VPOP3 Lua Scripting Documentation

Table Of Contents

VPOP3LUA

VPOP3LUA is a version of the Lua 5.0 programming language (see www.lua.org) for the VPOP3 email server. See the **Programming in Lua** book (from http://www.lua.org/pil/) for details of programming the Lua language.

This allows scripting of several sections of VPOP3 to allow enhanced functionality and customisation.

There are several modifications to the base Lua language to make it more appropriate for running in the VPOP3 environment.

- the **os.exit** function is disabled
- the **os.setlocale** function is disabled
- the **os.execute** function is disabled
- the **os.popen** function is disabled
- the **print** function now outputs to the **LUA.OUT** file in the VPOP3 folder, rather than printing to screen
- the **stdin**, **stdout**, and **stderr** standard file handles are not created
- a **safety mode** has been created to allow restricted functionality scripts to be written by users while being unable to do some 'dangerous' things.

Also, any errors that occur when running Lua scripts are written to the **LUAERRORS.LOG** file in the VPOP3 folder

Several extra functions have been added - see the **VPOP3 Helper Functions** section for more details

The Lua scripting engine can be called from several different places in VPOP3:

- SMTP Server
- POP3 Client
- Error Handling
- Password Validation
- When a fax is sent (with VPOP3 Fax Server)
- When a message is about to be delivered to a user to allow customised routing options
- When a message is sent by a local user to generate a customised email signature for that user
- When VPOP3 is about to send messages out to a remote SMTP server to determine which messages should be sent and in what order.
- On demand (from command line)

This documentation is valid for VPOP3 2.5.1 and later. Earlier versions may not use certain Lua scripts.

In these cases VPOP3 will load a certain Lua script file, then, at certain points during its processing will call into specific Lua functions in order to allow Lua to override or enhance its behaviour.

VPOP3 Helper Functions

VPOP3 Helper Functions

VPOP3 adds several standard functions to its Lua implementation to help with tasks that are often needed by scripts within the VPOP3 environment.

There are two types of helper functions: those which are always available and those which are only available within certain callback functions.

Helper Functions which are always available

- VPOP3.GetSetting()
- VPOP3.SetSetting()
- VPOP3.ExpandAddress()
- VPOP3.GetUserList()
- VPOP3.GetUserSetting()
- VPOP3.SetUserSetting()
- VPOP3Net.GetDNSRecords()
- VPOP3Net.GetHTTP()
- VPOP3Net.GetWHOIS()
- os.sleep()
- os.timedexecute()
- os.getdirectory()
- os.setsafety()
- os.clearsafety()

Helper Functions which are available from the POP3CLT.LUA and SMTPSVR.LUA 'ProcessMessage' function

- VPOP3.GetMessagePart()
- VPOP3.MessageReset()
- VPOP3.MessageSeek()
- VPOP3.GetMessageLine()
- VPOP3.SaveAttachment()

VPOP3.GetSetting()

```
VPOP3.GetSetting("<setting name>")
```

Gets a VPOP3 global setting

if setting exists:
    Returns **true, <setting value>**
otherwise:
    Returns **false, NIL**

VPOP3.SetSetting()

```
VPOP3.SetSetting("<setting name>", "<new setting value>")
```

Sets a VPOP3 global setting

If setting exists:
    Returns **true**
otherwise:
    Returns **false**

VPOP3.ExpandAddress()

```
VPOP3.ExpandAddress("<email address>", <in mail server>)
```

Expands an email address into the recipients a message to that email address would go to (expanding mappings, lists etc)

The '<in mail server>' setting is the index number of the In Mail setting which is being used, or '-1' for incoming or local SMTP.

Returns a table of entries. The entries are indexed by number, with the following sub-entries per entry

- **Name** = the mailbox name or email address of the expanded recipient
- **List** = boolean - is the Name a list?
- **DeferredExpansion** = boolean - (only exists if 'List' is true) - if this is 'true' then the list hasn't been expanded yet, if it's 'false' then the list has been expanded into this table
- **Remote** = boolean - is the Name a remote email address/user
- **Forwarded** = boolean - is the Name a LAN forwarded email address/user

VPOP3.GetUserList()

```
VPOP3.GetUserList()
```

This returns a list of users defined in VPOP3 as a table, indexed by number. The users are not sorted in any particular order.

