

UNIT 4

Service Recommendation

(16 Hours) — Network Administration (CACS406)

⚠ Please Note

These notes were prepared with the help of AI from the TU BCA syllabus and general knowledge. They may contain errors, omissions or outdated details. Please verify important points against your textbook, class notes and the latest TU syllabus before relying on them in an exam.

4.1 Server Upgrade

A **server upgrade** replaces or improves the hardware, operating system or software of a server so that it keeps meeting the needs of its users. Upgrades are risky because users depend on the service, so a good administrator plans them carefully instead of doing them in a hurry.

4.1.1 Why Upgrade?

- **Performance:** the server is slow because of growth in users, data or load.
- **Security:** the OS or application no longer receives security patches (end of life).
- **Features and compatibility:** new clients or applications need a newer version.
- **Reliability:** old hardware fails more often; spare parts are hard to find.
- **Cost:** a newer server may use less power and space than several old ones.

4.1.2 Upgrade Process

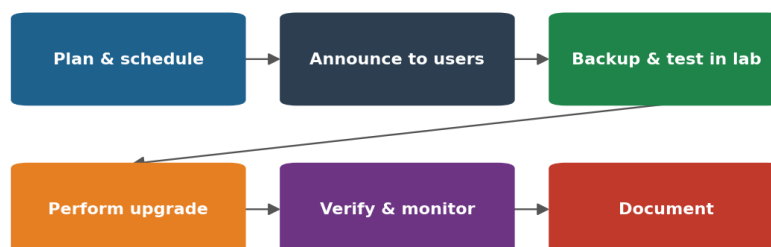


Figure 4.1 — Typical server upgrade process

- **Plan:** list what will change, who is affected, the time window and the success criteria.
- **Announce:** tell users early; choose a low-usage time (night or holiday).

- **Backup and test:** take a full backup; rehearse the upgrade on a test machine or a clone.
- **Perform:** follow a written step-by-step checklist.
- **Verify:** run test cases, check logs, ask a few users to confirm.
- **Rollback plan:** always decide in advance how to return to the old version if the upgrade fails.

4.1.3 Upgrade Strategies

Strategy	Description	Good for
In-place upgrade	Upgrade the same server directly	Small services; simple; higher risk
Side-by-side (migration)	Build a new server, move data/config, switch users over, retire old one	Critical services; easy rollback
Rolling upgrade	Upgrade one node of a cluster at a time	Clustered/load-balanced services with no downtime
Virtualize / P2V	Convert physical server to a virtual machine, then upgrade	Consolidation; snapshots for safe rollback

Note: Fix one thing at a time: do not upgrade hardware, OS and application all at the same moment; if something breaks you will not know the cause.

4.2 Centralizing a Service

Centralizing means running a service from one (or a few) well-managed servers instead of many separate copies spread over departments or desktops. Examples: one mail system for the whole company instead of each office running its own; one print server, one file server, one authentication (directory) service.

Advantages of Centralization	Disadvantages / Risks
Easier management, patching and backup	Single point of failure (needs redundancy)
Consistent policies and better security	Network dependency: all users need good links to the central site
Lower cost: fewer servers, licences, admins	Slower response for remote offices (latency)
Better monitoring and capacity planning	Heavier load; needs careful capacity design
Easier to give users a uniform experience	Local departments may lose flexibility/control

How to centralize safely: first understand the current situation; design for high availability (clustering, redundant power and network); migrate group by group; keep the old service

running in parallel until the new one is proven; update documentation and inform users.

A middle path is **hub-and-spoke**: one central master service with local replicas or caches at remote sites (for example, a DNS or DHCP relay, or a local file cache).

4.3 Service Monitoring

Monitoring means continuously watching services to find problems *before* the users complain, and to collect data for planning. A basic rule: **if you are not monitoring it, you do not know that it is working.**

4.3.1 What to Monitor

- **Availability:** is the service up? (ping, port check, test transaction such as sending a test email)
- **Performance:** response time, throughput, queue length, CPU, memory, disk I/O, network usage.
- **Capacity:** disk space, database size, licences, number of users/connections.
- **Errors and security:** log messages, failed logins, certificate expiry, backup success/failure.

