DUMPSBOSS.

303-200: LPIC-3 Exam 303: Security, version 2.0

LPI 303-200

Version Demo

Total Demo Questions: 10

Total Premium Questions: 105

Buy Premium PDF

https://dumpsboss.co

support@dumpsboss.co

support@dumpsboss.co
dumpsboss.co

Topic Break Down

Topic	No. of Questions
Topic 1, Security	5
Topic 2, Host Security	20
Topic 3, Network Security	27
Topic 4, Cryptography	26
Topic 5, Access Control	27
Total	105

QUESTION NO: 1

Which of the following sections are allowed within the Kerberos configuration file krb5.conf? (Choose THREE correct answers.)

A. [plugins]

B. [crypto]

[domain]

C. [capaths]

D. [realms]

ANSWER: A C D

Explanation:

The sections `[plugins]`, `[capaths]`, and `[realms]` are valid top-level sections in the MIT Kerberos `krb5.conf` configuration file. The `[realms]` section defines settings for individual Kerberos realms, such as KDC hosts, administrative servers, default domain mappings, and realm-specific behavior. It is therefore a central section of a client or server Kerberos deployment.

The `[capaths]` section defines authentication paths between client and service realms when cross-realm authentication requires an explicitly configured traversal path. This is useful where a direct cross-realm trust is unavailable or where the desired path must be controlled.

The `[plugins]` section configures Kerberos plugin behavior, including plugin module directories and enablement settings. MIT Kerberos documents it as a supported configuration relation for controlling dynamically loaded plugin modules. These sections use the standard bracketed-section syntax of `krb5.conf`. See the MIT Kerberos krb5.conf documentation and its reference for cross-realm authentication paths.

QUESTION NO: 2 - (SIMULATION)

Which option in an Apache HTTPD configuration file enables OCSP stapling? (Specify ONLY the option name without any values or parameters.)

ANSWER: See the explanation for the answer

Explanation:

Answer: `SSLUseStapling`

This Apache HTTP Server `mod_ssl` directive enables OCSP stapling (typically configured as `SSLUseStapling on`).

QUESTION NO: 3

How are SELinux permissions related to standard Linux permissions? (Choose TWO correct answers.)

A. SELinux permissions overnde standard Linux permissions.

B. standard Linux permissions override SELinux permissions.

C. SELinux permissions are verified before standard Linux permissions.

D. SELinux permissions are verified after standard Linux permissions.

ANSWER: B D

Explanation:

Standard Linux discretionary access control permissions are evaluated before SELinux policy permissions. The kernel first applies conventional ownership, user/group identity, and mode-bit checks to determine whether the requested access is permitted under DAC. If those standard permissions deny the request, access does not proceed to a successful SELinux authorization. In this ordering sense, standard Linux permissions take precedence over SELinux permissions.

When the ordinary Linux permission check permits the requested operation, SELinux performs its mandatory access control authorization using the security contexts of the subject and object and the applicable policy rules. SELinux is therefore verified after standard Linux permissions, and it can still deny an operation that normal Unix permissions would otherwise allow. Effective access consequently requires both the standard Linux permission mechanism and SELinux policy to permit the operation. This layered model lets administrators retain familiar ownership and mode-based controls while enforcing additional, policy-defined confinement. Red Hat's SELinux documentation describes SELinux as an additional access-control mechanism and notes that access must be allowed by both DAC and SELinux policy. See Red Hat SELinux User's and Administrator's Guide and Getting started with SELinux.

QUESTION NO: 4

Given that this device has three different keys, which of the following commands deletes only the first key?

A. cryptsetup luksDelKey /dev/sda 1 0

B. cryptsetup luksDelkey /dev/sda 1 1

C. cryptsetup luksDelKey / dev /mapper/crypt- vol 1

D. cryptsetup luksDelKey / dev /mapper/crypt- vol 0

E. cryptsetup luksKillSlot /dev/sda 0

ANSWER: E

Explanation:

`cryptsetup luksKillSlot /dev/sda 0` is the correct command because LUKS stores each enrolled passphrase or key in a numbered key slot, and slot numbering starts at zero. Therefore, the first of three configured keys is in key slot `0`. The `luksKillSlot` action is specifically designed to remove the key material from one selected LUKS key slot while leaving the remaining slots intact. The target must be the block device containing the LUKS header, such as `/dev/sda`, rather than an already-open mapped device under `/dev/mapper`. In normal use, cryptsetup requests authentication using another valid active key so that removal of the selected slot does not make the encrypted volume inaccessible. Administrators should verify the active slot layout first with `cryptsetup luksDump /dev/sda` and ensure that a working recovery key remains before removing any slot. The cryptsetup manual documents `luksKillSlot <device> <key slot>` as the command for erasing a specified LUKS key slot. See the cryptsetup manual page and the cryptsetup FAQ.

QUESTION NO: 5

How does TSIG authenticate name servers in order to perform secured zone transfers?