VPOP3.GetUserSetting()

```
VPOP3.GetUserSetting("<user name>", "<setting name>")
```

This returns the relevant setting for the specified user.

VPOP3.SetUserSetting()

```
VPOP3.SetUserSetting("<user name>", "<setting name>", "<new value>")
```

This sets a setting for the specified user.

VPOP3Net.GetDNSRecords()

```
VPOP3Net.GetDNSRecords("<domain name>", <DNS entry type> [, "<dns server list>"])
```

Returns the DNS entries of the specified type for the specified domain name. If the <dns server list> is specified, it will query those DNS servers (separate multiple DNS server IP addresses by semicolons). If the <dns server list> is not specified it will use the Windows ones if they can be detected.

It will return the values from the first DNS server which responds, NOT the union of the responses from all the listed DNS servers.

The **DNS entry type** parameter can have the following values:

- 1 = A records
- 2 = NS records
- 5 = CNAME records
- 6 = SOA records
- 11 = WKS records
- 12 = PTR records
- 15 = MX records
- 16 = TXT records

(if you are unsure what these mean, refer to some DNS documentation. A records are the most common)

If the query succeeds:
Returns **TABLE, true**
where the TABLE value will be as below
If the query fails:
Returns **<DNS error code from last DNS server queried>, false**

TABLE is a table of entries indexed on sequential numbers, each entry has the following subentries:

- **Domain**
- **Section** - the response section where the entries occurred (AR, AN or NS - refer to DNS documentation if you want to know what these mean)
- **Type** - the type of response record (A, NS, CNAME, MX, TXT, PTR, SOA, SRV or a numeric type)
- **TTL** - the TTL value from the response record
- **Value** - the response record value
- **MXPriority** - the MX priority value (only if **Type=MX**)
- If **Type = SOA**, the following values as well: (see DNS documentation for the meanings)
    - Serial
    - Refresh
    - Retry
    - Expiry
    - MinimumTTL
    - PrimaryNS
    - ResponsibleEmail

VPOP3Net.GetHTTP()

```
VPOP3Net.GetHTTP("<HTTP page to retrieve>")
```

This function retrieves a web page from a web server.

If it succeeds:
    Returns **true, TABLE**
otherwise:
    Returns **false, NIL**

TABLE is:

- **Host** - web server host being queried
- **File** - file being retrieved from that host
- **Headers** - table containing header lines indexed by sequential numbers
- **Body** - table containing body lines indexed by sequential numbers
- **ErrorInfo** - extra information about any errors
- **HTTPResult** - HTTP result code (eg 200 for success, 404 for page not found etc)
- **StatedLength** - the stated length of the returned Body from the header 'Content-Length' field
- **ActualLength** - the actual length of the returned Body

VPOP3Net.GetWHOIS()

```
VPOP3Net.GetWHOIS("<server>", "<query>")
```

Sends a WHOIS query to the specified server

If it succeeds:
    Returns **true, "<WHOIS return data>"**
otherwise:
    Returns **false, <optional error detail>**

os.sleep()

```
os.sleep(<time to sleep in ms>)
```

Pauses the current thread for the specified amount of time (in milliseconds)

os.timedexecute()

```
- os.timedexecute("<Program to run>", <time to allow> [,
<show program>])
```

Executes the specified program and wait for it to finish. If it doesn't finish in the time allowed, the program will be terminated.
If '<show program>' is true, then the executed program will be shown on screen, otherwise it will be hidden

Returns **exitcode, errorcode**

- **ExitCode** is the exit code of the program itself
- **ErrorCode** is the error code from trying to launch the program

os.getdirectory()

```
os.getdirectory("<directory name>")
```

Get the contents of the specified directory name

Returns a table of entries indexed by the name of the file/directory in the specified directory. Each entry has the following sub-entries:

- - Size - size of the file
- - Attributes - file attributes as a string (A = archive, C = compressed, D = directory, E = encrypted, H = hidden, R = readonly, S = system)
- - Creation - creation time
- - LastAccess - last access time
- - LastWrite - last write time

os.setsafety()

```
os.setsafety("<key>")
```

Enables Safety Mode with the specified security key

os.clearsafety()

```
os.clearsafety("<key>")
```

Disables Safety Mode. The <key> value must be identical to the use specified with the preceding os.setsafety() function

