

Apache 2.x Module

MOD_BUT

November 21, 2006

Document Name:	mod_but_v2.9.doc
Version:	V 2.9
Author(s):	Ivan Buetler, Compass Security AG
Date of Delivery:	November 21, 2006
Classification:	PUBLIC

Table of Contents

1 INTRODUCTION.....	4
1.1 Preamble	4
1.2 Goals of this Project	4
1.3 About the Author	5
1.4 Web Application Security Lab	5
1.5 Solution Provider for Commercial Entry Servers	6
1.6 Version Control	6
 2 MOD_BUT APACHE 2.X MODULE.....	 8
2.1 Introduction	8
2.2 Demo Application	10
2.3 Test Cases	12
2.3.1 Cookie Support	13
2.3.2 Login and Logout Test Cases	14
2.3.3 Cookie Store	17
2.3.4 Service Authorization	22
2.3.5 Backend Cookie Domains	23
2.3.6 Authentication Strength	26
2.3.7 Single Sign On	28
2.3.8 Important URLs	32
2.4 UML Sequence Diagram	33
2.5 Compatibility	34
2.6 Limitations	34
2.7 Mailing List	35
2.8 Browsers with disabled Cookie Support	35
 3 MOD_BUT CONFIGURATION DIRECTIVES.....	 36
3.1 List of Configuration Directives	36
3.2 Enable/Disable	37
3.3 Cookie Test	37
3.4 Cookie Settings	37
3.5 Session Settings	38
3.6 Shared Memory Settings	40
3.7 Global Authentication and Authorization	41
3.8 Authentication and Authorization (Location)	43
3.9 Service List Authorization	43
3.10 Authentication Level	44
3.11 Backend Application Session Handling	46
3.12 DLS (Delegated Login Service)	47
 4 INSTALLING MOD_BUT	 48
4.1 Introduction	48
4.2 Preparation	48
4.3 Disable “TRACE” in the Apache Source Code	49
4.4 Hide Banner Info in the Apache Source Code	51
4.5 Enable mod_replace	51
4.6 Compile your Reverse proxy (without MOD_BUT)	52



4.7 Install and Test	52
4.8 Compilation MOD_BUT with Unix	53
4.9 Compilation MOD_BUT with Microsoft Windows	55
4.10 Sample Configuration MOD_BUT	56

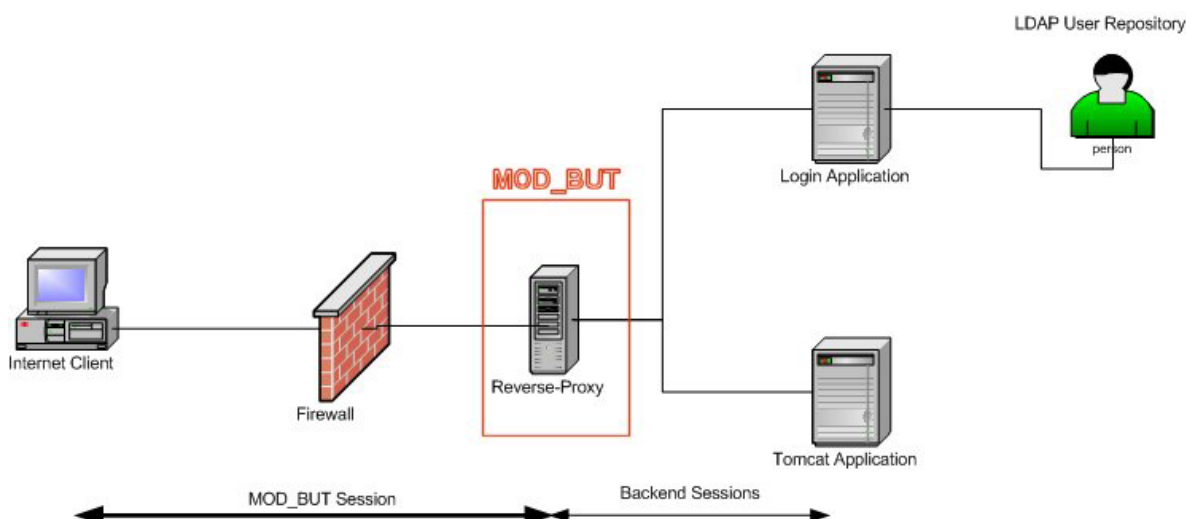
1 Introduction

1.1 Preamble

Apache has been the most popular web server on the Internet since April of 1996. The February 2005 Netcraft Web Server Survey¹ 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.