A. Both servers mutually verify their X509 certificates.

B. Both servers use a secret key that is shared between the servers.

C. Both servers verify appropriate DANE records for the labels of the NS records used to delegate the transferred zone.

D. Both servers use DNSSEC to mutually verify that they are authoritative for the transferred zone.

ANSWER: B

Explanation:

Both servers use a secret key that is shared between the servers. TSIG (Transaction Signatures) authenticates individual DNS messages by adding a TSIG resource record containing a keyed message authentication code (MAC). The sending name server calculates the MAC using the DNS message, TSIG metadata, and a preconfigured shared secret; the receiving server uses its copy of that same secret to validate the MAC. A valid result establishes that the message came from a party possessing the shared key and that its protected contents were not modified in transit. For zone transfers, the primary and secondary name servers are configured with the same TSIG key name, algorithm, and secret. The secondary includes TSIG when requesting an AXFR or IXFR transfer, and the primary validates it before authorizing the transfer; transferred DNS messages can likewise be signed and checked. TSIG is a symmetric-key DNS message authentication mechanism, commonly using HMAC-based algorithms, and is designed to protect DNS transactions such as zone transfers and dynamic updates. RFC 2845 defines TSIG and its shared-secret authentication model. BIND's administrator documentation also describes configuring identical keys on the communicating servers and applying them to transfer permissions. See RFC 2845: Secret Key Transaction Authentication for DNS (TSIG) and BIND 9 key statement documentation.

QUESTION NO: 6

Which command, included in BIND, generates DNSSEC keys? (Specify ONLY the command without any path or parameters.)

A. dnssec-keygen

ANSWER: A

Explanation:

The correct command is `dnssec-keygen`. BIND provides this utility specifically to generate DNSSEC key pairs for a DNS zone. It creates the public and private key files used by an authoritative name server to sign zone data and publish DNSSEC-related records. The command supports generating both key-signing keys and zone-signing keys through its command-line options, as well as selecting the signing algorithm, key size, and other key properties. In a normal DNSSEC workflow, the resulting key material is then used with BIND's DNSSEC signing tools or automated signing features to produce signed zone content. The requested response is only the executable name, so `dnssec-keygen` is the complete answer without a path or parameters. BIND's official manual documents `dnssec-keygen` as the program that generates keys for DNSSEC. See the BIND dnssec-keygen manual page and the BIND DNSSEC Guide for the tool's role in creating and managing DNSSEC signing keys.

QUESTION NO: 7 - (SIMULATION)

Which command is used to run a new shell for a user changing the SELinux context? (Specify ONLY the command without any path or parameters.)

ANSWER: See the explanation for the answer

Explanation:

Answer: `runcon`

The `runcon` command runs a command, including a new shell, under a specified SELinux security context.

QUESTION NO: 8

Which option of `sshd_config` prevents direct SSH login as root while still allowing an administrator to obtain root privileges after logging in as an ordinary user?

A. `PermitRootLogin no`

B. `PasswordAuthentication no`

C. `AllowUsers root`

D. `UsePAM no`

ANSWER: A

Explanation:

`PermitRootLogin no` disables direct SSH authentication as the root account. An administrator can still log in using an ordinary account and, where authorized, use `sudo` or `su` to obtain elevated privileges. This improves accountability by associating remote access with an individual user account rather than a shared root login.

`PasswordAuthentication no` disables password authentication for all relevant users but does not specifically prohibit root public-key authentication. `AllowUsers` restricts login to named users, while `UsePAM` controls PAM integration. See the sshd_config(5) manual page.

QUESTION NO: 9

Which of the following statements are valid wireshark capture filters? {Choose TWO correct answers.)

A. port range 10000:tcp-15000:tcp

B. port-range tcp 10000-15000

C. tcp portrange 10000-15000

D. portrange 10000/tcp-15000/tcp

E. portrange 10000-15000 and tcp

ANSWER: C E

Explanation:

`tcp portrange 10000-15000` is a valid libpcap capture-filter expression. Capture filters use the libpcap/BPF filter syntax, where `portrange` accepts a numeric range written as two port numbers separated by a hyphen. Prefixing it with `tcp` restricts the port-range test to TCP traffic, so the expression selects TCP packets whose source or destination port is within the inclusive range 10000 through 15000.

`portrange 10000-15000 and tcp` is also valid. The unqualified `portrange` primitive matches either the source or destination port in the stated range, and the Boolean `and` operator combines that condition with the TCP protocol test. Consequently, it expresses the same intended traffic selection as the other valid filter, although the protocol qualifier is written in a different position. Wireshark capture filters are compiled using libpcap syntax rather than Wireshark's display-filter language. The Wireshark User's Guide documents capture filtering and links it to the libpcap filter grammar, while the pcap-filter manual specifies both the `portrange` primitive and protocol-qualified forms. See Wireshark User's Guide: Building capture filters and pcap-filter(7).

QUESTION NO: 10

Which of the following statements are true regarding Kerberos keytab files? (Choose TWO correct answers.)

A. A keytab can permit a service to authenticate without an interactive password prompt.

B. A keytab file must be protected because it contains secret key material.

C. A keytab contains only public keys and may safely be published in DNS.

D. A keytab replaces the Kerberos KDC database.

E. Every Kerberos client must use the same keytab file as the KDC.

ANSWER: A B

Explanation:

A keytab file stores one or more Kerberos principals together with cryptographic keys derived for those principals. A service can use its keytab to authenticate to the Kerberos infrastructure without requiring an interactive password prompt. For example, a host running an HTTP service can use a keytab entry for an `HTTP/hostname` service principal.

Keytab files must be protected because their keys can be used to obtain Kerberos credentials for the contained principals. They are normally readable only by the account that runs the associated service or by root. The `klist -k` command can display the principal entries held in a keytab, while `ktadd` in the Kerberos administration interface is commonly used to create or update service keytab entries. See the MIT Kerberos keytab documentation and the MIT Kerberos klist documentation.