WASD in high availability environment

Jean-Pierre PETIT

jpp@esme.fr

WASD

Reference site

http://wasd.vsm.com.au/wasd/

WASD

- WASD means
  - Wide Area Surveillance Division
- It is the name of an Australian Defense division that accepted in 1996 that this product, developed for internal use, was made available to the VMS community (GNU GPL model).

The author

- Since 1996, Mark Daniel has invested a lot of his free time to further develop the product.
- He deserves a lot of thanks for:
  - the quality of his work (code quality is exemplary)
  - his reactivity
  - the way he responds to enhancement requests

WASD / environment

- WASD is available on:
  - VAX
  - Alpha
  - Itanium
- It runs on any VMS version from 6.0
- It is compatible with different IP stacks:
  - Compaq/HP: TCPIP et UCX any version
  - Process Software: MultiNet et TCPware

WASD performance features

- specifically designed for VMS
- doesn’t use *threads*
- relies upon the AST mechanism
- has its own performance enhancers:
  - file cache
  - DNS cache
  - authentication cache (shared between instances)

WASD

How does ESME-Sudria use it ?

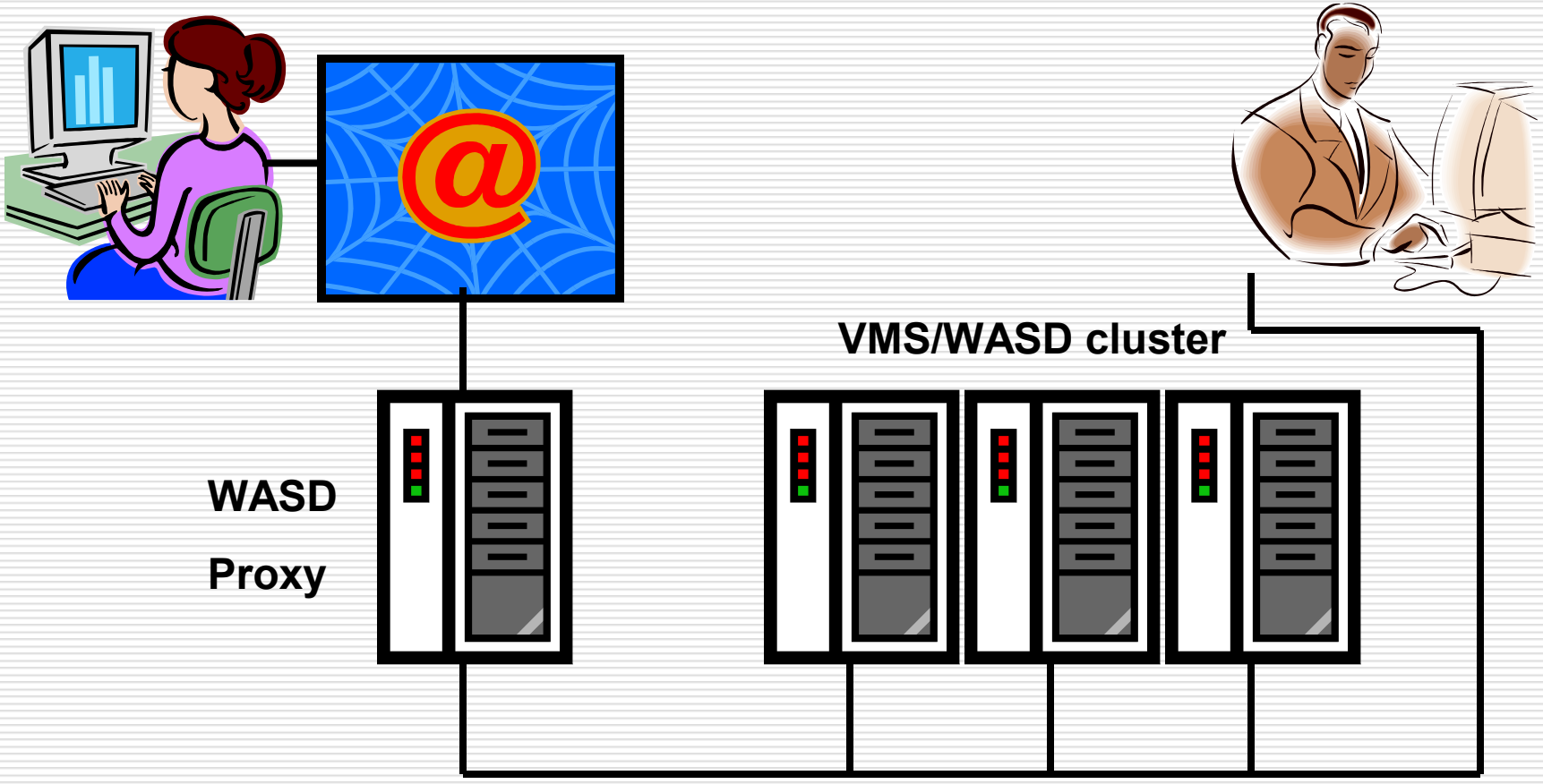
HTTP services at ESME-Sudria

- The Intranet has been for years the official source of information for both the students and the staff.
- It must be securely available through the Internet to allow students access from their home.

HTTP services at ESME-Sudria

- Different products have been tested/used in the past:
  - Purveyor
  - Netscape FastTrack
  - Apache
- Since the beginning of 2001, WASD has been the only web server running on VMS production platforms.

HTTP services at ESME-Sudria



Intranet servers

Intranet
WASD

Proxy
WASD

Intranet servers

- 3 servers in the same cluster
  - 2 Alpha servers / 1 Itanium server
- All 3 servers share the same disks and hence have the same:
  - configuration files
  - static pages
  - platform-neutral scripts (DCL, PHP, Python,…)
- One set of compiled scripts (.EXE) per architecture.