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

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 “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 About the Author

Ivan Buetler co-founded Compass Security AG² in February 1999 where he works as a Security Analyst and Managing Director. Compass Security is active in the domains of security assessments and forensic investigations and is located in Rapperswil Switzerland. Additionally, Ivan teaches at the University of Applied Sciences Rapperswil³ and the HSW Lucerne⁴. He has published several security reports in the security field.

1.4 Web Application Security Lab

The strength of Compass Security is its web application-training infrastructure. This enables our customers and trainees to learn about web risks both theoretically and with practical exercises. The Application Security Lab⁵ course deals with the security of Web applications. During the three-day course, the trainee dives into the world of browsers, cookies, sessions and secure coding.

Compass Security has incorporated the most common web developer mistakes into its training web applications. These applications are the target of lab exercises which deal with authentication, encryption, session handling, input validation, secure coding, design decisions, database access and authorization aspects. The new MOD_BUT will bring entry server knowledge to this training.

² Compass Security Network Computing AG: <http://www.csnc.ch>

³ HSR Hochschule für Technik Rapperswil: <http://www.hsr.ch/>

⁴ HSW Hochschule für Wirtschaft Luzern: <http://www.hsw.fhz.ch/>

⁵ Application Security Lab: <http://www.csnc.ch/estatic/services/training/asl.html>

1.5 Solution Provider for Commercial Entry Servers

There are several commercial entry server products available.

- Seclutions AirLock⁶
- AdNovum Nevis Web⁷
- tetrade secure entry server⁸
- IBM Tivoli Access Manager (a.k.a. WebSEAL)⁹

Owing to the early development stage of MOD_BUT we recommend evaluating one of the above products, if you plan to use an entry server.

1.6 Version Control

Version	Date	Changes	Author
1.0	18.11.2005	Initial Version Register module in modules.apache.org Publication via Compass Security	Ivan Buetler
1.1	18.01.2006	Reduce default SHM size in mod_but.h Delete mod_but_input_filter.c from source	Ivan Buetler
2.0	28.02.2006	Support for Authorization URL Compiles with Microsoft Windows	Ivan Buetler
2.1	03.03.2006	Add description how to compile mod_but with Microsoft Windows	Ivan Buetler
2.2	01.10.2006	Change MOD_BUT_LOGON_REQUIRED from yes/no to On/Off	Ivan Buetler
2.5	03.10.2006	Delete cookie from cookie store implementation if a Set-Cookie: jsessionid=deleted is found. Deletion of cookies from the cookiestore was not supported before.	Ivan Buetler
2.6	05.10.2006	Send Cookies to URL's where they were set only (instead of sending the cookies to all backend sessions)	Ivan Buetler
2.7	09.10.2006	Bugfix Backend Cookie Delivery (cookies delimiter was set to “,” instead of “;”)	Ivan Buetler

⁶ Seclutions AirLock - Application Security Gateway: http://www.seclutions.ch/en/ct_products_en.htm

⁷ Adnovum Nevis Web: <http://www.adnovum.ch/sol/nevis.html>

⁸ tetrade secure entry server: <http://www.tetrad.ch/tt/de/competence/si.html>

⁹ IBM Tivoli Access Manager for e-Business: <http://www-306.ibm.com/software/tivoli/products/access-mgr-e-bus/>

Version	Date	Changes	Author
2.8	17.10.2006	AUTH_STRENGTH is now available. This control helps to manage the required authentication strength for a given URI. The auth_strength is important if you have pages requiring username/password authentication (AUTH_STRENGTH=1) or strong authenticated pages (AUTH_STRENGTH=2)	Ivan Buetler
2.9	21.11.2006	Support for SSO. A so-called DLS (delegated login service) automatically authenticates the user into its backend applications. By special notifications, the cookie based sessions from the real backend applications can be injected into the real shared memory store via MOD_BUT_BACKEND_SESSION cookie.	Ivan Buetler

2 MOD_BUT Apache 2.x Module

2.1 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)
- HttpOnly¹⁰ 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”
- 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.

