the current system.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24





SDN Security Measures

The layers of an SDN system must be secured to prevent exposing an organization to attacks.

- **Securing the application plane layer**
  - Strengthen end-user security by implementing in-line mode security functions in the application.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

- Use authentication and encryption to secure the communications from the applications and services that request services or data from the controller.
- Northbound applications requesting SDN resources should implement secure coding practices.
- Secure public-facing Internet web applications.

- **Securing the data plane layer**
  - Implement the cryptographic transport layer security (TLS) protocol to authenticate and encrypt traffic between network devices, SDN agents, and controller, to prevent eavesdropping and spoofed southbound communications.
  - Secure communication by selecting one of the available options based on the southbound protocol currently in use. For example,
    - Use protocols within TLS sessions.
    - Use shared secret passwords or use nonce to avoid replay attacks.
    - Use SNMPv3 protocols instead of SNMPv2c.
    - Use secured shell (SSH) instead of telnet.
    - Use passwords and shared-secrets to authenticate tunnel endpoints, and secure tunneled traffic based on the data center interconnect (DCI) protocol currently in use.

- **Securing the controller layer**
  - Ensure the SDN systems permit configuration of secure and authenticated administrator access to controller. Ensure the controller administrators use role-based access control (RBAC) policies.
  - Implement logging and audit trails to check for unauthorized changes made by administrators or attackers.
  - Use a high-availability (HA) controller architecture if a risk of DoS attacks on the controller exists.
  - Avoid SDNS that use redundant controllers, as this may enable an attacker to cause DoS in all the controllers in the SDN system, while leaving the attacker undetected.

- **Securing the SDN layer**
  - Use an out-of-band (OOB) network for control traffic, which can serve as a convenient and inexpensive protection measure for data centers. Using this network for northbound and southbound communications can secure the protocols for controller management.
  - Use TLS, SSH, or any other method to protect northbound communications.
  - Secure the management of the controller.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

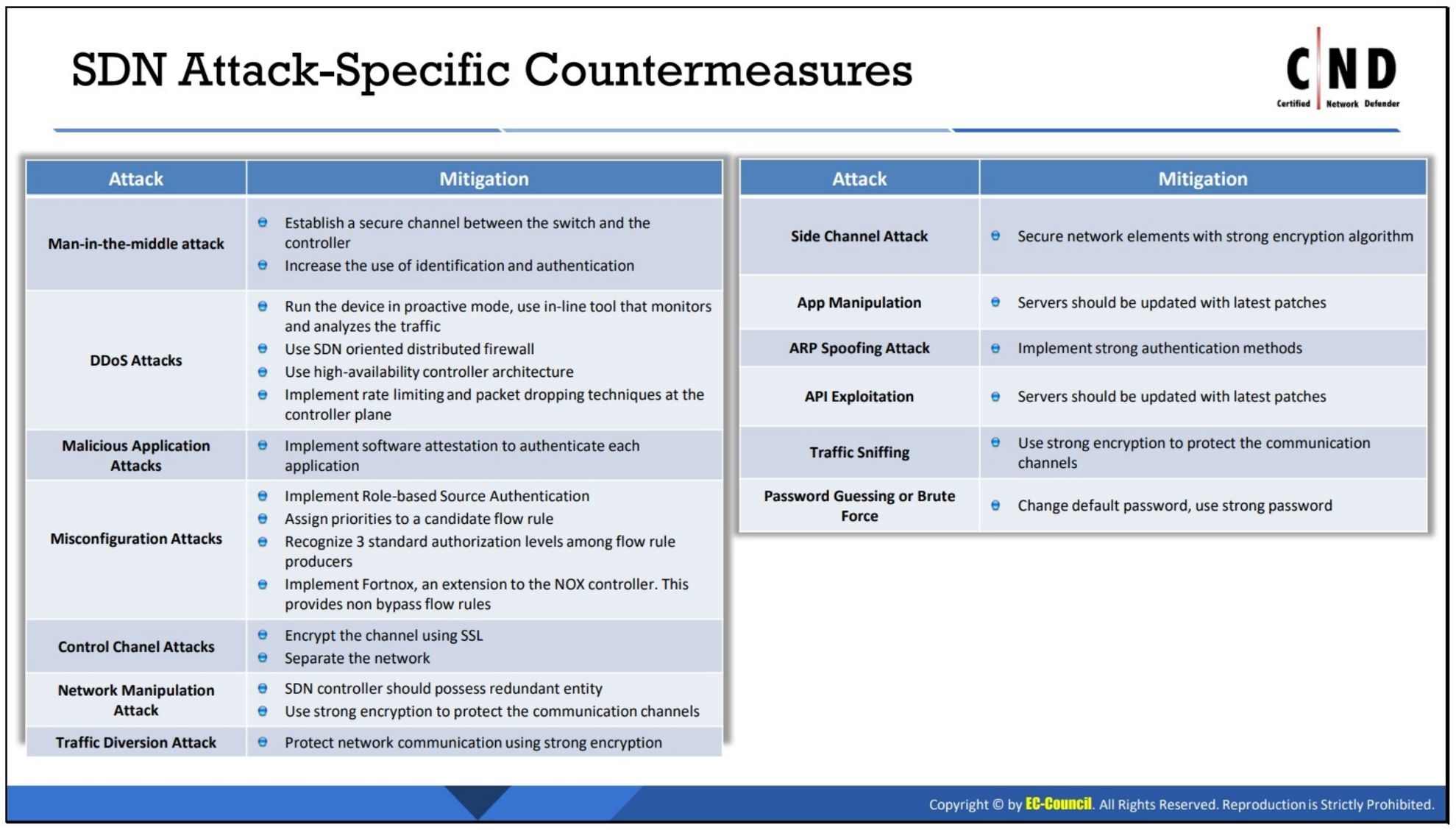
https://t.me/learningnets

Technet24

- Secure the communications from the applications and services requesting services or data from the controller by implementing authentication and encryption methods.
- Secure coding practices for northbound applications such as public-facing internet web applications that request SDN resources.
- Secure coding practices for northbound SDN connections.
- Use the SDN system's ability to validate flows in network device tables against controller policy. This can help identify any discrepancies that may be caused by an attack.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



SDN Attack-Specific Countermeasures

The following are additional recommended security measures that restrict access to the SDN controller to avoid unauthorized activity and prevent various attacks.

- Establish a secure channel between the switch and the controller.
- Increase the use of identification and authentication.
- Use audited and reviewed role-based access policies consistently.
- Use audited and reviewed configuration changes regularly.
- Monitor controllers for malicious activity and implement alerts to security staff.
- Follow best practices for hardening and patching the system. Otherwise, document, measure, and approve the risk and its impact.
- Prevent DDoS attacks by implementing high-availability controller architecture.
- Use high-availability (HA) in the architecture to examine the changes or updates in the production environment.
- Allow immediate failover with an option if a change fails to function.
- Use TLS/SSH (or UDP/DTLS) to encrypt northbound communication.
- Secure coding of northbound applications.
- Use authentication for northbound applications instead of default passwords before they communicate with the controller.
- Authenticate endpoints using TLS for southbound communication.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

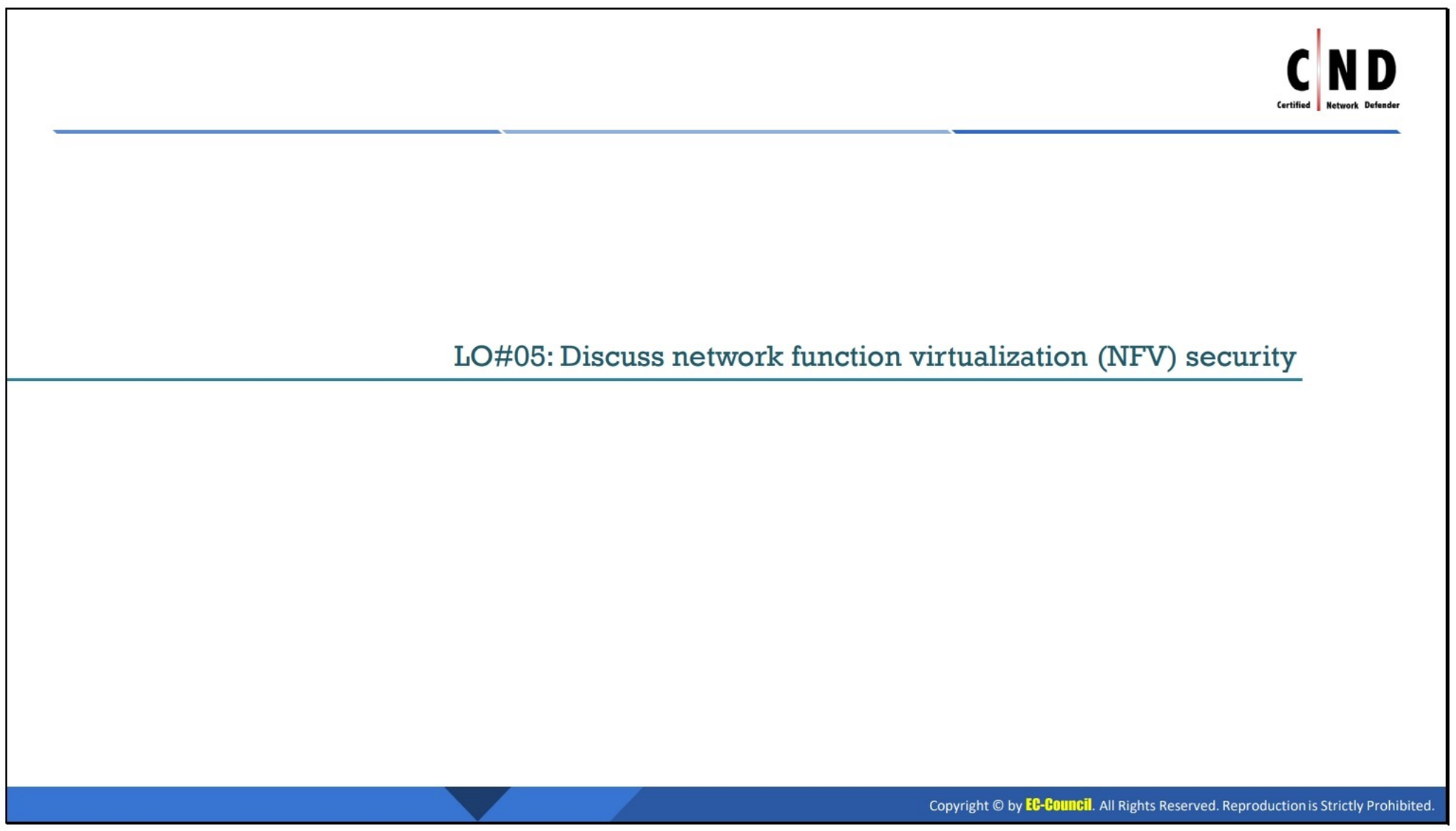
https://t.me/learningnets

Technet24

- Separate the control protocol traffic and primary data flows through an out-of-band (OOB) network.
- Authorize tunnel endpoints and protect tunneled traffic using data center interconnect (DCI) protocols.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



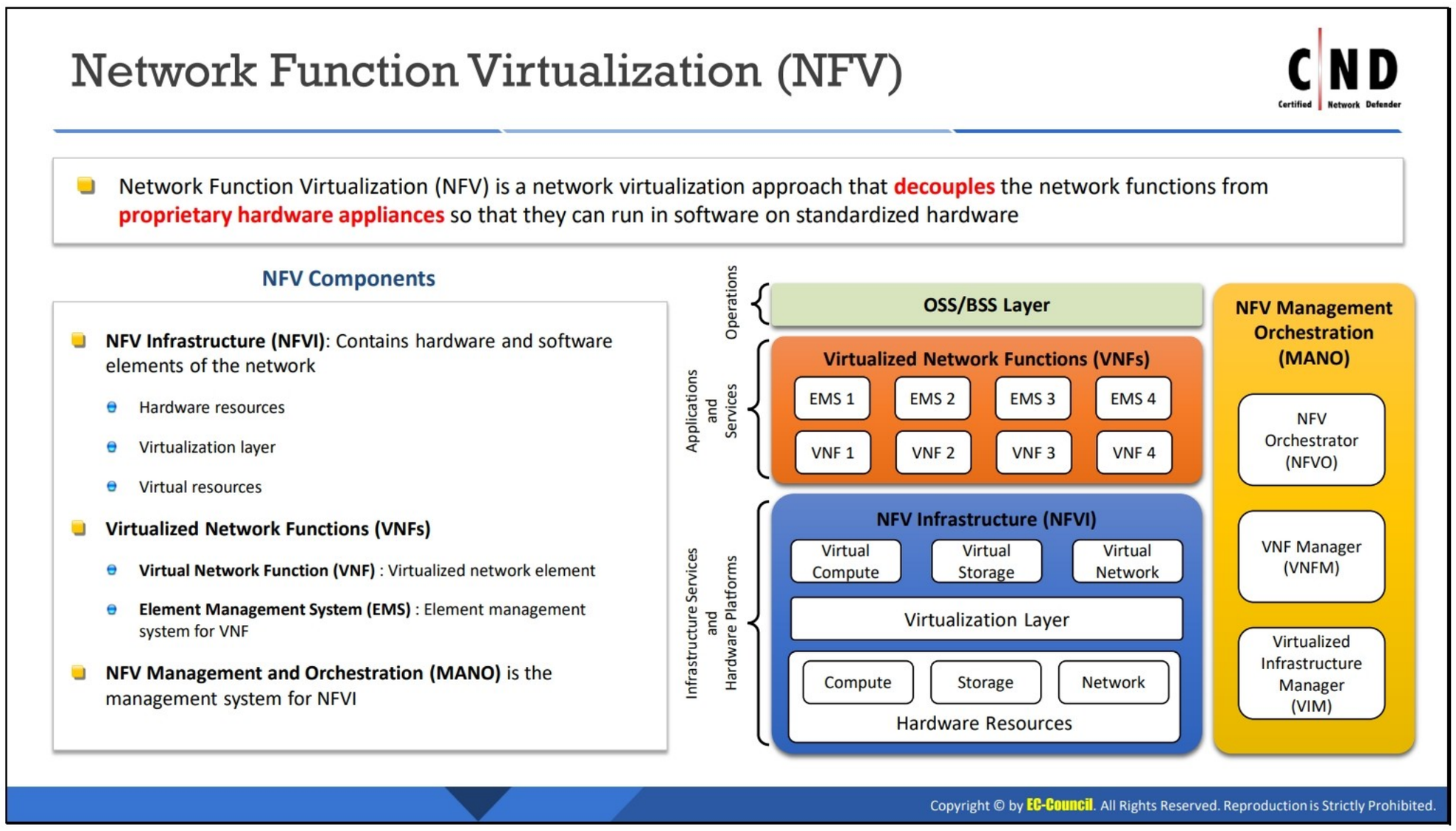
LO#05: Network Function Virtualization (NFV) Security

The objective of this section is to explain network function virtualization (NFV) and its components. The module also discusses vulnerabilities and attacks, and security measures for securing each NFV component.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24



Network Function Virtualization (NFV)

Network function virtualization (NFV) is a network virtualization approach, which separates network functions (NFs) such as firewalls, traffic control, and virtual routing from physical devices and runs them as software in virtual resources. By decoupling NFs from the proprietary hardware appliances, NFV minimizes operating expenses (OPEX) and capital expenses (CAPEX), and enables easy deployment of new services.

NFV Components: The NFV architecture consists of three principal elements

- **NFV Infrastructure (NFVI)**: This is the main component of the NFV architecture, and describes all the hardware and software components on which virtual network functions (VNFs) are deployed. NFVI standards enhance the interoperability of VNF components. There are three subparts in NFVI
  - **Hardware Resources**: These include hardware of network devices, servers, storage, etc., which are used by VNFs.
  - **Virtualization Layer**: This is the layer containing the hypervisor in which virtualization is implemented.
  - **Virtual Resources**: This includes virtual networks, virtual storages, and virtual servers.
- **Virtualized Network Functions (VNFs)**: This is the implementation of software for the virtualization of network functions on virtual resources. The specific network functions running in virtual machines are handled by VNFs. There are two key elements in VNFs:
  - **Virtual Network Function (VNF)**: This virtualizes the network function. For example, if a router is virtualized, it is termed as a VNF router. VNFs are deployed on VMs. A VNF can

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

be deployed on multiple VMs in which each VM can host a single function of VNF, or the entire VNF can be deployed on a single VM.

- **Element Management System (EMS)**: This handles the functions related to the management of a VNF such as accounting, configuration, performance, and security management. The EMS uses a proprietary interface to manage the VNF. Multiple VNFs can be managed by a single EMS.

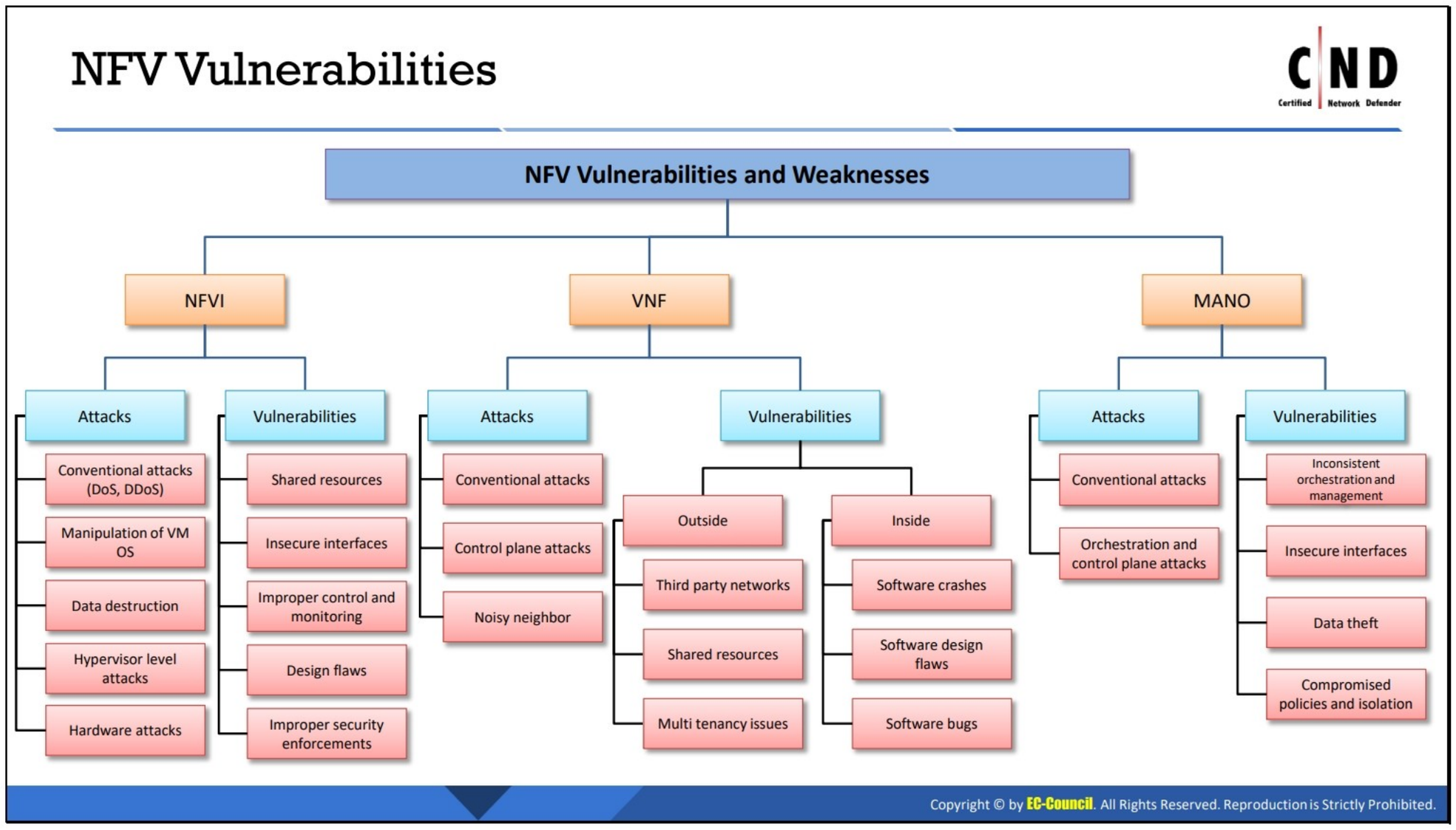
- **NFV Management and Orchestration (MANO)**: This communicates with the NFVI and VNF layers and manages the resources of NFVI and the allocation of VNFs. Standard interfaces are used to combine NFV MANO with the decoupled operation support subsystem (OSS) or business support system (BSS) of an operator. MANO consists of three components:

  - **Virtualized Infrastructure Manager (VIM)**: The function of a VIM is to control and manage the communication from the VNF to computing, storage, and network resources along with virtualization.
  - **VNF Manager**: This manages the actions in the life cycle of a VNF such as updates, query, installation, termination, and scale-up/down.
  - **Orchestrator**: This controls orchestration, and manages software resources and NFV infrastructure.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24



NFV Vulnerabilities

Each of the components of an NFV are vulnerable to various attacks, as described next.

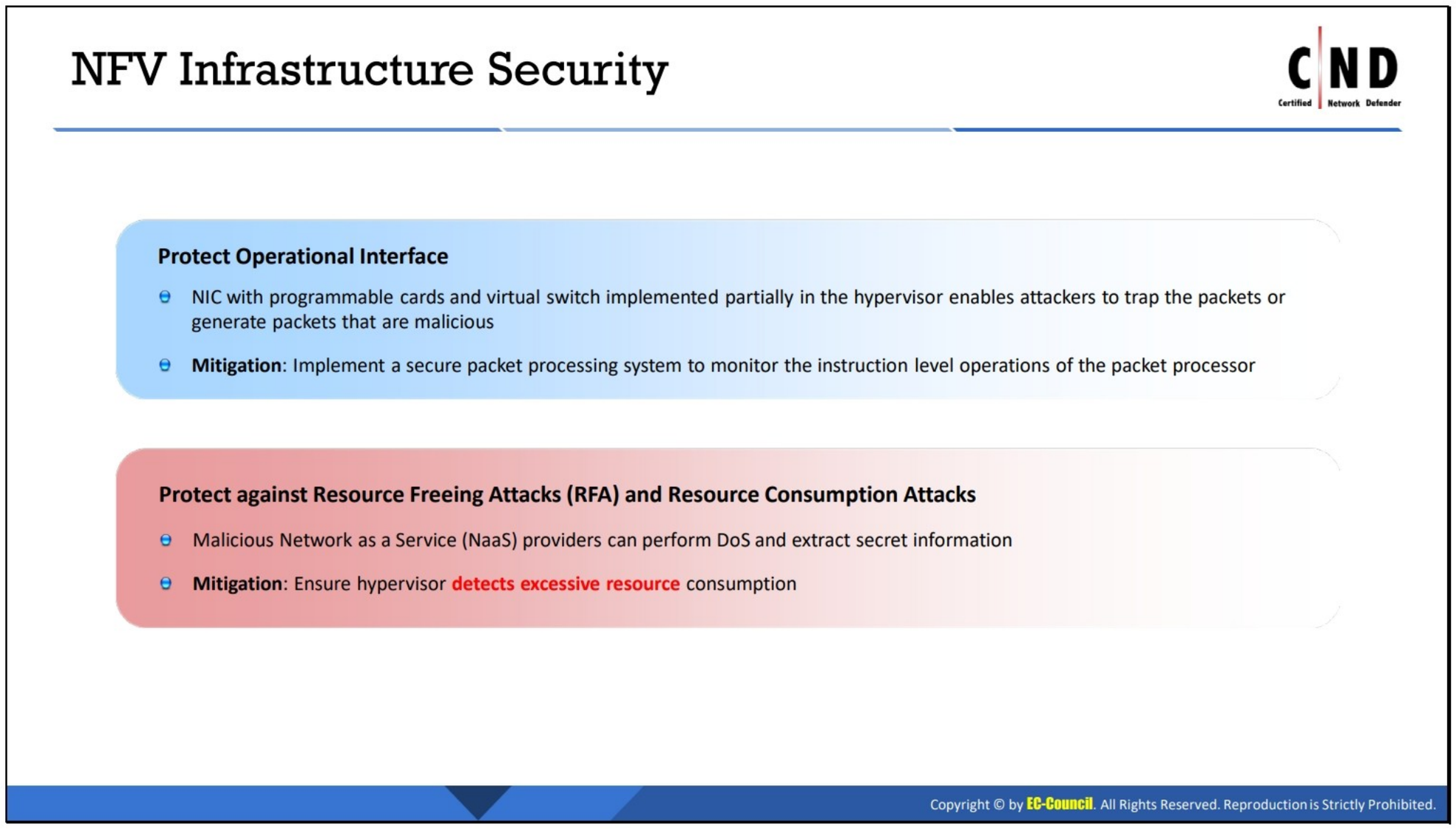
NFVI vulnerabilities and attacks: NFVI resources that are used by virtual network functions are managed by a VIM, which can be a target of attacks. The vulnerabilities in NFVI are shared resources, insecure interfaces, improper control and monitoring, design flaws, and improper security enforcements. The different types of attacks that can target NFVI are conventional attacks (DoS/DDoS), manipulation of VM OS, data destruction, hypervisor level attacks, and hardware attacks. The adversary targets the vulnerabilities of the hypervisors and compromises confidentiality, integrity, and the availability of the VNFs resources.

VNF vulnerabilities and attacks: The VNF could either be the source of an attack, or the target of an attacker. Since a VNF is a software component provided by a vendor, it can have software vulnerabilities, or may even be a malware designed to execute an attack. The vulnerabilities that may affect a VNF are software crashes, software design flaws, and software bugs. The resources that may be vulnerable in the event of a compromised VNF are shared resources, third party networks, and other tenants in the server.

MANO vulnerabilities and attacks: An adversary may either eavesdrop or modify the communications within an NFV MANO and the traffic between the NFVI and the NFV MANO. The vulnerabilities of MANO are inconsistent orchestration and management, insecure interfaces, data theft, compromised policies, and isolation. The potential attacks that could occur on the MANO are conventional attacks, orchestration, and control plane attacks. The adversary targets the orchestrator or the VNF manager in order to affect the network services or individual VNFs.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



NFV Infrastructure Security

A number of measures can be adopted to protect the NFV infrastructure. Broadly, in the hypervisor domain, ensure that the authentication is controlled and managed by the virtual machines to prevent unauthorized access and data leaks. In the compute domain, data should be encrypted and accessed only by the VNFs in shared computing resources. In the network domain, secure networking techniques such as TLS, IPSec and SSH must be adopted. The following are specific strategies and measures that can be adopted to secure the NFV infrastructure.

- **Defense-in-depth and well-defined policy enforcement**: These are required for operating and maintaining NFVIaaS layer security.
- **Network isolation and segmentation:** This is an effective strategy to protect the NFVIaaS layer and can be executed utilizing VLANs and VXLAN techniques. It safeguards external virtual machines from sniffing or monitoring internal traffic.
- **Security monitoring and intrusion detection**: This is a key countermeasure used to detect threats, identify suspicious events, and minimize security attacks on the NFVIaaS layer.
- **Security services on demand**: Security services like firewalls, IDS/IPS, and DPI minimize the risks of malicious attacks and enhance the security at the hypervisor and NFVIaaS layer.
- **Regular VM updates and patches**: This maintains a stable host environment and safeguards against virtualized attacks such as hyper jacking and VM DoS.
- **Protection of Operational Interface**: NICs with programmable cards and virtual switches are implemented partially in the hypervisor. This may enable an adversary to trap the

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

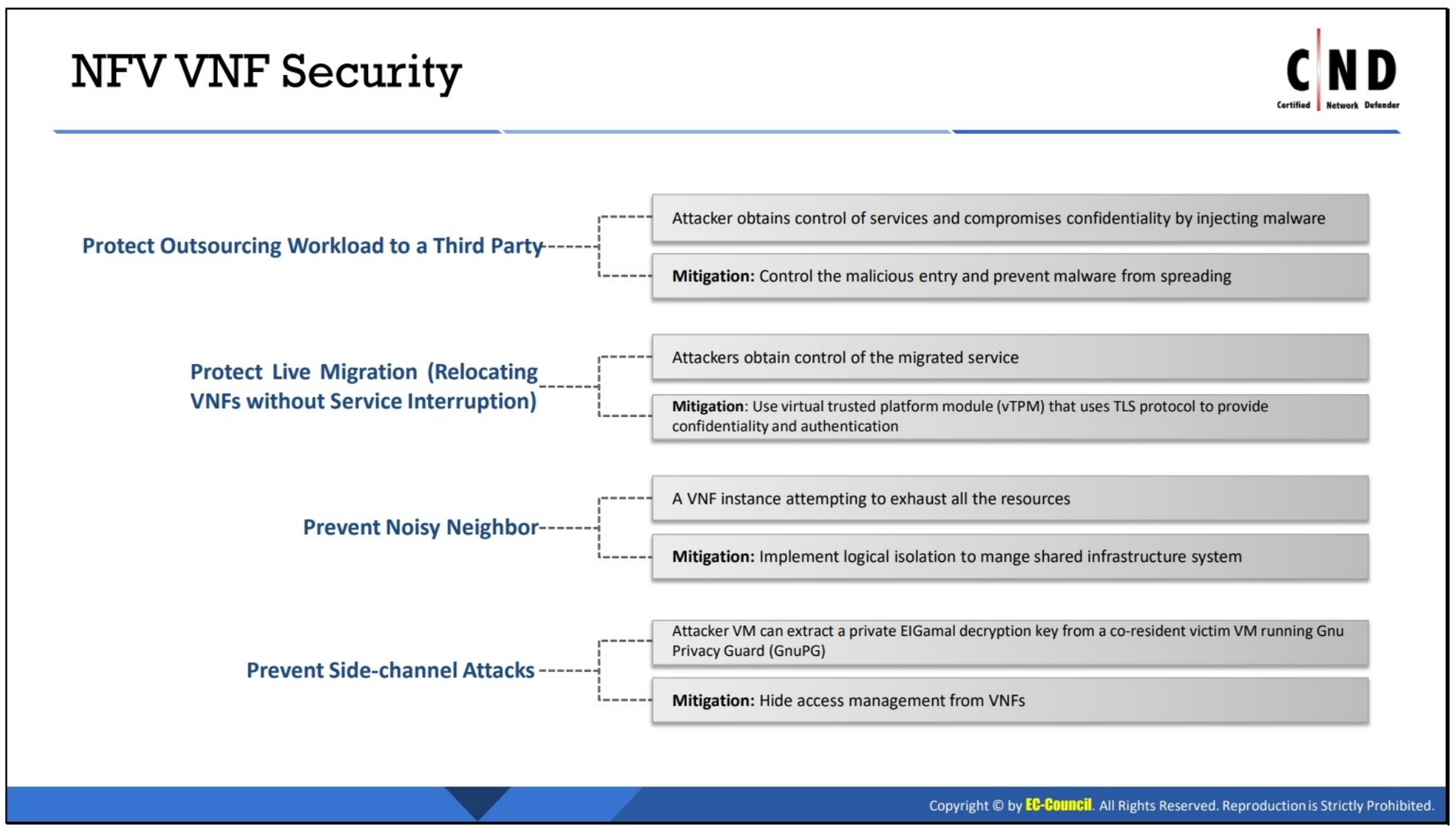
https://t.me/learningnets
Technet24

packets of the host or generate packets that lead to network congestion or packet retransmission. This security concern can be addressed by implementing a secure packet processing system to monitor the instruction level operations of the packet processor.

