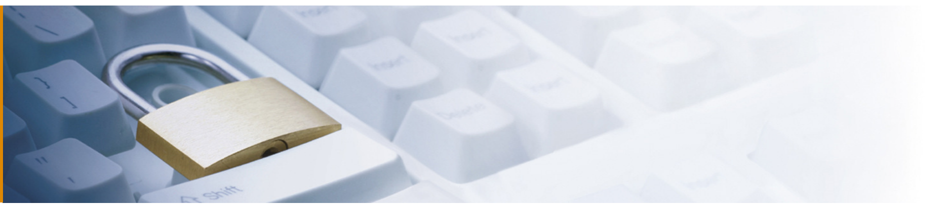
Compass Security AG
Glärnischstrasse 7
Postfach 1628
CH-8640 Rapperswil

T +41 55 214 41 60
F +41 55 214 41 61
team@csnc.ch
www.csnc.ch

# MOD_BUT Documentation

10. August 2009

| | |
|---|---|
| Name des Dokumentes: | mod_but.doc |
| Version: | V 1.0 |
| Autor(en): | Ivan Buetler, Compass Security AG |
| Lieferungsdatum: | 10. August 2009 |
| Klassifikation: | PUBLIC |



# MOD_BUT – Documentation – V 1.0

## Inhaltsverzeichnis

Compass Security AG
Glärnischstrasse 7
Postfach 1628
CH-8640 Rapperswil

T +41 55 214 41 60
F +41 55 214 41 61
team@csnc.ch
www.csnc.ch

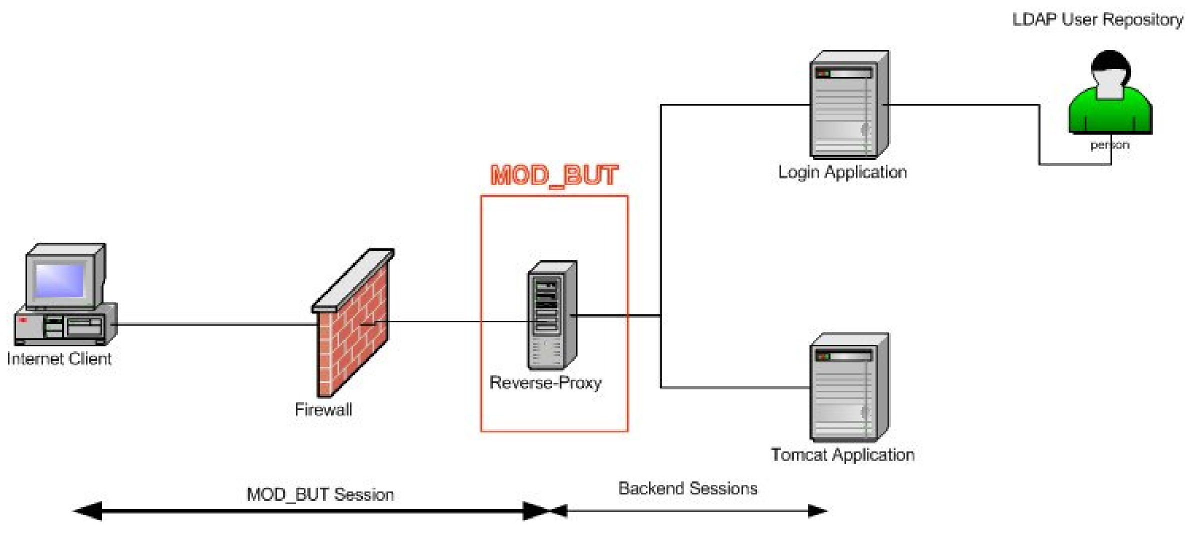
# 1 Apache Module "mod_but"

## 1.1 Preamble

Apache has been the most popular web server on the Internet since April of 1996. The February 2005 Netcraft Web Server Survey[1] found that more than 68% of the web sites on the Internet are using Apache, thus making it more widely used than all other web servers combined.