VPOP3.GetMessagePart()

```
VPOP3.GetMessagePart("<Path>")
```

Retrieves the entire content of a message part as a string based on the 'Path' value
The message section content is NOT decoded in any way (eg Quoted-Printable -> text)

VPOP3.MessageReset()

```
VPOP3.MessageReset()
```

Sets the internal pointer to the start of the message

VPOP3.MessageSeek()

```
VPOP3.MessageSeek(<offset>)
```

Sets the internal pointer to the specified offset from the start of the message

VPOP3.GetMessageLine()

```
VPOP3.GetMessageLine(<num lines>)
```

Retrieves the specified number of lines from the message starting from the current pointer position, and moves the pointer position to the end of those lines

Returns **<number of lines retrieved>, "<string containing requested lines>"**

VPOP3.SaveAttachment()

```
VPOP3.SaveAttachment(<attachment number>, "<filename>")
```

Saves the specified attachment from the current message to the specified filename.

On success:

Returns **true**
otherwise:
Returns **false**

Safety Mode

Lua allows access to files, the ability to call other programs, DLLs etc, all of which could potentially be problematic on a server.
To allow user based scripts to be written without them being able to use these functions, whilst still allowing administrator scripts to use them, there is a 'safety' mode which can be enabled.

To use this, call 'os.setsafety("<string>")'. This turns on safety mode

To exit safety mode, call 'os.clearsafety("<string>")'. The "<string>" value must match that specified in the os.setsafety call

In safety mode the following standard LUA library functions are disabled

- loadlib
- io.close
- io.open
- io.input
- io.output
- io.lines
- io.read
- io.write
- io.flush
- os.timedexecute
- os.remove
- os.rename
- os.getenv
- VPOP3.SetSetting
- VPOP3.GetSetting

Note that although the '**io**' file I/O functions are disabled, the '**file**' functions aren't. This means that it is possible for an 'administrator' script to open a file and pass the file handle to a user written script

SMTP Server Script

SMTP Server Script

The SMTP Server Script is called **SMTPSVR.LUA** and should be placed in the VPOP3 folder. This script is called whenever an instance of the VPOP3 SMTP server is launched - ie whenever someone opens a connection to port 25 on the VPOP3 server to send it a message, whether from a local user or an incoming SMTP message.

Each SMTP server instance has its own instance of the SMTPSVR.LUA script. No threading conflicts will occur.

During the lifetime of the SMTP server instance, there are several global variables which are available to the Lua script. These may change as the script is called on different callbacks, and may be changed by the Lua script itself in order to change the VPOP3 behaviour.

VPOP3 will call the following Lua functions in the script at the following times:

- **Start()** - called when the SMTP service session starts up, before the welcome message is sent to the SMTP client
- **RBLResults()** - (VPOP3 Enterprise Only) - called after the RBL checks have been performed
- **GetMaxMessageSize()** - called at startup, and when the *HELO* and *EHLO* commands are received
- **GetEHLOCapabilities()** - called when the *EHLO* command is received
- **DoHELO()** - called when the *HELO* or *EHLO* command is received
- **DoHELP()** - called when the *HELP* command is received
- **DoQUIT()** - called when the *QUIT* command is received
- **DoNOOP()** - called when the *NOOP* command is received
- **DoRSET()** - called when the *RSET* command is received
- **DoMAILFROM()** - called when a valid *MAIL FROM* command is received
- **DoRCPTTO()** - called when a valid *RCPT TO* command is received
- **DoDATAStart()** - called when VPOP3 receives a valid *DATA* command
- **DoUnrecognised()** - called when VPOP3 receives an unrecognised SMTP command
- **ProcessMessage()** - called when an entire message has been received by VPOP3
- **End()** - called when the SMTP service session ends

SMTP Server Global Variables

The following variables are defined for the SMTP Server Lua script. They are accessible to all callback functions in the script, and can often be modified by the script to modify the VPOP3 behaviour.

If the variable is marked as **(IN)** then it should be treated as read-only by the script. If it is marked as **(IN/OUT)** then the script can modify it if desired.

