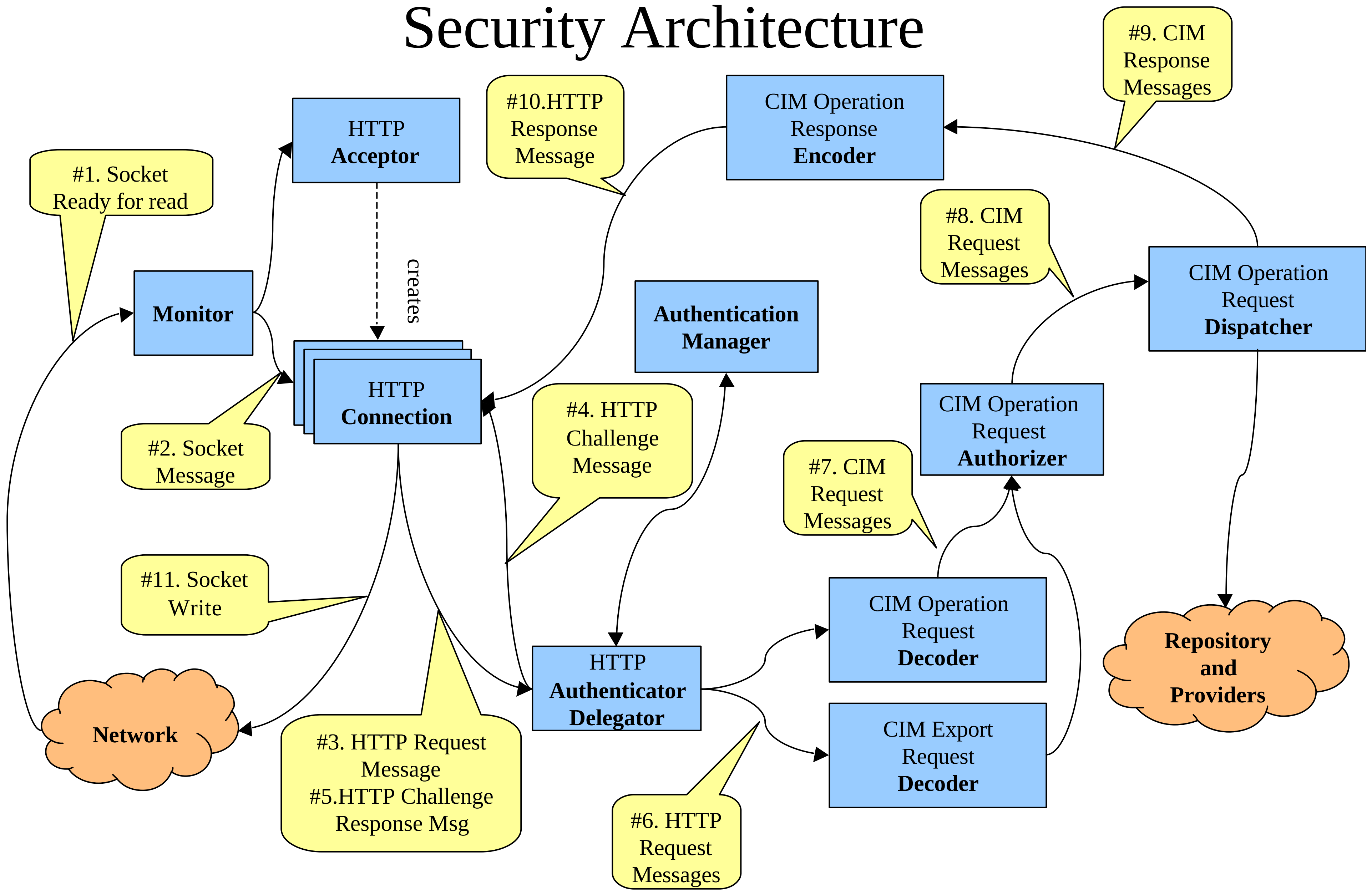
Pegasus Security Architecture

Author: Nag Boranna

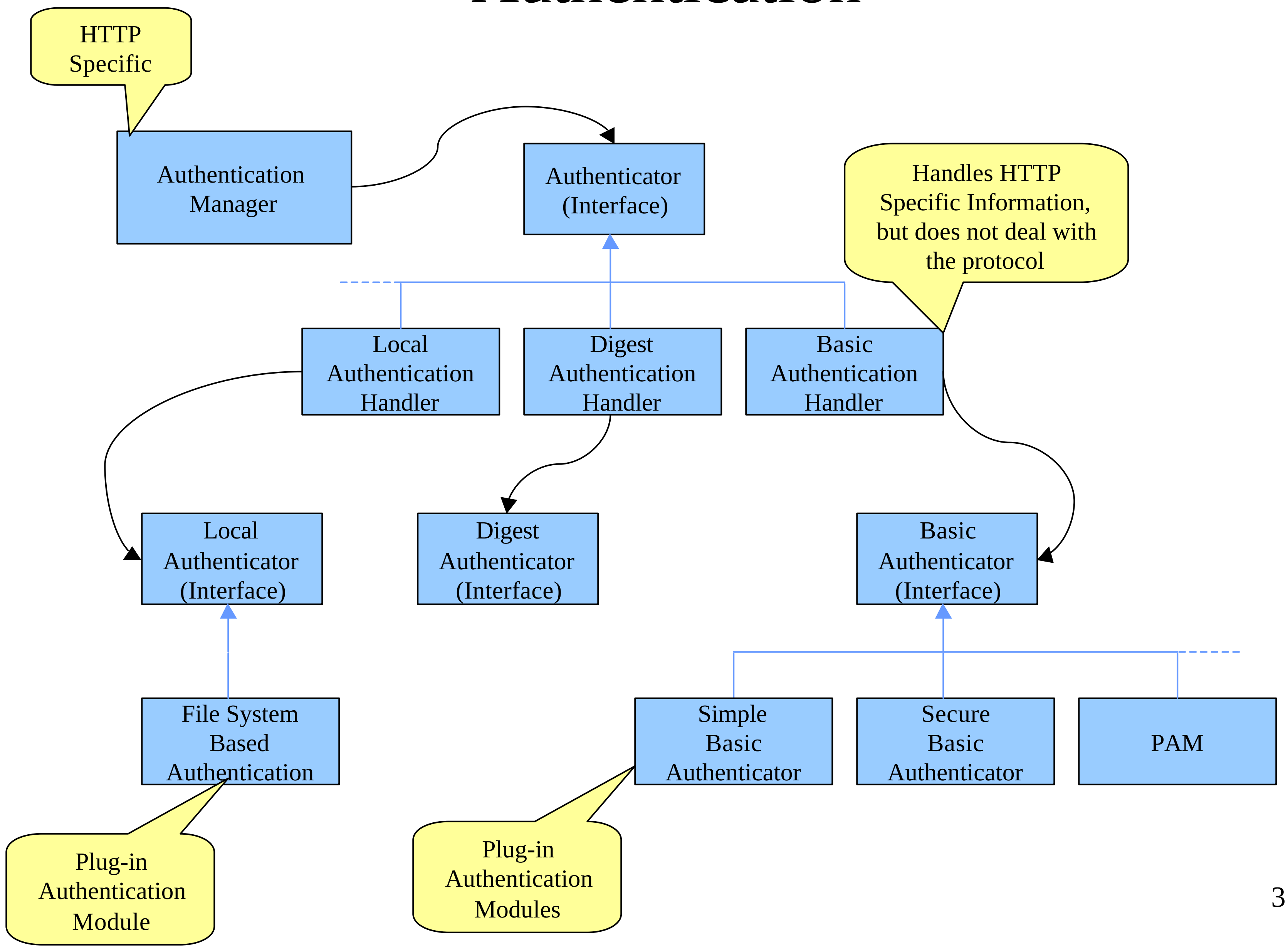
Hewlett-Packard Company

Security Architecture



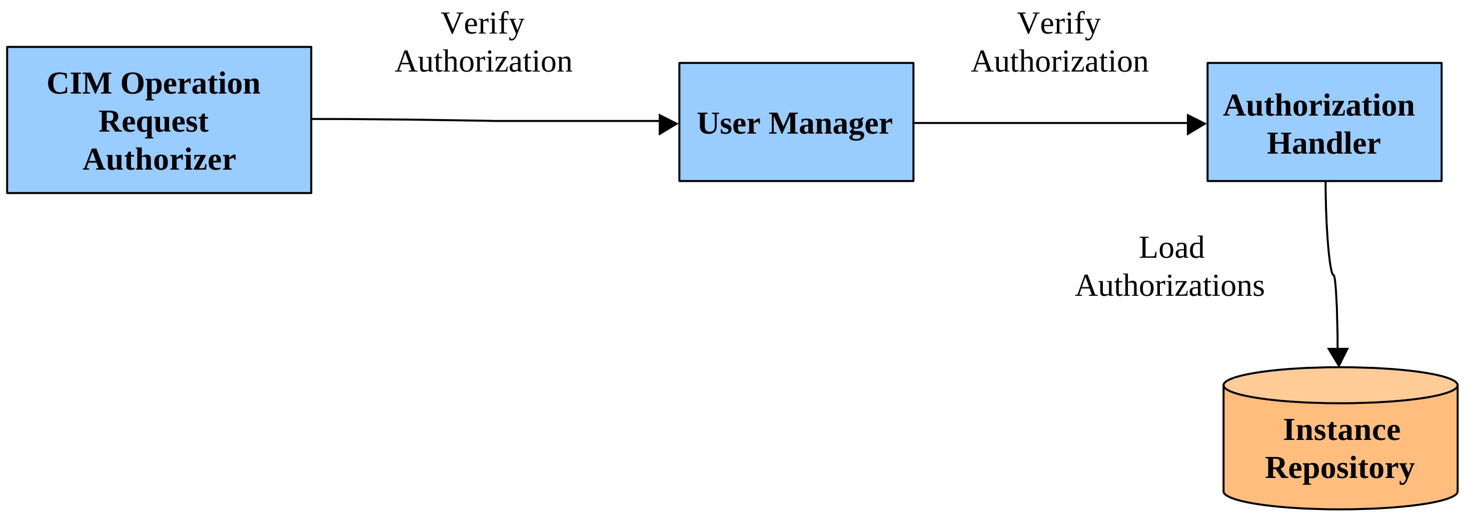
#9. CIM Response Messages
#10.HTTP Response Message
CIM Operation Response **Encoder**
HTTP **Acceptor**
#1. Socket Ready for read
#8. CIM Request Messages
creates
CIM Operation Request **Dispatcher**
Monitor
Authentication Manager
HTTP **Connection**
CIM Operation Request **Authorizer**
#2. Socket Message
#4. HTTP Challenge Message
#7. CIM Request Messages
#11. Socket Write
CIM Operation Request **Decoder**
Repository and Providers
HTTP **Authenticator Delegator**
Network
CIM Export Request **Decoder**
#3. HTTP Request Message
#5.HTTP Challenge Response Msg
#6. HTTP Request Messages

Authentication



HTTP Specific
Authentication Manager
Authenticator (Interface)
Handles HTTP Specific Information, but does not deal with the protocol
Local Authentication Handler