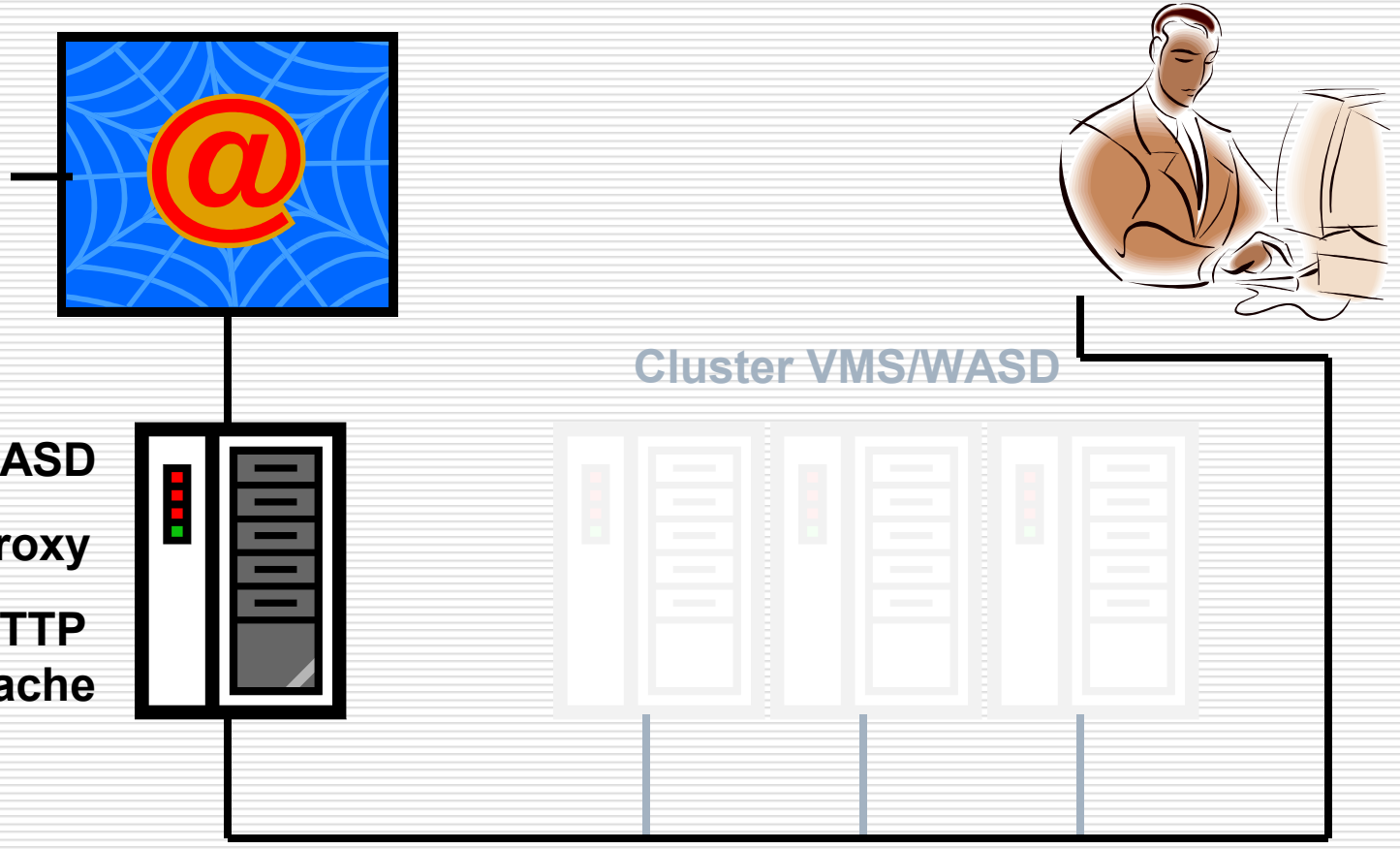
Intranet servers

- *DNS round robin* is used to dispatch the clients over the available servers
- WASD can be managed by triggering cluster-wide actions using:
  - /CLUSTER command line option
  - CLUSTER checkbox in the web interface

Proxy server

- Assumes multiples functions described hereafter
- Two instances running on the same host in order to:
  - immediate failover whether an instance crashes
  - totally transparent restart in case of:
    - configuration changes
    - new version installation

Regular Internet proxy access



Regular Internet proxy access

- In addition to allowing access to the Internet, the proxy server allows for:
  - Global caching of web content
  - Filtering out unwanted sites
  - Filtering out ads
  - CONNECT method restriction to HTTPS standard port

Web content cache

- contains ~200000 files (daily renewal rate: 15%).
- may undergo a very important I/O rate
- needs a dedicated disk
- a RAID 0 array is used for good performances
- array is defragmented each night

Web content cache

- No specific index is used for implementing the cache.
- RMS attributes are used to store creation and access dates.
- File names are generated by *hashing* the URL.
- Finding a document in the cache is only a matter of using the XQP to locate a file.
- Large XQP cache values are needed for good performances.

XQP parameters

ACP_DIRCACHE parameter information:

- Old value was 10240, New value is 10240
- Hit percentage: 94%
- Attempt rate: 392 attempts per 10 sec.

ACP_DINDXCACHE parameter information:

- Old value was 2048, New value is 2048
- Hit percentage: 96%
- Attempt rate: 137 attempts per 10 sec.