- **IPAddress** - **(IN)** - *string* - dotted notation client IP address (eg "*192.168.1.1*")
- **ValidClientAddress** - **(IN/OUT)** - *boolean* - TRUE if the client IP address is in an 'allowed' IP address range
- **LocalClientAddress** - **(IN/OUT)** - *boolean* - TRUE if client IP address is in an 'allowed' IP address range and anti-relay proteciton is 'check client IP address', and no SMTP authentication is needed
- **AuthenticationRequired** - **(IN/OUT)** - *boolean* - TRUE if SMTP authentication is required
- **AllowedUsers** - **(IN/OUT)** - *string* - List of allowed usernames (separated by spaces) if there are any username restrictions for this client IP address

In VPOP3 Enterprise Only

- **CheckRBL**
- **RBLFound**
- **RBLBlackhole**
- **RBLAccept**
- **RBLRedirect**
- **RBLRejectMessage**
- **RBLHeader**

SMTP Server "Start()"

```
Start()
```

This function is called by VPOP3 when the SMTP Server session starts, before the welcome message is sent to the SMTP client

Can return "" to give the default welcome message, or another string (including the SMTP result code).

VPOP3 will terminate the connection if a **5xx** or **4xx** result code is given

In VPOP3 Enterprise, there are also the following global variables defined. These can be changed by the **Start()** function if desired:

- **RBLServers** - table containing a list of RBL servers to query

- **RBLWhitelist** - table containing a list of white-listed servers (wildcards are supported)
- **RBLCheckAllServers** - if TRUE then all RBL servers will be checked, if FALSE, then VPOP3 will stop testing as soon as one returns a match result.

SMTP Server - RBLResults()

```
RBLResults()
```

(Only supported in VPOP3 Enterprise)

This function is called by VPOP3 after the RBL (Realtime Black List) processing has taken place

For this function the standard SMTP Server Global Variables will give the result of the RBL processing.
Also there is a table called **RBLResultInfo** which contains the raw result information in the form of a table of entries indexed on the name of the RBL database, with the entry data being the IP address resulting from the RBL query in the form of a single number (so a resulting value of '127.0.0.1' which be stored as 0x0100007f or 16777343)

(Note a bug in VPOP3 2.1.0 and earlier means that this will not work. It will be fixed in 2.1.0a and later)

This function should return a text string which can be used to block connections. This text string is used as the 'welcome' response from the VPOP3 SMTP server. If the return string is "", then the default VPOP3 welcome text will be used, or the return value from the **Start()** function if any. Otherwise the text returned from this function will be used. The text string returned should contain the SMTP result code - VPOP3 will terminate the connection if a **5xx** or **4xx** result code is given

SMTP Server - GetMaxMessageSize()

```
GetMaxMessageSize("<hostname>", <current max size in bytes>)
```

This function is called by VPOP3 when the SMTP service session starts, and also when VPOP3 receives a *EHLO* or *HELO* command

At session start the **<hostname>** value is the SMTP client IP address. At the *EHLO/HELO* time it's the parameter to the *EHLO* or *HELO* command.

This function should return the new maximum message size in bytes (0 = no limit)

SMTP Server - GetEHLOCapabilities

```
GetEHLOCapabilities("<hostname>", "<current capabilities>")
```

This function is called by VPOP3 when it receives a *EHLO* command

The **<hostname>** value is the value of the parameter to the *EHLO* command.
The **<current capabilities>** value is the current capabilities string which will be returned to the SMTP client

This function should return the new capabilities string (see the relevant RFCs for the syntax)

SMTP Server - DoHELO()

```
DoHELO("<hostname>", bRefuse, bEHLO)
```

This function is called by VPOP3 when it receives a *HELO* or *EHLO* command

The **<hostname>** value is the value of the parameter to the *EHLO* or *HELO* command.
The **<bRefuse>** value is if VPOP3 is going to refuse this SMTP connection already
The **<bEHLO>** value is TRUE if this function is being called after a *EHLO* command, or FALSE if it's being called after a *HELO* command