Digest Authentication Handler
Basic Authentication Handler
Local Authenticator (Interface)
Digest Authenticator (Interface)
Basic Authenticator (Interface)
File System Based Authentication
Simple Basic Authenticator
Secure Basic Authenticator
PAM
Plug-in Authentication Module
Plug-in Authentication Modules

Authorization



Authentication Implementation

Startup:

- CIMSever creates the HTTPAuthenticatorDelegator queue.
- HTTPAuthenticatorDelegator creates AuthenticationManager
- AuthenticationManager loads the AuthenticationHandlers (Currently it is creating instances of Authentication Handlers, but it should be changed to dynamically load those handlers)

On Client Request:

- HTTPConnection queue gets created when there is connection request from a client
- HTTPConnection creates AuthenticationInfo object to keep track of the authentication information for this connection. HTTPConnection and AuthenticationInfo object gets created for each new connection.
- HTTPConnection gets CIM requests, adds AuthenticationInfo object reference & passes to Delegator
- Delegator parse the request and looks for 'Authorization' or 'PegasusAuthorization' header tags
- If the 'Authorization' and 'PegasusAuthorization' tags are not found:
    * Calls getAuthResponseHeader() of the AuthenticationManger
    * Sends challenge to the HTTPConnection queue, and eventually to the client
- If the 'Authorization' tag is found:
    * Calls performHttpAuthentication() method of AuthenticationManager
    * If authentication is successful, then it passes the request to Decoders. Else calls getAuthResponseHeader() of the AuthenticationManger and sends challenge back.
- If the 'PegasusAuthorization' tag is found:
    * Calls performPegasusAuthentication() method of AuthenticationManager
    * If authentication is successful, then it passes the request to Decoders. Else calls getAuthResponseHeader() of the AuthenticationManger and sends challenge back.