4.3.2 Types and Tools

Type	Meaning	Examples
Active (synthetic)	The monitor tests the service by acting like a user	Nagios, Zabbix, PRTG, Uptime Kuma
Passive	The device/service sends events to the monitor	SNMP traps, syslog
Log-based	Analyse log files for patterns/errors	rsyslog, Graylog, ELK stack
Metrics/graphing	Store numbers over time and draw graphs	Grafana, Prometheus, MRTG, Cacti

4.3.3 Alerts and Good Practice

- Set **thresholds** (for example, warning at 80% disk, critical at 90%).
- Send alerts by email/SMS/chat to the right person; avoid **alert fatigue** (too many false alarms make people ignore alerts).
- Define **escalation**: if nobody responds in a set time, notify the next person.

- Keep historical data for **trend analysis** and capacity planning.
- Monitor the monitoring system itself; keep it independent of the systems it watches.

Example 4.1 — Choosing thresholds

A file server has a 1 TB data disk that grows about 20 GB per month. When should the administrator plan an expansion?

Solution

Warning at 80% = 800 GB used; critical at 90% = 900 GB used. From 800 GB to 1000 GB is 200 GB, i.e. about 10 months at 20 GB/month; from 900 GB it is about 5 months. Ordering and installing storage takes weeks, so the warning at 80% gives enough time. Graphs over time also let the administrator predict this date automatically.

4.4 Namespaces

A **namespace** is a system of names in which every name is unique and maps to something (a host, user, file, printer, etc.). Good namespaces let people and programs find resources by *name* instead of by address or location.

4.4.1 Common Namespaces in a Network

Namespace	Names → Objects	Service / Protocol
Host names	www.example.com → IP address	DNS (or hosts file, WINS in old networks)
User accounts	username → UID/SID, password, groups	LDAP, Active Directory, NIS
Email addresses	user@domain → mailbox	SMTP + MX records in DNS
File shares	\\server\share or /mnt/data → files	SMB/CIFS, NFS, DFS
Printers	printer name → queue/device	CUPS, Windows print server
Groups / roles	group name → set of users	Directory service

4.4.2 Principles of a Good Namespace

- **Unique** — no two objects share the same name in the same scope.
- **Consistent and meaningful** — follow a naming standard, e.g. *ktm-web-01* (site-role-number) rather than random or joke names.
- **Scalable** — the scheme should still work after the company grows tenfold.
- **Stable** — names should not need to change when a person moves or hardware is replaced (use aliases/CNAMEs for services, e.g. *mail.example.com*).
- **Single authoritative source** — one central directory, not many copies that disagree.
- **Documented and protected** — control who may create or change names.

DNS is a hierarchical distributed namespace: root (.) → top-level domain (.com, .np) → domain (example) → host (www). **Directory services** such as LDAP/Active Directory organise objects in a tree (domain → organisational units → users, groups, computers) and provide **Single Sign-On**: one login for many services.

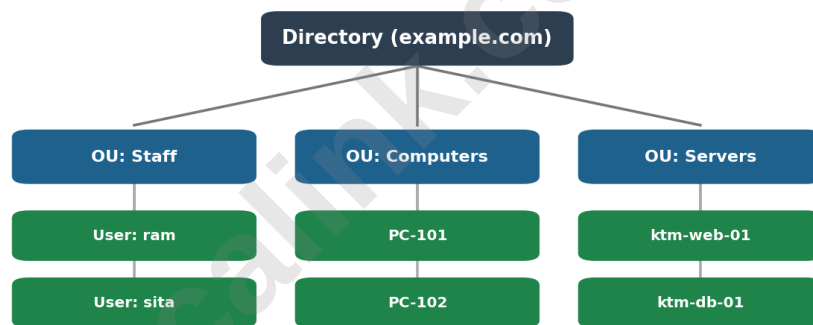


Figure 4.2 — Directory namespace organised into organisational units

4.5 Email Service

Email is one of the most important business services; users notice immediately when it fails. It is also the main path for spam, phishing and malware, so security matters.