XQP parameters

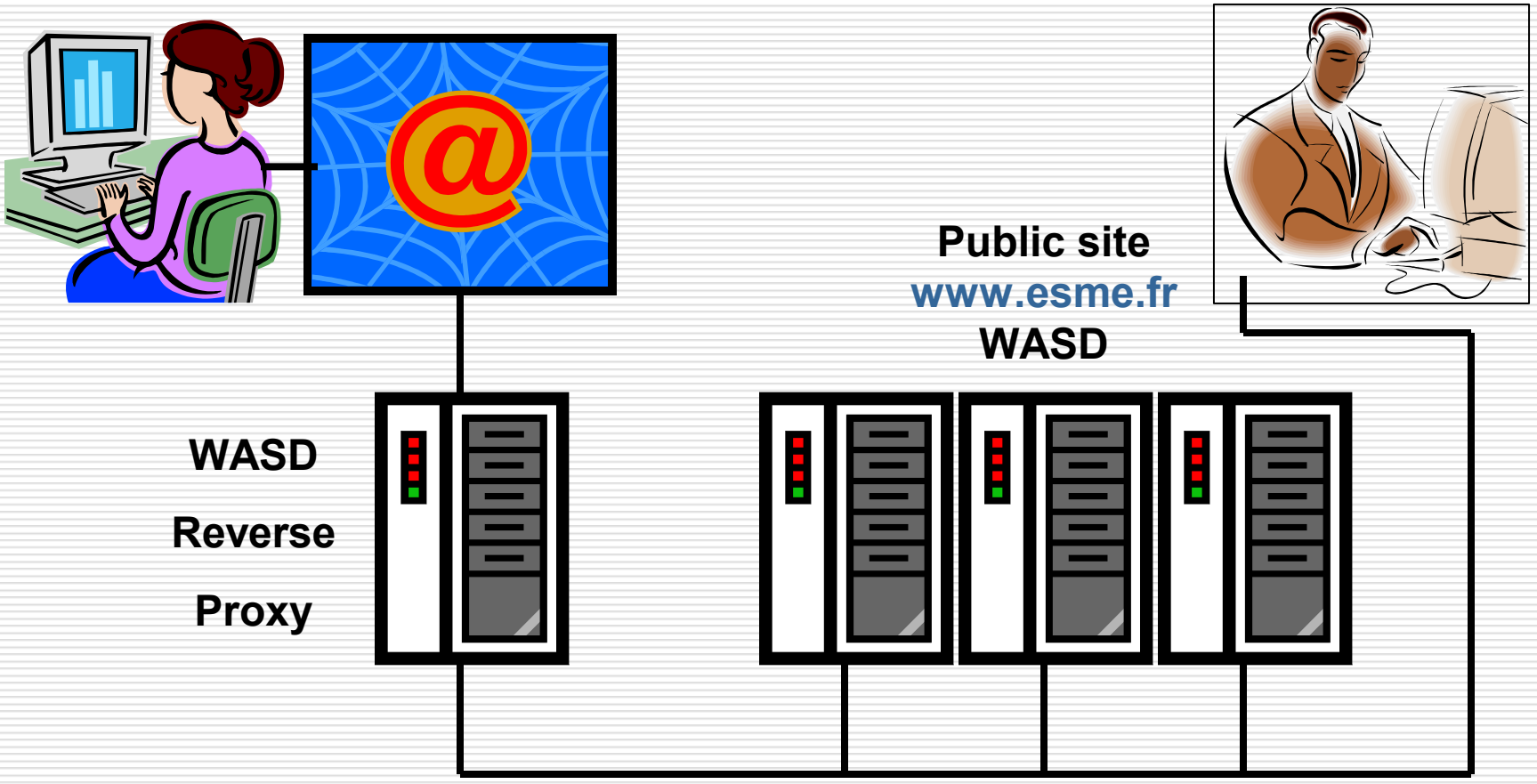
ACP_HDRCACHE parameter information:

- Old value was 16384, New value is 16384
- Hit percentage: 71%
- Attempt rate: 126 attempts per 10 sec.

ACP_MAPCACHE parameter information:

- Old value was 1024, New value is 1024
- Hit percentage: 97%
- Attempt rate: 13 attempts per 10 sec.