- **Protection against resource freeing attacks (RFA) and resource consumption attacks:** Malicious network as a service (NaaS) providers can conduct DoS attacks and extract secret information. To safeguard against this type of attack, excessive resource consumptions and malicious virtual networks should be detected by the hypervisor.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



NFV VNF Security

VNF weaknesses can be exploited by attackers to perform various attacks. The following are some techniques that can be used to mitigate common VNF weaknesses.

- **Establish a trust relationship among intra-VNF, inter-VNF, and an extra-VNF domains:** The trust relationship between all VNF instances are evaluated based on policy rules, contracts, and guidelines. The trust relationship is not workable if one of the instances is not faithful. Therefore, establish a trust relationship within VNFs, between VNFs, and with external VNFs.
- **Securing the network, end-to-end security, and authentication:** VNF applications are vulnerable to networking attacks such as eavesdropping, spoofing, and man-in-the-middle attacks. Security measures like encryption, policy enforcement, access control, security zoning, and segmentation can safeguard VNF applications against such attacks.
- **Control access when outsourcing workload to a third party:** An adversary can gain control of services and compromise confidentiality when workload is outsourced to a third party. To mitigate this security risk, control the entry of third parties to prevent malicious entities from spreading malware or gaining unauthorized access.
- **Use TLS protocol:** During live migration (relocating VNFs without service interruption) an adversary can gain control of the services that are migrated between hypervisors. Therefore, the virtual trusted platform module (vTPM) should use the TLS protocol to provide confidentiality and authentication.
- **Logical isolation:** A VNF instance may attempt to exhaust all the resources. Hence, implement logical isolation for improving the control and manageability of the shared infrastructure system.

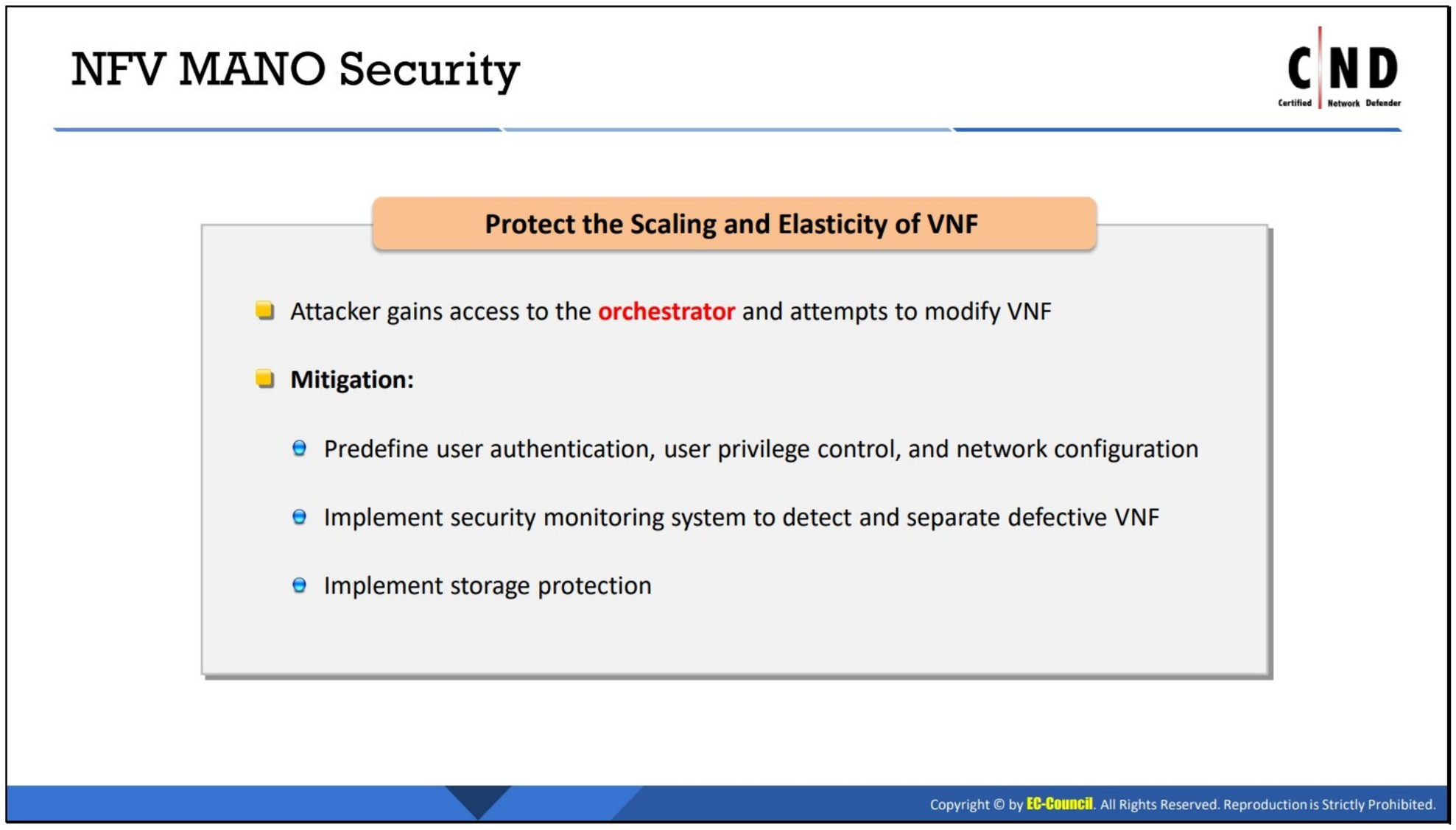
Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets
Technet24

- **Hide access management form VNFs to prevent Side-Channel Attacks**: If access management is visible to the VNF, an adversary may gain control of the service and compromise confidentiality. For example, an attacker VM can extract a private EIGamal decryption key from a co-resident victim VM running Cnu Privacy Guard (GnuPG). In order to mitigate this security threat hide access management form VNFs.
- **Monitoring**: Lack of control and monitoring may cause the NFVIaaS and VNFaaS layers to be prone to attacks. A lack of transparent monitoring may enable an adversary to gain access and exploit the network services, VNF instances, virtual assets, and sensitive data.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



NFV MANO Security

Numerous security vulnerabilities and threats might be potentially introduced in the NFV MANO. The following are some security measures that can be adopted to mitigate them.

- **Secure management, orchestration, and automation:** The security mechanisms for all the functions of the NFV MANO should be automated and agile for quick deployment at different security policy enforcement points (PEPs).
- **Ensuring controller availability:** An adversary may target the controller as it is the centralized decision point and, if compromised, can cause a wide network impact. Therefore, access to the controller should be stringently monitored and controlled.
- **Attack on Orchestrator:** The adversary gains access to the orchestrator and instantiates a modified VNF, which might break access privileges and VNF isolation. To mitigate this type of attack, a defective VNF should be detected and separated by the security monitoring system. Further, user authentication, privilege control, and network configuration must be predefined.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

NFV Security Best Practices

Boot Integrity Measurement Leveraging TPM

- Use **TPM** as a hardware root of trust. TPM's launch control policy (LCP) requires validation of platform measurements

Security Zoning

- Separate **VM traffic** and **management traffic** to prevent the impact of one VM on another VM or host

Hypervisor Introspection

- Identify **abnormal activities** in the VMs and enhance VM security

Copyright © by EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited.

NFV Security Best Practices (Cont'd)

Hypervisor and Virtual Network Security

- Keep the hypervisor **up-to-date**
- **Disable** all the services and enable only necessary services based on requirement
- Implement a **strong password** policy on the cloud administrator account.

Linux Kernel Security

- Implement **SELinux** module in the kernel to separate the users in the host virtual environment
- Use tools such as **hidepd** and **GRSecurity** to secure the Linux kernel

Copyright © by EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited.

NFV Security Best Practices

The following best practices for NFV security should be implemented in order to provide security and resiliency against threats:

Boot Integrity Measurement Leveraging TPM: Use a trusted platform module (TPM) as a hardware basis of trust. TPM's launch control policy (LCP) requires the validation of the platform measurements.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Hypervisor and Virtual Network Security: Keep the hypervisor up-to-date. Disable all services and enable only necessary services based on requirements. Implement a strong password policy on the cloud administrator account.

Security Zoning: Separate VM traffic and management traffic to prevent the impact of a compromised VM on another VM or host. VMs with the same functionalities should be grouped into specific zones and their traffic must be isolated. Each zone must be protected with access control policies and firewalls such as the demilitarized zone (DMZ).

Linux Kernel Security: Implement the SELinux module in the kernel to separate the users in the host virtual environment. sVirt, a new form of SELinux, separates the VM processes and data files, and safeguards Linux based hypervisors. In addition, tools such as hidepd and GRSecurity can be used to secure the Linux kernel.

Hypervisor Introspection: Identifying abnormal activities in the VMs can be invaluable towards enhancing the security of the VMs.

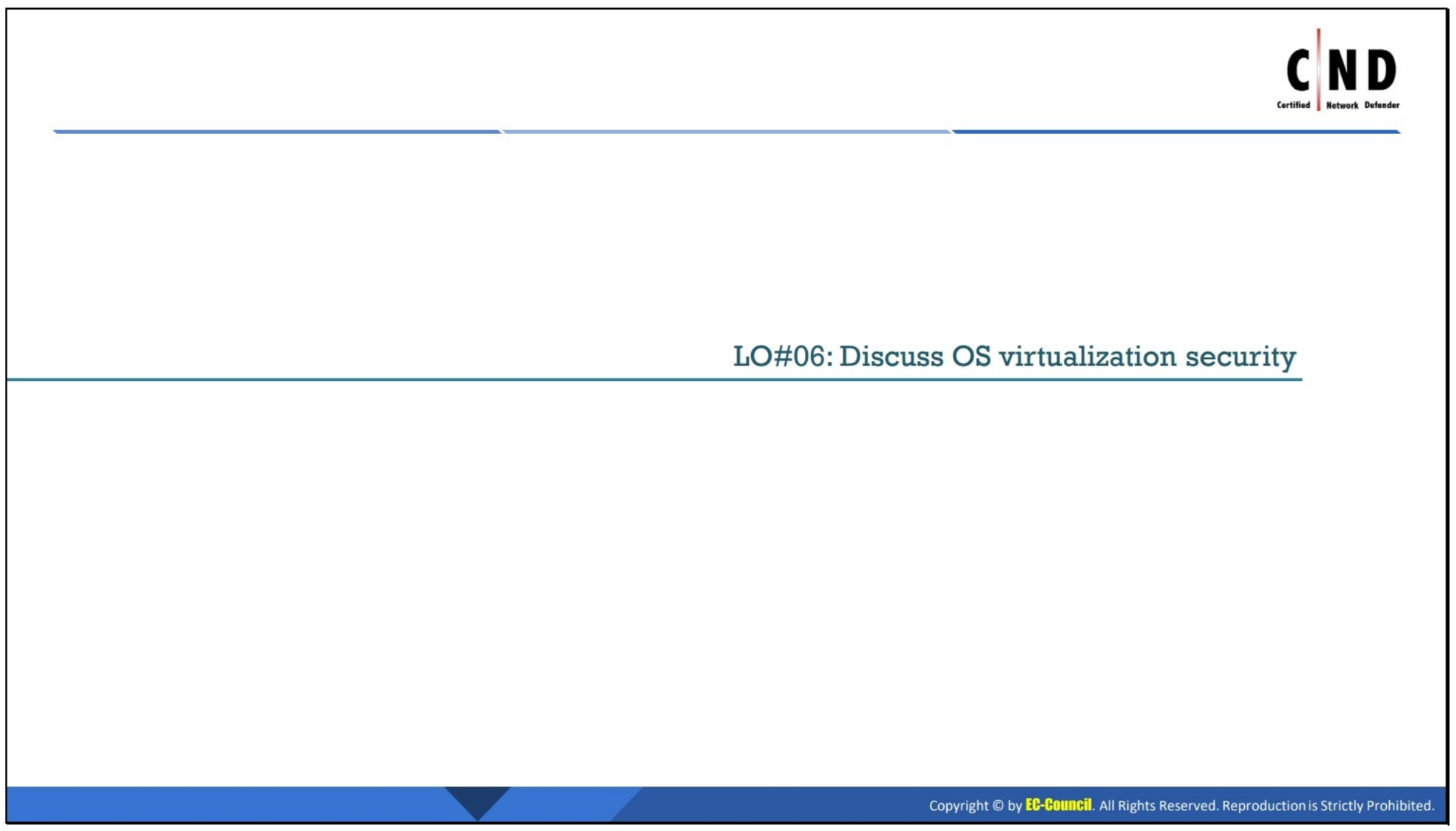
In addition to these best practices, traditional best practices such as the following must also be implemented to ensure NFV security.

- Implement defense in depth
    - Implement log management with accurate date/time stamps
    - Monitor software bugs
- Implement layer 2 security
    - Isolate VLANS for secure zones
- Implement layer 3 security
- Implement access control via access lists and firewall rules
- Implement IPS/IDS tools to secure the network

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24



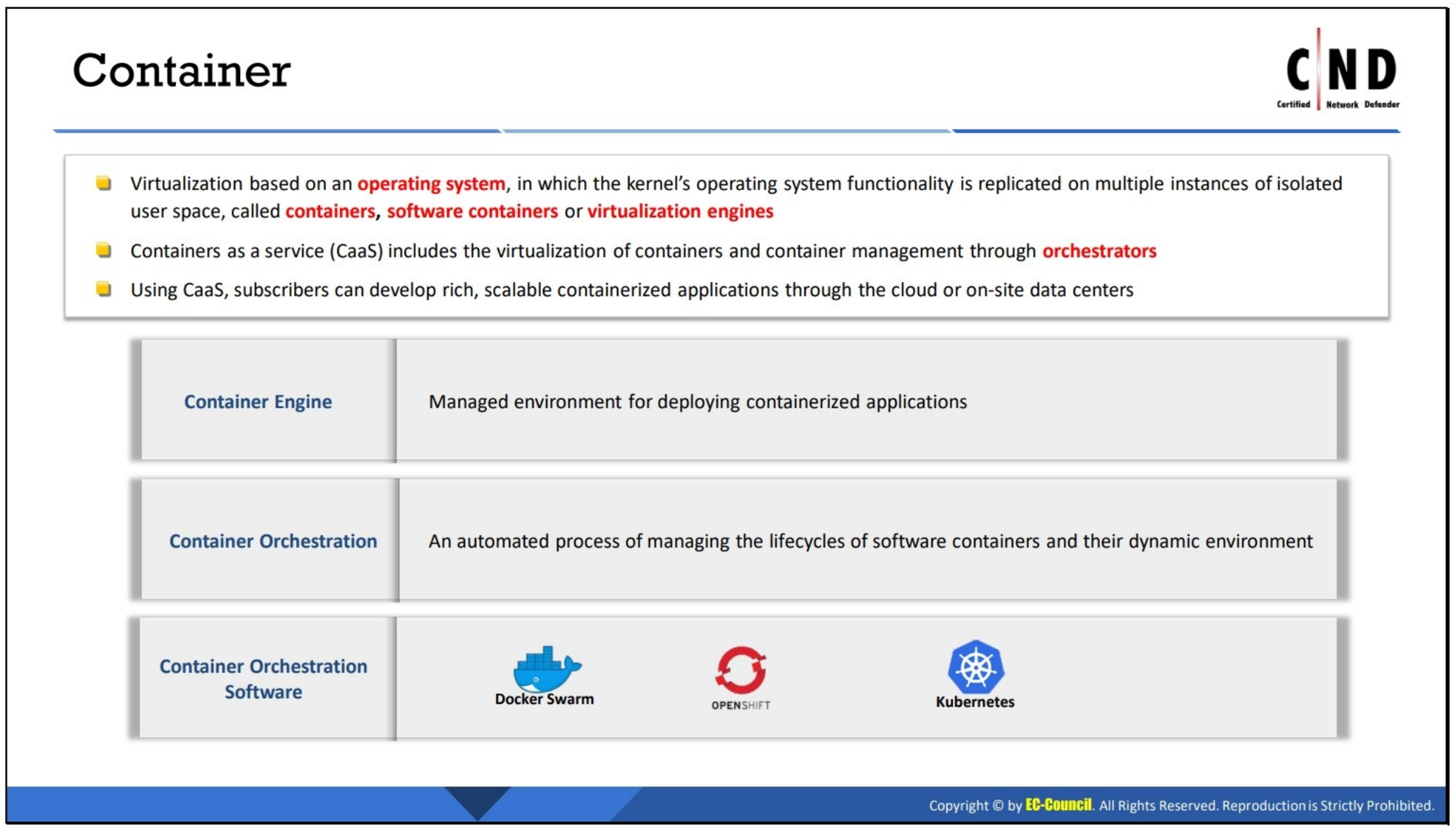
LO#06: Discuss OS Virtualization Security

In OS virtualization, the host operating system's kernel is virtually replicated in multiple instances of isolated user space, called containers, software containers, or virtualization engines, thereby lending (virtualized) operating system functionality to each container.

A container is widely used for encapsulating an application and its dependencies in its own environment, and runs in isolation from other containers and applications while utilizing the same resources and operating system. This section discusses vulnerabilities, attacks and security challenges associated with containers. The section also explains vulnerabilities, attacks and security challenges associated with Docker and Kubernetes, which are widely used for developing, packaging, running, and managing applications and all their dependencies in the form of containers.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Container

Containers (also called software containers or virtualization engines) refer to virtualization based on an operating system, in which the kernel's operating system functionality is replicated on multiple instances of isolated user space.

This can be used, for example, in a virtual hosting environment that requires segmentation of the physical resources among multiple users to enable each user to have their own virtual space. Containers help to manage the users and their respective resources, while keeping them isolated. The containers, are monitored and managed by the administrator having full admin rights to all the containers.

Many virtualization problems are effectively resolved with containerization. In containerization, although each user space instance runs in isolation, resources are not wasted since the actual operating system runs independently of the containers. A container encapsulates an application and its dependencies in its own environment while utilizing the same resources and operating system as other containers. Compared to VMs, each container image is more easily migrated and shared because of their smaller sizes. As only one operating system is involved, a container can be easily maintained. Containers also minimize hardware costs since multiple applications run on the same hardware, increasing the utilization of the hardware.

The following are some services and technologies that can be used to deploy and manage containers.

Containers as a service (CaaS): This refers to services that enable the deployment of containers and container management through orchestrators. Using CaaS, subscribers can develop rich, scalable containerized applications through the cloud or on-site data centers.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

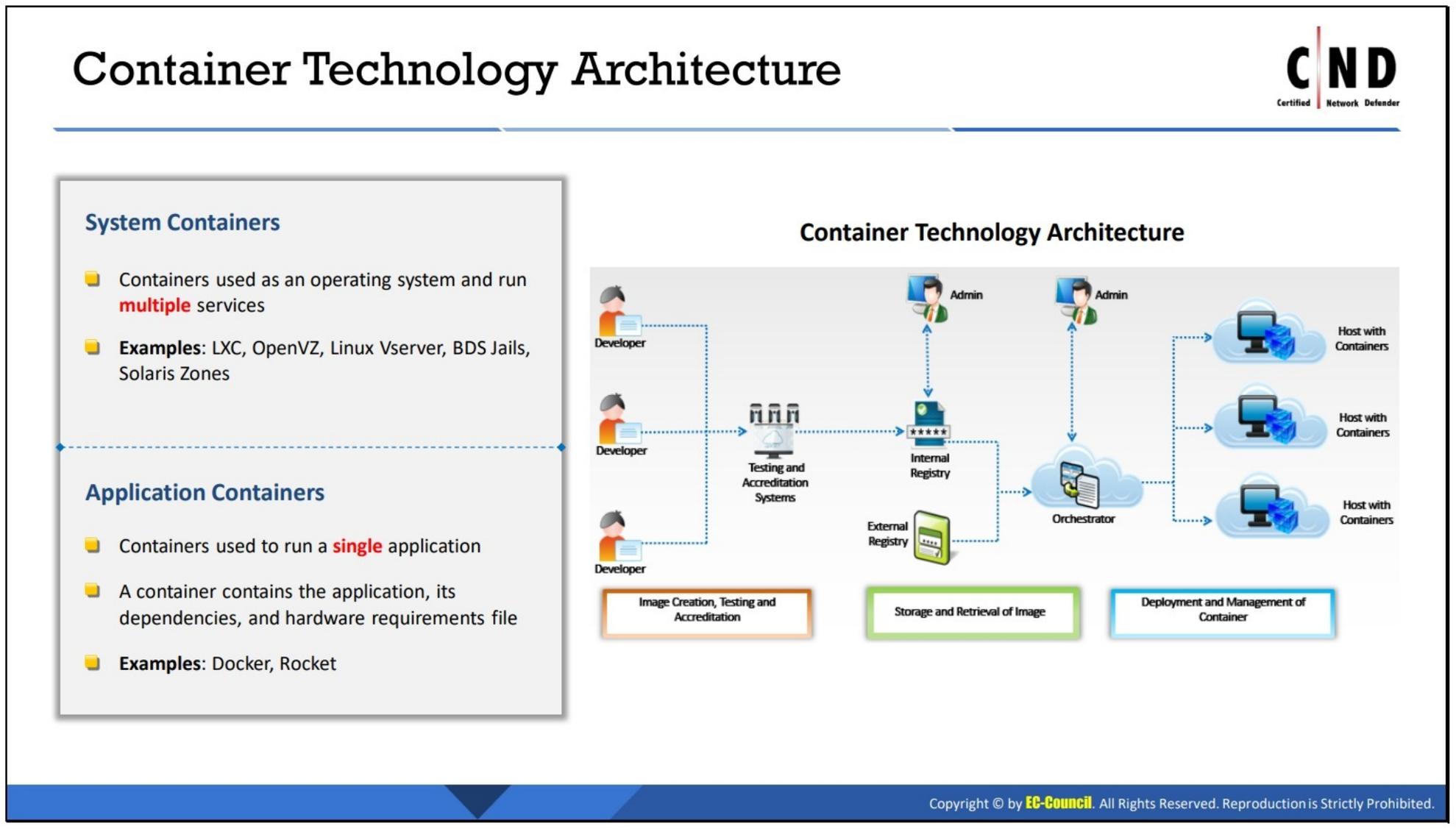
Technet24

Container Engine: A container engine can be used to create, add, and remove containers as per requirements. It manages the environment for deploying containerized applications.

Container Orchestration: This refers to an automated process of managing the lifecycles of software containers and their dynamic environment. Currently available open-source container orchestrators are Kubernetes and Docker Swarm, and a commercial container orchestrator is OpenShift by Red Hat.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Container Technology Architecture

Container technology architecture comprises the following five tiers:

- The developer creates the images and sends them for testing and accreditation.
- The testing and accreditation systems validate, verify, and sign the images and send them to the registry.
- At registries, the images are stored and distributed upon request from an orchestrator.
- At orchestrators, the images are converted into containers and deployed to the hosts.
- The host runs and stops the containers on the direction of the orchestrator.

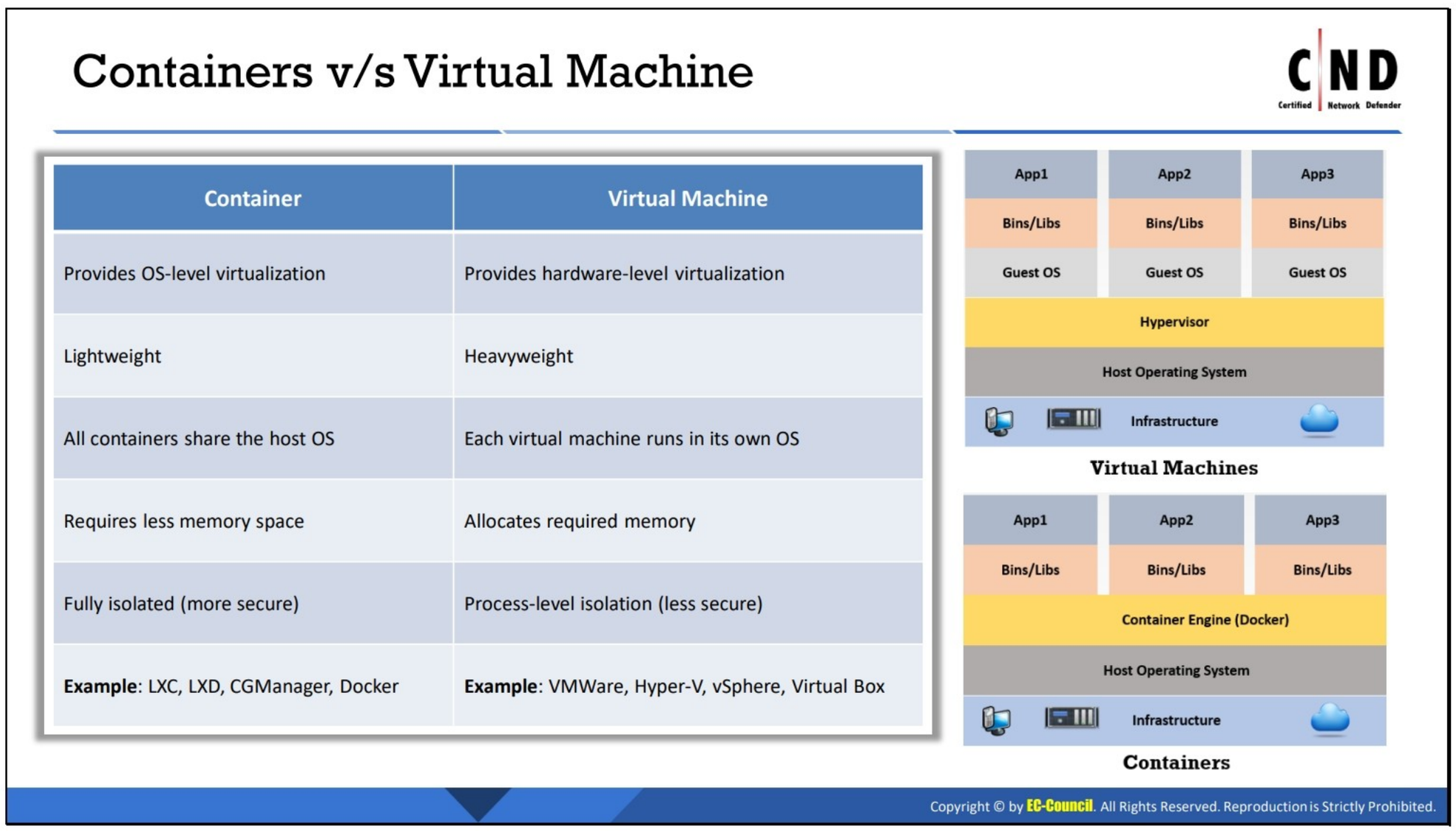
Types of Containers

- **OS Containers**: OS containers are virtual environments sharing the kernel of the host environment that provides them isolated user space. The user can install, configure, and run different applications, libraries, etc. in OS containers. OS containers run multiple services and processes. OS containers are suitable for users that require an operating system to install various libraries, databases, etc. Examples of OS containers are LXC, OpenVZ, Linux Vserver, BSD Jails, and Solaris Zones.
- **Application Containers**: These are containers used to run a single service. They have layered file systems and are built on top of OS container technologies. Application containers are suitable for users that require to package an application and its components together for distribution. Examples of application containers are Docker and Rocket.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24



Containers v/s Virtual Machine

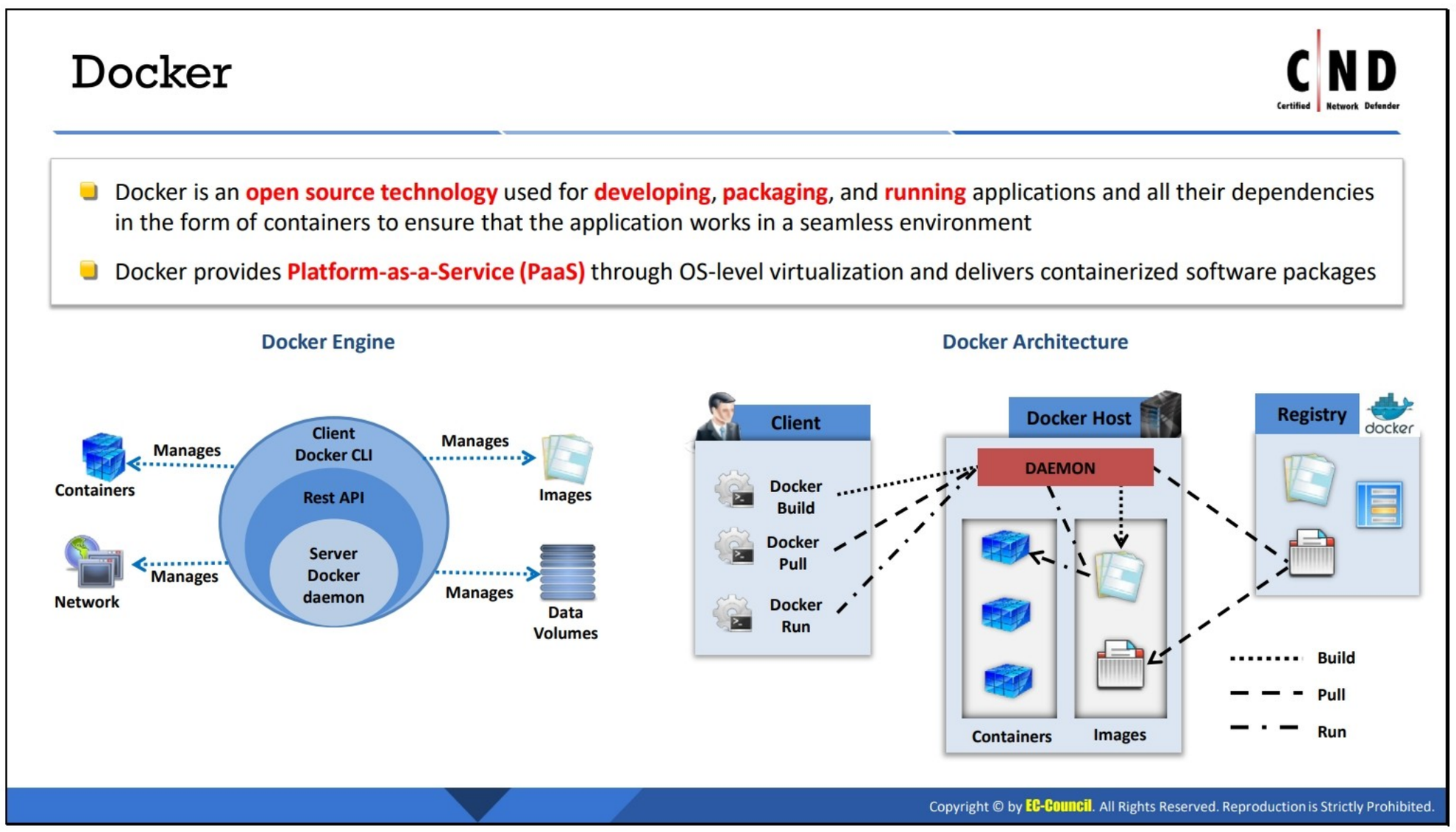
Containers and virtual machines decrease resource requirements and increase functionality. The differences between a container and a virtual machine are as follows.