Apache offers flexible programming interfaces, by which the core functionality is expandable. If such an additional functionality, in terms of Apache it is called a "module", is rather successful, it could be added to the main distribution. Examples of successful modules are mod_rewrite, mod_proxy or mod_ssl.

This report introduces a new module, so-called MOD_BUT. If placed in front of "traditional" web servers, it can act as secure, single-sign on authentication and entry server.



## 1.2 Goals of this Project

Large e-business infrastructures take advantage of so-called web application firewalls. Such firewalls are placed in front of web infrastructures and they help armoring your web based applications. In Switzerland, we use the term "Entry Server" for such products and concepts. MOD_BUT is an open-source entry server, based on Apache 2.x web server APR (Apache Programming Runtime) interfaces.

Additionally, the module can be used in conjunction of penetration tests and web application security assessments, because it simplifies session handling and tracking mechanisms. The assessment idea was the basic driver behind MOD_BUT.

Once MOD_BUT is in place, it can add additional security by pre-authentication, session hiding and single sign on technique. If pre-authentication is configured, only authenticated user requests will be sent to the

[1] Web Server Survey: http://news.netcraft.com/archives/web_server_survey.html

"real" application. This increases the security, because anonymous users will not be able requesting trusted applications without being authenticated any more. Furthermore, auditing and logging requirements fit better, because MOD_BUT protected sites always know the senders identity.

Single Sing On is a new feature of MOD_BUT version 2.9. Before version 2.9, users have to authenticate twice. The first time, the user authenticates at MOD_BUT and if successfully authenticated, they login at the final backend application again. This is not very user friendly. MOD_BUT solves this problem by the Single Sing On enhancement component. Once the user is authenticated at MOD_BUT, access to MOD_BUT protected application will be granted without a second login procedure.

## 1.3 MOD_BUT Introduction

This document describes the core functionality of MOD_BUT, an Apache 2.x module designed to operate as authentication, secure entry server and Single-Sign ON service. MOD_BUT integrates with other standard modules (e.g. mod_rewrite, mod_proxy, mod_ssl).

MOD_BUT implements the following functionality:

- Authentication. Requests from authenticated users are sent to backend systems. This document refers other servers "behind" the reverse proxy as backend systems. Requests from unauthenticated users are denied from being sent to backend systems. By doing so, only authenticated users are allowed to access an entry server-protected web site. MOD_BUT allows you to define special URLs for which authentication is not enforced. This mechanism is also called "pre-authentication".
- Cookie based session handling between the client and MOD_BUT. This session is referred to as "***MOD_BUT session***".
- Cookie hiding of backend application cookies within a shared memory store. This type of shared memory segment is referred as "Cookie Store", "Cookie Bag" or "Session Store". The session between the reverse proxy and the backend system is referred as "***Backend Session***". For example the *jsessionid* is the backend session for Java based web application servers, like Tomcat, WebSphere or BEA WebLogic.
- Centralized session termination (logout feature)
- HttpOnly[2] cookie flag support.
- Support for "free" cookies which are not processed by MOD_BUT and are sent transparently between the client and the backend system.
- MOD_BUT has another URL-based authorization layer. Once the user is authenticated at MOD_BUT, one will have access to any backend system URL. The login application **can** (mandatory) setup a URL restriction for which the user is authenticated. By doing so, the configuration of per-user access restrictions to backend systems can be applied. For example; a user might be authenticated at MOD_BUT for the url /webmail, but not authorized for the url /freesms. Such restriction can be applied.
- Deletion of cookies from the cookie store if the cookie value is equal "deleted"

[2] Mitigating Cross-site Scripting With HTTP-only Cookies: http://msdn.microsoft.com/workshop/author/dhtml/httponly_cookies.asp