¹⁰ Mitigating Cross-site Scripting With HTTP-only Cookies:
http://msdn.microsoft.com/workshop/author/dhtml/httponly_cookies.asp

- 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:

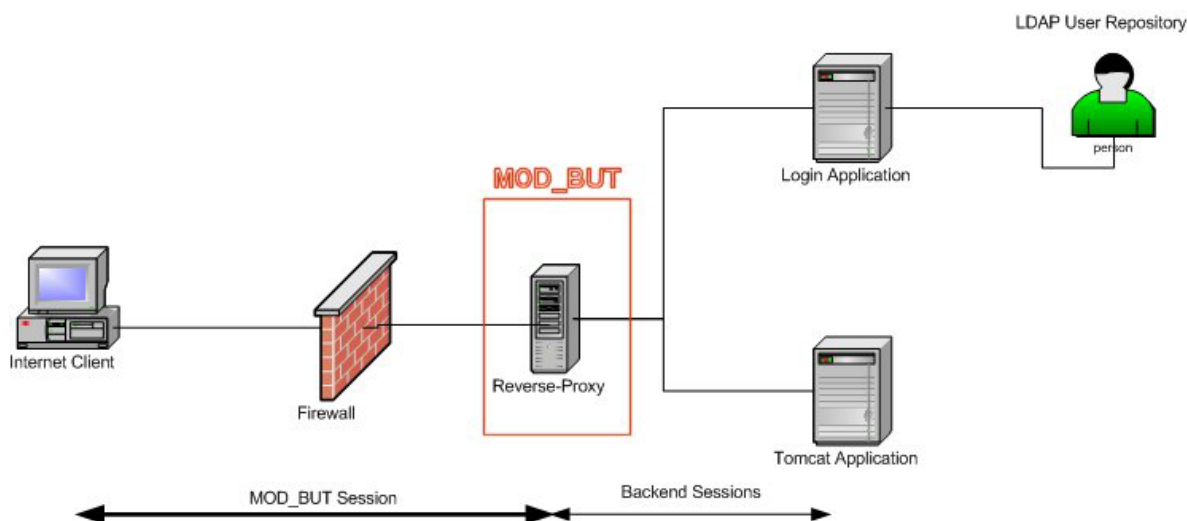
- MOD_BUT is not compatible with Apache 1.x web servers.
- Request filtering is not implemented by MOD_BUT. We recommend using mod_security instead.
- MOD_BUT only handles cookie-based sessions.
- MOD_BUT does not have nice configuration interfaces like Seclutions Airlock. Configuration is achieved through 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.

2.2 Demo Application

There is an Internet demonstration application available at http://www.but.ch/mod_but/. It will help you fully understanding the functionality of MOD_BUT. However – before really testing MOD_BUT, we recommend reading through this chapter first, which describes the demonstration setup and its architecture. Especially when you step through the test cases it is important to know, in what setup and infrastructure you are.



The demo application consists of the following components:

- Firewall
- Apache 2.0.59 Web Server with enabled MOD_BUT. This component is configured as reverse proxy. It does nothing more than forwarding requests to backend systems.
- Tomcat Test Servlet
 - Login Application
 - EchoRequest Servlet – this implements the test backend system
 - SetCookie Servlet – this implements the test backend system
- OpenLDAP user repository

MOD_BUT is designed as an entry server and uses its own session (MOD_BUT session) between the client computer and itself. If a backend application (in our case the demonstration servlets) wants to set a cookie, it will not be wired to the client by default. The cookie (in our case we regard the cookie as a backend system session) will be stored in the user's cookie store shared memory segment on the reverse proxy.