	Container	Virtual Machine
Definition	Virtualization based on an operating system, in which the kernel's operating system functionality is replicated on multiple instances of isolated user space.	An operating system or application environment that runs on a physical machine.
Type	Lightweight.	Heavyweight.
Virtualization	Provides OS virtualization.	Provides hardware-level virtualization.
Memory Space	Requires less memory space.	Requires more memory space.
Security	Process-level isolation (less secure).	Fully isolated (more secure).
Start-up Time	Start-up time is in milliseconds	Start-up time is in minutes.
Operating System	Host OS is shared.	Each VM has its own OS.
Providers	Examples: LXC, LXD, CGManager, Docker.	Examples: VMware, Hyper-V, vSphere, Virtual Box.

Table 11.2: Containers Vs Virtual Machines

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Docker

Docker is an open source technology used for developing, packaging, and running applications and all their dependencies in the form of containers, to ensure that each application works in a seamless environment. Docker provides platform-as-a-service (PaaS) through OS-level virtualization and delivers containerized software packages.

Docker Engine: This is an application installed on the host machine and uses the following components to develop, assemble, ship, and run applications.

- **Docker Daemon**: This manages the Docker images, containers, networks, and storage volume, and processes the requests of the Docker API. It is responsible for container-related actions and communicates with other daemons in order to manage its services.
- **Docker Engine REST API**: This API is used by an application to communicate with the Docker daemon.
- **Docker CLI**: This is a command line interface that is used to interact with the Docker daemon. Using CLI, users can execute commands (build, run, and stop applications) to a Docker daemon.

Docker Systems Working Mechanism

The Docker client interacts with the Docker daemon using a REST API through Unix sockets or a network interface. The Docker client and the Docker daemon can run on the same system, or the user can connect a Docker client to a remote Docker daemon.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

Docker Architecture

The Docker architecture is based on a client-server model and has the following components:

- **Docker Client:** This enables the users to communicate with the Docker environment. The key function of the Docker client is to retrieve the images from the registry and run them on the Docker host. Some of the common commands of the Docker client are:

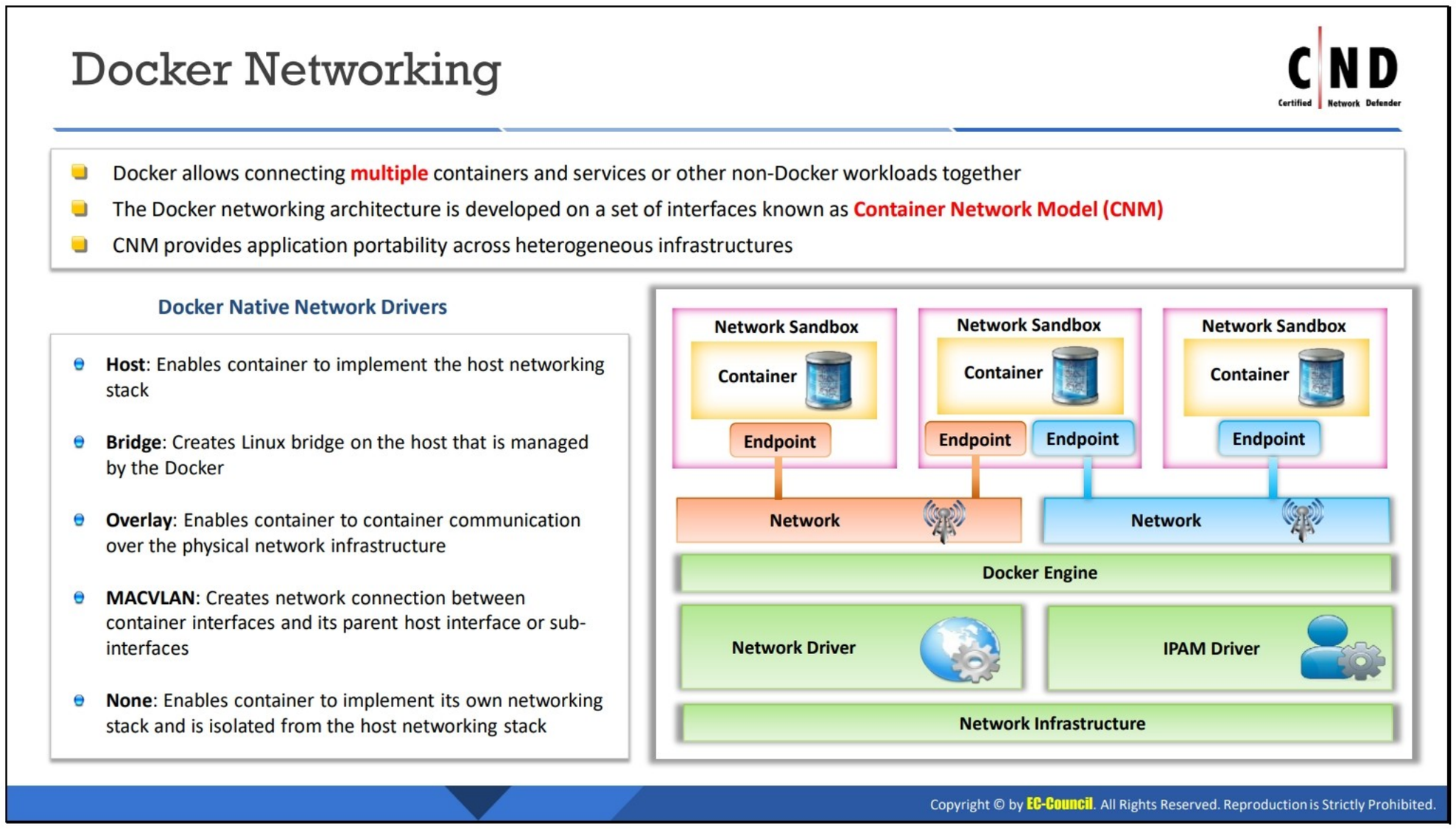
```
docker build
docker pull
docker run
```

- **Docker Host**: This provides the user an environment to run an application. The Docker host consists of Docker daemon, images, containers, networks, and storage. The following components are objects of the Docker host.
- **Images**: An image is a read-only binary template for building a container. Images are used to build a container or to configure the container with additional features. The container capabilities and requirements rely on the metadata of the images. Docker images are hosted by Docker registries.
- **Containers**: A container is an encapsulated environment to run an application. A container's access to resources are defined by the image. The user can also create a new image depending on the state of the container.
- **Networking**: Docker has networking drivers to support networking containers. It implements networking in an application-driven manner.
- **Docker Registries**: These are services that provide locations for storing and downloading images. While working with registries, frequently used commands are:

```
docker push
docker pull
docker run
```


Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Docker Networking

Docker allows connecting multiple containers and services or other non-Docker workloads together. The Docker networking architecture is developed on a set of interfaces known as container network model (CNM). CNM provides application portability across heterogeneous infrastructures. CNM consists of the following five objects:

- **Sandbox**: This contains the configuration of a container's network stack such as routing table, management of container's interfaces, and DNS settings. It may have multiple endpoints from various networks. CNM sandbox can be implemented for Windows HNS, Linux network namespace, or a FreeBSD jail.
- **Endpoint**: This connects a sandbox to a network and abstracts the actual connection to the network from the application. An endpoint aids in maintaining portability, to enable the service to utilize various types of network drivers.
- **Network**: A network is a collection of endpoints that have connectivity between them. When a network is created or updated, the corresponding driver is notified. A CNM network can be used to implement a Linux bridge, VLAN, etc.
- **CNM Driver Interfaces**: CNM has two pluggable and open interfaces for the users, vendors, community, etc. to drive additional functionality, visibility, and control in the network. The following are the drivers in the CNM model:
  - **Network Drivers**: These are pluggable and provide the actual implementation for the functioning of the network. Multiple network drivers can be simultaneously used on a Docker engine or cluster, but each Docker network is represented by a single driver. There are two types of CNM network drivers.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

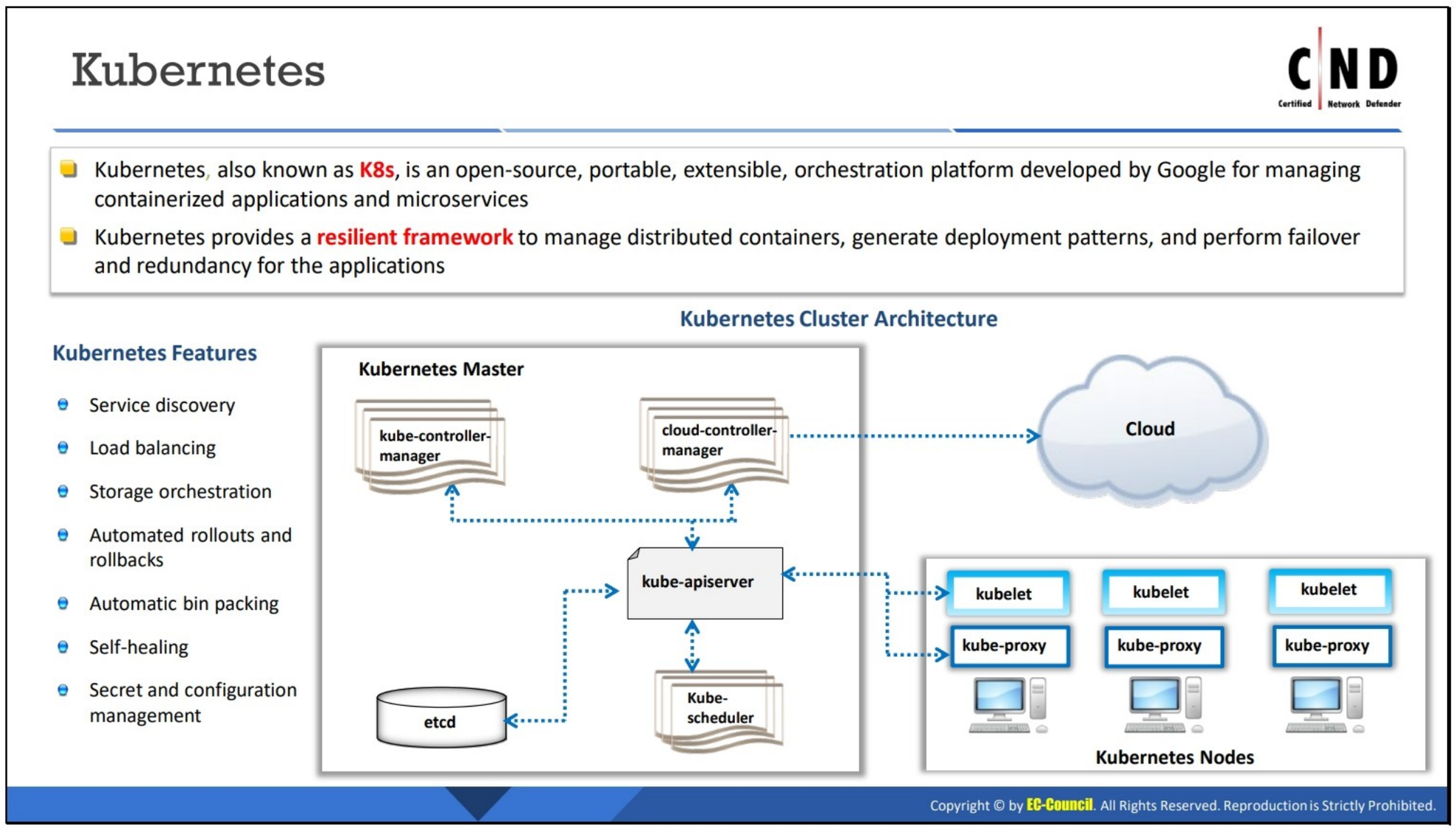
- **Native Network Driver**: These drivers are provided by Docker.
- **Remote Network Driver**: These drivers are created by the community and vendors based on their requirements.

- **IPAM Drivers**: IP address management (IPAM) drivers in Docker provide default subnets or IP addressing to the network and the endpoints. A user can also assign an IP address manually through the network, container, and service create commands.

- **Docker Native Network Drivers:** These are native drivers in the Docker engine and can be used through Docker network commands. The following are the various Docker native network drivers.
  - **Host**: The host driver enables the container to use the host networking stack.
  - **Bridge**: With the bridge driver, a Linux bridge is created on the host, which is managed by the Docker.
  - **Overlay**: An overlay network is created with the overlay driver and enables container-to-container communication over the physical network infrastructure.
  - **MACVLAN**: With the MACVLAN driver, a network connection is created between container interfaces and its parent host interface (or sub-interfaces).
  - **None**: The none driver enables the container to implement its own networking stack and is isolated from the host networking stack.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Kubernetes

Kubernetes, also known as K8s, is an open-source, portable, extensible, orchestration platform developed by Google for managing containerized applications and microservices. Kubernetes provides a resilient framework to manage distributed containers, generate deployment patterns, and perform failover and redundancy for the applications.

Kubernetes Cluster Architecture

In the Kubernetes cluster, the Kubernetes nodes are worker machines, which run the containerized applications. There should be at least one worker node in a cluster. The worker nodes host the pods, which refer to groups of containers that are deployed together on the same host. The components of a Kubernetes cluster are as follows:

- **Control Plane Components**

  The components of the control plane perform decisions for the Kubernetes cluster such as scheduling or staring a new pod. The following are the control plane components of the cluster:

  - **Kube-apiserver**: The Kubernetes control plane has an API server in its front end. Kube-apiserver is the implementation of the Kubernetes API server. The user can run multiple instances of kube-apiserver to facilitate the maintenance of the traffic between the instances.
  - **etcd**: This is a backing store for the data in the Kubernetes cluster. For example, if the user specifies that three instances of a specific pod should be executed, this information is stored in etcd. The data stored in etcd is used to determine the number

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

of instances that are running. If an instance is not working, Kubernetes creates an additional instance of the same pod.

- **Kube-scheduler**: The kube-scheduler monitors newly created pods that do not have any assigned nodes, and assigns each of them a node to run on.
- **kube-controller-manager:** Kube-controller-manager runs the controller processes. It consists of a node controller, replication controller, endpoints controller, service account, and token controllers. To minimize complexity, all these controllers are compiled and run as a single process.
- **cloud-controller-manager:** This runs the controller that communicates with the cloud providers. The cloud-provider-specific controller loops are run by cloud-controller-manager. When a kube-controller-manager is initialized, the user can disable the controller loops by setting the -cloud-provider flag to external. The controllers with cloud provider dependencies are node controller, route controller, service controller, and volume controller.

- **Kubernetes Node**

  A Kubernetes node is a worker machine that contains the services required to run the pods, and is managed by master components. The services on Kubernetes nodes are as follows:

  - **Kubelet:** This is a node agent that ensures that the containers are running in a pod. A pod contains a PodSpec, which is a YAML or JSON object. The kubelet ensures that the containers mentioned in the PodSpecs are running and healthy.
  - **Kube-proxy:** This is a network proxy that runs and maintains network rules on each node.
  - **Container Runtime:** This is a software that downloads the images and runs the containers. Kubernetes supports container runtimes such as Docker, CRI-O, and the Kubernetes container runtime interface (CRI).

- **Kubernetes Features**

  - **Service Discovery and Load Balancing**: Kubernetes represent a container using the DNS or its own IP address. In case the network traffic to the container is high, Kubernetes utilizes load balancing to distribute the traffic.
  - **Storage Orchestration**: Kubernetes enables the user to choose between storage systems such as local storage, public cloud providers (AWS or GCP), or a network storage system.
  - **Automated Rollouts and Rollbacks:** Kubernetes enables the user to change the actual state of the container to the desired state of the container at a controlled rate.
  - **Automatic bin packing:** To utilize resources effectively, Kubernetes fits the containers into nodes depending on the specifications provided by the user.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

- **Self-healing:** Kubernetes self-heal the containers by restarting failed containers. If a node is dead, Kubernetes replaces and reschedules the containers. When a container fails to respond to user-defined health checks, Kubernetes kills the container. Kubernetes also advertises the containers that are in working condition.
- **Secret and Configuration Management:** Kubernetes enables users to store and manage confidential information such as passwords, OAuth tokens, and SSH keys.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

Container Security Challenges

While containerization provides fast and continuous delivery of applications to developers and DevOps teams, there are certain security challenges associated with it. The primary security challenges for containers are as follows:

- **Inflow of Vulnerable Source Code**: Since containers are open source**,** the images that are created by the developers are frequently updated, stored, and used as needed. This causes an inflow of source code that may potentially harbor vulnerabilities and unexpected behaviors, into an organization.
- **Large Attack Surface:** In cloud or on-premises, there are many containers than run on multiple machines. This provides a large attack surface, and therefore causes challenges in the tracking and detection of anomalies.
- **Lack of Visibility:** The abstraction layer created by the container engine masks the activity of a particular container.
- **DevOps Speed:** On average, the lifespan of a container is four times less than virtual machines. Containers can be created instantly, run for a short duration of time, stopped, and removed. Due to this ephemerality, an attacker can execute an attack and disappear quickly.
- **Noisy Neighboring Containers:** The behavior of one container can potentially cause a DOS for another container. For example, opening sockets frequently can freeze up the host machine.
- **Container Breakout to the Host:** Containers that run as the root user can breakout and access the host's operating system.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

- **Network-based Attacks:** A jeopardized container is vulnerable to network-based attacks, especially in outbound networks with unrestricted raw sockets.
- **Bypassing isolation/Lack of isolation**: Any inadequacy in the isolation between containers can be a security challenge, since an attacker who compromises one container can then easily access another container in the same host.
- **Ecosystem complexity**: The tools utilized to build, deploy, and manage containers are provided by different sources. Therefore, a user should keep the components secure and up-to-date.
- **Lack of governance:** Software errors and identity and access management (IAM) backdoor shortcuts can lead to significant security gaps in containers. As a result, container security incidents now account for more security threats, with most of these incidents being major attacks.
- **Lack of standardization:** It can be challenging to incorporate existing security standards based on alternative, out-of-date approaches into containers. The combination of multiple security standards along with growing container, tool, and platform infrastructures raises security concerns.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

Container Security Threats

Image Threats

- **Image vulnerabilities**: As images are static archive files, **lack of updates** in the image components or **missing of critical security updates** make the image vulnerable
- **Image configuration defects**: Images with **misconfigurations** lead to privilege escalation
- **Embedded malware**: Image packages may contain malicious files
- **Embedded clear text secrets**: An application packed into an image containing secrets stored in plain-text
- **Use of untrusted images:** Use of untrusted images can introduce malware and leak data

Registry Threats

- **Insecure connections to registries:** Insecure connection to the registries can lead to a **man-in-the-middle** attack
- **Stale images in registries:** Vulnerable or out-of-date images in the registry if deployed accidentally expose the container to various vulnerabilities
- **Insufficient authentication and authorization restrictions**: Since registry runs sensitive or proprietary software, **insufficient authentication** and **authorization** leads to unauthorized access to sensitive information

Copyright © by EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited.

Container Security Threats (Cont'd)

Container Threats

- **Vulnerabilities within the runtime software**: Attackers exploit the vulnerabilities present within the runtime software
- **Unbounded network access from containers:** Compromised container with permissions to access other containers increases the risk to other containers in the environment
- **Insecure container runtime configurations:** Runtime misconfigurations make the containers **insecure**
- **App vulnerabilities:** Application **flaws** during runtime make the container vulnerable
- **Rogue containers:** Unplanned or unsanctioned containers, when not properly configured and passed through rigorous vulnerability scanning, are easily exploited

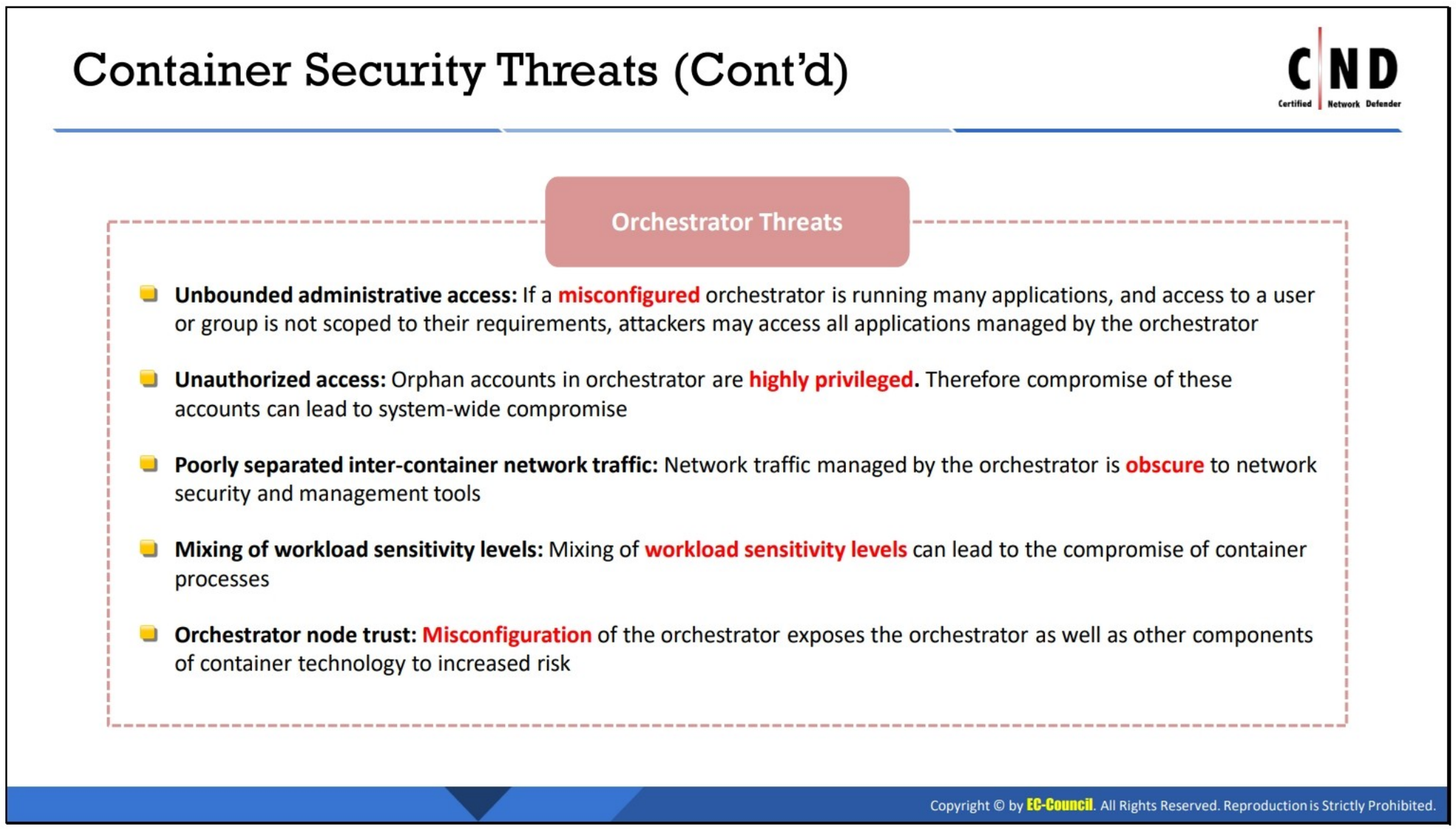
Host OS Threats

- **Large attack surface:** Attackers **exploit** host OS vulnerabilities
- **Shared kernel:** Usage of shared kernel increases the inter-object attack surface
- **Host OS component vulnerabilities:** Host OS component **vulnerabilities** impact all the containers and applications running on the host
- **Improper user access rights:** **Improper user access** affects the host system along with other containers present in it
- **Host OS file system tampering:** Host OS file system **tampering** affects the stability and security of the host and containers running on it

Copyright © by EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Container Security Threats

Containers are among the most important technologies in DevOps, and many organizations are adopting this technology to develop, test, package, and deploy the applications. Nevertheless. the increased use of containers is exposing companies to new security threats. Therefore, there is a need to secure containers throughout the development pipelines from these security threats.

The threats that are associated with container ecosystems are as follows:

Image Threats

- **Image Vulnerabilities**: Since images are static archive files consisting of components that run an application, lack of updates in the image components or the missing of critical security updates make the image vulnerable. If the version of the image that is utilized to make a container has vulnerabilities, it poses a risk to the containerized environment.
- **Configuration Defects**: In addition to software defects, the image of a container is also subject to configuration defects. For example, an image may fail to configure with a specific user account, and instead run with greater privileges than required.
- **Embedded Malware**: As an image is a collection of files that are packed together, there is a likelihood that malicious files are included in the image package intentionally or inadvertently. Such embedded malware has the same privileges as other components of the image, and could be used to attack other containers or hosts.
- **Embedded Clear Text Secrets**: Most applications require secrets to securely communicate with other components. For example, a web application requires a username and password to connect to the backend database. When an application is packed into an image, these secrets get embedded into the image. This embedded clear

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

text secret poses a security risk as a user having access to the image can parse the image to extract these secrets.

- **Use of Untrusted Image**: The use of untrusted images can introduce malware, leak data, or introduce components with vulnerabilities. It is recommended that running any image in a container from untrusted sources is avoided.

Registry Threats

- **Insecure Connections to Registries**: Images may contain sensitive components such as proprietary software and embedded secrets. An insecure connection to the registries can enable a man-in-the-middle attack.
- **Stale Images in Registries**: A registry contains all images that an organization deploys. Over time, it is possible that the registry contains images that are vulnerable or out-of-date. These vulnerable images would not pose a threat during their storage in the registry, but can increase the likelihood of accidental deployment of the vulnerable image.
- **Insufficient Authentication and Authorization Restrictions**: Since registries contain images that may run sensitive or proprietary software, insufficient authentication and authorization expose the technical details of the application to the attacker.

Orchestrator Risks

- **Unbounded Administrative Access**: Most orchestrators are designed presuming that all the users that are interacting with them are administrators. In most cases, a single orchestrator runs many applications, each managed by different teams. If access to a user or group is not scoped to their requirements, then a malicious user can affect the functionality of different containers managed by the orchestrator.
- **Unauthorized Access:** The orchestrators have their authentication directory service, which could be separated from other directories of an organization, leading to orphan accounts in orchestrator. These accounts are highly privileged, and their compromise can subsequently lead to system-wide compromise. The data storage volumes of a container are managed by the orchestration tool. Many organizations encrypt the stored data to prevent unauthorized access.
- **Poorly Separated Inter-Container Network Traffic:** The network traffic between each node is routed through a virtual overlay network, which is managed by the orchestrator and is obscure to network security and management tools.
- **Mixing of Workload Sensitivity Levels**: The orchestrators' primary focus is to enhance the density of workloads. By default, the orchestrator places the workloads of different sensitivities on the same host. For example, in a default state, the orchestrator may place a container running a public-facing web server and another processing financial data on the same host because the host has more available resources at the time of deployment. Due to the mixing of workload sensitivity levels, the container processing financial data can be easily compromised.
- **Orchestrator Node Trust:** Maintaining trust between the nodes in an environment is crucial. As the orchestrator is the foundational node, if the configuration of the

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

orchestrator is weak, it can expose the orchestrator as well as other components of the container technology to increased risk.

Container Risks

- **Vulnerabilities within the Runtime Software:** The attacker can exploit vulnerabilities within the runtime software to compromise it. Subsequently, the attacker can utilize the compromised runtime software to perform other activities such as attack other containers and monitor container-to-container communication.
- **Unbounded Network Access from Containers:** In the default state, most containers during runtime access other containers or the host OS through the network. In case a container is compromised, permitting it access to the network traffic significantly increases the risk to other containers in the environment.
- **Insecure Container Runtime Configurations:** During container runtime, the administrator is exposed to many configurable options. The security of the system can be lowered if these options are improperly set.
- **App Vulnerabilities:** Containers can be compromised because of flaws in an application that they run. This, however, is not a fault in the container, but rather that the vulnerabilities in the application within the container environment compromise the container.
- **Rogue Containers:** Rogue containers are unplanned or unsanctioned containers. They originate in the development environment when the application developer creates a container to test the code. If these containers are not properly configured or are not passed through the rigors of vulnerability scanning, they can be exploited.

Host OS Risks

- **Large Attack Surface:** The attack surface of the host OS is the collection of all possible points through which the adversary can attempt to gain access to and exploit the host OS vulnerabilities. A large attack surface increases the potential for an attacker to gain access to and compromise the host OS and the containers running on that host.
- **Shared Kernel:** Compared to general-purpose OSes, container-specific OSes have a smaller attack surface area. However, a container has only software-level isolation of resources, and the usage of a shared kernel increases the inter-object attack surface.
- **Host OS Component Vulnerabilities:** Vulnerabilities of the host OS components impact all the containers and applications running on the host.
- **Improper User Access Rights:** An organization can be at risk when users sign in directly on the host to manage containers. Improper user access rights not only affect the host system but also all other containers present in it.
- **Host OS File System Tampering**: If the configuration of a container is not secure, it exposes the host volumes to significant risk of file tampering. Host OS file system tampering affects the stability and security of the host and other containers running on it.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

Docker Security Threats

There are specific parts of the Docker infrastructure that are vulnerable to attacks. The following are common Docker security threats.

- **Escaping**: In this Docker-specific security threat, the adversary escapes the container and gains root access on the host server. The attacker then attempts to compromise other machines within the local network. The following factors may facilitate container breakouts:
  - Insecure defaults and weak configuration.
  - Information disclosure.
  - Weak network defaults.
  - Working with the root user (UID 0).
  - Mounting host directories inside containers.
- **Cross-Container Attacks:** In this type of attack, the adversary gains access to a container and utilizes it to attack other containers of the same host or within the local network. Cross-container attacks lead to DoS attacks (e.g. XML bombs), ARP spoofing and stealing of credentials, compromising of the sensitive container, etc. The following are common causes for cross-container attacks:
  - Weak network defaults.
  - Weak cgroup restrictions.
  - Working with the root user (UID0).