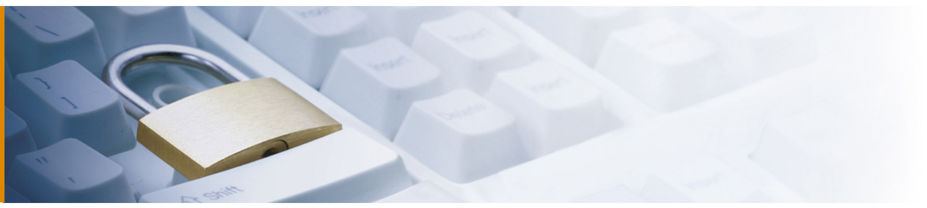
- Support for different authentication levels. If a user requests a page where a higher authentication strength is required, mod_but will do a step-up. This means, the user will be redirected to the strong authentication login application. The authentication strength is set by the login application and stored to the MOD_BUT session.
- MOD_BUT v2.9 offers Single Sign On. If configured, the user can login at MOD_BUT once and a DLS (delegated login service) authenticates the user for the users backend applications.

MOD_BUT does *not* offer the following features:

- MOD_BUT is not compatible with Apache 1.x web servers.
- Request filtering is not implemented by MOD_BUT. We recommend using mod_security instead.
- MOD_BUT only handles cookie-based sessions.
- MOD_BUT does not have nice configuration interfaces like Seclutions Airlock. Configuration is achieved through changes in the Apache configuration files.
- MOD_BUT is unable to use the SSL session id as additional session information. This is something we consider implementing later.

The core of MOD_BUT is its shared memory management, which enables a standard Apache reverse proxy to handle its own sessions. Shared memory is implemented via Apache APR interfaces. The source code of MOD_BUT is absolutely based on APR, without foreign libraries and codes.

MOD_BUT uses a variety of configuration directives, and is not one of the easy "download, configure, make, make install" modules. Therefore, we have created a demonstration application, where you can play around with MOD_BUT before using it. Additionally, this PDF introduces training test cases for a better understanding of MOD_BUT. It will help you really finding the real power of MOD_BUT.



## 1.4 List of Configuration Directives

| | | |
|---|---|---|
| MOD_BUT_ENABLED | Chapter 1.5 | |
| MOD_BUT_CLIENT_REFUSES_COOKIES_URL | Chapter 1.6 | |
| MOD_BUT_COOKIE_NAME | Chapter 1.7 | |
| MOD_BUT_COOKIE_DOMAIN | Chapter 1.7 | |
| MOD_BUT_COOKIE_PATH | Chapter 1.7 | |
| MOD_BUT_COOKIE_EXPIRATION | Chapter 1.7 | |
| MOD_BUT_COOKIE_SECURE | Chapter 1.7 | |
| MOD_BUT_COOKIE_HTTPONLY | Chapter 1.7 | |
| MOD_BUT_SESSION_FREE_URL | Chapter 1.8 | |
| MOD_BUT_SESSION_TIMEOUT_URL | Chapter 1.8 | |
| MOD_BUT_SESSION_RENEW_URL | Chapter 1.8 | |
| MOD_BUT_SESSION_HARD_TIMEOUT | Chapter 1.8 | |
| MOD_BUT_SESSION_INACTIVITY_TIMEOUT | Chapter 1.8 | |
| MOD_BUT_SESSION_DESTROY | Chapter 1.8 | |
| MOD_BUT_SESSION_DESTROY_URL | Chapter 1.8 | |
| MOD_BUT_SESSION_STORE_FREE_COOKIES | Chapter 1.8 | |
| MOD_BUT_ALL_SHM_SPACE_USED_URL | Chapter 1.9 | |
| | | |
| MOD_BUT_AUTHORIZATION_ENABLED | Chapter 1.10 | |
| MOD_BUT_GLOBAL_LOGON_SERVER_URL | Chapter 1.10 | |
| MOD_BUT_GLOBAL_LOGON_SERVER_URL_1 | Chapter 1.10 | **New since Version 2.8** |
| MOD_BUT_GLOBAL_LOGON_SERVER_URL_2 | Chapter 1.10 | **New since Version 2.8** |
| MOD_BUT_GLOBAL_LOGON_AUTH_COOKIE_NAME | Chapter 1.10 | |
| MOD_BUT_GLOBAL_LOGON_AUTH_COOKIE_VALUE | Chapter 1.10 | |
| MOD_BUT_AUTHORIZED_LOGON_URL | Chapter 1.10 | **New since Version 2.0** |
| | | |
| MOD_BUT_AUTH_STRENGTH | Chapter 1.13 | **New since Version 2.8** |
| | | |
| MOD_BUT_LOGON_SERVER_URL | Chapter 1.11 | |
| MOD_BUT_LOGON_REQUIRED | Chapter 1.11 | |
| | | |
| MOD_BUT_SERVICE_LIST_ENABLED | Chapter 1.12 | **New since Version 2.0** |
| MOD_BUT_SERVICE_LIST_COOKIE_NAME | Chapter 1.12 | **New since Version 2.0** |
| MOD_BUT_SERVICE_LIST_COOKIE_VALUE | Chapter 1.12 | **New since Version 2.0** |
| MOD_BUT_SERVICE_LIST_AUTH_ERROR_URL | Chapter 1.12 | **New since Version 2.0** |
| | | |
| MOD_BUT_LOCATION_ID | Chapter 1.14 | **New since Version 2.6** |