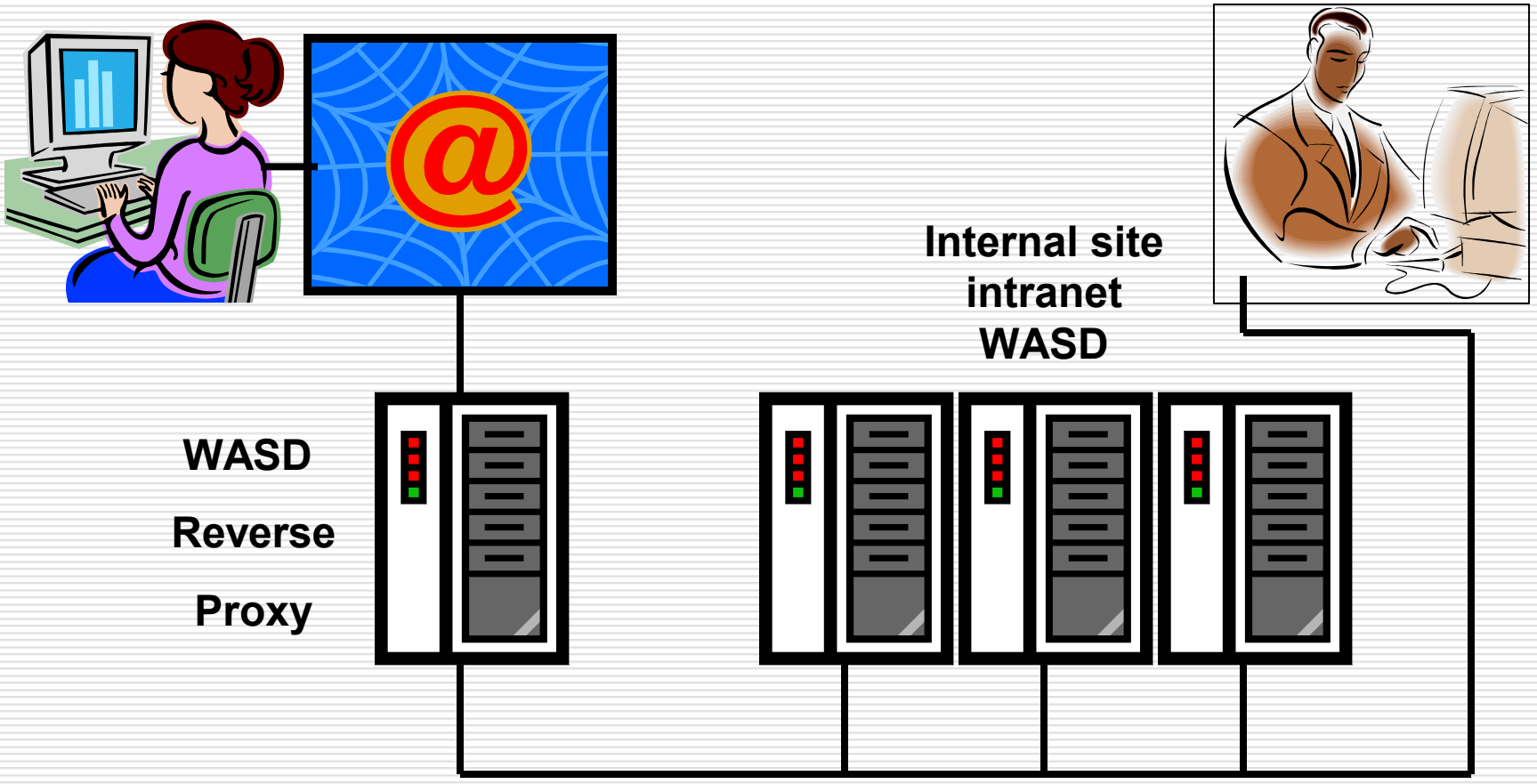
HTTP Reverse proxy



Reverse proxy HTTP

- 2 cluster members share the same configuration and serves www.esme.fr
- The reverse proxy provides for automatic failover from one server to the other.
- The internals servers are protected form the Internet but directly accessible by internal users.
- *DNS round robin* is used for load balancing between servers.