4.5.1 Components and Protocols



Figure 4.3 — Path of an email message

Term / Protocol	Role	Port (default)
MUA	Mail User Agent — the client (Outlook, Thunderbird, webmail)	—
MTA	Mail Transfer Agent — moves mail between servers (Postfix, Exim, Sendmail, Exchange)	—
MDA	Mail Delivery Agent — puts mail in the user's mailbox (Dovecot, procmail)	—
SMTP	Sends/relays mail between servers and from clients	25 (server), 587 (submission), 465 (SMTPS)
POP3	Downloads mail to one device (usually deletes from server)	110 / 995 (SSL)
IMAP	Keeps mail on the server; sync across many devices	143 / 993 (SSL)
MX record	DNS record that names the mail server for a domain	—

4.5.2 Design and Administration Issues

- **Spam and malware filtering:** SpamAssassin, Rspamd, antivirus (ClamAV), greylisting, RBLs.
- **Email authentication:** SPF (which servers may send for a domain), DKIM (digital signature), DMARC (policy on failures).
- **Encryption:** use TLS for SMTP/IMAP/POP3; users may use S/MIME or PGP for end-to-end security.
- **Storage and quotas:** mailbox size limits, archiving, retention rules.
- **Reliability:** backup mail store, secondary MX, monitoring of mail queue.
- **Hosted or self-hosted?** Cloud email (Google Workspace, Microsoft 365) reduces admin work; self-hosting gives control but needs skill and time.

4.6 Print Services

Printing looks simple but consumes much administrative time: drivers, paper jams, toner, queues and cost control. A **print server** centralises these tasks.

4.6.1 Architecture

- **Print server:** receives jobs from clients, queues (spools) them, and sends them to the printer. Examples: CUPS (Linux/macOS), Windows Print Server role.
- **Printer:** connected by network (IP address) or USB; network printers are preferred.
- **Protocols:** IPP (Internet Printing Protocol, port 631), LPD/LPR, raw/JetDirect (port 9100), SMB printing.
- **Drivers:** the print server can push the correct driver to clients automatically.

4.6.2 Good Practices

- Standardise on few printer models to reduce spares, toner types and driver problems.
- Use clear naming (e.g. *ktm-fl2-laser01*) and put the name label on the printer.
- Use **print accounting/quotas** and secure release (card or PIN) to reduce waste and protect confidential printouts.
- Prefer multifunction devices (print + scan + copy) shared by a department; monitor toner and page counts (SNMP).
- Give users simple instructions to find and add printers; place printers where they are convenient but secure.

4.7 Data Storage

Data is the most valuable asset of most organisations. The administrator must provide storage that is **fast enough, large enough, reliable and secure**.

4.7.1 Storage Architectures

Type	Description	Typical use
DAS — Direct Attached Storage	Disks inside or directly cabled to one server	Small setups; OS/boot disks
NAS — Network Attached Storage	Storage device sharing files over the network (NFS, SMB)	File shares, home directories, backups

Type	Description	Typical use
SAN — Storage Area Network	Dedicated high-speed network giving servers block-level disks (Fibre Channel, iSCSI)	Databases, virtualization clusters
Cloud / Object storage	Storage as a service accessed via API (S3-like)	Archive, backup, web content

4.7.2 RAID

RAID (Redundant Array of Independent Disks) combines several disks for speed and/or fault tolerance.

Level	Method	Min. disks	Fault tolerance	Usable capacity
RAID 0	Striping	2	None	100%
RAID 1	Mirroring	2	1 disk	50%
RAID 5	Striping + single parity	3	1 disk	$(n - 1)/n$
RAID 6	Striping + double parity	4	2 disks	$(n - 2)/n$
RAID 10	Mirror + stripe	4	1 per mirror pair	50%

Note: RAID protects against disk failure only. It is NOT a backup: it does not protect against deletion, ransomware, fire or controller errors.