## 1.5 Enable/Disable

It is possible to Enable/Disable MOD_BUT per VirtualHost.

Context: virtual host

| Key | Parameter | Description |
|---|---|---|
| MOD_BUT_ENABLED | On, Off | Enable/Disable MOD_BUT per VirtualHost<br><br>Default: Off |

## 1.6 Cookie Test

MOD_BUT requires HTTP cookies to work properly. If the client multiple time denies Set-Cookie headers, MOD_BUT will send an error page. The configuration directive configures the URL of this error page.

Context: virtual host

| Key | Parameter | Description |
|---|---|---|
| MOD_BUT_CLIENT_REFUSES_COOKIES_IN_URL | String | Configure error URL, if browser denies Set-Cookie headers<br><br>Default: none |

**Note**: Be aware that **MOD_BUT_CLIENT_REFUSES_COOKIES_IN_URL** must be part of the **FREE URL section**. Otherwise Apache will **loop with URL redirections**.

## 1.7 Cookie Settings

The following configuration directives define the MOD_BUT_SESSION. This is the cookiethat is used between the client and MOD_BUT.

Context: virtual host

| Key | Parameter | Description |
|---|---|---|
| MOD_BUT_COOKIE_NAME | String | Configure cookie name<br><br>Default: MOD_BUT |
| MOD_BUT_COOKIE_DOMAIN | String | Configure cookie domain<br><br>Default: not specified |

| Key | Parameter | Description |
|---|---|---|
| MOD_BUT_COOKIE_PATH | String | Configure cookie path<br><br>Default: / |
| MOD_BUT_COOKIE_EXPIRATION | String | Configure cookie expiration date<br><br>Default: not specified |
| MOD_BUT_COOKIE_SECURE | On, Off | Configure cookie secure<br><br>Default: On |
| MOD_BUT_COOKIE_HTTPONLY | On, Off | Configure HttpOnly (IE 6.0 SP1)<br><br>Default: On |

**Note**: Be aware that **MOD_BUT_COOKIE_EXPIRATION** should be left empty so the session will reside within the browser's memory only and is not stored to the hard disk. HttpOnly is a special flag for Microsoft Internet Explorer 6.0, SP1. It denies JavaScript from accessing cookies.