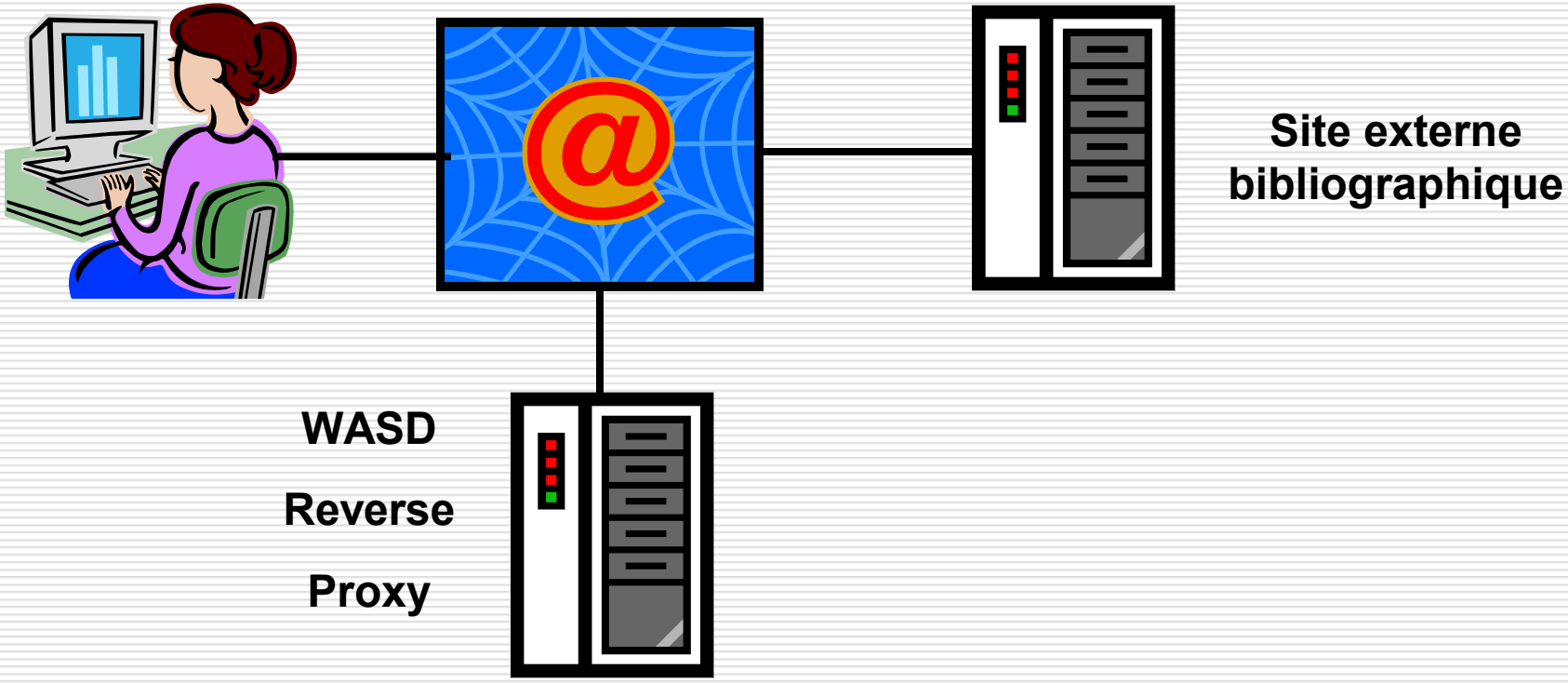
Reverse proxy HTTPS



HTTPS Reverse proxy

- The reverse proxy server controls access to the internal servers and encrypts all traffic over the Internet using SSL .
- Users authentication is done using a locally written module IMAP that check user credentials by querying an IMAP server in the cluster (the proxy server is not part of the cluster and can’t have access to its UAF).
- The reverse proxy servers provides for automatic failover among the 3 internal servers.

DNS wildcard proxy



DNS wildcard proxy

- The school has subscribed to an on-line library which only accept requests originating from the proxy server's IP address.
- The *DNS wildcard proxy* feature is based on the following type of definition in the DNS:
  - *.proxy.esme.fr IP_address_of_proxy

DNS wildcard proxy

- The site www.site.fr is referenced through an URL like www.site.fr.proxy.esme.fr whoch, of course, requires authentication.
- Hopefully, the requested page may be found in the content cache and will not require a request to the origin server.
- The page content is then returned to the client, using the GZIP compression provided, the client supports it. This means the client can get a compressed page from a server that doesn’t support compression.

Transparent proxy to IIS

- Some IIS servers have been «hidden» behind the Intranet servers.
- Authentication is performed by WASD using UAF data.
- IIS applications get the requesting username from HTTP headers (HTTP_REMOTE_USER).
- The reverse proxy filters out classical IIS attacks.

WASD

Most appreciated features

CGIplus scripts

- WASD offers a great extension to the basic concept of CGI scripting.