Example 4.2 — RAID capacity

A server has 6 disks of 2 TB each. Find the usable capacity in RAID 5 and RAID 10.

Solution

RAID 5: $(n - 1) \times \text{disk size} = (6 - 1) \times 2 \text{ TB} = \mathbf{10 \text{ TB}}$.

RAID 10: half of the total = $(6 \times 2 \text{ TB}) \div 2 = \mathbf{6 \text{ TB}}$.

RAID 5 gives more space; RAID 10 gives better write speed and faster rebuild.

4.7.3 Other Storage Issues

- **Quotas** limit space per user/group; **tiering** puts hot data on SSD and cold data on cheaper disks.
- **Access control**: permissions/ACLs, encryption at rest, audit logs.
- **Monitoring and capacity planning**: track growth, SMART disk health, latency.
- **Data lifecycle**: create → use → archive → delete, following retention policy and law.

4.8 Backup and Restore

A **backup** is a copy of data kept separately so it can be recovered after loss (deletion, disk failure, ransomware, disaster). **Restore** is the process of getting the data back. A backup that has never been tested for restore cannot be trusted.

4.8.1 Backup Types

Type	What is copied	Backup time	Restore needs
Full	Everything selected	Longest	Only the latest full backup (fastest restore)
Incremental	Only changes since the last backup of any kind	Shortest	Last full + all incrementals in order (slowest)
Differential	Changes since the last full backup	Medium (grows daily)	Last full + latest differential
Synthetic full	Full built on the server from earlier backups	Low load on clients	Like full

Example 4.3 — Restore chain

A full backup is taken every Sunday. Data is lost on Thursday morning. Which backups are needed if (a) incremental and (b) differential backups run Monday to Wednesday?

Solution

- (a) Incremental: Sunday full + Monday + Tuesday + Wednesday incrementals (4 sets).
- (b) Differential: Sunday full + Wednesday differential only (2 sets).

4.8.2 The 3-2-1 Rule and Other Concepts

- **3-2-1 rule:** keep **3** copies of data, on **2** different media, with **1** copy off-site (many add: 1 offline/immutable copy against ransomware).
- **RPO** (Recovery Point Objective): how much data loss is acceptable (decides backup frequency).
- **RTO** (Recovery Time Objective): how quickly service must be restored (decides method and hardware).

- **Retention:** how long backups are kept (e.g. daily 2 weeks, weekly 3 months, monthly 1 year).
- **Media:** disk, tape (LTO), NAS, cloud; **tools:** rsync, Bacula, Veeam, Amanda, Windows Server Backup.
- **Backup window:** the time available for backup without hurting users; use snapshots to shorten it.
- **Security:** encrypt backups, restrict access, store tapes safely, and protect backup servers from ransomware.

4.8.3 Restore Testing

Schedule **regular restore tests** (for example every quarter), restore random files and one full system, measure the time taken, verify the data is usable, and record results. Also keep instructions to restore when the backup system itself is lost.

4.9 Software Repository

A **software repository** is a central, trusted location that stores installation packages, updates and licences so all computers install software from one place instead of from random internet sources.

- **Package repositories:** Linux (apt/deb, yum/dnf/rpm) with local mirrors (apt-cacher-ng, Nexus, Artifactory).
- **Windows:** WSUS or Microsoft Endpoint Configuration Manager for updates; software distribution via Group Policy or Intune; winget/Chocolatey with internal feed.
- **Benefits:** saves internet bandwidth, ensures the same approved versions everywhere, allows testing patches before roll-out, blocks unauthorised/malicious software, and helps licence tracking.
- **Practices:** verify package signatures/checksums, keep a test group before production, maintain an inventory of installed software, plan for rollback of bad updates.

4.10 Web Services

A **web service** provides content and applications through HTTP/HTTPS. It includes the web server, the application/database back-end, DNS, certificates and security.