## 1.8 Session Settings

Context: virtual host

| Key | Parameter | Description |
|---|---|---|
| MOD_BUT_SESSION_FREE_URL | String | Regular expression: Configure patterns, for which MOD_BUT will not apply its session tests<br><br>Default: not specified |
| MOD_BUT_SESSION_HARD_TIMEOUT | Integer | Configure max session time of MOD_BUT (elapsed time).<br><br>Default: 3600 (seconds) |
| MOD_BUT_SESSION_INACTIVITY_TIMEOUT | Integer | Configure inactivity timeout. This timeout is below the session timeout<br><br>Default: 900 (seconds) |
| MOD_BUT_SESSION_TIMEOUT_URL | String | Error URL in case the session is timed out (hard or inactivity timeout)<br><br>Default: not specified |

| Key | Parameter | Description |
|---|---|---|
| MOD_BUT_SESSION_RENEW_URL | String | Regular expression: Renew pattern for which MOD_BUT will create a new session, independent of what the client sent previously<br><br>Default: not specified |
| MOD_BUT_SESSION_DESTROY | String | Regular expression: Logout pattern (session destroy)<br><br>Default: |
| MOD_BUT_SESSION_DESTROY_URL | String | Error URL in case the client has logged out<br><br>Default: |
| MOD_BUT_SESSION_STORE_FREE_COOKIES | String | Regular expression: Configure cookie names, which are not handled by MOD_BUT. They pass the reverse proxy without being kept within the shared memory session store<br><br>Default: not specified |

## 1.9 Shared Memory Settings

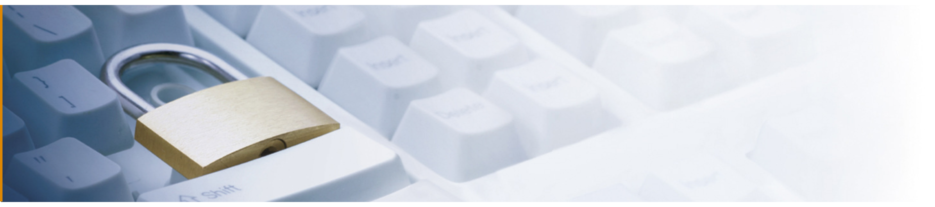
Context: virtual host

| Key | Parameter | Description |
|---|---|---|
| MOD_BUT_ALL_SHM_SPACE_USED_URL | String | Error URL in case MOD_BUT is not able to store a session to the shared memory segment<br><br>Default: not specified |

**Note**: The shared memory size is not configurable in httpd.conf, because the configuration file is parsed after Apache is started. Therefore, some configuration settings need to be configured in mod_but.h before compiling MOD_BUT.

mod_but.h

| Key | Parameter | Description |
|---|---|---|
| MOD_BUT_SESSION_COUNT | Integer | Define number of MOD_BUT sessions (most important)<br><br>Default: 1000 |
| MOD_BUT_COOKIESTORE_COUNT | Integer | Define cookie store count. This shared memory segment stores the backend cookies (sessions)<br><br>Default: 5000 |



## 1.10 Global Authentication and Authorization

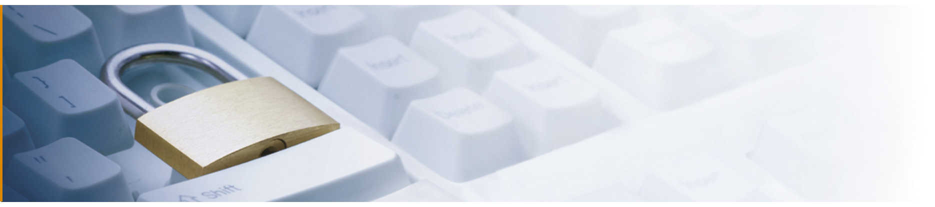
Context: Virtual Host

| Key | Parameter | Description |
|---|---|---|
| MOD_BUT_AUTHORIZATION_ENABLED | On, Off | If set to "On", MOD_BUT will test the MOD_BUT session for authentication. Only authenticated users are allowed requesting protected URL's<br><br>If set to "Off", MOD_BUT will not enforce authentication to any URL.<br><br>Default: off |
| MOD_BUT_GLOBAL_LOGON_SERVER_URL | String | Defines the URL where the login application resides. The GLOBAL value is used, if the Location directive does not configure it's own login service URL.<br><br>Default: not specified |
| MOD_BUT_GLOBAL_LOGON_SERVER_URL_1 | String | Defines the URL where the login application resides. The GLOBAL value is used, if the Location directive does not configure it's own login service URL.<br><br>URL_1 is used for medium security strength (medium = 1)<br><br>Default: not specified |