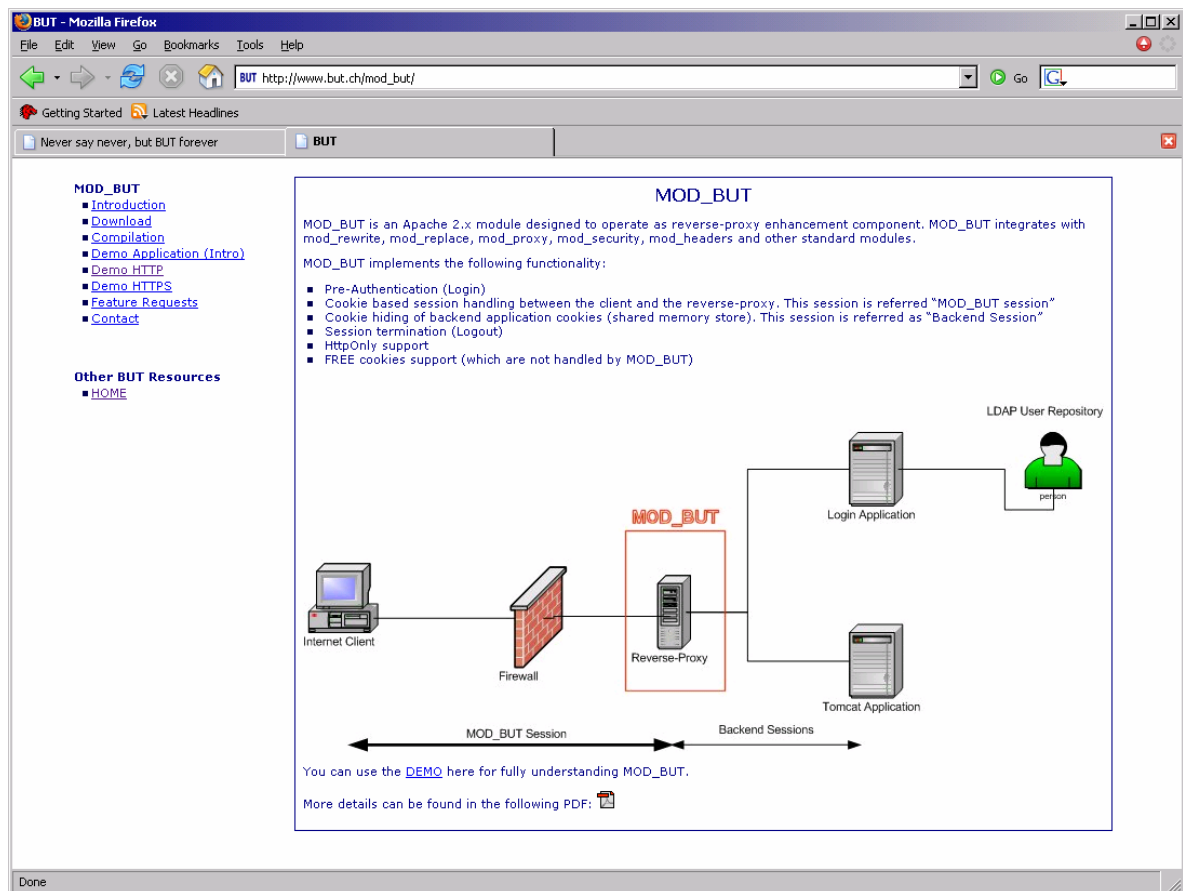
Later, while the user requests a backend server URL, MOD_BUT will first verify the existence and validity of the MOD_BUT session and if it is correct, lookup further backend system cookies from its cookie store and insert them into the final mod_proxy backend HTTP request header.

Additionally, MOD_BUT checks whether the user is already authenticated or not. If the requesting URL requires authentication and the user is not authenticated yet, the user will be redirected to a

login page. If provided login credentials match, the Login Application will set a special cookie telling MOD_BUT to flag the user's session as authenticated.

The demonstration application stores user credentials within the OpenLDAP database. But this is independent of MOD_BUT and not really interesting in this demo application. Information exchange between the Login Application and MOD_BUT is implemented via HTTP headers.

The demo suite has the following look & feel:



2.3 Test Cases

This chapter introduces some test cases that can be run from your local computer using a cookie-enabled browser. These test cases will help you fully understand MOD_BUT and its pros and cons.

The demo is available via HTTP. The demonstration page will give you the chance snooping network packets to verify the traffic content. We recommend *Wireshark* or *Ethereal*¹¹ as a network sniffer. It is available for all major UNIX derivatives and Microsoft Windows systems. Another valuable client side analyzing tool is *Paros*¹². It acts as client proxy and you get the chance to view, modify request and responses between your client and MOD_BUT. Please refer to the *Ethereal* and *Paros* page if you plan using this software. Another nice Firefox plug-in is called *LiveHTTPHeaders*¹³. It opens another Firefox window containing request header and response header information. Please use the *CookieCutter* Firefox plug-in, if you want to change cookie values in real-time.

The demo application requires authentication! Please use the following user credentials while testing MOD_BUT:

Username = **mod_but**
Password = **demo**

Now – we wish you happy testing using the test cases below...