This function should return **<bRefuse>, "<response string>"**

- **bRefuse** is true if VPOP3 should refuse this SMTP connection
- the **response string** is the SMTP response which VPOP3 should give to the *EHLO/HELO* command. (Leave blank to use the standard VPOP3 response)

SMTP Server - DoHELP()

```
DoHELP("<help text>")
```

This function is called by VPOP3 when it receives a *HELP* command

The **<help text>** value is the text which VPOP3 will return to the SMTP client

This function should return the new help text to return to the client (see the RFCs for the syntax)

SMTP Server - DoQUIT()

```
DoQUIT()
```

This function is called by VPOP3 when it receives a *QUIT* command

This function should return the text to return to the client (or "" to use the default VPOP3 text)

SMTP Server - DoNOOP()

```
DoNOOP()
```

This function is called by VPOP3 when it receives a *NOOP* command

This function should return the text to return to the client (or "" to use the default VPOP3 text)
.

SMTP Server - DoRSET()

```
DoRSET()
```

This function is called by VPOP3 when it receives a *RSET* command

This function should return the text to return to the client (or "" to use the default VPOP3 text)
..

SMTP Server - DoMAILFROM()

```
DoMAILFROM("<data>", "<mailfrom>", <parameters>)
```

This function is called by VPOP3 when it receives a valid *MAIL FROM* command

The **<data>** is the raw data after the *MAIL FROM:* command
The **<mailfrom>** is the address after the *MAIL FROM:* command

The **<parameters>** is a table of parameters to the *MAIL FROM:* command (if any) with the table entry key being the parameter name and the table entry value being the parameter value

This function should return **<SMTP Result string>**, **<new parameters>**
.

SMTP Server - DoRCPTTO()

```
DoRCPTTO("<data>", "<recipient>", <parameters>, <recipient list>)
```

This function is called by VPOP3 when it receives a valid *RCPT TO* command

The **<data>** is the raw data after the *RCPT TO:* command
The **<recipient>** is the address after the *RCPT TO:* command
The **<parameters>** is a table of parameters to the *MRCPT TO:* command (if any) with the table entry key being the parameter name and the table entry value being the parameter value
The **<recipient list>** is a table containing the list of recipient email addresses for this message so far (not including this one)

This function should return **<SMTP Result string>**, **<new parameters>**

SMTP Server - DoDATAStart()

```
DoDATAStart(<recipient list>)
```

This function is called by VPOP3 when it receives a valid *DATA* command

The **<recipient list>** is a table containing the list of recipient email addresses for this message

This function should return **<SMTP Result string>**, **<Header lines to ADD to the start of the message header>**

SMTP Server - DoUnrecognised()

```
DoUnrecognised(<line>)
```

This function is called by VPOP3 when it receives a command it doesn't recognise

The **<line>** is a raw command line that VPOP3 received

This function should return **<SMTP Result string>**

SMTP Server - ProcessMessage()

```
ProcessMessage("<mailfrom>", "<subject>", <recipient list>,
<current actions>, <message MIME structure>, <message
size>, <attachments>)
```

This function is called by VPOP3 when a message has been received by the SMTP service.

- **<mailfrom>** is the SMTP 'MAIL FROM' address
- **<subject>** is the message subject line
- **<recipient list>** is a table containing a list of the recipient email addresses (from the RCPT TO envelope)
- **<current actions>** is a table with the following entries
    - **SendToOriginalRecipients** *boolean* - should the message be sent to the originally specified recipients
    - **Delete** *boolean* - should the message be deleted after retrieving (see below)
    - **Ignore** *boolean* - should the message be ignored
    - **Reason** *string* - text string to use if a reason is needed for the action
    - **Recipients** *table* - table containing new/replacement recipients for the message
    - **HeaderModifiers** *table* - list of message header modifications to make
- **<message MIME structure>** is a table with one or more of the following entries
    - **Path** *string* - the IMAP style path to the message section
    - **ContentType** *string* - the main content type (eg 'text', 'image' etc)
    - **ContentSubtype** *string* - the content subtype (eg 'html', 'plain', 'gif', etc)
    - **ContentTypeFilename** *string* - the filename from the **ContentType** header
    - **ContentDisposition** *string* - usually 'inline' or 'attachment' or blank
    - **ContentDispositionFilename** *string* - the filename from the **ContentDisposition** header
    - **Start** *number* - the offset from the start of the message where this section starts
    - **End** *number* - the offset from the start of the message where this section ends
- **<message size>** is the size of the message in bytes
- **<attachments>** is a table with zero or more of the following entries, referring to the attachments in the message
    - **MIMESection** *number* - the MIME section that this attachment occurred in
    - **Name** *string* - the name of the attachment