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

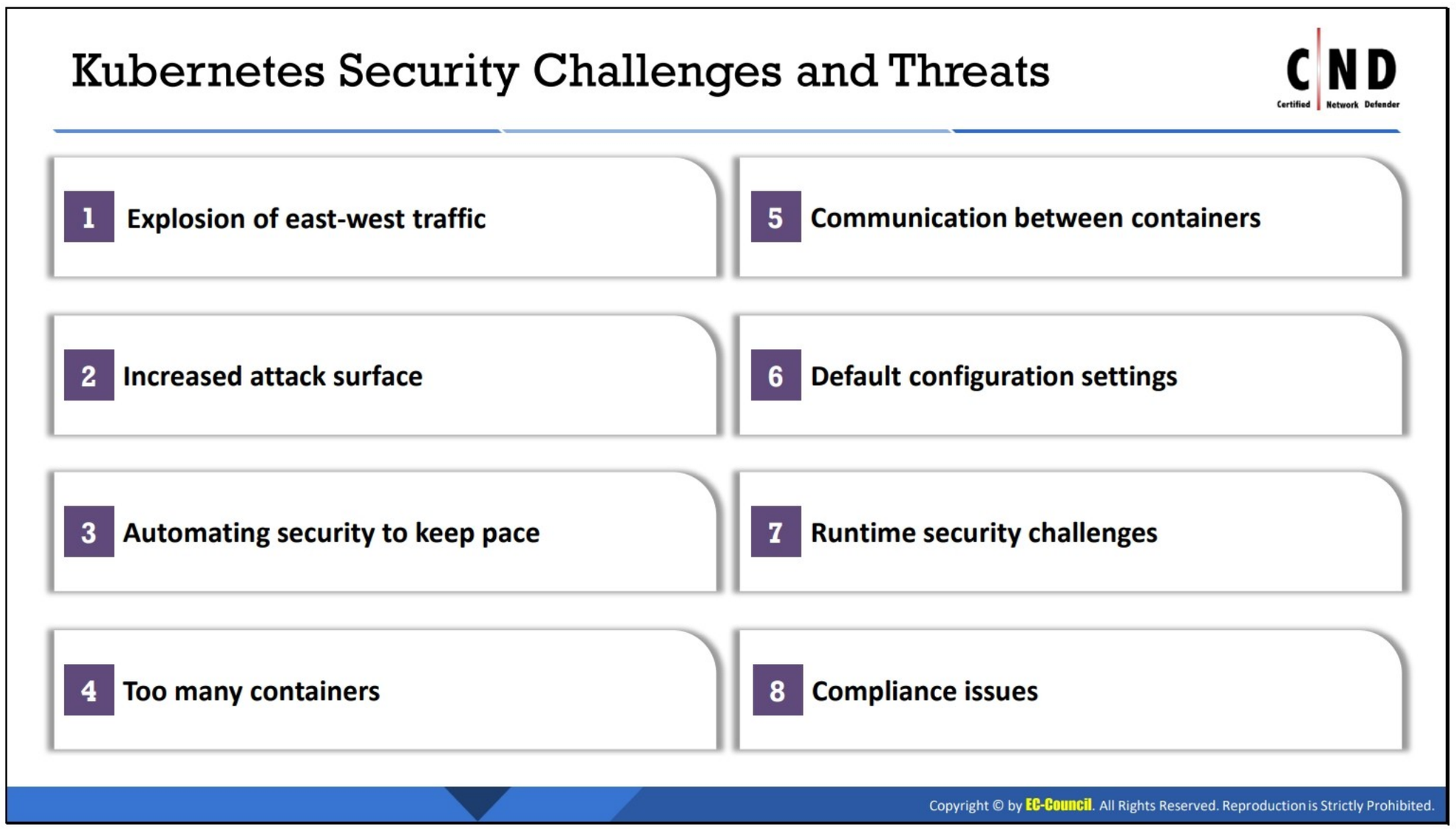
https://t.me/learningnets

- **Inner-Container Attacks:** in this attack, the attacker gains unauthorized access to a single container. The potential causes for inner-container attacks are:
  - Overage software.
  - Exposure to insecure/untrusted networks.
  - Use of large base images.
  - Weak application security.
  - Working with the root user (UID 0)
- **Docker registry Attacks**
  - **Image forgery**: In an image forgery attack, the adversary gains access to the registry server and tampers the Docker image.
  - **Replay Attack**: In this type of attack, the adversary gains access to the registry server and provides outdated content.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24



Kubernetes Security Challenges and Threats

Kubernetes enables an organization to automate application deployment, which leads to tremendous growth in business. However, these deployments are vulnerable to attacks and exploitation from adversaries. Thus, security is a critical component for all deployments of Kubernetes. Containerization in clouds can be targeted through attacks such as ransomware attacks, cryptomining, data stealing, and service disruption. The hyperdynamic nature of containers is responsible for the following Kubernetes security challenges.

- **Explosion of East-West Traffic**: Since containers are dynamically deployed in multiple hosts or clouds, east-west traffic (traffic flow within a data center) and internal traffic should be monitored for attacks.
- **Increased Attack Surface:** Every container has an attack surface and vulnerabilities, which can be exploited by an adversary. Further, container orchestration tools like Docker and Kubernetes also increase the attack surface of the container.
- **Automating Security to Keep Pace:** Due to the dynamic nature and the constantly changing environment of the container, old models and security tools cannot provide complete protection. Therefore, there is a need to automate security for securing containers.

Kubernetes Vulnerabilities and Attacks

Kubernetes containers are vulnerable to attacks externally through the network or internally by an insider. The following are potential Kubernetes vulnerabilities and attacks:

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

- **Container Compromise**: The misconfiguration of an application enables the attacker to gain access to the container and search for weaknesses in the network, process controls, or file system.
- **Unauthorized Connections Between Pods**: Due to unauthorized connections between pods, an adversary may launch an attack by connecting the compromised container to the pods in the same host or different hosts.
- **Data Exfiltration from a Pod**: This can be conducted using techniques such as reverse shell in a pod connecting to a command/control server, and using network tunneling for hiding sensitive information.
- **Compromised Container Running Malicious Process**: The container normally runs a well-defined set of processes. However, a compromised container may run malicious processes such as cryptomining, network scanning, and port scanning.
- **Container File System Compromised**: An adversary may install vulnerable libraries/packages to compromise the container file system. Subsequently, the attacker may attempt privilege escalation to root or attempt other breaks out.
- **Compromised Worker Node**: The host running the containers can also be compromised through vulnerabilities such as the dirty cow Linux kernel vulnerability that enables user privilege escalation to root.

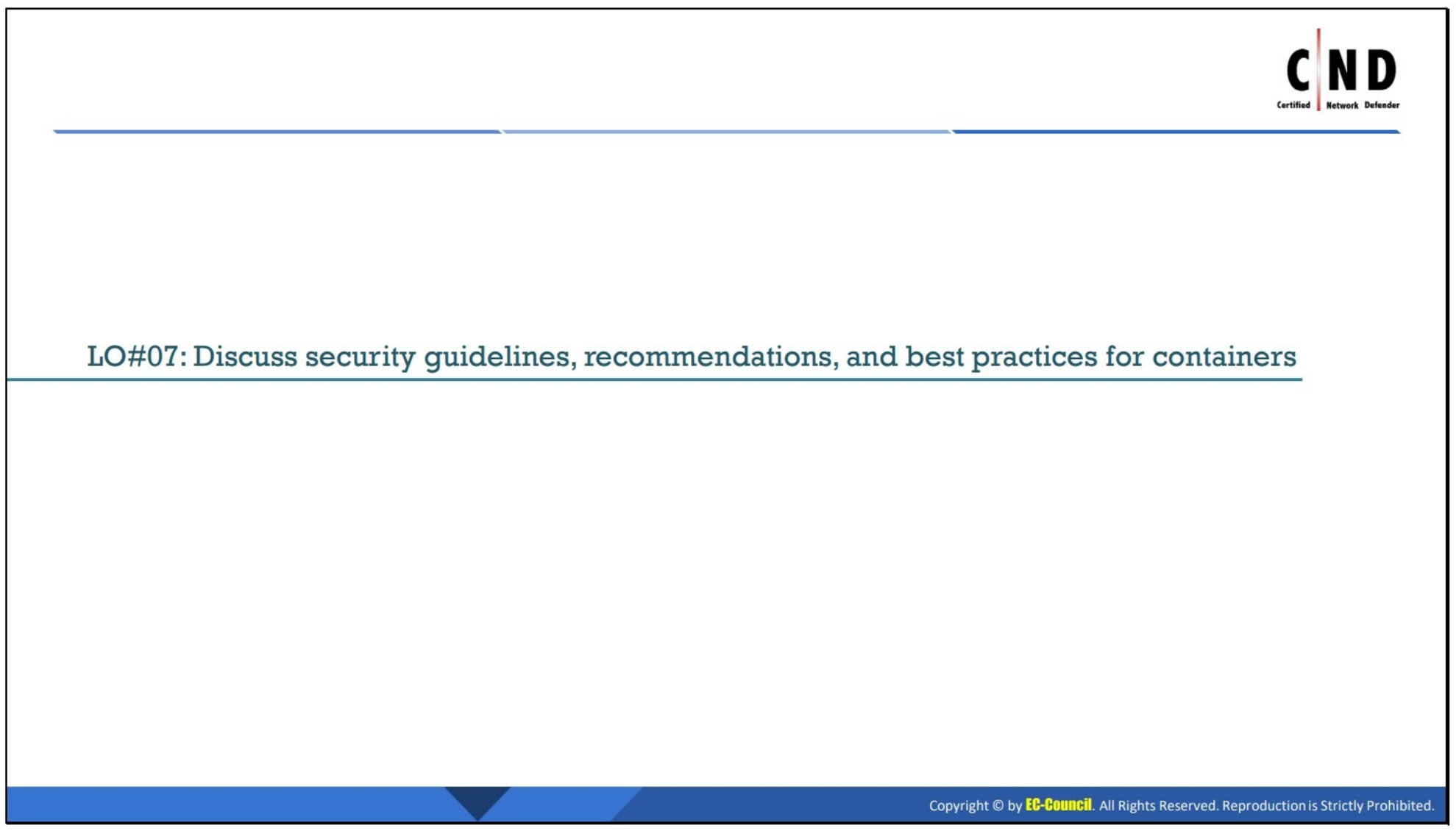
  The following components of Kubernetes are vulnerable to various attacks

  - Management server UI
  - API services
  - Etcd
  - Kuberlets
  - Compromised nodes, pods and accounts
  - Exposed dashboard and Kuberlets

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

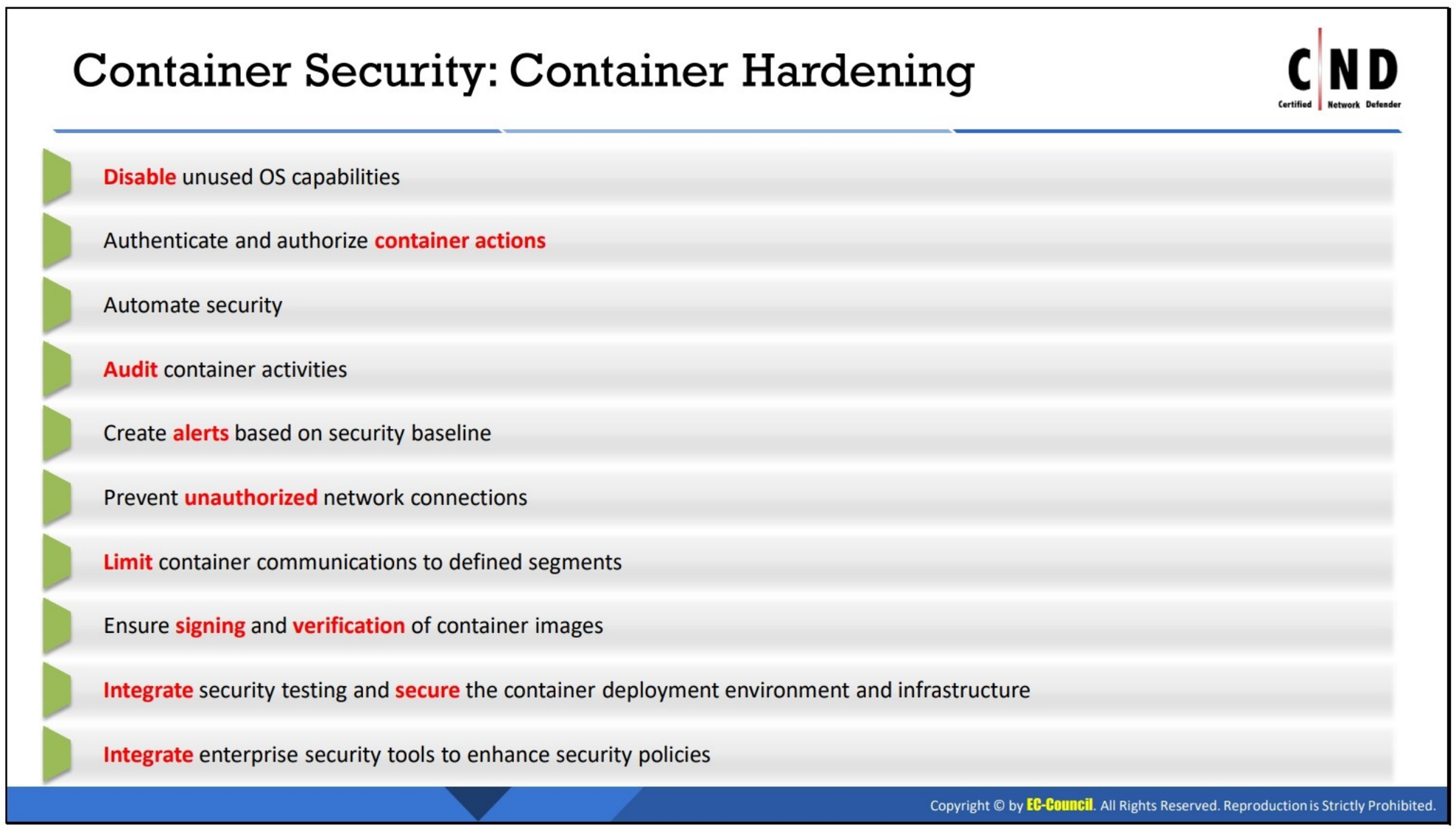


LO#07: Security Guidelines, Recommendations, and Best Practices for Containers

The objective of this section is to explain the security measures, best practices, and recommendations for the security of containers.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Container Security: Container Hardening

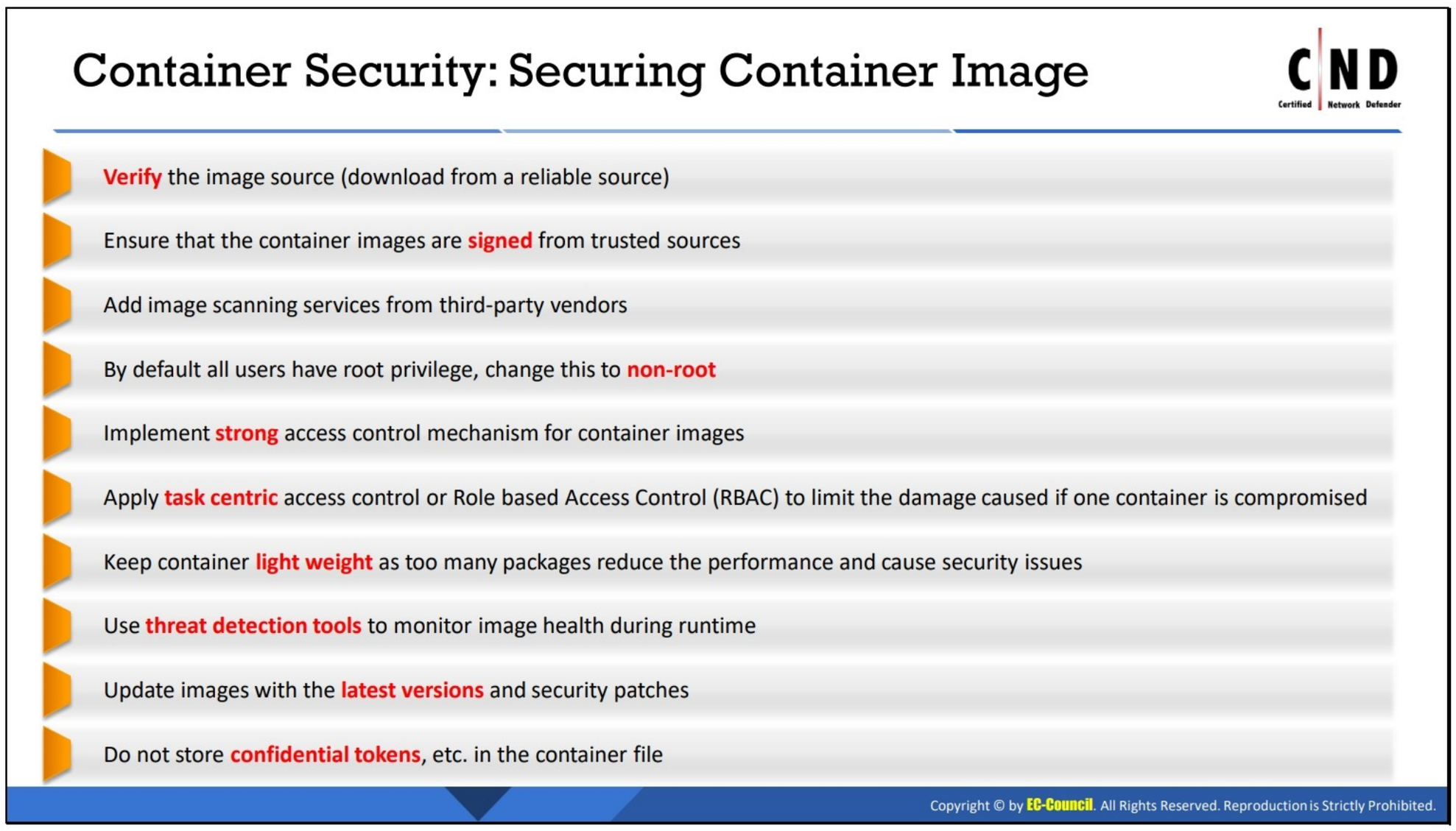
To harden containers against vulnerabilities, the following measures must be implemented:

- Limit container communications to defined segments. This prevents unauthorized connections.
- Prevent unauthorized network connections by utilizing network firewall technology. This protects running containers.
- Alerts based on security baselines - create a runtime security policy for the prompting of alerts and remedies when suspicious activity is observed.
- Audit container activity to resolve issues pertaining to an organization's containers, by utilizing operational logs, configuration data, and process documents.
- An increasing number of organizations have been adopting containers. Therefore, automated security can be used to facilitate prompt identification of vulnerabilities in containers, making container management less challenging.
- Authenticate and authorize container actions to safeguard the container and prevent unauthorized access.
- Disable unused OS capabilities. This reduces vectors of attack to a significant extent.
- Container image signing and verification serve as a means to validate container images and enforce policies while retrieving them into the system.
- Enforce fine-grained access control as needed for granting and managing permissions.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24



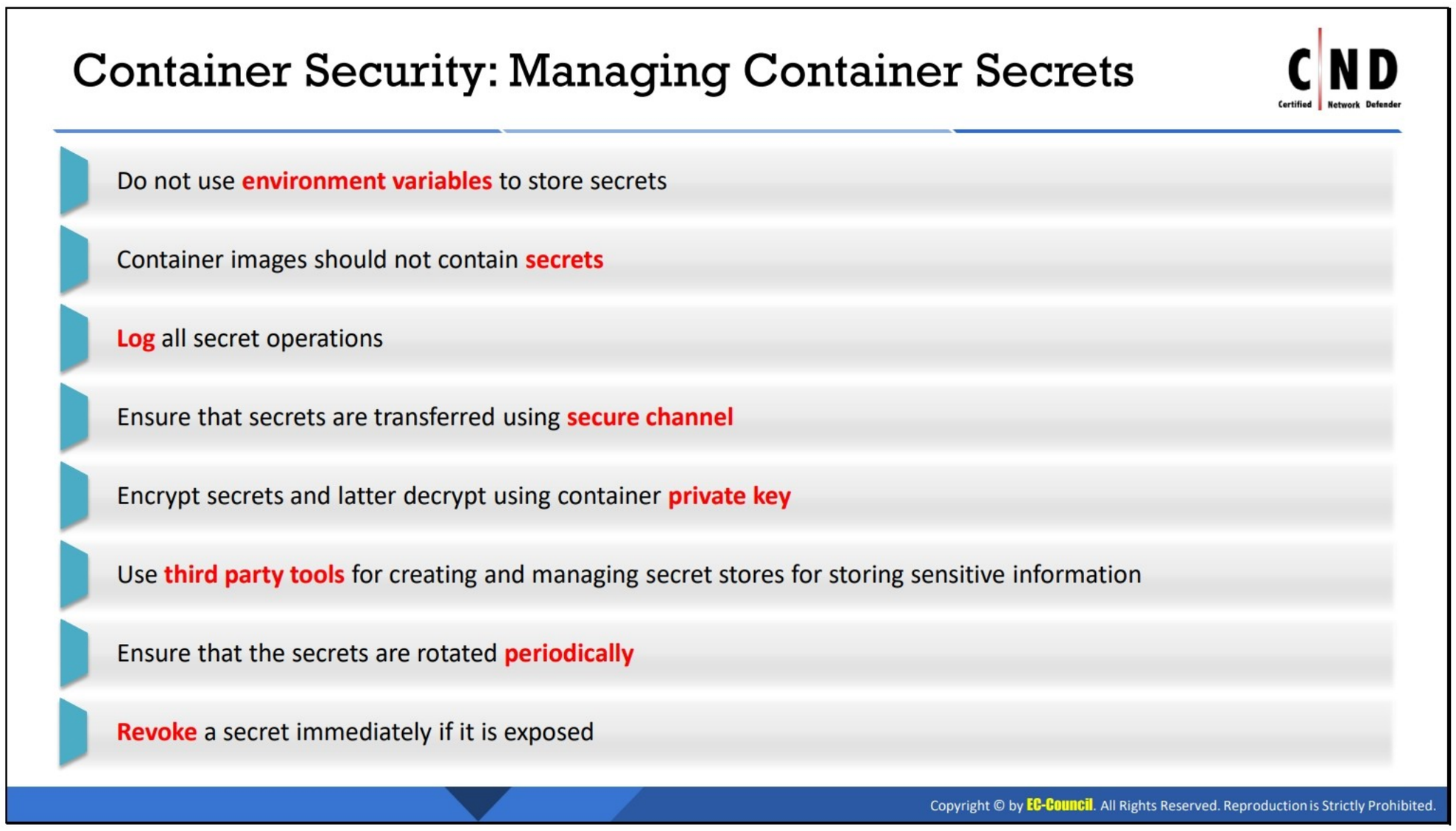
Container Security: Securing Container Image

The security of container images is crucial when a user migrates to Docker. The exposure of containers to various vulnerabilities can be prevented by adopting the following container image security measures:

- The source of the container image should be verified (i.e., download from a reliable source).
- Enable Docker content trust (DCT), which enables the image publishers (individuals or organizations) to sign the image and assure consumers that the image is authentic.
- Use default Docker options by signing the tagged version, so that Docker image integrity is implemented.
- Use image scanning services from third-parties.
- Implement strong access control mechanisms for container images. Change the default root privilege of the user to non-root.
- To limit damages, adopt task centric access control or role-based access control.
- Keep containers lightweight to increase performance and thereby decrease attack surface area.
- Use threat detection tools to monitor image health during runtime.
- Keep the images up-to-date with the latest versions and security patches.
- Avoid storing confidential tokens and other secrets in the container file or Docker file.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Container Security: Managing Container Secrets

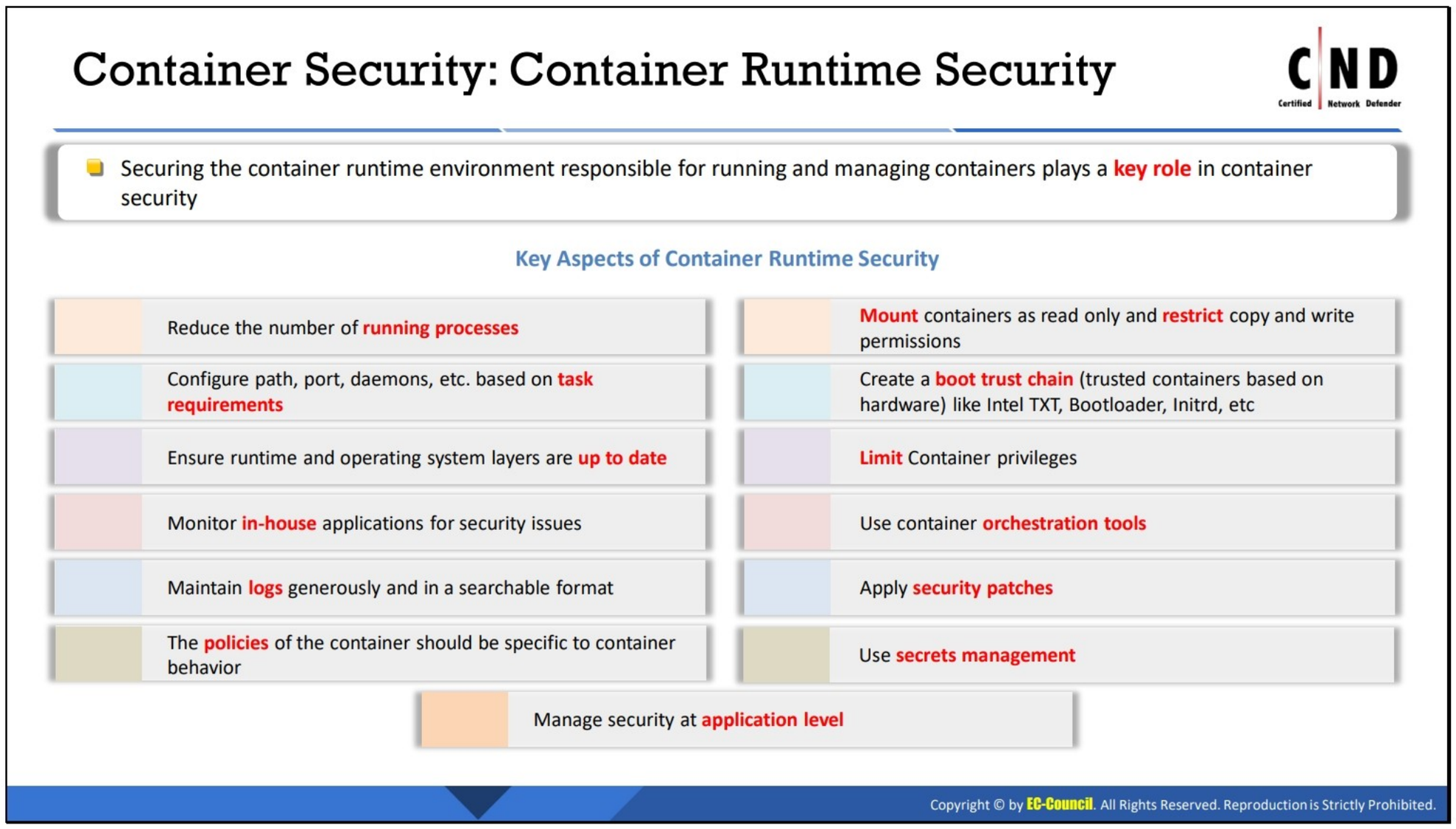
Secrets like passwords, access tokens, and API keys must be secured to prevent them from being accessed by unauthorized users with malicious intent. The following are some best practices to follow to protect container secrets.

- The container image should not contain a secret.
- Ensure that all secret operations are logged.
- The secret should be transferred through a secure channel.
- The secret should be encrypted and decrypted by the containers private key.
- Credential management tools or secret storage must be adopted while using third party vendor products.
- Each application must assume responsibility for authentication and authorization.
- Secrets should be rotated on a regular basis.
- Exposed secrets must be revoked.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24



Container Security: Container Runtime Security

Key aspects of secure container runtime are as follows.