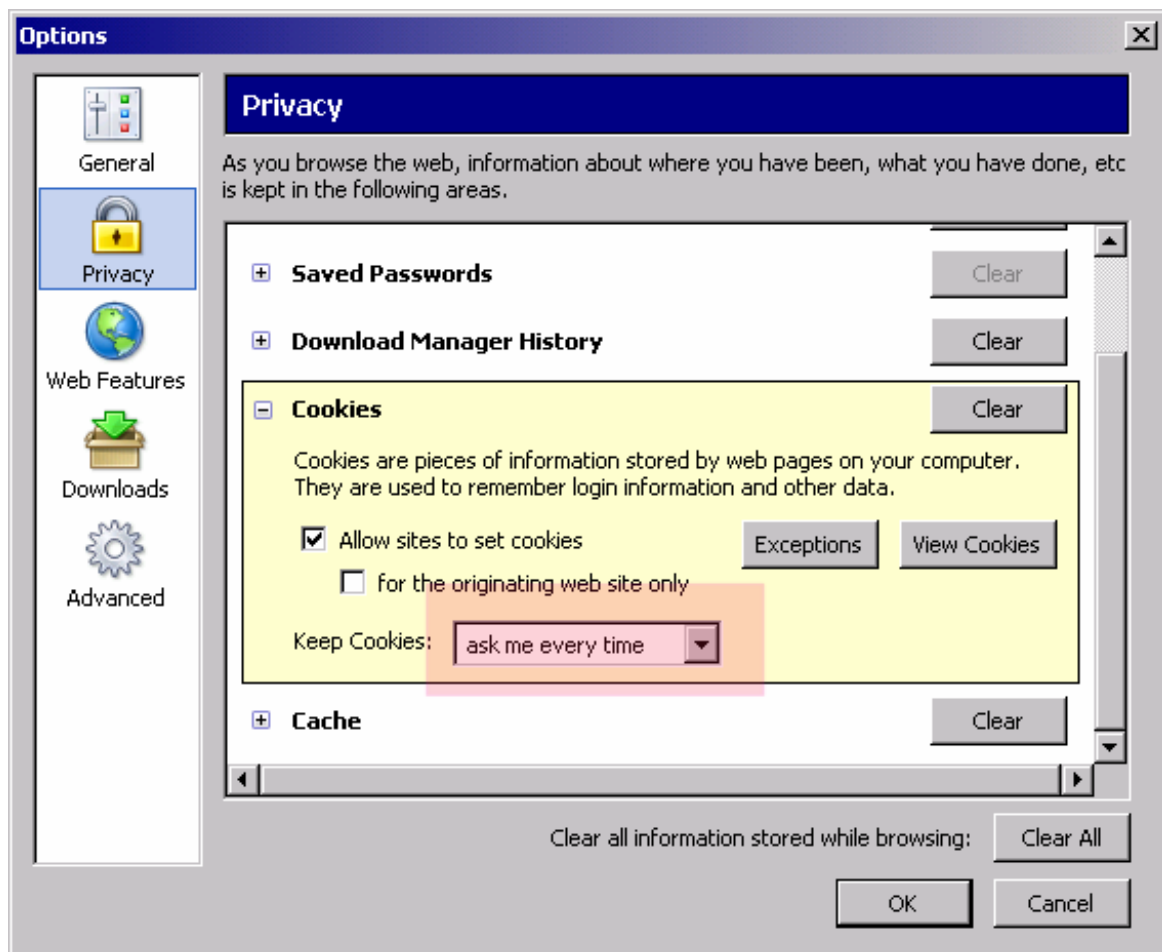
¹¹ *Ethereal*: <http://www.ethereal.com/>

¹² *Paros Proxy*: <http://www.parosproxy.org/>

¹³ *LiveHTTPHeaders*: <http://livehttpheaders.mozdev.org/installation.html>

2.3.1 Cookie Support

Goal	Browser must have cookie support enabled
Preparation	Configure your browser – Ask me every time