- In all these cases the AuthenticationInfo object is updated with the authentication status for that connection session.

Authentication Components

Authentication Manager

- AuthenticationManager loads the Authentication Handlers
- When perforHttpAuthentication() or performPegasusAuthentication() methods are called
    * it parses the HTTP headers to get the auth type, user name & password information
    * calls authenticate() method of the Authentication Handler module
    * On successful authentication, updates AuthenticationInfo object and returns true else false
- When getAuthResoponseHeader() method is called, it gets the header from Authentication Handlers

Authentication Handlers (LocalAuthenticationHandler)

- Implements the Authenticator interface
- Creates response header when getAuthResoponseHeader() method is called (may get the header information from the authenticator module).
- Extracts user name and password (or digest string) from the header info passed with the authenticate() and calls the authenticate() method of the loaded authenticator module.

Authenticators (SecureLocalAuthenticator)

- Implements specific authenticator interface (e.g, BasicAuthenticator)
- The getAuthResponseHeader() method will return the authentication challenge header info.
- The authenticate() method will verify the user name and password or the digest string and return true on successful authentication, false otherwise.

Authorization Implementation

Startup:

- CIMSever creates the CIMOperationRequestAuthorizer queue if requireAuthorization config property is set to true.
- CIMOperationRequestAuthorizer gets an instance of UserManager

- UserManager creates AuthirizationHandler
- AuthenticationHandler loads authorizations from the Repository.

On Client Request:

- CIMOperationRequestAuthorizer queue receives a request message from Decoder

- CIMOperationRequestAuthorizer calls verifyAuthorization( ) method of UserManager.

- UserManager calls verifyAuthorization( ) method of AuthorizationHandler.

- AuthorizationHandler verifies the authorizations and return true if authorization were found else false

Authorization Components

CIMOperationRequestAuthorizer

- Gets the user name, namespace, authentication type and CIM method names from the CIM request
- Calls UserManager’s verifyAuthorization( ) method

User Manager

- Creates AuthirizationHandler and calls its verifyAuthorization( ) method

Authorization Handler

- Loads authorizations from the repository
- Verifies user authorizations when verifyAuthorization( ) method is called

Security Configurations

Configuration Properties required for Authentication Module

requireAuthentication = true | false

httpAuthType = Basic | Digest

passwordFilePath = cimserver.passwd (use the file in PEGASUS_HOME directory)

Configuration Properties required for Authorization Module

requireAuthorization = true | false

enableRemotePrivilegedUserAccess = true | false (enable or disable remote root access to the cimom)

Enabling Authentication

1. Set the following configuration properties in the cimserver_planned.conf file

   requireAuthentication = true

   httpAuthType = Basic (to enable HTTP Basic Authentication)

   passwordFilePath = cimserver.passwd (to create/use the password file in PEGASUS_HOME directory)

3. Start cimserver

4. Add new users to cimom (Refer to pegasus/src/Clients/cimuser/doc/cimuser.htm)

   To add user ‘nag’ with password ‘nag’ (user ‘nag’ must be valid system user on that system)

   cimuser -a -u nag -w nag

5. Run any CIM clients to do CIM operations with the cimom.

Note:

(1) On Unix systems, the cimuser CLI can only be run locally as ‘root’.

(2) Basic/Digest authentication are not fully implemented on the CIMServer and the CIMClient API.

Enabling Authorization

1. Set the following configuration properties in the cimserver_planned.conf file

    requireAuthorization = true

    enableRemotePrivilegedUserAccess = false (to disable remote root user access to the cimom)

2. Start cimserver

3. Add authorizations to the CIM users (Refer to pegasus/src/Clients/cimuser/doc/cimauth.html)

    To add both read and right authorizations to user ‘nag’ on namespace ‘root/cimv2’

    ```
    cimauth -a -u nag -n root/cimv2 -R -W
    ```

4. Run any CIM clients to do CIM operations with the cimom.

Note:

(1) On Unix systems, user ‘root’ by default will have all the authorizations for local clients (the clients that use ConnectLocal() of CIMClient API.

(2) The cimauth CLI can only be run locally as ‘root’ on Unix systems.