- **A container should have only a few running processes**: A container should have only a few running processes. Maintaining multiple processes in a container makes it complicated to manage and troubleshoot.
- **Be task-specific**: Specifically in terms of paths, ports, configuration of daemons, mount points, etc.
- **Ensure runtime and operating system layers are up to date**: Outdated containers can pose compatibility, performance, and security risks. By being up to date, they offer portability, isolation, and scalability.
- **Adopt security monitoring of in-house applications, as they may have issues**: This helps monitors performance and health with full visibility into the application. Monitoring the application helps find vulnerabilities lurking in the container image.
- **Maintain generous logs and store them in searchable format for future analysis**: By analyzing logs, unused containers can be identified and removed from the host.
- **Create policies on containers**: Implement a policy on a monitored image to impose specific behavior constraints and enhance the security of the container.
- **Manage security at the application level to enhance container security**: Various techniques can be used that protect application code, find vulnerabilities, and provide insights into the security of applications.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

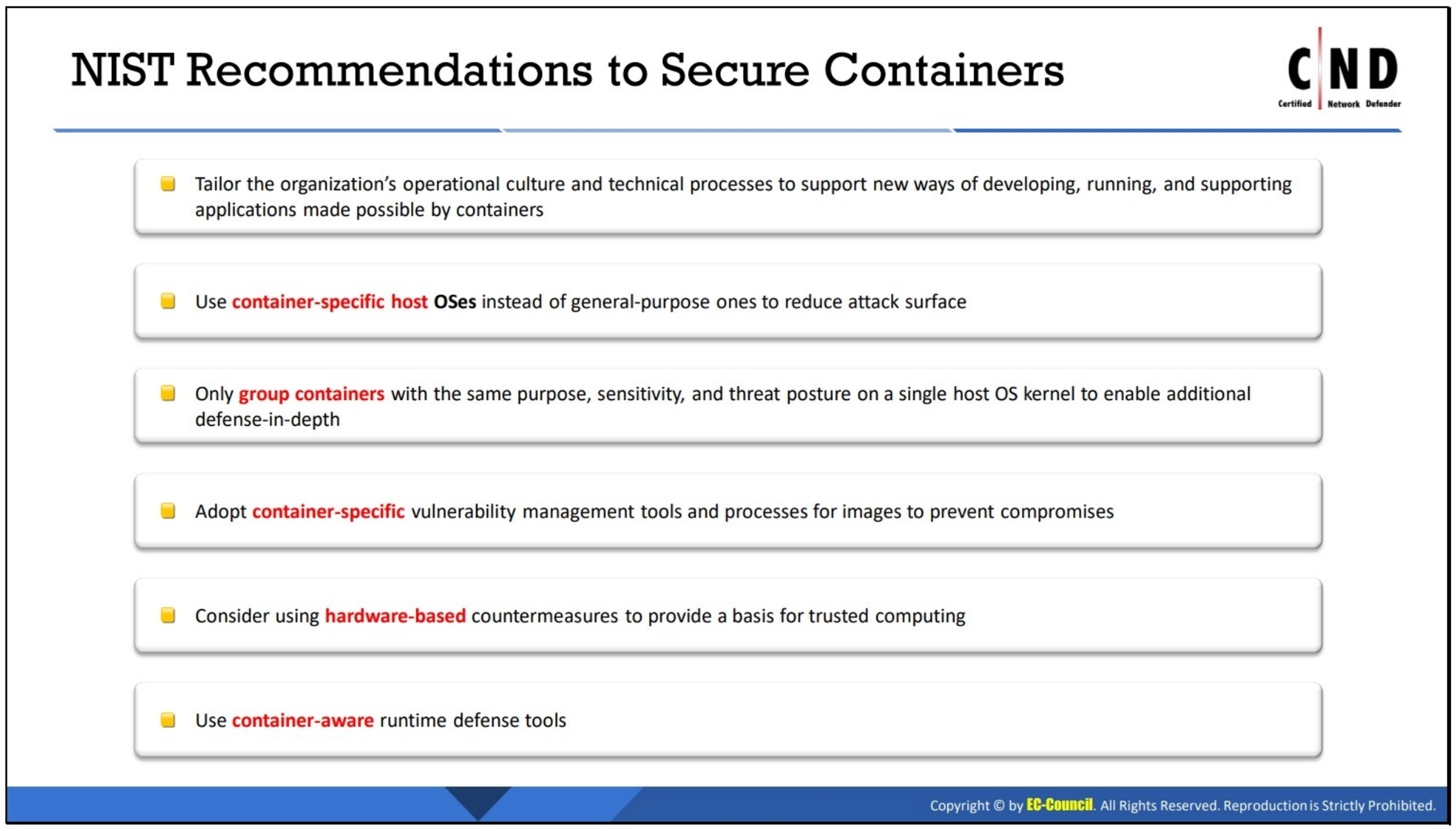
https://t.me/learningnets

- **Mount as read-only**: Ensure writing prevention in cases when only reading is required. This ensures that the container's file system is immutable. It reduces the risk of unauthorized changes or tampering with critical files during runtime.
- **Create a trust chain** (Intel TXT, Bootloader, Initrd, etc.): This automates container security.
- **Limit container privileges**: To successfully run the container, the privileges should be limited only to those that are required. It is secure to provide fine-grained privileges by granting specific capabilities instead.
- **Use container orchestration tools**: By doing this the threats are directly uncovered within the container code and the running environment.
- **Apply security patches and keep container runtimes up to date**: This ensures the latest security improvements and bug fixes.
- **Use secrets management, which a container needs at runtime**: Understand what authentication and authorization secrets are required to control and where they are needed in the container. Know how the secrets are configured, stored, and managed.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24



NIST Recommendations to Secure Containers

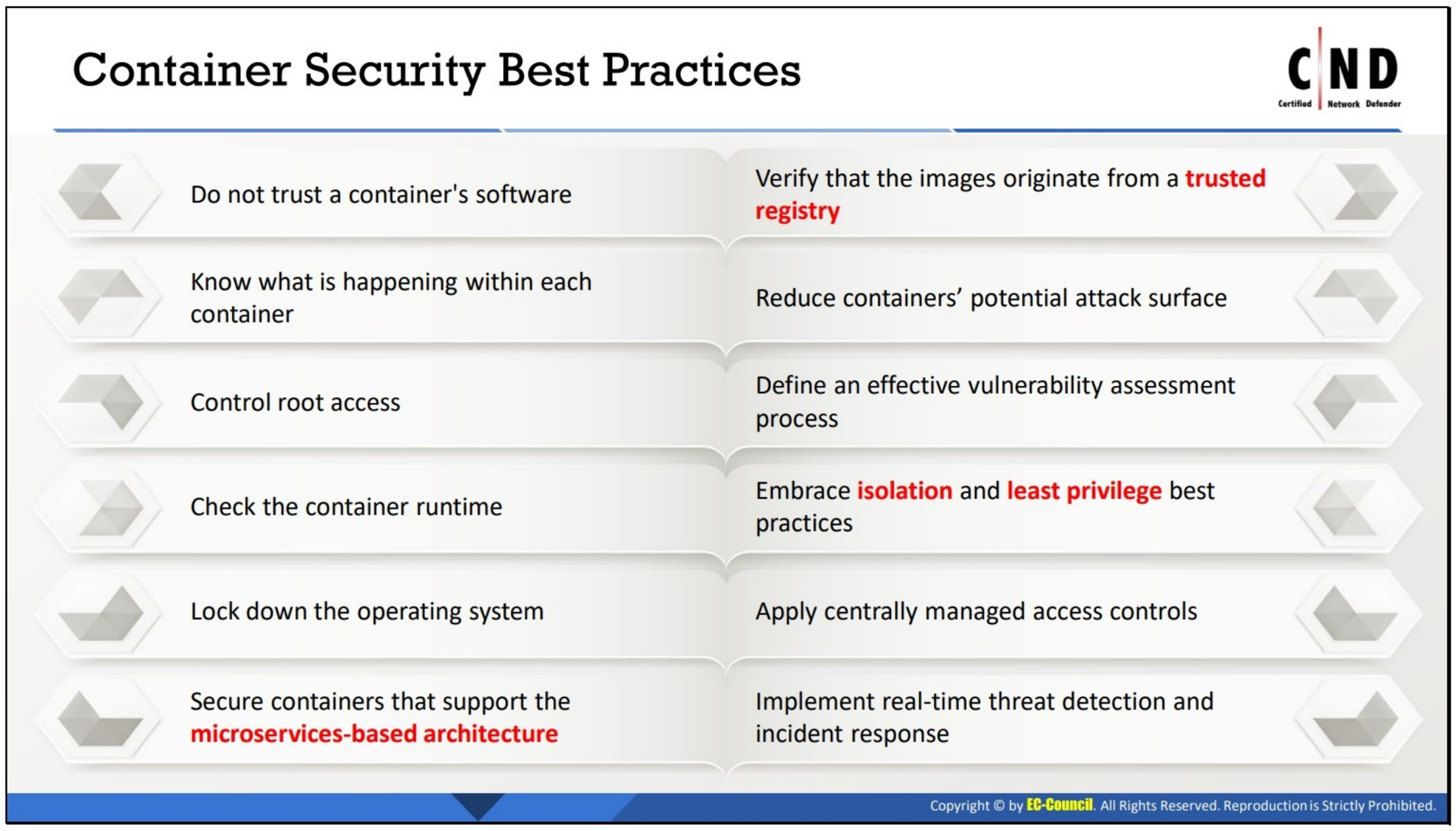
Source: https://nvlpubs.nist.gov

NIST recommends the following measures to secure the containers:

- Tailor the organization's operational culture and technical processes to support new ways of developing, running, and supporting applications made possible by containers.
- Use container-specific host OSes instead of general-purpose OSes to reduce attack surfaces.
- Only group containers with the same purpose, sensitivity, and threat posture on a single host OS kernel to allow for additional defense in depth.
- Adopt container-specific vulnerability management tools and processes for images to prevent compromise.
- Consider using hardware-based countermeasures to provide a basis for trusted computing.
- Use container-aware runtime defense tools.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Container Security Best Practices

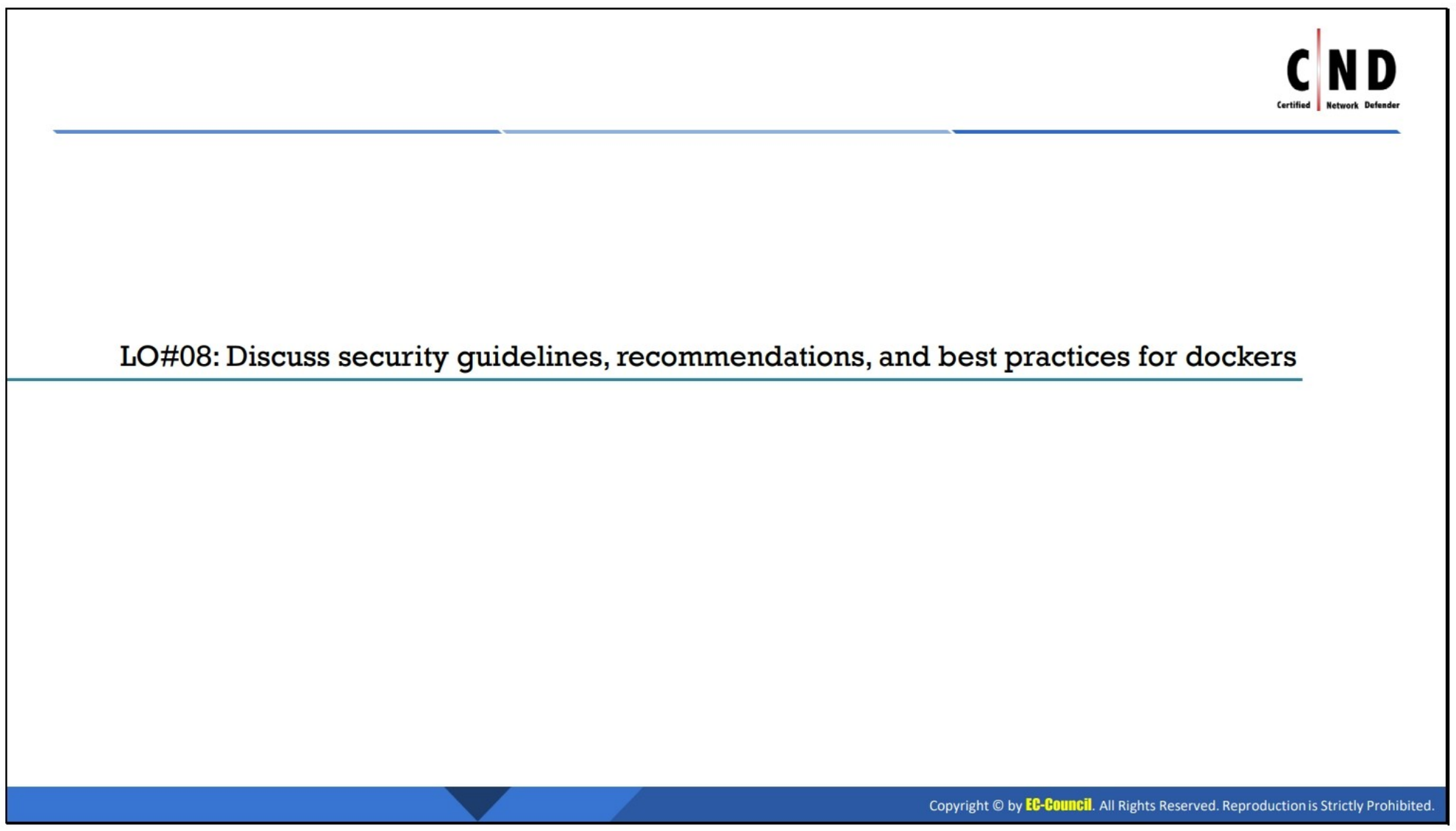
The following are some of the best practices to be implemented to secure a container:

- Hardening — Containers must be configured against benchmarks, adopt control features for host/daemon/kernel, avoid privileged mode execution, avoid noisy neighbors, limit resources such as CPU/memory, permit network traffic only on default bridge.
- Adopt minimal OS — To minimize surface area of attack (e.g.: Canonical Light OS, Redhat Atomic).
- Image Security — Adopt multistage build of an image and image scanning (from registry), use code level security, ensure authenticity, enable Docker content trust.
- Secret management – Separate until runtime, encrypt at rest.
- Health check — Ensure appropriate life cycle management, delete drifted containers, control container sprawl, adopt continuous monitoring of container traffic, ensure service log management.
- Process, file and device restrictions — Adopt privileges based on roles, RBAC, ensure authentication/authorization, avoid using AUFS driver, enable user namespace.
- Trusted containers based on hardware — Secure the trust chain.
- Use third party tools for threat control; facilitate configuration best practices with tools like Docker bench audit tool.
- Compliance — Ensure compliance with situation specific laws, frameworks, and benchmarks (FISMA, NIST, etc.)
- Adopt a holistic approach and perform security benchmarking.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

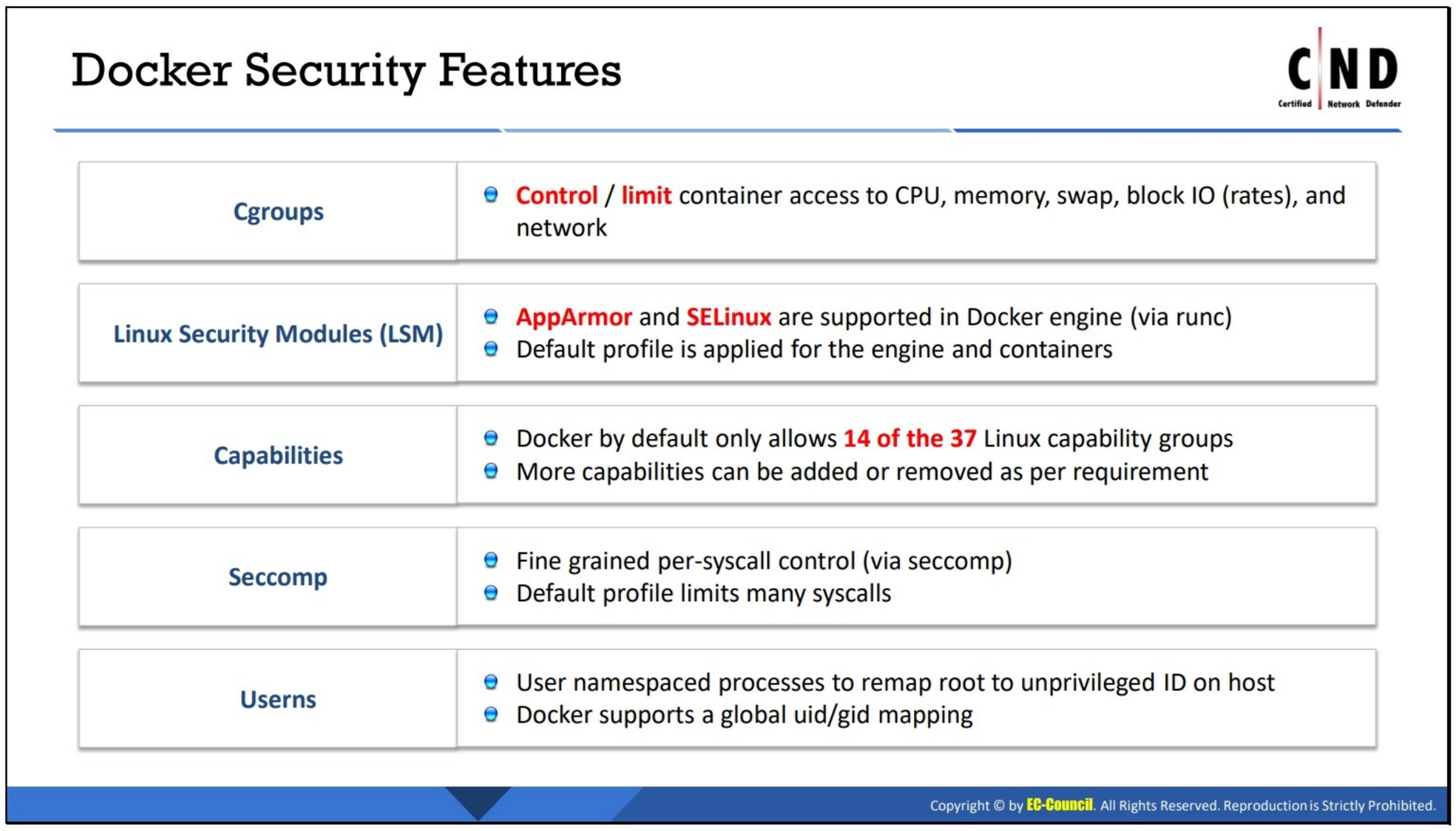


LO#08: Security Guidelines, Recommendations, and Best Practices for Dockers

While the previous section discussed security for containers in general, there are certain security issues and measures that pertain particularly to Dockers. The objective of this section is to explain these security measures, best practices, and recommendations for Docker security.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Docker Security Features

The adoption of Docker among organizations using containers for development or production has tremendously increased. Therefore, Docker specific security measures are crucial to understand in order to safeguard containers. Dockers can be secured by using the following Docker security tools.

Cgroups: This a Linux kernel feature to manage, restrict, and audit groups of processes. In Docker, cgroups control and limit container access to the CPU, memory, swap, block IO (rates), and network. For example, the cgroup can be used to set a memory limit.

LSMs: Linux security modules (LSMs) like AppArmor and SELinux are supported in the Docker engine (via runc). A default profile is applied for the engine and containers. These can be used, for example, to limit access to specific filesystem paths in the container.

Capabilities: Linux kernel capability can be used to split the root privileges and distribute them individually on a thread basis without granting all permissions to a certain process at a time. Docker by default only allows 14 of the 37 Linux capability groups. Additional capabilities can be added or removed as per requirement. These can be used for the security of the system. For example, unnecessary capabilities can be dropped from a container.

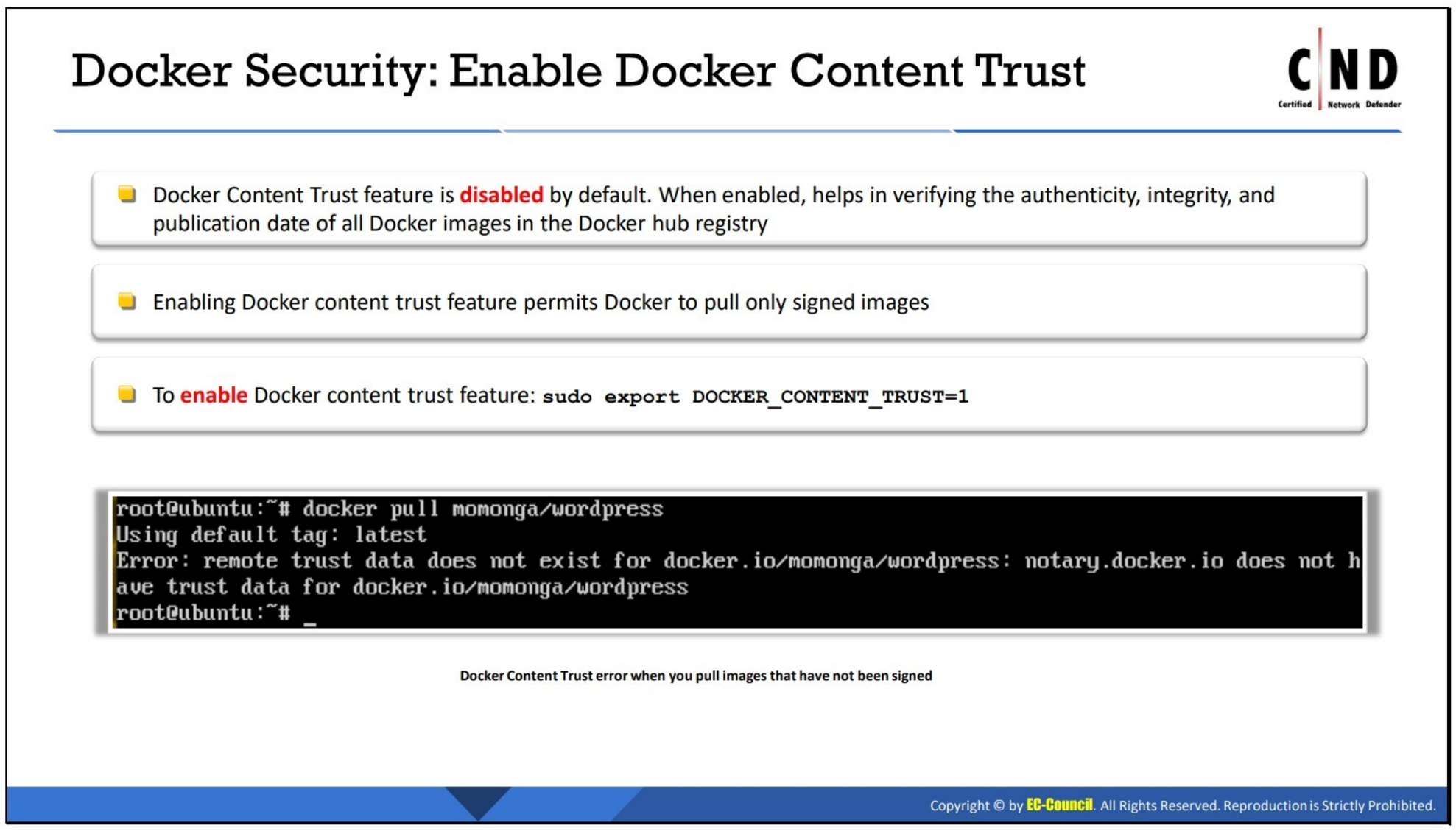
Seccomp: The secure computing mode (Seccomp) Linux kernel feature can be used to restrict actions within a container. It enables fine-grained per-syscall control. The default profile limits many syscalls. This feature can be used to block specific syscalls from being used by container binaries.

Userns: This Linux kernel feature isolates running process and limits access to system resources. User namespaced processes remap root to unprivileged IDs on the host. Docker supports global uid/gid mapping. This feature can be used enable user namespaces on the Docker daemon for all containers.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24



Docker Security: Enable Docker Content Trust

With Docker content trust (DCT), a user can verify the authenticity, integrity, and publication date of the Docker images present in the Docker hub registry. The DCT feature is disabled by default. Enable DCT for signing and verifying images that users build, push to, or pull from the Docker Hub. The DCT guarantees to the image user that the images are reliable and signed. With DCT enabled, the signed images are retrieved by Docker pull.

To enable DCT use the following command

```
Sudo export DOCKER_CONTENT_TRUST=1
```

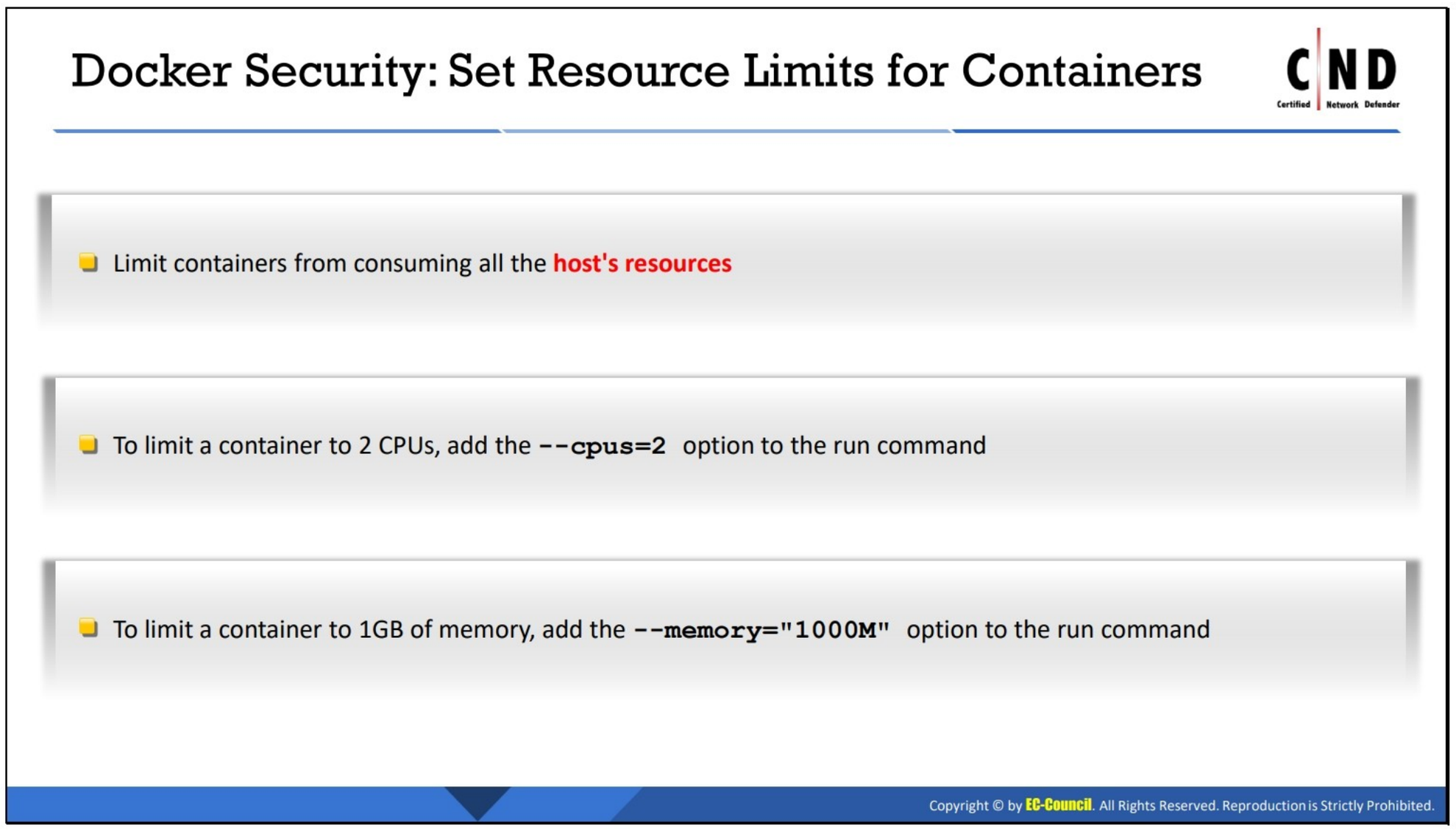
DCT displays the following error message when the user attempts to pull images that have not been signed:

```
root@ubuntu:~# docker pull momonga/wordpress
Using default tag: latest
Error: remote trust data does not exist for docker.io/momonga/wordpress: notary.docker.io does not h
ave trust data for docker.io/momonga/wordpress
root@ubuntu:~# _
```

Figure 11.31: Enable DCT

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Docker Security: Set Resource Limits for Containers

In the default state, a container has no limit on resources and can utilize as much resource as the host scheduler provides. Docker controls the amount of memory or CPU provided to the container. For setting these limits, the limit setting capabilities of Linux should be supported by the kernel. The details of how to set these limits are provided below.

Limit a container's access to memory:

If a running container consumes a considerable amount of memory from the host, the kernel may throw an out of memory exception (OOME) and kill other processes to free the memory. However, this can potentially cause the entire system to collapse if a crucial process is killed.

To prevent such issues, Docker enables the imposition of either hard memory limits that permit a container to only use a particular amount of system memory, or soft memory limits that permit the container to use memory without constraints under certain conditions such as the detection of overall low memory usage by the kernel.

Set resource limit for memory in a container

-m or --memory=

For example, to limit a container to 1 GB of memory:

`--memory="1000m"` (here *m* represents megabytes)

Set resource limit for CPU in a container

`--cpus=<value>`

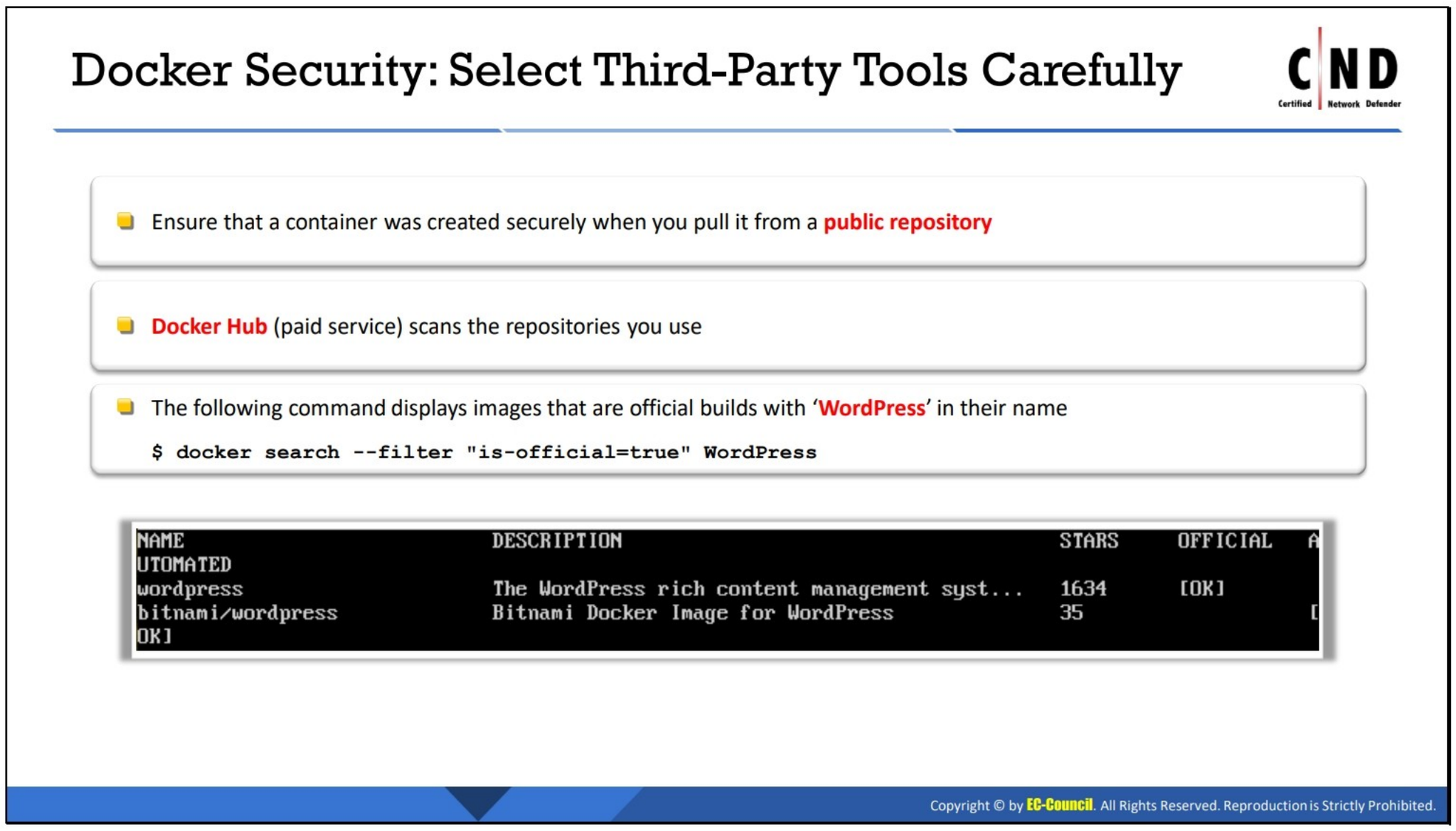
For example, to limit a container to 2 CPUs:

`--cpus="2"`

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24



Docker Security: Select Third-Party Tools Carefully

While using Docker, a user can pull containers from public repositories without knowing whether the containers were created securely. A container might have malicious or corrupt files. To safeguard against such files, a user should pull containers only from reliable sources such as the Docker Hub.