- **MIMEType** *string* - the MIME type of the attachment
- **Size** *number* - the size of the attachment
- **Type** *string* - the type of attachment (**MIME**, **UUE**, **BINHEX**, **MIMEUUE**, **MIMEBINHEX**, **TNEF**)

In **<actions>**, 'Delete' is implied for all SMTP messages (it's not possible to *not* delete a mesage). If **Delete** and **Ignore** are both TRUE then the message is rejected with a **5xx** SMTP error code. If **Ignore** is TRUE, but **Delete** is FALSE, then the message is accepted, but not processed (ie 'blackholed')

This function should return **<new actions table>**, **<results string>**

In this function the VPOP3 ProcessMessage helper functions can be used.

SMTP Server - End()

```
End()
```

This function is called by VPOP3 when the SMTP service session ends

Error Handling

The Error Handling Script is called **ERRORS.LUA** and should be placed in the VPOP3 folder. This script is called whenever certain error messages are about to be generated.

VPOP3 will call the **ProcessError()** function in the script

Error Handling - ProcessError()

```
ProcessError("<itemid>", "<error text>", <connection
error>, <subject resource id>, "<subject text>", <message
format resource id>, "<message format text>", <message text
parameters>, "<error message target>", "<error message
sender>")
```

This function is called by VPOP3 when certain error messages are about to be generated.

- **<Itemid**> this is the 'item' reporting an error, eg the InMail item index
- **<error text>** this is the original text of the error code – eg the ISP's error result

- **<connection error>** this is a Boolean value saying whether the error is due to a problem during establishing the connection (true) or afterwards (false)
- **<subject resource id>** this is a number which is the resource ID of the message subject to be used. This is normally not of much use to the Lua script, but it may be useful as an index, especially since the resources may have been internationalised meaning the subject text may vary across installations, but the resource ID will be constant.
- **<subject text>** this is the text to be used as the error message subject
- **<message format resource id>** this is a number which is the resource ID of the message content format to be used. See the subject resource id above for more information
- **<message format text>** this is the text to be used as the error message text format. This can contain values like %1, %2 etc to indicate replaceable text from the **<message text parameters>** list
- **<message text parameters>** this is a list of data to be placed in replacement points in the message format text.
- **<error message target>** this is the email address which the message will be sent to (usually the VPOP3 Main Administrator)
- **<error message sender>** this is the email address which the message will be sent from (usually 'Mailer_Daemon')

This function should return 5 entries:

- New message subject
- New message format text
- New message target
- New message sender
- 'Force message' (Boolean)

If the 'New message format text' is blank, no error will be sent
If 'Force Message' is set to true, then the error message will be sent regardless of any other checks to say whether the message should be sent or not.

Password Validation

The Password Validation Script is called **PASSWORDCHECK.LUA** and should be placed in the VPOP3 folder. This script is called whenever a user changes their password.

VPOP3 will call the **Check()** function in the script

Password Validation - Check()

```
Check("<user name>", "<new password>", "<min password
length>")
```

This function is called by VPOP3 whenever a user changes their password.

- **<user name>** this is the name of the user whose password is being changed
- **<new password>** this is the new password being set
- **<min password length>** this is the minimum password length configured in VPOP3

This script can do things such as complexity checking, eg if you have any requirements for a certain mix of characters in the password, although note that if passwords are too complex it may lead to users writing them down and sticking them to their computer monitors, which is probably less secure than having a simpler password!

The function should return a Boolean which is true if the password is allowed, or false if not.

Message Routing

(Not in versions prior to 2.5.1)
The Message Routing Script is called **USER_REDIRECT.LUA** and should be placed in the VPOP3 folder. This script is called whenever a message is about to be delivered to a local user. It can be used to modify how the message is routed.