4.10.1 Architecture

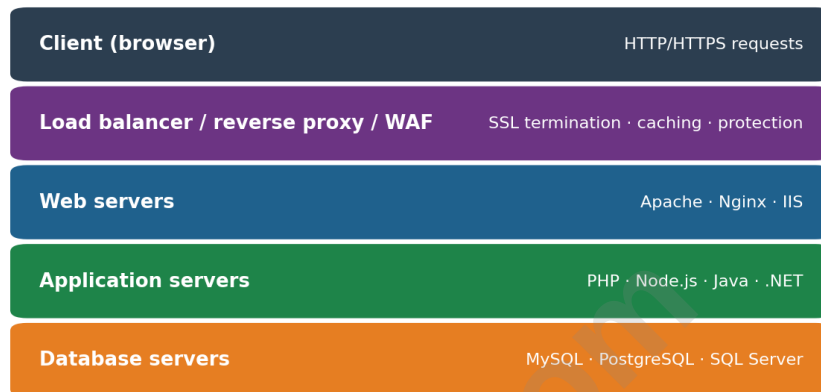


Figure 4.4 — Layers of a typical web service

4.10.2 Administration Topics

Topic	Notes
Web server software	Apache HTTP Server, Nginx, Microsoft IIS; virtual hosts allow many sites on one server
Protocols and ports	HTTP 80, HTTPS 443; use HTTPS everywhere with valid TLS certificates and renew them before expiry (Let's Encrypt automates this)
Scalability and availability	Load balancing, caching/CDN, clustering, database replication; test with load tools
Security	Patch OS and software, disable unused modules, firewall/WAF, least-privilege accounts, secure headers, protect against SQL injection and XSS
Monitoring and logs	Access/error logs, response-time checks, certificate expiry, uptime monitors
Backup and staging	Back up content and databases; keep separate development, testing and production environments

4.11 Desktop Computers

Desktop computers (and laptops) are the most numerous devices, so a small saving or a small problem is multiplied by the number of machines. The administrator aims for **standardised, easily managed and secure** desktops.

- **Standardisation:** few hardware models and one or two standard OS images make support and spare parts simple.
- **Automated installation:** imaging (Clonezilla, Windows Deployment Services, MDT), PXE boot, kickstart/preseed; use a golden image plus post-install scripts.
- **Central management:** join domain; use Group Policy or MDM for settings, updates, software and security baselines; remote support tools.
- **Users and data:** keep user data on servers (roaming profiles, redirected folders, cloud sync) so a desktop can be replaced quickly; users get standard rights, not administrator rights.
- **Security:** antivirus/EDR, disk encryption (BitLocker), host firewall, automatic patching, USB/port control.
- **Life cycle:** plan replacement every 3–5 years, keep an asset inventory, dispose of old equipment and wipe data securely.
- **Support:** help desk with ticket system, knowledge base and clear SLAs.

Key Takeaways — Unit IV

- ✓ Server upgrades need planning, announcement, backup, testing, a rollback plan and verification; change one thing at a time.
- ✓ Centralizing a service simplifies management and security but adds single-point-of-failure and latency risks, so design for redundancy.
- ✓ Monitor availability, performance, capacity and errors; use thresholds, escalation and trend graphs (Nagios, Zabbix, Grafana).
- ✓ Namespaces must be unique, consistent, scalable and centrally managed (DNS, LDAP/Active Directory).
- ✓ Email uses MUA/MTA/MDA with SMTP, POP3/IMAP; protect with SPF, DKIM, DMARC, TLS and spam filters.
- ✓ Print servers (CUPS, IPP) centralise queues, drivers and accounting; standardise printers.
- ✓ Storage: DAS, NAS, SAN, cloud; RAID gives fault tolerance but is not a backup.
- ✓ Backups: full, incremental, differential; 3-2-1 rule; define RPO/RTO; test restores regularly.
- ✓ Software repositories and update servers deliver approved, signed software; web services need HTTPS, scaling, security and monitoring.
- ✓ Desktops should be standardised, image-based, centrally managed and secured.

— *End of Unit 4* —

bcalink.com