For example, when a user searches for Wordpress containers using the following command

```
Sudo docker search WordPress
```

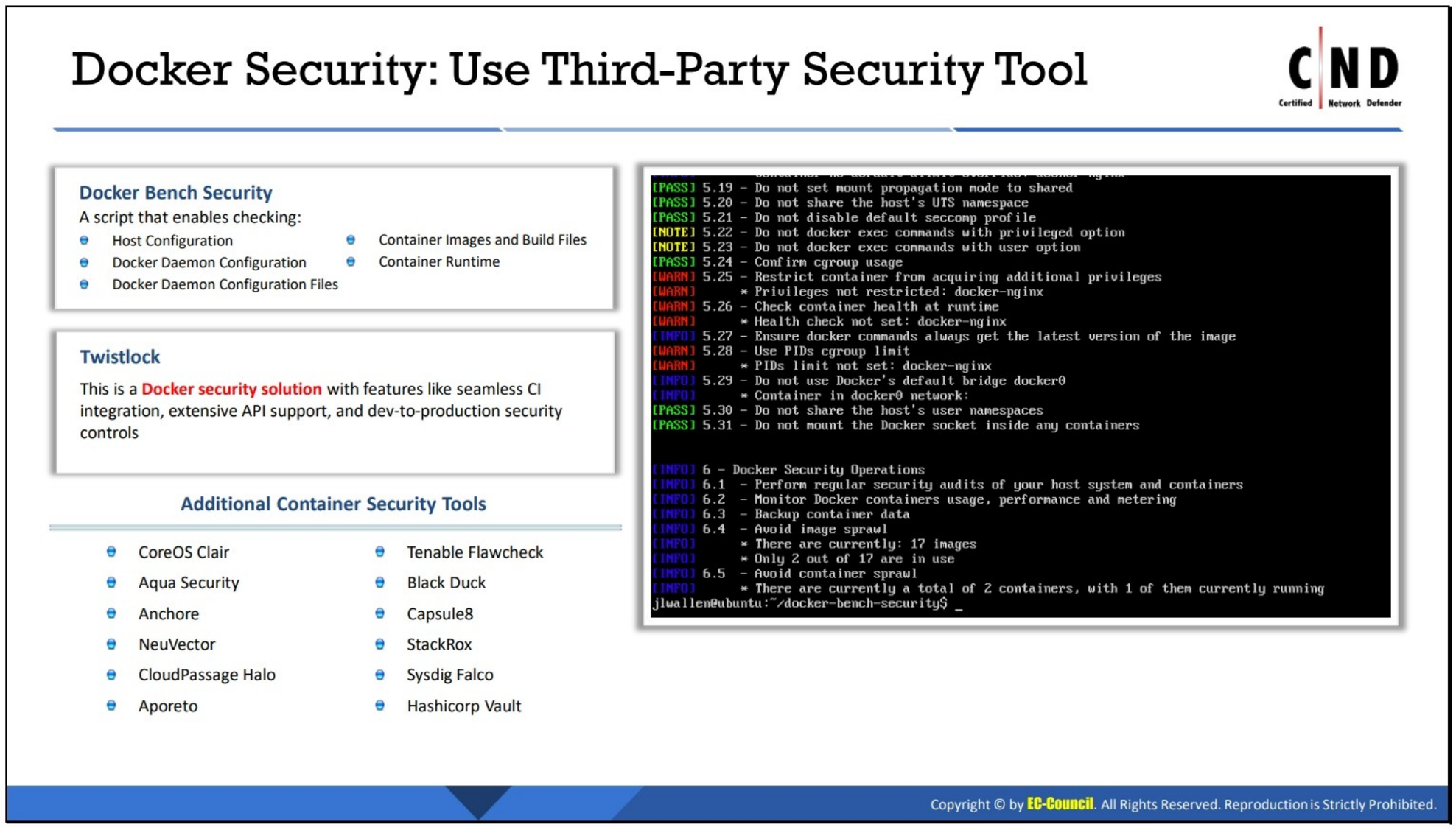
the results display an entry of Wordpress, along with other potentially relevant entries such as bitnami/wordpress and appcontainers/wordpress, as shown in the screenshot below. The results also indicate that the first entry in the list is the official image. This information can be used to distinguish official sources from third-party sources and tools.

```
NAME                            DESCRIPTION                                     STARS     OFFICIAL
UTOMATED
wordpress                       The WordPress rich content management syst...   1634      [OK]
bitnami/wordpress               Bitnami Docker Image for WordPress              35
OK]
appcontainers/wordpress         Centos/Debian Based Customizable Wordpress...   34
OK]
kaihofstetter/wordpress-cli     Installs a configured and ready to use Wor...   14
OK]
centurylink/wordpress           Wordpress image with MySQL removed.             12
OK]
trafex/wordpress                Lightweight WordPress container with Nginx...   5
OK]
maximiliend/wordpress           A enhanced docker image of WordPress with ...   5
OK]
scjalliance/wordpress           WordPress with GD and FreeType                  3
OK]
devgeniem/wordpress-server      Minimal wordpress server with nginx and ph...   3
OK]
```

Figure 11.32: Search for Wordpress Containers

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Docker Security: Use Third-Party Security Tool

A number of third-party tools can assist in the maintenance of Docker security. The following are examples of such tools.

Twistlock

Source: www.twistlock.com

Twistlock can secure VMs, containers, serverless functions, and service meshes, or combinations of these. It is a holistic platform for protecting hosts, networks, and applications. Twistlock provides real-time intervention, blocking, and prevention for in-process runtime attacks. It also provides insights and visibility into attempted attacks with detailed forensics, auditing, and real-time log analytics. It enables granular access control across all pivots and segments of the environment.

Aqua

Source: www. aquasec.com

Aqua provides full dev-to-prod security across the entire CI/CD pipeline and runtime environment. It provides the user with end-to-end visibility and protects applications against attacks. Aqua's security controls are consistently enforced across orchestrators, whether on premises or in cloud environments. It adopts least-privilege whitelisting to detect and prevent anomalous behavior, privilege escalation, or code injection.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

Anchore

Source: https://anchore.com

As part of the build pipeline, Anchore analyzes images and creates software container bills of materials. When new code is committed and pushed, the CI/CD system is triggered to build a new container image. Anchore can be integrated with the Kubernetes environment using admission controllers, ensuring that only images that meet the organization's policies are deployed.

NeuVector

Source: https://neuvector.com

NeuVector delivers the cloud-native Kubernetes security platform with uncompromising end-to-end protection, ranging from DevOps vulnerability protection to automated run-time security, and featuring a true layer 7 container firewall. It prevents exploits and breakouts by detecting and blocking suspicious processes and file system activity, and prevents damage with the industry's only container firewall. It also prevents attacks with cloud-native automated segmentation, deep packet inspection (DPI), and attack detection for DDoS, DNS, SQL injection, and DLP breaches.

CloudPassage

Source: www.cloudpassage.com

CloudPassage Halo is a security automation platform that delivers comprehensive visibility, protection, and continuous compliance monitoring to reduce cyber security risks. It assesses the security of the cloud compute, storage, and other infrastructure services such as containers, server instances, and serverless functions. It establishes and maintains compliance with CIS benchmarks and other best practices and regulations using built-in, customizable policies.

Tenable

Source: www.tenable.com

Tenable.io container security delivers end-to-end visibility of Docker container images, providing vulnerability assessment, malware detection, and policy enforcement across the software development life cycle, from the development to operation phases. By integrating with developer build systems, Tenable.io enables proactive visibility to solve security challenges of containers at the pace of DevOps. It protects containers from newly identified threats via monitoring of a wide range of external vulnerability databases. It also addresses new cyber risks that emerge after deployment.

Capsule8

Source: https://capsule8.com

Capsule8 enables attack detection and response for Linux environments – whether containerized, virtualized, or bare-metal – both on-premises and in the cloud. It provides distributed, streaming analytics combined with high-fidelity data detection, and can automatically respond to attacks the instant they are attempted. Continuous updates by a team of security experts ensure that the user is secured against the latest zero-day attacks.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

StackRox

Source: www.stackrox.com

StackRox protects cloud-native apps across the full life cycle including the build, deploy, and runtime phases. It monitors, collects, and evaluates system-level events such as process execution, network connections and flows, privilege escalation, and files launched within each container in Kubernetes environments to enable the user to detect and scrutinize suspicious activity more quickly. StackRox applies pre-defined policies to detect threats including cryptocurrency mining, privilege escalation, and various exploits.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

Docker Image Security Best Practices

Favor minimal base images

- Choose images with fewer OS libraries and tools to minimize risk and attack surface of the container
- Favor alpine-based images over full-blown system OS images

Sign and verify images

- To mitigate MITM attacks, pull only images that are pushed by the **publisher**, and have not been **tampered**
- Sign images with the help of notary
- Ensure that the pulled image is trustworthy and authentic

Implement least privileged policy

- The dedicated user and group should be created on the image, with **minimal** permissions to run the application
- Use the same user to run the process. **For example**, Node.js image has a built-in node generic user:

```
FROM node:10-alpine
USER node
CMD node index.js
```

Find, fix, and monitor for open source vulnerabilities

- Docker images should be scanned for known vulnerabilities as part of continuous integration.
- Use tool such as Snyk to scan for security vulnerabilities in open source application libraries and Docker images.
- Use **Snyk** to scan a Docker image:

```
$ snyk test --docker node:10 --file=path/to/
Dockerfile
```

- Use Snyk to monitor and alert for newly disclosed vulnerabilities in a Docker image: `$ snyk monitor --docker node:10`

Copyright © by EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited.

Docker Image Security Best Practices (Cont'd)

Use multi-stage builds for small secure images

- Use multi-stage build to create small and clean images with minimized attack surface and vulnerabilities

Use a linter

- Use a **static code analyzer** tool like hadolint linter, to detect and alert for issues found in a Dockerfile and enforce Dockerfile best practices

Use labels for metadata

- Image labels and metadata contain security details
- Use a **SECURITY.TXT** policy file to implement a security disclosure policy. Provide this information in your image label

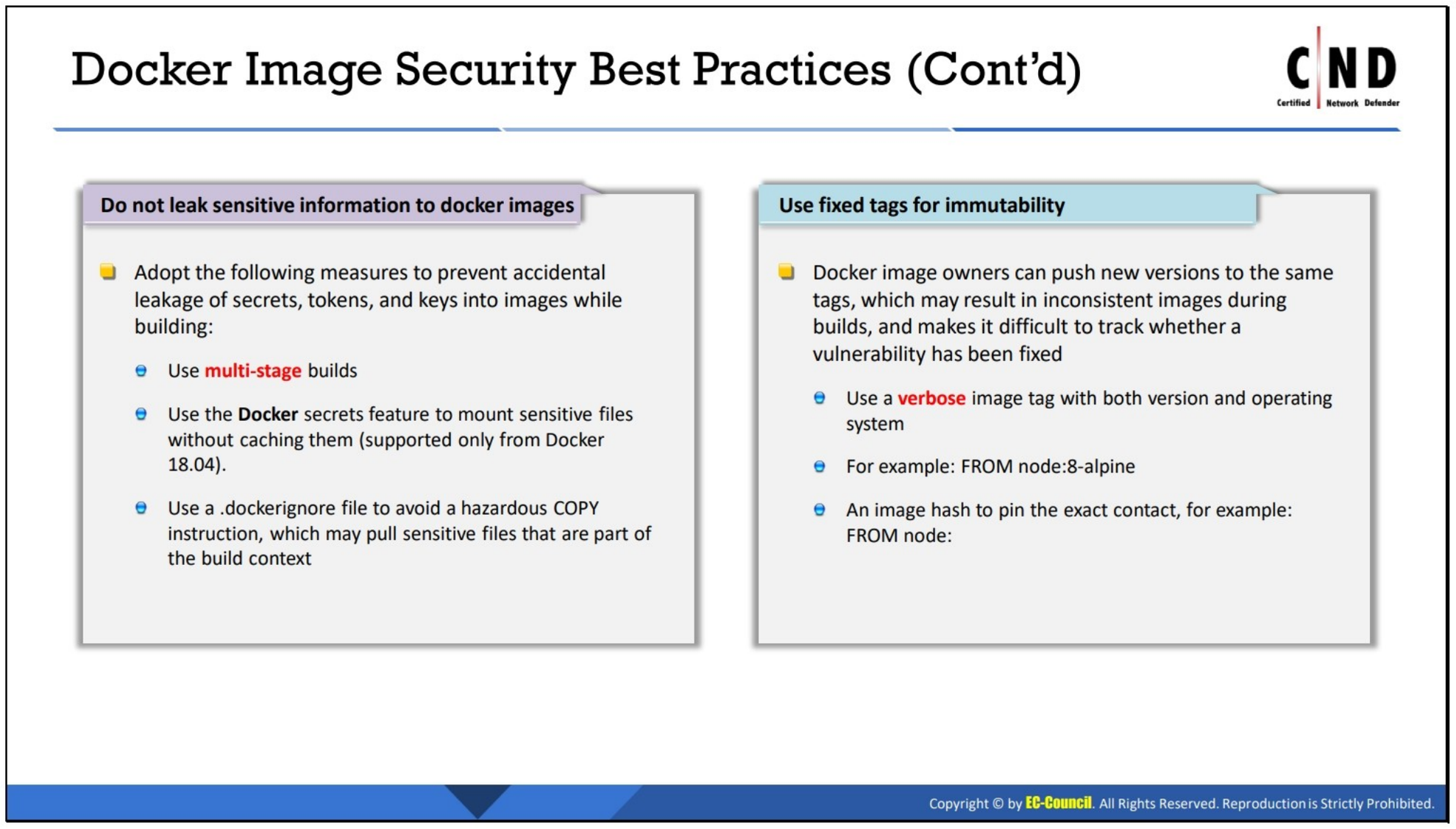
Use COPY instead of ADD

- ADD is vulnerable to MITM attacks, as arbitrary URLs specified could be malicious data sources
- ADD implicitly unpacks local archives, which could result in path traversal or zip slip vulnerabilities
- Use COPY, unless ADD is specifically required

Copyright © by EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Docker image security best practices

- **Favor minimal base images**: Favor images with fewer number of OS libraries and tools. This not only decreases risk, but also reduces the attack surface area of the container.
- **Minimize user privilege**: The user and group created on the image should be given minimum permission to run the application, and the same user should be used to run the process.
- **Sign and verify images to mitigate MITM attacks:** The user must verify the trust and authenticity of an image before pulling, and sign images with notary.
- **Find, fix and monitor for open source vulnerabilities**: A Docker image should be scanned for vulnerabilities as part of continuous integration (CI).
- **Do not leak sensitive information to Docker images**: Prevent unintentional leakage of secrets, tokens, and keys into images while creating them. To ensure this, utilize multi-stage builds, Docker's secret feature, and the .dockerignore file.
- **Use fixed tags for immutability**: Pushing new versions with the same tags can lead to inconsistency in images during builds, and thus cause hindrances while tracking fixed vulnerabilities. To avoid this, use a verbose image tag and an image hash.
- **Use COPY instead of ADD:** The arbitrary URLs specified for ADD can enable man-in-the-middle attacks, path traversal, and Zip Slip vulnerabilities. To avoid this, utilize COPY unless ADD is specifically required.
- **Use labels for metadata**: Use labels in image metadata, to provide important information such as security details to users. Implement a responsible security disclosure policy and store this information in images labels.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

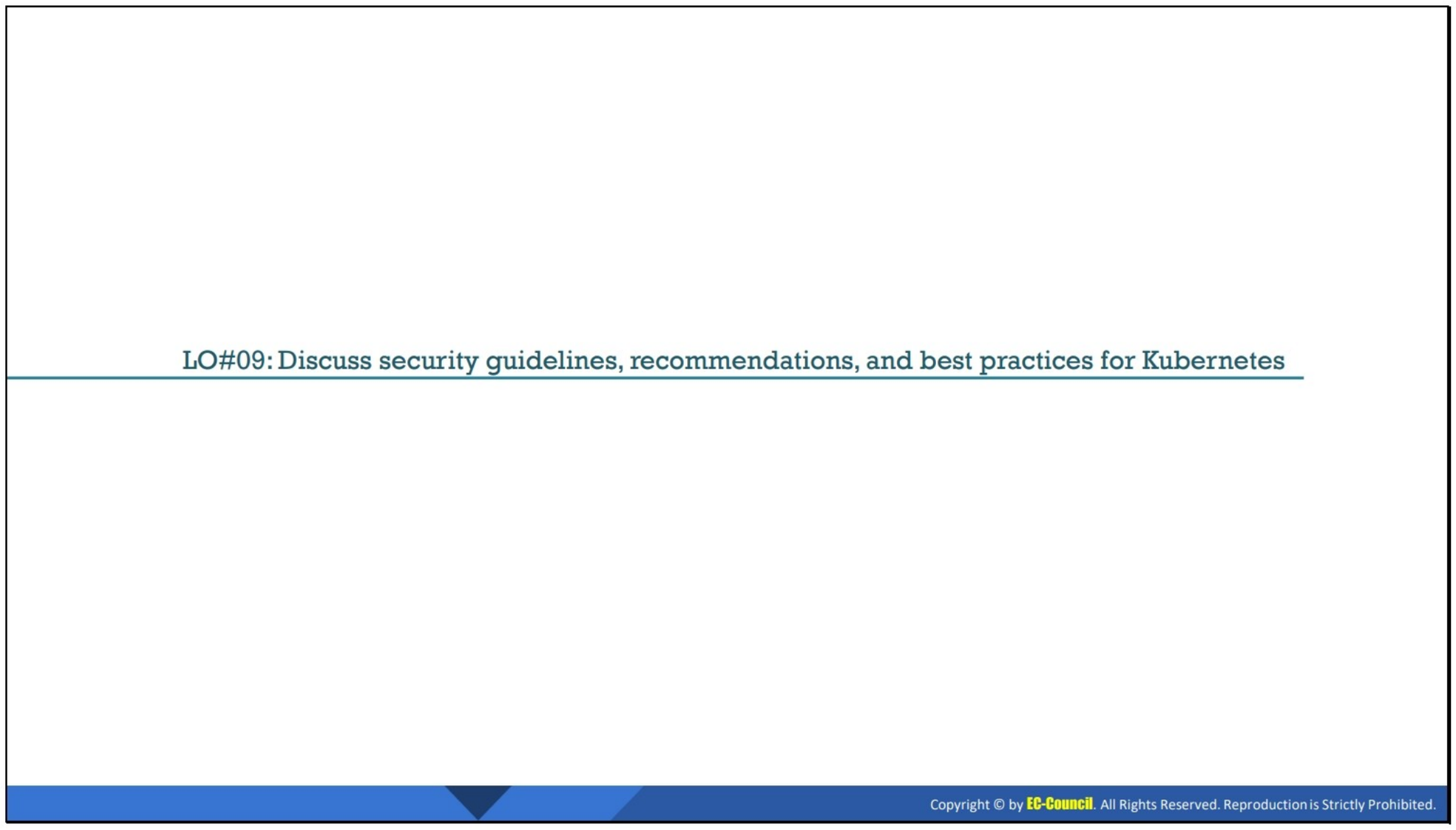
https://t.me/learningnets

Technet24

- **Use multi-stage builds for small secure images**: Use multi-stage builds to reduce the attack surface for Docker image dependencies.
- **Use a linter**: Use the hadolint linter to resolve concerns related to Docker files.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



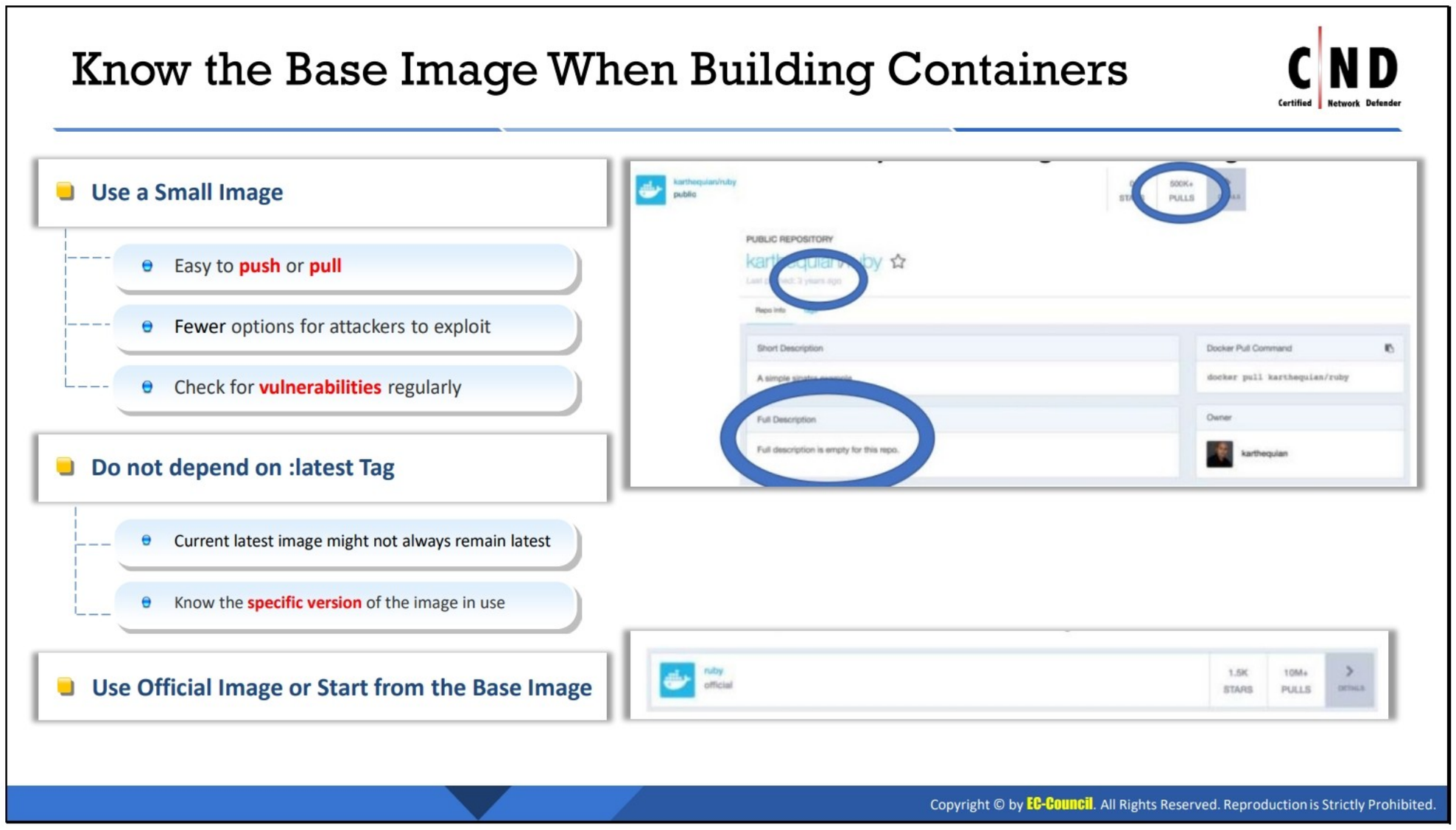
LO#09: Security Guidelines, Recommendations, and Best Practices for Kubernetes

The objective of this section is to explain guidelines, best practices, and recommendations for the security of Kubernetes. Each of the following subsections explains a guideline, best practice, or recommendation in detail.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24



Know the Base Image When Building Containers

This is a best practice that prescribes being aware of and understanding all the component software in the container. This enables the building of a secure container. The following two guidelines facilitate the adoption of this best practice.

Use a small image

Use minimal base image that does not contain unnecessary software, which can create a bigger attack surface. Minimizing the components in containers can restrict the attack vectors. The images must be checked for vulnerabilities regularly.

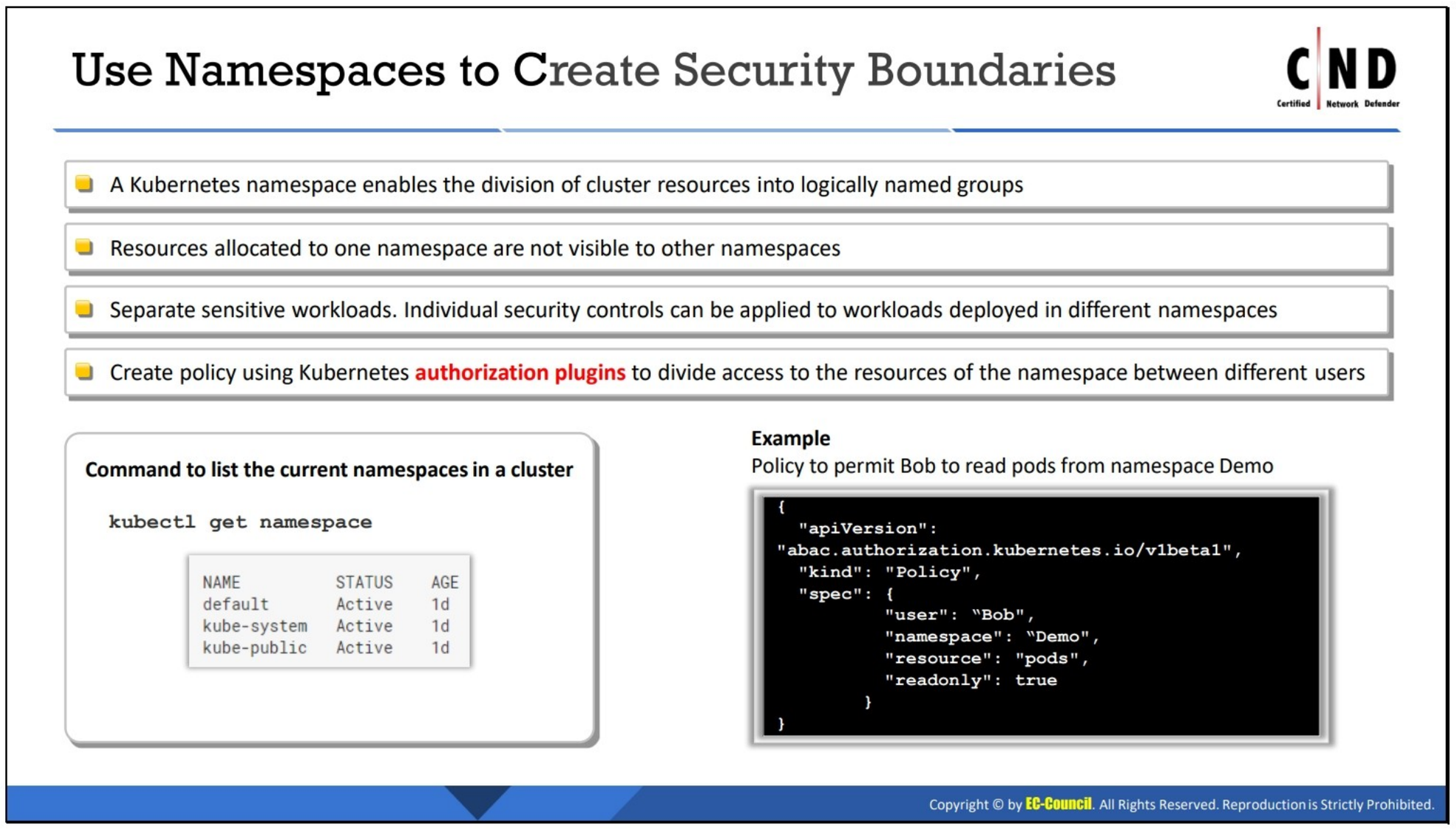
Smaller images can also enhance performance by consuming only a few bytes on disk and reducing network traffic. BusyBox and Apline are examples of tools that are helpful for building small images.

Do not depend on :latest tag

Do not depend on the tag **:latest** as this may cause the building of container images against a version. It is recommended that particular version number is used as the image's tag and the version number is updated.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Use Namespaces to Create Security Boundaries

To isolate components, create namespaces. This secures the cluster by dividing the resources logically. Security controls like network policies can be applied on different workloads deployed in various namespaces. Resources allocated to one namespace are not visible to other namespaces. Therefore, namespaces help reduce the impact of a compromise and can separate sensitive workloads.