VPOP3 comes with a default USER_REDIRECT.LUA script, which will call a USER_REDIRECT.LUA script in the user's folder if one exists. This allows user specific routing rules to be scripted. The default script allows time/date based routing options.

VPOP3 will call the **MainDoRouting()** function in the script when a message comes in. It will also call **MailGetSettings()** when a user's settings pages are displayed in the VPOP3 settings. This allows configuration of the script's routing settings through the settings page.

Message Routing – MainDoRouting()

```
MainDoRouting("<user id>")
```

This function is called by VPOP3 whenever a user is to be routed to a VPOP3 user.

- **<user id>** this is the receiving user id. If the message is to be delivered to several local users, the script is called once for each user.

There are also lots of global variables set in the script:

- **ShouldForward** – this is set to true if VPOP3 would usually forward the message, depending on the spam score, and whether VPOP3 should forward spam or not
- **Assistants** – this is the user's Assistants setting
- **RedirectToAssistant** – this is set to true if VPOP3 would only send the message to the assistant, not to the user's mailbox as well
- **SendToNormalRecipient** – this is true if VPOP3 would send the message to the normal recipient
- **Forwards** – this is the user's 'Forward To' setting
- **UseForwards** – this is true if VPOP3 is set to use the Forward To setting
- **BigRedirect** – this is the settings for the large message redirection settings for the user (see below)
- **SmallRedirect** – this is the settings for the small message redirection settings for the user (see below)
- **MessageSize** – this is the raw size of the message
- **SpamScore** – this is the spam score for the message
- **SpamScoreValid** – if this is false, then the spam score has not been read for some reason (eg spam filtering is disabled) so it should not be used
- **Sender** – this is the email address of the message sender
- **User** – this is the user id of the recipient
- **Subject** – this is the subject of the email
- **UserSettings** – this is an associative array of any custom user settings, such as those specified specifically for the script in the **MainGetSettings** function
- **Headers1** – this is an associative array of the message headers, so that you can use **Headers1["Subject"]** to get the message subject, for instance. If there are two header lines with the same name, only the last one will be in this array.
- **Headers2** – this is an array indexed by number of all the message header lines in order

This function should modify the following global variables to modify how the message is routed.

- **ShouldForward**
- **Assistants**
- **RedirectToAssistant**
- **SendToNormalRecipient**
- **Forwards**
- **UseForwards**
- **BigRedirect**
- **SmallRedirect**

Note that changing these global variables won't change the user's settings, just their current values for processing this particular message.

Signature Generation

The Signature Generation Script is called **SIGNATURE.LUA** and should be placed in the VPOP3 folder. This script is called whenever a local user sends an outgoing message, to determine what signature (if any) to put on their email.

VPOP3 will call the **GetSignature()** function in the script

Signature Generation – GetSignature()

```
GetSignature("<format>", "<auth user>", "<sender email>")
```

This function is called by VPOP3 whenever a user sends an outgoing email.

- **<format>** this is either HTML or PLAIN
- **<auth user>** this is the username of the authenticated user, if any. If the sender did not use SMTP authentication, then this is blank
- **<sender email>** this is the email address of the message sender. Note that it is possible for this to be forged, unlike the <auth user> value

This function should return a string which is the value of the signature to use, if any. If the returned string is blank, then VPOP3 will check the signature values set in the VPOP3 settings.

See http://kbase.pscs.co.uk/index.php?article=339 for an example of this type of script.

Outgoing Mail – SMTP Relay

(Not in versions prior to 2.5.1)
The Outgoing Mail script for outgoing SMTP Relay mail is called **RELAYOUT.LUA** and should be placed in the VPOP3 folder. This script is called before VPOP3 start sending out mail to an external SMTP relay server. It may also be called again whilst mail is being sent. This script can be used to determine which messages should be sent, and in what order.

VPOP3 will call the **CheckFile()** function in the script

Outgoing Mail - Relay – CheckFile()

```
CheckFile ("<filename>", "<actions>", <size>, "<file
creation time>", "<return path>", <recipients>, <header
data>, <header lines>)
```

This function is called by VPOP3 for every message which it is about to send out to the Internet.