- CGIplus scripts are persistent CGI scripts that stay waiting for the next request.
- There is no latency induced by:
  - image activation
  - database initialization (i.e. SQL attach)

CGIplus scripts

- It is very easy to write scripts that may be executed in CGI or CGIplus mode.
- Activation mode is simply selected by a mapping rule (EXEC or EXEC+).
- Interpreting environments (PHP, Python, Java…) can also use CGIplus mode to minimize system overhead and response time.

CGIplus scripts

- void main (int argc, char *argv[])
- {
- CgiLibInit (argc, argv, script);
- }

- void script (void)
- {
- ... script body
- ... called for each request
- }

Impersonation

- A CGI or CGIplus script may execute under different accounts:
  - HTTP$NOBODY (default account)
  - a given user specified by a mapping rule
  - The account of the VMS user authenticated through the browser authentication dialog box.

Dynamic configuration

- Any modification to the mapping rules (HTTPD$MAP) can be dynamically loaded using:
  - HTTPD$EXE /DO=MAP
  - a mouse click in the web interface
- The change is immediately taken into account with no other user visible effect.

Dynamic configuration

- Any modification to the authorization file (HTTPD$AUTH) can be dynamically loaded using:
  - HTTPD$EXE /DO=AUTH=LOAD
  - a mouse click in the web interface
- The change is immediately taken into account but implies a new authentication dialog for all clients.

WATCH

- WASD has an unmatched capability allowing to trace all events inside the server and even script to server dialog.
- The whole code has been instrumented.
- Events may be selected according to client address, service, server path, event category,…

Questions