To list the current namespaces in a cluster use the following command

```
kubectl get namespace
```

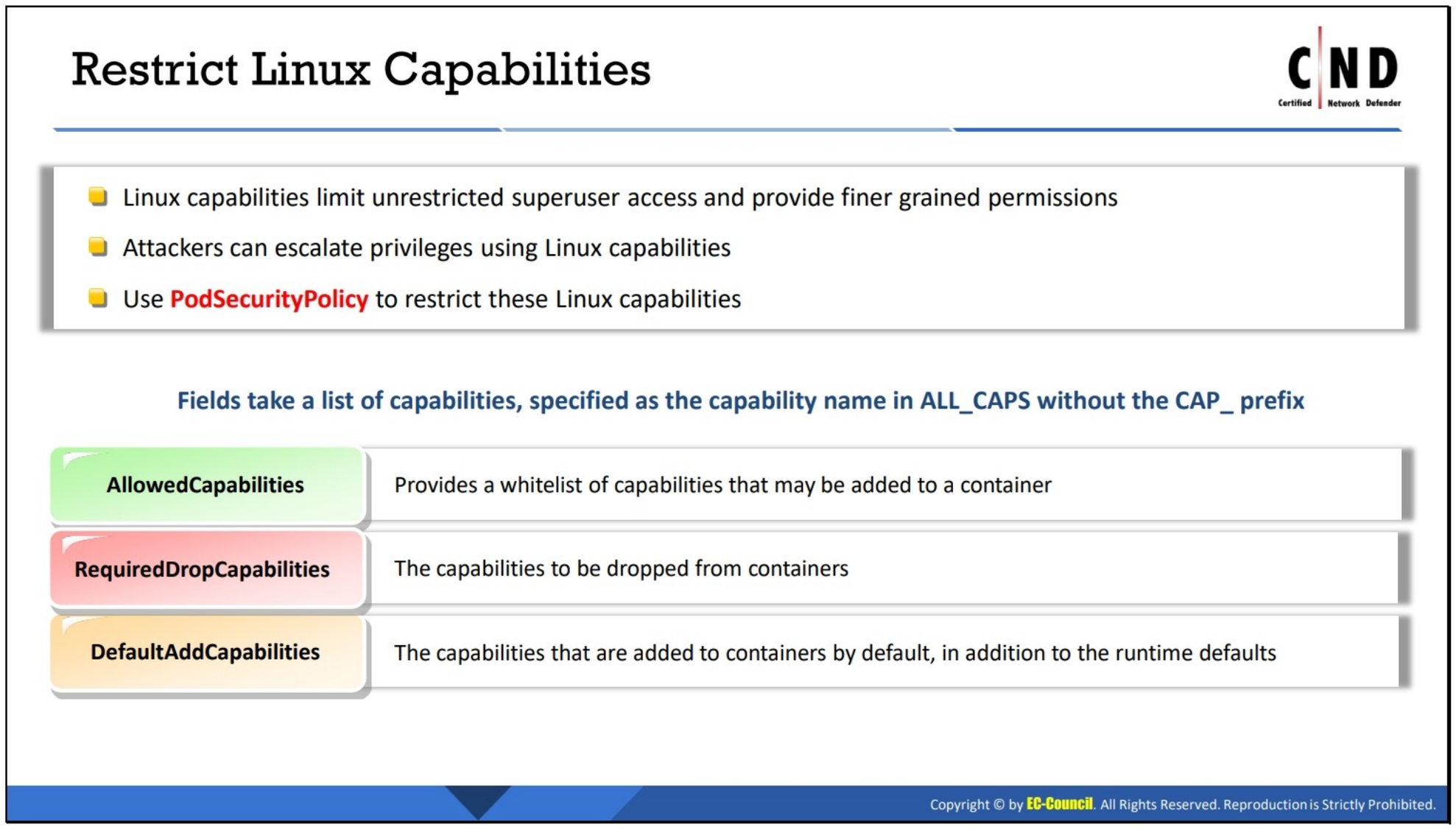
A policy can divide access to the resources of the namespace between different users. The following is an example of a policy to allow Bob to read pods from namespace Demo:

```
{
  "apiVersion":  "abac.authorization.kubernetes.io/v1beta1",
  "kind": "Policy",
  "spec": {
     "user": "Bob",
     "namespace": "Demo",
     "resource": "pods",
     "readonly": true
        }
}
```


Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24



Restrict Linux Capabilities

Linux capabilities enable the limiting of unrestricted superuser access to users and provide finer grained permissions. Attackers can escalate privileges using Linux capabilities. To prevent such attacks, these Linux capabilities can be restricted using PodSecurityPolicy. The following fields in PodSecurityPolicy take a list of capabilities, specified as the capability name in ALL_CAPS without the CAP_ prefix:

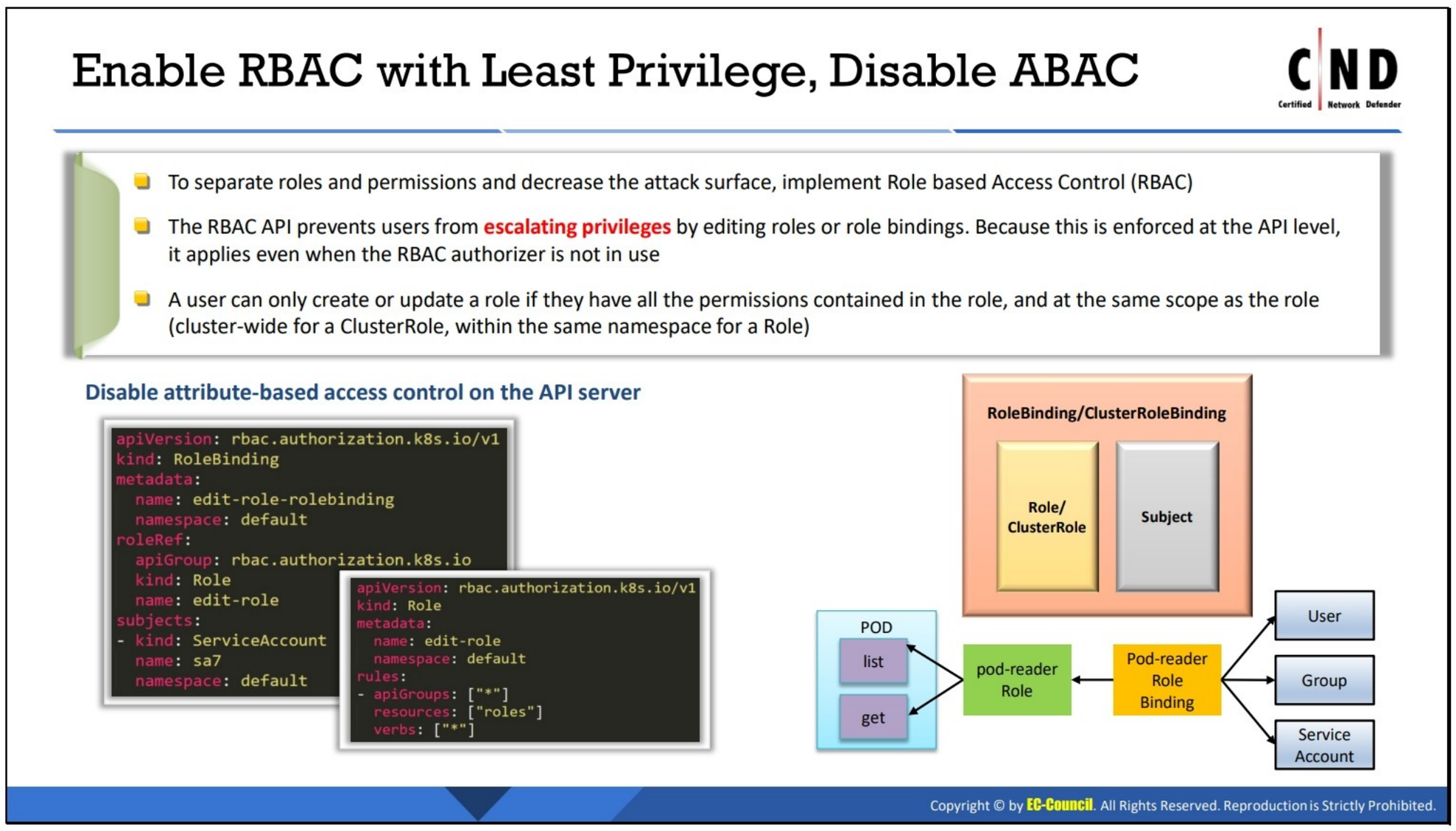
AllowedCapabilities: Provides a whitelist of capabilities that may be added to a container.

RequiredDropCapabilities: The capabilities to be dropped from containers. These capabilities are removed from the default set, and must not be added. Capabilities listed in the RequiredDropCapabilities must not contain AllowedCapabilities or DefaultAddCapabilities.

DefaultAddCapabilities - The capabilities that are added to containers by default, in addition to the runtime defaults.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Enable RBAC with Least Privilege, Disable ABAC

This recommendation prescribes enabling role-based access control (RBAC) with least privilege, and disabling the attribute-based access control (ABAC), as described below.

The RBAC feature of Kubernetes' authorization module is used for fine-grained policy management and restricting access to resources such as namespaces on the network.

The following is an example usage of this feature.

For a specific namespace, a User **Smith** wants to be able to list the pods in a namespace **demo**. The following are the steps to follow to achieve this:

- Assuming **Smith** is already authenticated, create a Role **listpods** for namespace **demo** and verb list.
- Create a RoleBinding to bind the User Smith and Role listpods.
- Once this is in place, User Smith can list the pods in namespace demo.

For a cluster, a User David wants to be able to list the pods across the cluster. The following are the steps to follow to achieve this:

- Assuming David is already authenticated, create a ClusterRole democlusterrole for verb get and list.
- Create a **ClusterRoleBinding** to bind User David and ClusterRole democlusterrole.
- Once this is in place, User David can list the pods cluster-wide.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

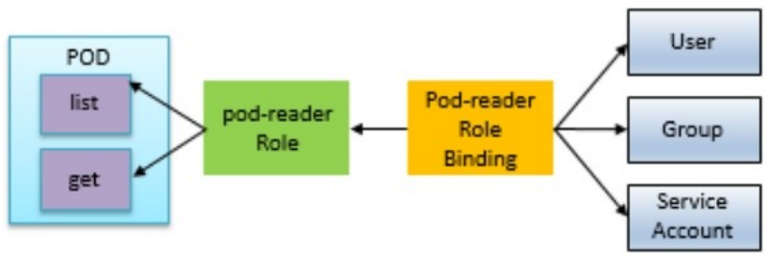


Figure 11.33: RBAC Example

RBAC Structure

The permissions of RBAC consist of three parts:

- **Role or ClusterRole –** This is the original permission, and consists of rules. In turn, these rules include permissions. Every rule consists of resources and verbs, where verbs are actions applied on the resources. Otherwise stated, rules are defined to perform verbs on resources.

  Roles pertain to namespaces and ClusterRoles pertain to clusters.

- **Subject (User, Group, or ServiceAccount) –** This is an entity (user, group, or a service account) that requires permissions.
- **RoleBinding or ClusterRoleBinding –** This is the connection between Role/ClusterRole and the subject. If a subject needs to access a resource based on a rule, define bindings between RoleBinding and ClusterBinding. Here, RoleBinding attaches a Role to a subject for resource access within a namespace and ClusterRoleBinding attaches a ClusterRole to a subject for resource access cluster-wide.

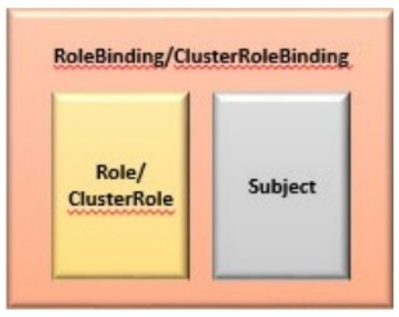


Figure 11.34: RBAC structure

Built-in Privileged Escalation Prevention

The RBAC API prevents users from accessing resources without the requisite permissions by applying fine-grained privileges in the form of editing roles and role bindings. As this is done via the API, the restriction remains even when the RBAC authorizer is not active. A user can only **create or update a role if they already have all the permissions contained in the role**, and at the same scope as the role (cluster-wide for a ClusterRole, and within the same namespace or cluster-wide for a Role).

Disable Attribute-based access control on the API server

Kubernetes' ABAC is swapped with RBAC since release 1.6. Disable ABAC on the API server and use RBAC instead.

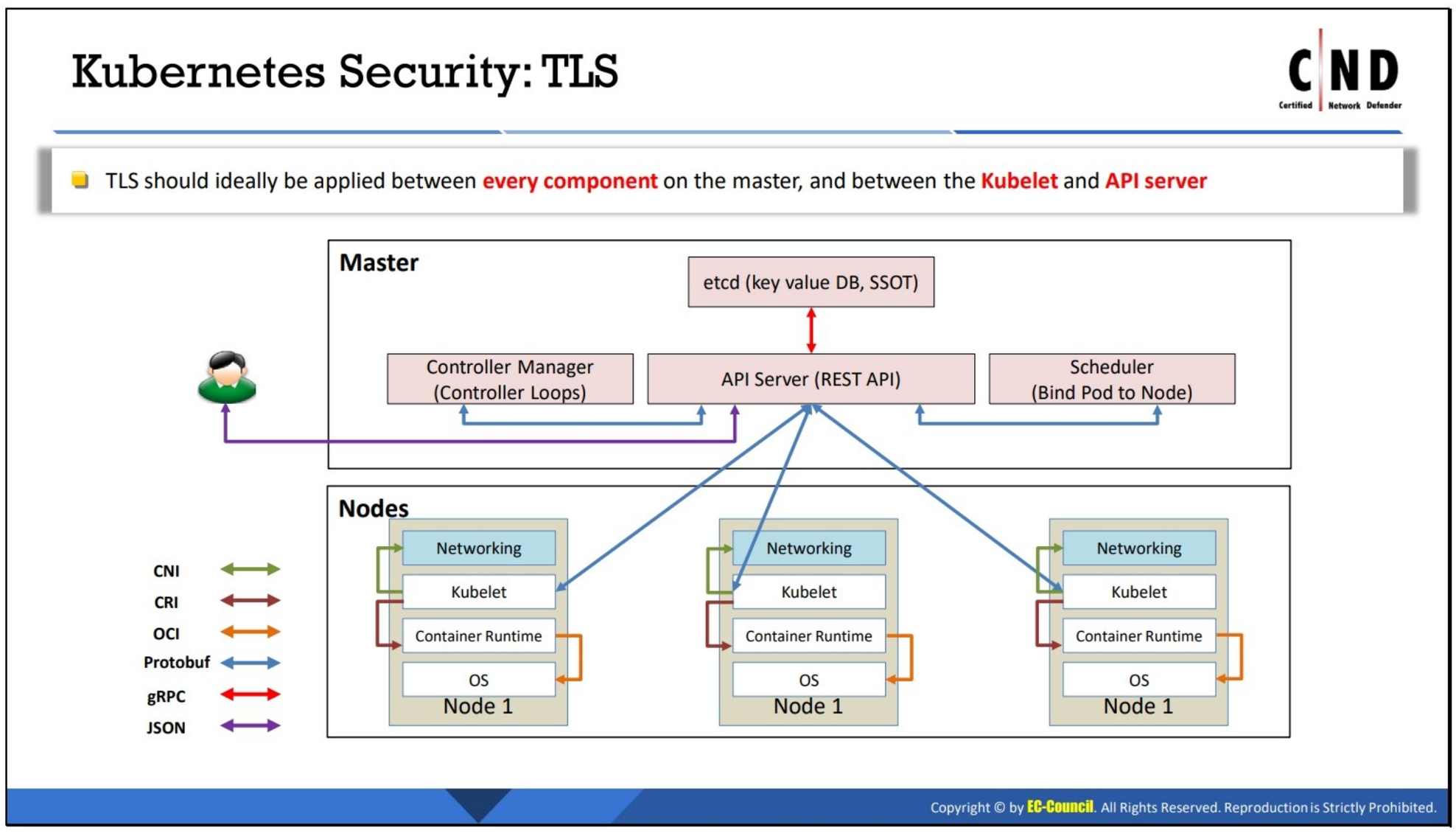
```
--authorization-mode=RBAC
```

(or) use this flag to disable it in GKE:

```
--no-enable-legacy-authorization
```


Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Kubernetes Security: TLS

Enable transport layer security (TLS) for the components that support TLS, for traffic sniffing prevention, server identity verification, and client identity verification.

Kubernetes expects encryption with TLS by default for the cluster's API communications. Most installation methods include the creation and distribution of necessary certificates for the components of the cluster. However, local ports over HTTP may sometimes be enabled by some components and installation methods. It is recommended that administrators familiarize themselves with the component's settings for the identification of unsecured traffic.

The following network diagram demonstrates the application of TLS between every component on the master/control plane, and between the Kubelet (the primary agent that runs on each node) and API server.

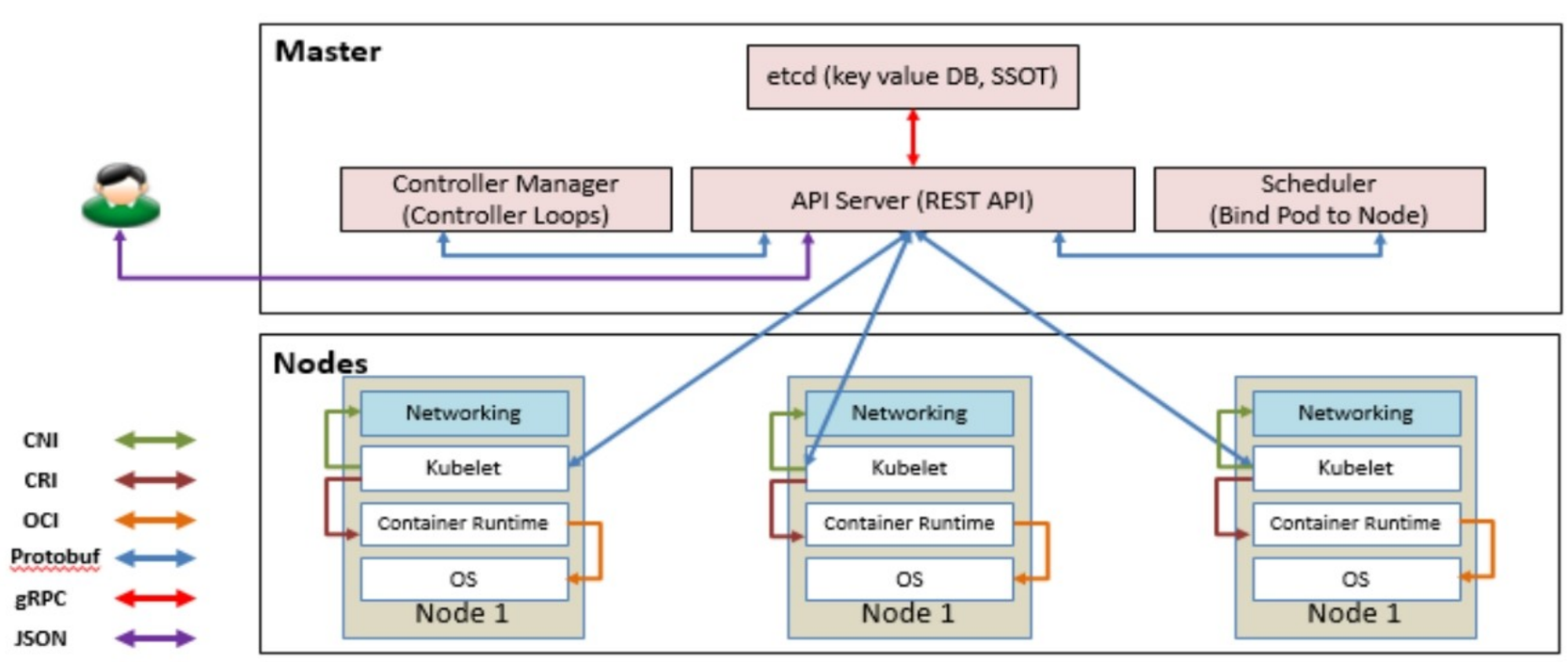


Figure 11.35: Application of TLS Between Components on the Master/Control plane, and Between the Kubelet and API server

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

Control plane

The components of the control plane can run on any machine in the cluster. Set up scripts start all the components of the control plane on the same machine. It is recommended that user containers are not run on this machine. The components of the control plane focus on the global decisions about the cluster, detecting cluster events, and responding to cluster events, such as scheduling or starting a new pod if a deployment's replicas field is unsatisfied.

API Server

The front end of the Kubernetes control plane, the Kubernetes API Server, is kube-apiserver, which scales by deploying more instances (horizontal scaling). By running many kube-apiserver instances, traffic can be balanced between the instances.

etcd

The etcd can serve as Kubernetes' backing store for the cluster data. Ensure a backup plan for the cluster data if the cluster uses etcd as its backing store.

Scheduler

The scheduler monitors for new pods with no assigned node, and assigns a node for such newly created pods to run on. The following are the factors that influence scheduling decisions:

- Resource requirements
- Hardware, software, and policy constraints
- Affinity and anti-affinity specifications
- Data locality
- Inter-workload interference and deadlines

Controller manager

The controller manager runs controller processes. All the controller processes are combined into a binary and run as a single process to minimize complexity. Controllers are of the following kinds:

- **Node controller** detects and responds when nodes go down.
- **Replication controller** maintains the correct number of pods for the replication controller object in the system.
- **Endpoints controller** focuses on populating the endpoints object. For example, it combines services and pods.
- **Service account and token controllers** focus on creating default accounts and API access tokens for new namespaces.

Node components

Node Components run on all nodes. These components maintain running pods and provision the Kubernetes runtime environment.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Kubelet

Kubelet works as an agent, runs on each cluster node, and ensures the running of containers in a pod.

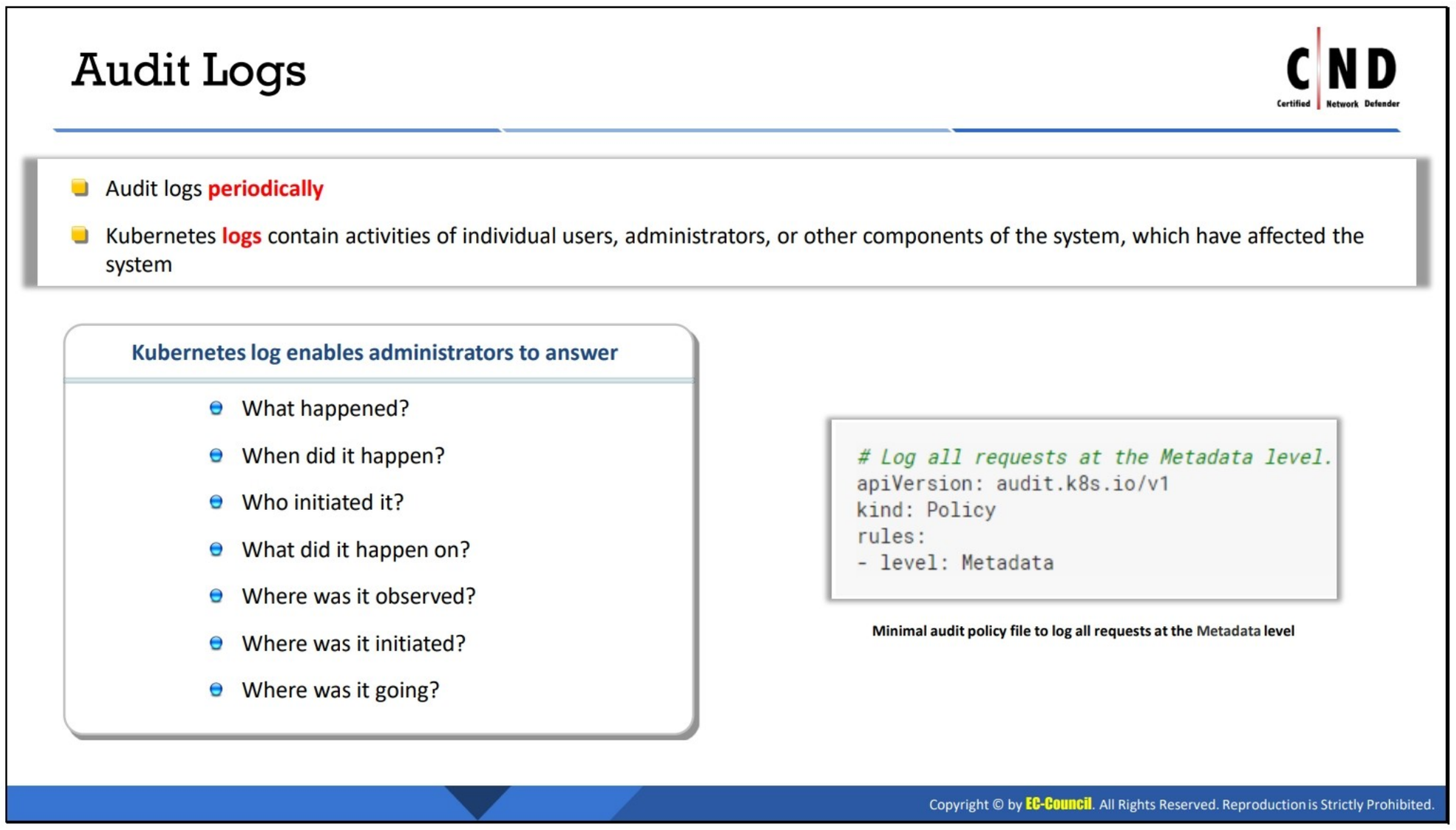
Container runtime

The container runtime is a software that focuses on running containers. Examples include Docker, container, Kubernetes container runtime interface (CRI) implementations, and CRI-O.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24



Audit Logs

Kubernetes audit logs contain a record of activities of users, administrators, or system components, which have affected the system. Audit logging features customize API logging both at metadata levels and the payload (for example, request and response). For requests involving read (get, list, and watch), the request object is exported to the audit logs. For requests that involve sensitive data (Secret and ConfigMaP), the metadata is saved in audit logs. For the remaining requests, the requests are exported to audit logs. These log levels can be set as per the policy of the organization.

Audit logs enable the cluster administrators to answer the following questions.

- What happened?
- When did it happen?
- Who initiated it?
- What did it happen on?
- Where was it observed?
- Where was it initiated?
- Where was it going?

Use a minimal audit policy file for logging all the requests at the metadata level:

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

```
# Log all requests at the Metadata level.
apiVersion: audit.k8s.io/v1
kind: Policy
rules:
- level: Metadata
```

Figure 11.36: Minimal Audit Policy File

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

Implement Network Policies

Kubernetes network policies are specifications describing how groups of pods can communicate with each other and other network endpoints

The **default network policy** permits each pod to talk toall other pods. Therefore, create a network policy to restrict the communication between pods

Example: Policy to deny all ingress traffic by default (service/networking/network-policy-default-deny-ingress.yaml)

```
---
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: default-deny-ingress
spec:
  podSelector: {}
  policyTypes:
  - Ingress
```

Copyright © by EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited.

Implement Network Policies (Cont'd)

Sample Network Policy

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: test-network-policy
  namespace: default
spec:
  podSelector:
    matchLabels:
      role: db
  policyTypes:
  - Ingress
  - Egress
  ingress:
  - from:
    - ipBlock:
        cidr: 172.17.0.0/16
        except:
        - 172.17.1.0/24
```

```
- namespaceSelector:
        matchLabels:
          project: myproject
    - podSelector:
        matchLabels:
          role: frontend
    ports:
    - protocol: TCP
      port: 6379
  egress:
  - to:
    - ipBlock:
        cidr: 10.0.0.0/24
    ports:
    - protocol: TCP
      port: 5978
```

Copyright © by EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited.

Implement Network policies

A network policy specifies the communication among groups of pods, and also the communication between the groups of pods and network endpoints. NetworkPolicy resources apply labels for selecting pods and defining rules, specifying what traffic is allowed among the selected pods. NetworkPolicy resources are additive. For example, if multiple policies select a pod, the pod is isolated based on the union of the policies' rules.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

The default network policy enables all pods to communicate with other pods, implying that pods are non-isolated by default. A NetworkPolicy can restrict this by isolating the pods. If a NeworkPolicy exists in a namespace and a particular pod is selected, the pod rejects any communication that is not allowed by the NetworkPolicy.

Steps to create a policy to restrict communication among pods

Configure the kubectl command-line tool to communicate with the Kubernetes cluster. If there is no cluster, create one by using a Kubernetes playground such as Minikube, Katacoda, or Play with Kubernetes.

Ensure the Kubernetes server's version is higher than version v1.8 by entering **kubectl version**.

Ensure the configuration of a network provider (for example, Calico, Cilium, Kube-router, or Romana) with network policy support.

Steps to create an nginx deployment and expose it through a service

- Create an **nginx** deployment to see how Kubernetes network policy works.

```
kubectl create deployment nginx --image=nginx
deployment.apps/nginx created
```

- Expose the deployment through a service called **nginx**.

```
kubectl expose deployment nginx --port=80
service/nginx exposeds
```

- The **nginx** and deployment exist in **default** namespace.

```
kubectl get svc,pod
```

```
NAME                         CLUSTER-IP    EXTERNAL-IP   PORT(S)    AGE
service/kubernetes           10.100.0.1    <none>        443/TCP    46m
service/nginx                10.100.0.16   <none>        80/TCP     33s

NAME                         READY         STATUS        RESTARTS   AGE
pod/nginx-701339712-e0qfq    1/1           Running       0          35s
```

Figure 11.37: See the nginx and Deployment Exist in "default" Namespace

Steps to test the service by accessing it from another pod

- To access the **nginx** service from another pod in the **default** namespace, start a busybox container:

```
kubectl run --generator=run-pod/v1 busybox --rm -ti --image=busybox --
/bin/sh
```

- Run the below command in shell.

```
wget --spider --timeout=1 nginx
Connecting to nginx (10.100.0.16:80)
remote file exists
```


Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

Steps to limit access to the nginx service

- To restrict the access to the **nginx** service so that only pods with the label **access: true** can query it, create a NetworkPolicy object like below:

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: access-nginx
spec:
  podSelector:
    matchLabels:
      app: nginx
  ingress:
  - from:
    - podSelector:
        matchLabels:
          access: "true"
```

Figure 11.38: Create a "NetworkPolicy" Object

Steps to assign the policy to the service

- Create a NetworkPolicy from the above **nginx-policy.yaml** file using **kubectl**:

```
kubectl apply -f https://k8s.io/examples/service/networking/nginx-policy.yaml

networkpolicy.networking.k8s.io/access-nginx created
```

Steps to test access to the service when access label is not defined

- The request times out while attempting to access the **nginx** service from a pod without the correct labels.

```
kubectl run --generator=run-pod/v1 demo--rm -ti --image=busybox -- /bin/sh
```

- Run the below command in shell.

```
wget --spider --timeout=1 nginx

Connecting to nginx (10.100.0.16:80)

wget: download timed out
```

Define access label and test again

- Create a pod with the correct labels to check whether the request is allowed:

```
kubectl run --generator=run-pod/v1 demo --rm -ti --
    labels="access=true" --image=demo-- /bin/sh
```

- Run the following command in the shell:

```
wget --spider --timeout=1 nginx

Connecting to nginx (10.100.0.16:80)

remote file exists
```


Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Secure Kubernetes Cluster With Pod Security Policies

If a Kubernetes cluster has the kubectl command-line tool installed and if support is enabled for pod security policies, follow the steps below to secure the cluster with pod security policies.

Use pod security policies only if they are enabled in the Kubernetes cluster's admission controller. Check this by running the command **kubectl get psp**. The command returns the following error if it does not support pod security policies:

```
the server doesn't have a resource type "podSecurityPolicies".
```

The following screenshot shows the difference in the command outputs between a server that does not support pod security policies and a server that does.