Compass Security AG
Glärnischstrasse 7
Postfach 1628
CH-8640 Rapperswil

T +41 55 214 41 60
F +41 55 214 41 61
team@csnc.ch
www.csnc.ch

| Key | Parameter | Description |
|---|---|---|
| MOD_BUT_GLOBAL_LOGON_SERVER_URL_2 | String | Defines the URL where the login application resides. The GLOBAL value is used, if the Location directive does not configure it's own login service URL.<br><br>URL_1 is used for high security strength (medium = 2)<br><br>Default: not specified |
| MOD_BUT_GLOBAL_LOGON_AUTH_COOKIE_NAME | String | Define cookie name of MOD_BUT, which changes the authentication status. This cookie is sent from a backend-system as response header<br><br>Default: LOGON |
| MOD_BUT_GLOBAL_LOGON_AUTH_COOKIE_VALUE | String | Define cookie value of MOD_BUT, which changes the authentication status.<br><br>Default: ok |
| MOD_BUT_AUTHORIZED_LOGON_URL | String | Regular expression. This is a new setting since V2.0. If configured, one can configure an URL, which is allowed to flag a session as authenticated. Without this setting, all URL's are allowed to manipulate the authentication state-<br><br>Default: (^/.*$) |



## 1.11 Authentication and Authorization (Location)

Context: location

| Key | Parameter | Description |
|---|---|---|
| MOD_BUT_LOGON_SERVER_URL | String | Defines its own login server URL within a "Location" directive. If set, it overwrites the GLOBAL_LOGON_SERVER_URL<br><br>Default: not specified |
| MOD_BUT_LOGON_REQUIRED | On, Off | If set to "On" within a Location directive, MOD_BUT will enforce an authenticated session<br><br>If set to "Off", the URL within the Location is open for all and authentication is not enforced.<br><br>Default: Off |

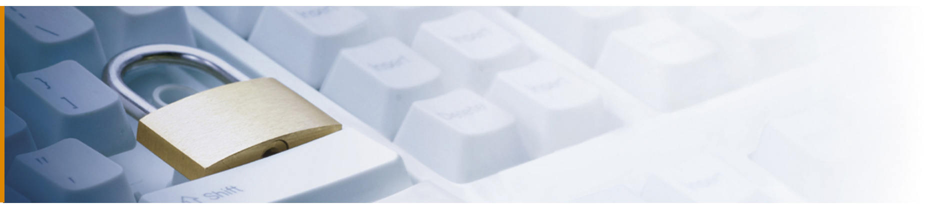
## 1.12 Service List Authorization

These are new configuration directives since version 2.0. The version 1.0 did not allow selectively access to backend systems based on the user's role after successful authentication. Once the user is authenticated, he or she is allowed to access any backend system. However – this concept is insufficient if the user of MOD_BUT exactly knows for what URL a use should have access granted and for which URL's access is denied. We are naming this concept as "service list authorization".

Since this version 2.0, MOD_BUT supports the concept of service lists. It must be turned on first and does only affect such URL's for which authentication is required. If the login application inserts a "special" service list authorization Set-Cookie header, the service list can be changed for the specific user.