- **<filename>** this is the filename on the VPOP3 disk. Usually this can be ignored
- **<actions>** this is a Lua array of 3 entries – SkipSend, Reason and Priority. See below for details
- **<size>** this is raw size of the message about to be sent
- **<file creation time>** this is the date/time which the file was created. This is in the format **YYYYMMDDHHMMSS**
- **<return path>** this is the return path value of the outgoing message (usually the email address of the sender)
- **<recipients>** this is an array of the message recipients
- **<header data>** this is an associative array of the header data, so you can use headerdata["subject"] to get the message subject, for instance. If there are several header fields with the same value, only the last will be present in this array.
- **<header lines>** this is an array of all the message header fields in order, indexed by number, so all header lines will be in this array, but it is less easy to use than the <header data> array above

In the **<actions>** array, there are three entries:

- **SkipSend** – if this is true, the message will not be sent. If it is false, it will be sent
- **Reason** – this value is only used for logging, if the script changes any of the action values
- **Priority** – this sets the order in which messages will be sent. Larger priorities will be sent first. All messages have a priority of 50 unless the script changes them. In this case they will either be sent in order of originally being sent, or in order of size, depending on the VPOP3 settings. If different priorities are set, then VPOP3 will use a 'stable sort' to reorder the messages, so that if the messages were originally sorted by size, and 10 messages are given a priority of 100, those messages will be sent before the other messages, but still sorted by size.

This function should return the 'Actions' array after any modifications have been made. If no modifications are made, simply return the Actions array as it was passed into the function.

Outgoing Mail – SMTP MX

(Not in versions prior to 2.5.1)
The Outgoing Mail script for outgoing SMTP MX mail is called **MXOUT.LUA** and should be placed in the VPOP3 folder. This script is called before VPOP3 start sending out mail to external SMTP MX servers. It may also be called again whilst

mail is being sent. This script can be used to determine which messages should be sent, and in what order.

VPOP3 will call the **CheckFile()** function in the script

Outgoing Mail - MX – CheckFile()

```
CheckFile ("<filename>", "<actions>", <size>, "<file
creation time>", <retries>, "<last try time>", "<return
path>", <recipients>, <header data>, <header lines>)
```

This function is called by VPOP3 for every message which it is about to send out to the Internet.

- **<filename>** this is the filename on the VPOP3 disk. Usually this can be ignored
- **<actions>** this is a Lua array of 5 entries – SkipSend, ForceRetry, ForceLastTry, Reason and Priority. See below for details
- **<size>** this is raw size of the message about to be sent
- **<file creation time>** this is the date/time which the file was created. This is in the format **YYYYMMDDHHMMSS**
- **<retries>** this is the number of times VPOP3 has previously tried to send the message
- **<last try time>** this is the last time that VPOP3 tried unsuccessfully to send this message. This is in the format **YYYYMMDDHHMMSS**
- **<return path>** this is the return path value of the outgoing message (usually the email address of the sender)
- **<recipients>** this is an array of the message recipients
- **<header data>** this is an associative array of the header data, so you can use headerdata["subject"] to get the message subject, for instance. If there are several header fields with the same value, only the last will be present in this array.
- **<header lines>** this is an array of all the message header fields in order, indexed by number, so all header lines will be in this array, but it is less easy to use than the <header data> array above

In the **<actions>** array, there are three entries:

- **SkipSend** – if this is true, the message will not be sent. If it is false, it will be sent
- **ForceRetry** – if this is true, VPOP3 will try to send this message in this session even if the normal retry settings would make it skip this message
- **ForceLastTry** – if this is true, VPOP3 will try this message in this session and then fail it immediately if the message can't be sent, rather than waiting for the configured amount of time before giving up
- **Reason** – this value is only used for logging, if the script changes any of the action values

- **Priority** – this sets the order in which messages will be sent. Larger priorities will be sent first. All messages have a priority of 50 unless the script changes them. In this case they will either be sent in order of originally being sent, or in order of size, depending on the VPOP3 settings. If different priorities are set, then VPOP3 will use a 'stable sort' to reorder the messages, so that if the messages were originally sorted by size, and 10 messages are given a priority of 100, those messages will be sent before the other messages, but still sorted by size.

This function should return the 'Actions' array after any modifications have been made. If no modifications are made, simply return the Actions array as it was passed into the function.