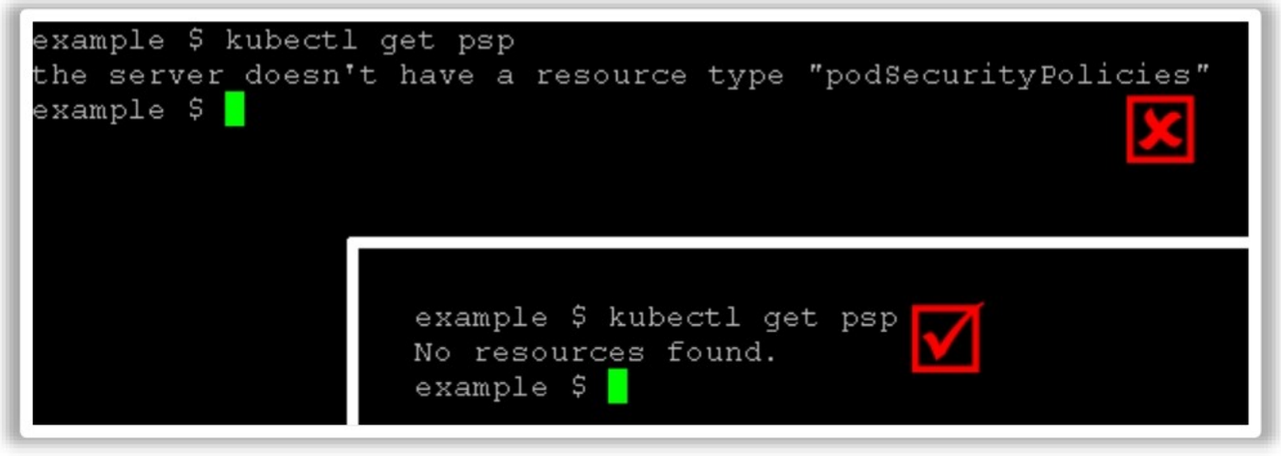


Figure 11.39: Difference in the Command Outputs Between a Server that does not Support pod Security Policies and a Server that does

Steps to enable support for pod security policies

- If using Minikube, use the below command to enable support for pod security policies and the other recommended admission controller plugins by starting Minikube.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

```
minikube start --extra-
config=apiserver.GenericServerRunOptions.AdmissionControl=NamespaceLif
ecycle,LimitRanger,ServiceAccount,PersistentVolumeLabel,DefaultStorage
Class,ResourceQuota,DefaultTolerationSeconds,PodSecurityPolicy
```

- If another platform is being used for the Kubernetes cluster, check the cluster vendor's documentation to find whether pod security policy support is enabled by default, and how to enable it.

The examples given below are tested by using a Minikube cluster running Kubernetes v1.6.4. However, they are also applicable to any Kubernetes cluster with pod security policy support.

Understand pod security policies

A PodSecurityPolicy resource represents a pod security policy in Kubernetes. This resource provides the conditions a pod must satisfy to run in the cluster. The following is an example of a pod security policy expressed in YAML.

```yaml
apiVersion: extensions/v1beta1
kind: PodSecurityPolicy
metadata:
  name: example
spec:
  privileged: false
  runAsUser:
    rule: MustRunAsNonRoot
  seLinux:
    rule: RunAsAny
  fsGroup:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  volumes:
  - 'nfs'
  hostPorts:
  - min: 100
    max: 100
```

Figure 11.40: Example of a pod

Security Policy Expressed in YAML

The following are the security rules implemented by the above pod security policy:

- Do not allow containers running in privileged mode.
- Do not allow containers that require root privileges.
- Do not allow containers that access volumes apart from NFS volumes.
- Do not allow containers that access host ports apart from port 100.

The following are common use cases for commands to add, view, and remove pod security policies in a cluster:

- **Steps to prevent pods from running with root privileges**

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

- Create the pod security policy provided below and save it as **restrict-root.yaml**.

```
apiVersion: extensions/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restrict-root
spec:
  privileged: false
  runAsUser:
    rule: MustRunAsNonRoot
  seLinux:
    rule: RunAsAny
  fsGroup:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  volumes:
  - '*'
```

Figure 11.41: pod Security Policy

- Activate the policy by the following command. This command creates a new policy from the file.

```
kubectl create -f restrict-root.yaml
```

- **Steps to prevent pods from accessing the volume type NFS**

Restrict the available storage choices for containers to reduce costs or avoid accessing information. This can be done by specifying the available volume types in the volumes key of a pod security policy, and installing this policy.

```
apiVersion: extensions/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restrict-volumes
spec:
  privileged: false
  runAsUser:
    rule: RunAsAny
  seLinux:
    rule: RunAsAny
  fsGroup:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  volumes:
  - 'nfs'
```

Figure 11.42: Specify Available Volume

Types in the Volumes Key of a pod Security Policy

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

- **Steps to prevent pods from accessing host ports**

  Cluster administrators can use pod security policies to restrict the containers' access to host resources such as host ports or network interfaces. The example below restricts a container from accessing host ports.

```
apiVersion: extensions/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restrict-ports
spec:
  privileged: false
  runAsUser:
    rule: RunAsAny
  seLinux:
    rule: RunAsAny
  fsGroup:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  volumes:
  - '*'
  hostPorts:
  - min: 0
    max: 0
```

Figure 11.43: Restrict a Container from Accessing Host Ports

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Use Kubernetes Secrets

- Do not use **Config** file for storing secrets
- Use **Kubernetes secrets** to store sensitive data

Kubernetes Secret Encryption Providers

Name	Encryption	Strength	Key Length	Other Considerations
identity	None	N/A	N/A	Resources written as-is without encryption. When set as the first provider, the resource will be decrypted as new values are written.
aescbc	AES-CBC with PKCS#7 padding	Strongest	32-byte	The recommended choice for encryption at rest, but may be slightly slower than secretbox.
secretbox	XSalsa20 and Poly1305	Strong	32-byte	A newer standard and may not be considered acceptable in environments that require high levels of review.
aesgcm	AES-GCM with random nonce	Must be rotated every 200k writes	16, 24, or 32-byte	Not recommended for use except when an automated key rotation scheme is implemented.
kms	Uses envelope encryption scheme: Data is encrypted by data encryption keys (DEKs) using AES-CBC with PKCS#7 padding, DEKs are encrypted by key encryption keys (KEKs) according to configuration in Key Management Service (KMS)	Strongest	32-bytes	The recommended choice for using a third-party tool for key management. Simplifies key rotation, with a new DEK generated for each encryption, and KEK rotation controlled by the user. The KMS provider must be configured.

Copyright © by EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited.

Use Kubernetes Secrets (Cont'd)

Command to read all secrets and update them to apply server side encryption

```
kubectl get secrets --all-namespaces -o json | kubectl replace -f -
```

Encryption Config File

```
apiVersion: apiserver.config.k8s.io/v1
kind: EncryptionConfiguration
resources:
  - resources:
    - secrets
    providers:
    - aescbc:
        keys:
        - name: key1
          secret: <BASE 64 ENCODED SECRET>
    - identity: {}
```

Copyright © by EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited.

Use Kubernetes Secrets

Kubernetes secrets enable the storing and managing of confidential data such as passwords, tokens, and keys. This is safer and more flexible than storing the sensitive data verbatim in a pod specification or a container image.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

Steps to enable and configure encryption of secret data at rest

Configure the kubectl command-line tool to communicate with the Kubernetes cluster. If there is no cluster, create one by using a Kubernetes playground such as Minikube, Katacoda, or Play with Kubernetes.

Ensure that the Kubernetes server's version is higher than or equal to version v1.13 by entering **kubectl version**.

In addition, use etcd version v3.0 or later.

- **Steps to configure and determine whether encryption at rest is enabled**

  The **`kube-apiserver`** process takes the argument **`--encryption-provider-config`,** which can be used to control the encryption of API data in etcd. The encryption at rest configuration can be understood with the following example:

```
apiVersion: apiserver.config.k8s.io/v1
kind: EncryptionConfiguration
resources:
  - resources:
    - secrets
    providers:
    - identity: {}
    - aesgcm:
        keys:
        - name: key1
          secret: c2VjcmV0IGlzIHNlY3VyZQ==
        - name: key2
          secret: dGhpcyBpcyBwYXNzd29yZA==
    - aescbc:
        keys:
        - name: key1
          secret: c2VjcmV0IGlzIHNlY3VyZQ==
        - name: key2
          secret: dGhpcyBpcyBwYXNzd29yZA==
    - secretbox:
        keys:
        - name: key1
          secret: YWJjZGVmZ2hpamtsbW5vcHFyc3R1dnd4eXoxMjM0NTY=
```

Figure 11.44: Encryption at Rest Configuration

Here,

- Every **`resources`** array item constitutes a complete configuration.
- The **`resources.resources`** field is an array of Kubernetes resource names.
- The **`providers`** array is an ordered list of the possible encryption providers (**`identity`** or **`aescbc`**).

Note: Delete the key from the etcd directly when a resource is not readable through the encryption config.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Encryption Providers

Kubernetes Secret Encryption Providers

Name	Encryption	Strength	Key Length	Other Considerations
identity	None	N/A	N/A	Resources written as-is without encryption. When set as the first provider, the resource will be decrypted as new values are written.
aescbc	AES-CBC with PKCS#7 padding	Strongest	32-byte	The recommended choice for encryption at rest but may be slightly slower than secretbox.
secretbox	XSalsa20 and Poly1305	Strong	32-byte	A newer standard and may not be considered acceptable in environments that require high levels of review.
aesgcm	AES-GCM with random nonce	Must be rotated every 200k writes	16, 24, or 32-byte	Is not recommended for use except when an automated key rotation scheme is implemented.
kms	Uses envelope encryption scheme: Data is encrypted by data encryption keys (DEKs) using AES-CBC with PKCS#7 padding, DEKs are encrypted by key encryption keys (KEKs) according to configuration in Key Management Service (KMS)	Strongest	32-bytes	The recommended choice for using a third-party tool for key management. Simplifies key rotation, with a new DEK generated for each encryption, and KEK rotation controlled by the user. Configure the KMS provider

Figure 11.45: Kubernetes Secret Encryption Providers

The above screenshot summarizes Kubernetes secret encryption providers. Here, all providers support multiple keys for encryption and decryption. Use **kms** provider for enhanced security, as **`EncrytpionConfig`** provides only moderate security for the stored encryption keys in it. The **`identity`** provider protects secrets in etcd but does not provide encryption.

- **Steps to encrypt data**
  - Create a new encryption config file.

```
apiVersion: apiserver.config.k8s.io/v1
kind: EncryptionConfiguration
resources:
  - resources:
    - secrets
    providers:
    - aescbc:
        keys:
        - name: key1
          secret: <BASE 64 ENCODED SECRET>
    - identity: {}
```

Figure 11.46: Create a New Encryption Config File

  - Steps to create a new secret include:
    - Generate a 32 byte random key and encode it to base64. For Linux and macOS, use the following command:

```
head -c 32 /dev/urandom | base64
```

    - Enter the resulting value in the secret field.
    - Set the **`--encryption-provider-config`** flag on the **`kube-apiserver`** to point to the location of the config file.
    - Restart API server.

Caution: Restrict permissions on master to ensure only the user who runs the kube-apiserver can read it, since the config file contains the keys for decrypting etcd content.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

- **Steps to verify whether the data is encrypted**
  - Use the **`etcdctl`** command line to retrieve the contents of a secret.
  - Create a new secret **`secret1`** in the **`default`** namespace:

```
kubectl create secret generic secret1 -n default --from-
literal=mykey=mydata
```

  - Read that secret out of etcd by using the etcdctl command line.

```
ETCDCTL_API=3 etcdctl get /registry/secrets/default/secret1 [...] |
hexdump -C
```

Here, **`[...]`** refers to additional arguments for connecting to the etcd server.

  - Ensure the stored secret is prefixed with **`k8s:enc:aescbc:v1:`**. This indicates that the **`aescbc`** provider encrypted the data.
  - Ensure the secret is decrypted correctly when retrieved through the API:

```
kubectl describe secret secret1 -n default
```

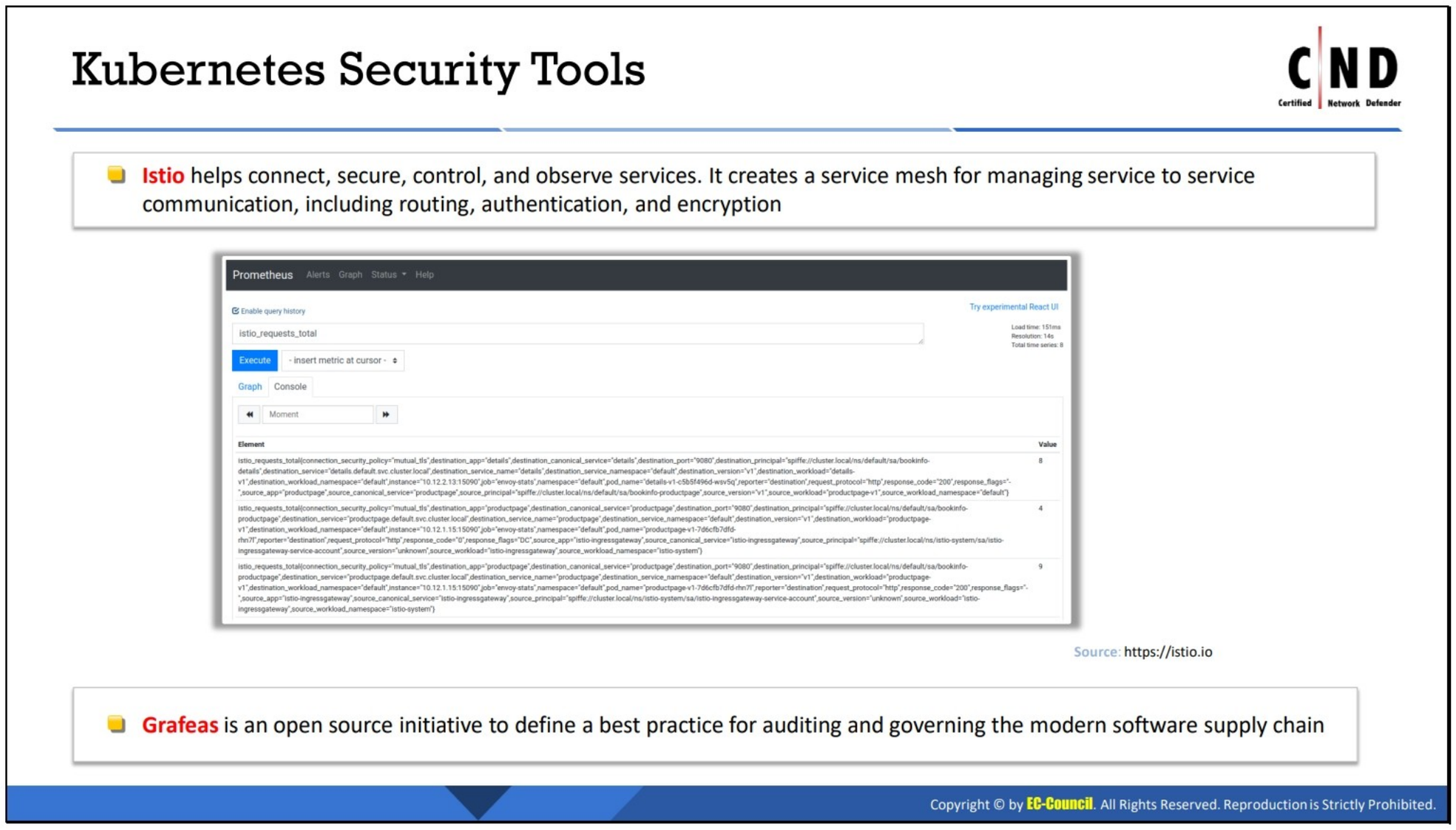
- **Steps to ensure all secrets are encrypted**

Update secrets used for encrypting content. The following command reads all secrets and updates secrets to apply server side encryption. Retry the command again if it triggers an error. Divide the secrets by namespace or use a script for the update in case of large clusters.

```
kubectl get secrets --all-namespaces -o json | kubectl replace -f -
```


Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Kubernetes Security Tools

The following are some security tools for Kubernetes:

Istio

Source: www.istio.io

Istio creates a service mesh for managing service-to-service communication, including routing, authentication, and encryption. It implements certifications and encryption for pod-to-pod communication using mutual transport layer security (TLS).

Grafeas

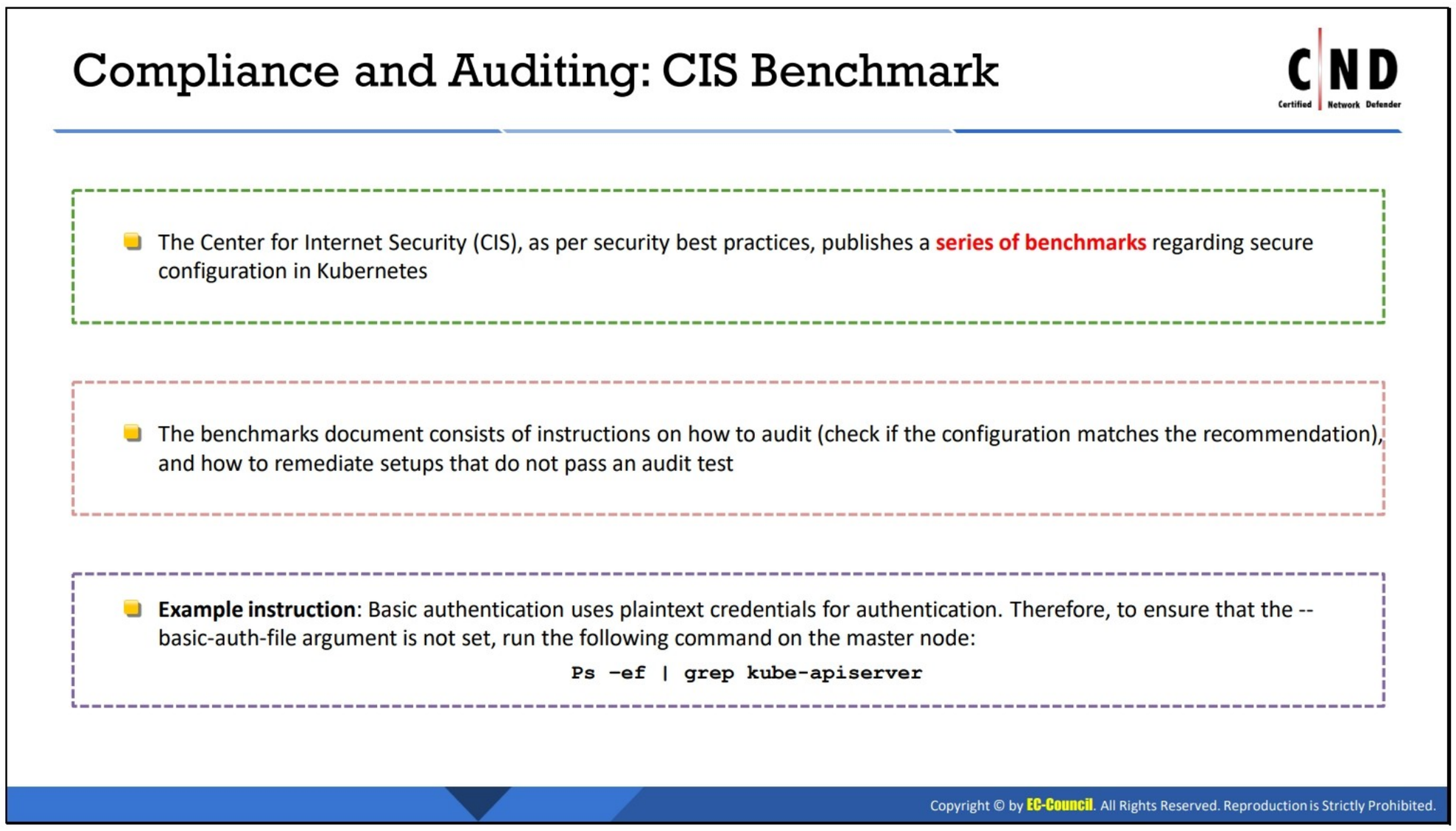
Source: www.github.com

Grafeas provides auditing and governing software supply chain, and defines an API spec for managing metadata about software resources such as container images, virtual machine images, JAR files, and scripts.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24



Compliance and Auditing: CIS Benchmark

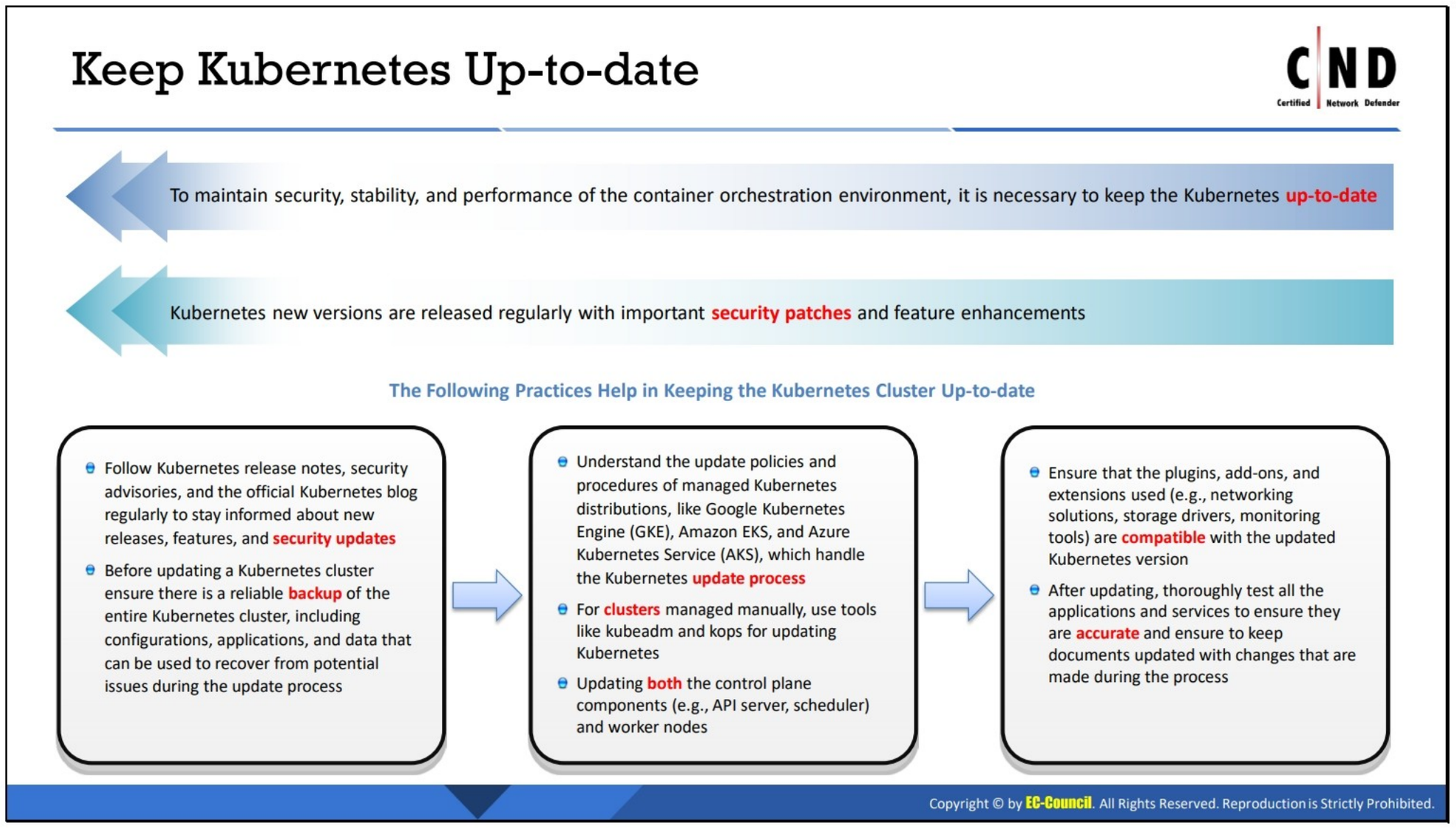
Kubernetes is a complex system that contains various components, each of which has numerous configuration parameters. To ensure that these parameters are configured securely, the Center for Internet Security (CIS) publishes a series of benchmarks regarding secure configuration in Kubernetes as per security best practices. The benchmarks document consists of instructions on how to audit (check if a configuration matches the recommendation), and how to remediate a setup that does not pass an audit test. The following is an example of such an instruction:

Basic authentication uses plaintext credentials for authentication. Therefore, to ensure that the --basic-auth-file argument is not set, run the following command on the master node

```
Ps –ef | grep kube-apiserver
```


Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Keep Kubernetes Up-to-date

To maintain the security, stability, and performance of the container orchestration environment, it is necessary to keep Kubernetes up to date. The following practices can help you achieve this:

- Regularly review the Kubernetes release notes, security advisories, and the official Kubernetes blog to stay informed about new releases, features, and security updates.
- Before updating a Kubernetes cluster, make sure you have a reliable backup of the entire cluster, including configurations, applications, and data. This backup can be used to recover from any potential issues during the update process.
- Familiarize yourself with the update policies and procedures of managed Kubernetes distributions, such as Google Kubernetes Engine (GKE), Amazon EKS (Elastic Kubernetes Services), and Azure Kubernetes Service (AKS), as they handle the Kubernetes update process.
- Utilize tools such as kubeadm and kops to securely update Kubernetes containers without data loss.
- Update both the control plane components (e.g., API server, scheduler) and worker nodes.
- Ensure that any plugins, add-ons, or extensions used (e.g., networking solutions, storage drivers, monitoring tools) are compatible with the updated Kubernetes version.
- For Ubuntu users, the instruction `sudo apt-get update && apt-get install -y kubeadm=1.26.3-00 &&\` can be used to update Kubernetes from one version to another. It is important to regularly check for available updates to keep Kubernetes up to date.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

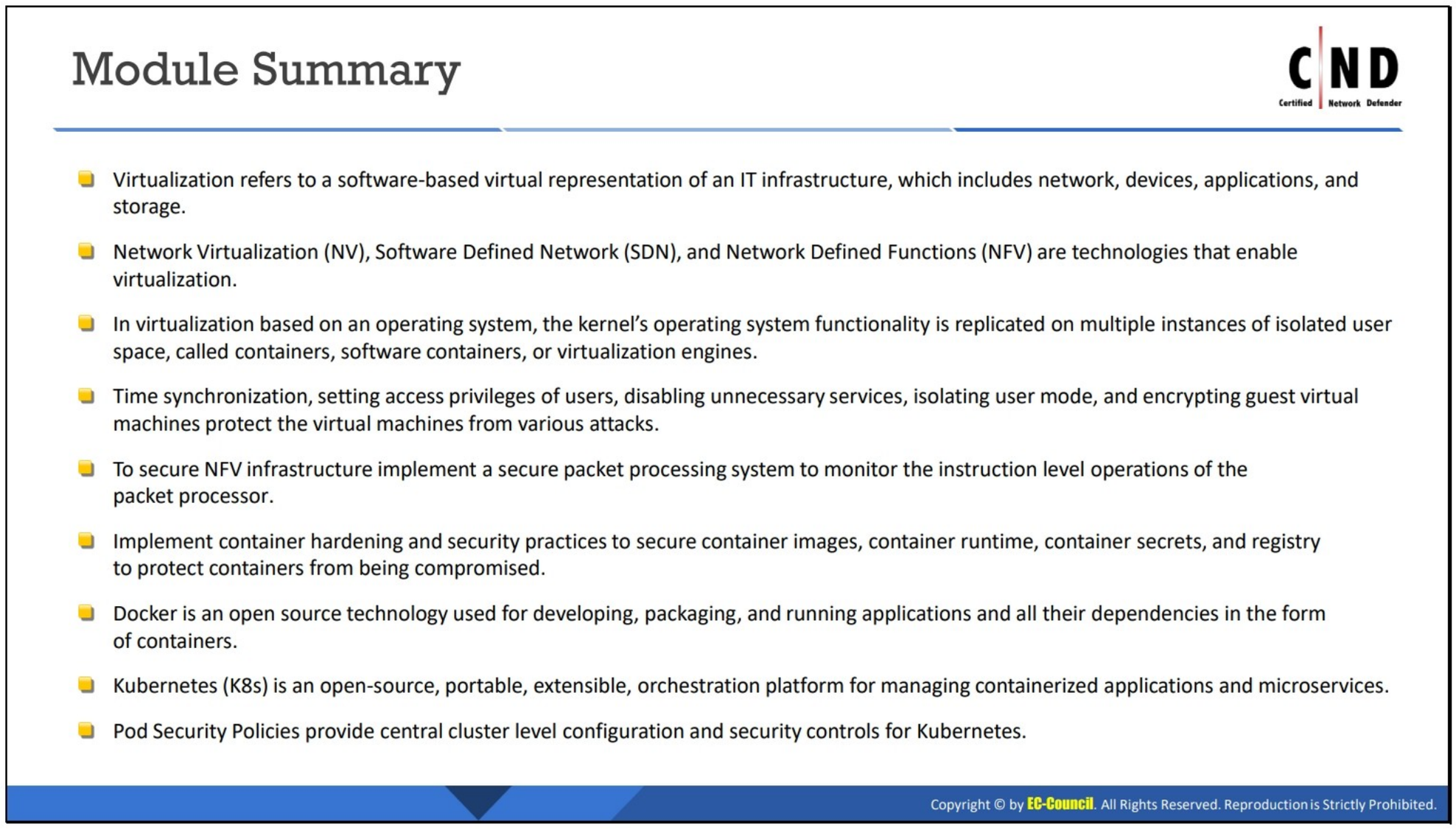
https://t.me/learningnets

Technet24

- Updating components such as controllers, network plugins, and monitoring tools within Kubernetes helps containers and clusters function safely in the environment. After updating, it is crucial to run container tests to ensure efficiency and identify any modifications made to the clusters during the process. Thoroughly test all applications and services to ensure their accuracy and remember to update any relevant documentation with the changes made during the update process.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets



Module Summary

This module focused on enterprise virtual network security. The module discussed essential concepts in virtualization, various vulnerabilities of and attacks that target network-virtualization (NV) technologies and security measures to protect them, software defined network (SDN), and network function virtualization (NFV). The module also explained the concept of OS Virtualization, and provided security guidelines, recommendations, and best practices to secure containers, Dockers, and Kubernetes.

The following are key points highlighted in the module:

- Virtualization refers to a software-based, virtual representation of an IT infrastructure, which includes network, devices, applications, and storage.
- Network virtualization (NV), software defined network (SDN), and network function virtualization (NFV) are key technologies that enable virtualization.
- In virtualization based on an operating system, the kernel's operating system functionality is replicated on multiple instances of isolated user space, called containers, software containers, or virtualization engines
- Time synchronization, setting access privileges for users, disabling unnecessary services, isolating user mode, and encrypting guest virtual machines protect the virtual machines from various attacks.
- To secure NFV infrastructure, implement a secure packet processing system to monitor the instruction level operations of the packet processor.
- Implement container hardening and security practices to secure container images, container runtime, container secrets, and registry. These protect containers from being compromised.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

Technet24

- Docker is an open source technology used for developing, packaging, and running applications and all their dependencies in the form of containers.
- Kubernetes (K8s), is an open-source, portable, extensible, orchestration platform for managing containerized applications and microservices.
- Pod security policies provide central cluster level configuration and security controls for Kubernetes.

Certified Network Defender Copyright © by **EC-Council**
All Rights Reserved. Reproduction is Strictly Prohibited.

https://t.me/learningnets

This page is intentionally left blank.

https://t.me/learningnets

Technet24