Context: Virtual Host

| Key | Parameter | Description |
|---|---|---|
| MOD_BUT_SERVICE_LIST_ENABLED | On, Off | Turns the service list authorization on and off.<br><br>Default: Off |
| MOD_BUT_SERVICE_LIST_COOKIE_NAME | String | Name of the service list authorization Set-Cookie header.<br><br>Default:<br>MOD_BUT_SERVICE_LIST |



| Key | Parameter | Description |
|---|---|---|
| MOD_BUT_SERVICE_LIST_COOKIE_VALUE | String | Regular expressions for the URL's the user is authorized. This regular expression has a default value, which allowes "all URL's". If the login application sets another regular expression value, for example ("^/webapp/asl\|^/webapp/bsl), the user is only authorized for the above URL's.<br><br>Default: (^/.*$) |
| MOD_BUT_SERVICE_LIST_AUTH_ERROR_URL | String | This value configures the error page URL, if the user is authenticated but unauthorized for requesting a certain URL.<br><br>Default: /mod_but/error/authorization_error.html |

## 1.13 Authentication Level

Since Version 2.8, mod_but support the so-called authentication_strength. A typical URL has now the following configuration directive:

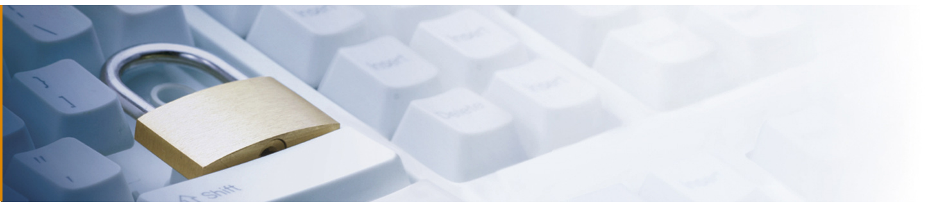
```
    <Location /mod_but/protected>
            ProxyPass
http://172.1.200.55:20060/mod_but/protected/
            ProxyPassReverse
http://172.1.200.55:20060/mod_but/protected/
            MOD_BUT_LOGON_REQUIRED  On
            MOD_BUT_AUTH_STRENGTH       1
    </Location>
```

The auth_strength belongs to the MOD_BUT session and once a new MOD_BUT session is created, the default auth_strength is set to 0. This means:

| | |
|---|---|
| o AUTH_STRENGTH=0 | anonymous access |
| o AUTH_STRENGTH=1 | username/password authentication |
| o AUTH_STRENGTH=2 | strong authentication |

If you leave the MOD_BUT_AUTH_STRENGTH from the per-directory configuration directive, the default auth_strengt=0 is applied.

If you want to authenticate users with username/password, one need to configure MOD_BUT_AUTH_STRENGTH=1 to the Location directive.



If you want to authenticate users using “strong” authentication schemes, one need to configure MOD_BUT_AUTH_STRENGTH=2 to the Location directive.

The Login Servlet is setting the AUTH_STRENGTH as part of the LOGON=ok message, after successful authentication. This auth_strength is stored to the MOD_BUT session and per any request checked, if the user has already the requiring auth_strength for the requesting url. If the user does have a lower auth_strength than the required per Location configured auth_strength, the user will be redirected via MOD_BUT_GLOBAL_LOGON_SERVER_URL_1 or MOD_BUT_GLOBAL_LOGON_SERVER_URL_2 to the setup login service.

Context: Virtual Host

| Key | Parameter | Description |
|---|---|---|
| MOD_BUT_AUTH_STRENGTH | Int | 0, 1 or 2 (other int values are discarded)<br><br>0=anonymous<br>1=username/password auth<br>2=strong auth<br><br>Default: 0 |

Compass Security AG
Glärnischstrasse 7
Postfach 1628
CH-8640 Rapperswil

T +41 55 214 41 60
F +41 55 214 41 61
team@csnc.ch
www.csnc.ch

## 1.14 Backend Application Session Handling

MOD_BUT was not aware of injecting specific cookies to specific backend applications before version 2.6. The problem firstly came up, when two backend applications were using the "jsessionid" as the backend session identifier. It was not possible to be authenticated in both applications, because the jsessionid was overwritten from one backend application to the other.

MOD_BUT introduces a new configuration directive, where one can group cookies into backend domains. Since this feature is available, one can work in two independent applications in parallel where both backend session identifier are named as "jsessionid".

Context: Virtual Host, LOCATION Directive

| Key | Parameter | Description |
|---|---|---|
| MOD_BUT_LOCATION_ID | Int | Groups backend-applications into cookie domains<br><br>Default: 0 |

```
Example of how we managed to solve our jsessionid problem within the mod_but demo
page:

    <Location /cms>
            ProxyPass                          http://127.0.0.1:36060/cms/
            ProxyPassReverse                   https://127.0.0.1:36060/cms/
            MOD_BUT_LOGON_REQUIRED  On
            MOD_BUT_LOCATION_ID           1
    </Location>

    <Location /smstool>
            ProxyPass                          http://127.0.0.1:36070/smstool/
            ProxyPassReverse                   https://127.0.0.1:36070/smstool/
            MOD_BUT_LOGON_REQUIRED  Off
            MOD_BUT_LOCATION_ID           2
    </Location>
```

If the content from /cms is delivered by an independent Tomcat servlet engine, using a sessionname like "jsessionid", this session is invisible to the URL /smstools, because the MOD_BUT_LOCATION id differs. It's like a different session scope of backend sessions.

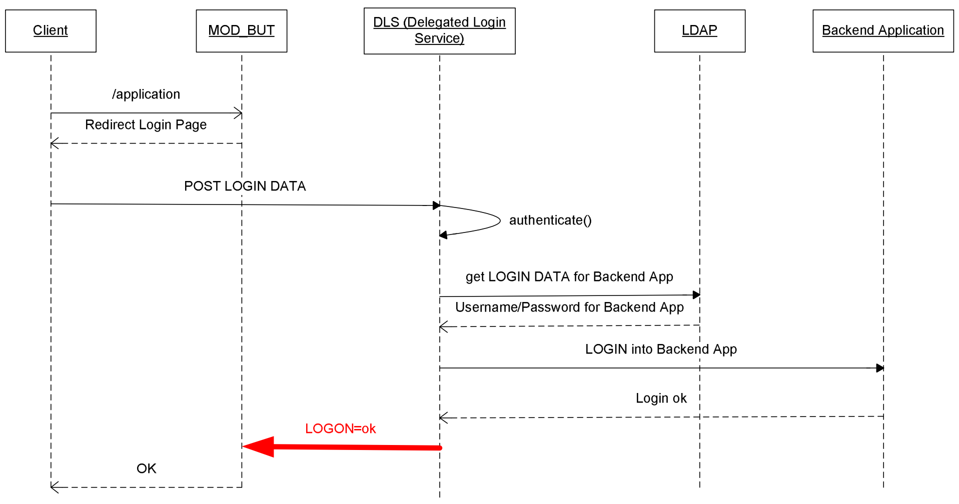
If you do not configure MOD_BUT_LOCATION_ID, the location will be configured in its default location_id = 0.

## 1.15 DLS (Delegated Login Service)

The DLS needs to set the following HTTP response headers for an successful authenticated user.

| Cookie Name | Cookie Value | Occurrence | Comment |
|---|---|---|---|
| LOGON | Ok | 1x | Key word is required if the user is successfully authenticated |
| MOD_BUT_AUTH_STRENGTH | <int> | 1x | Defines the authentication level the user has |
| MOD_BUT_SERVICE_LIST | String (regexp) | 1x | Defines the URL's, for which the user is authorized. |
| MOD_BUT_BACKEND_SESSION | bname=<string> ; bvalue=<string> ; bclearance=<int> ; | Multiple time | bname = backend cookie name<br><br>bvalue= backend cookie value<br><br>bclearance = backend LOCATION_ID clearance |

bclearance defines the MOD_BUT_LOCATION_ID's, for which the cookie is stored. Without DLS, the LOCATION_ID can be gathered from the backend url. But with DLS, MOD_BUT needs somehow be informed about the LOCATION_ID a backend session is configured. That is where the bclearance comes in.



Compass Security AG
Glärnischstrasse 7
Postfach 1628
CH-8640 Rapperswil

T +41 55 214 41 60
F +41 55 214 41 61
team@csnc.ch
www.csnc.ch