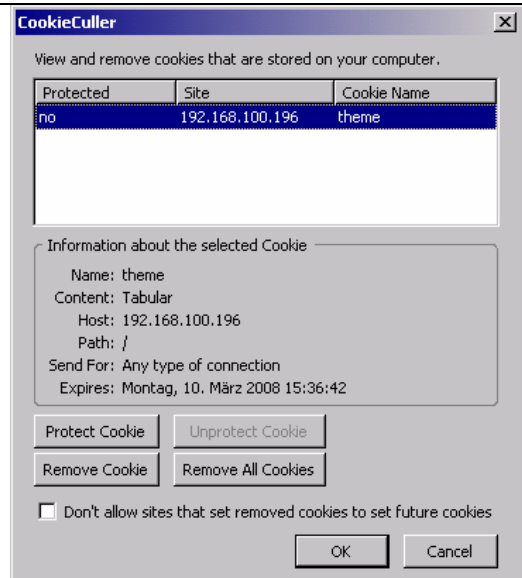
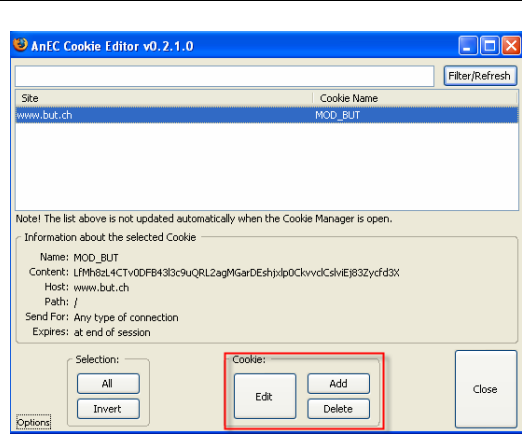
No	Description of Test	Expected Result	Actual Result	PASS FAIL
1	Go to the following URL and do not accept the set cookie for several times http://www.but.ch/mod_but/protected/	After 3 cookie tries, the browser is redirected to an "information" page. Cookies are required for MOD_BUT to run properly.		

2.3.2 Login and Logout Test Cases

Goal	Login and Logout
Preparation	Internet

First of all, let us test the login and logout behavior. Go through the test cases below and verify if our expected result meets your own test results. If they diverge, we will gladly receive suggestions for improvement. We recommend enabling the option “ask cookies before accepting” within your browser.

The Firefox CookieCuller¹⁴ plug-in will help you doing the next test cases:

	<p>CookieCuller plug-in helps you in removing cookies.</p>
	<p>Download the <i>Add N Edit Cookie Plug-in</i>¹⁵ for Firefox, which enables the client to manipulate cookie attributes within the client's browser instance.</p>

¹⁴ We use Firefox CookieCuller Plug-In for cookie deletion tasks. You can close/open your browser instead.

¹⁵ Add & Edit Cookies - Cookies Editor: <http://addneditcookies.mozdev.org/>

No	Description of Test	Expected Result	Actual Result (Please fill in here your test results)
1	<p>a) Delete all cookies from your browser¹⁶.</p> <p>b) Go to the following URL (this is a protected page)</p> <p>http://www.but.ch/mod_but/protected/index.html</p> <p>c) Authenticate in the login form correctly using the given credentials.</p>	<p>b) Redirect to login page</p> <p>c) Redirect to /protected/index.html</p> <p>After c) you should be logged into MOD_BUT.</p>	
2	<p>a) Delete all cookies from your browser.</p> <p>b) Go to the following URL and authenticate correctly</p> <p>http://www.but.ch/mod_but/protected/index.html</p> <p>c) Go to the following URL</p> <p>http://www.but.ch/logout/</p>	<p>c) You are logged out.</p> <p>Press "Browser BACK" button. You should not be able to "see" the content you saw before the logout. You are permanently logged out.</p>	
3	<p>a) Make sure you have done step 2) properly and you have clicked the logout button.</p> <p>b) Go to the following URL</p> <p>http://www.but.ch/mod_but/protected/index.html</p>	<p>You are logged out.</p> <p>This means, that MOD_BUT really destroys its session. You can click on "renew" session if desired.</p>	
4	<p>Download the <i>Add N Edit Cookie</i> Plug-in¹⁷ for Firefox, which enables the client to manipulate cookie attributes within the client's browser instance.</p> <p>a) Go to the following URL and authenticate correctly</p> <p>http://www.but.ch/mod_but/protected/index.html</p> <p>b) Modify the cookie value of the MOD_BUT session with the CookieCuller plug-in to "another" value. Just add/remove something to the cookie value.</p> <p>c) Type Refresh Button after you have saved b).</p>	<p>You are redirected to a "hacker attempting" web page.</p> <p>MOD_BUT thinks you are a hacker, because you have sent an unknown MOD_BUT session and this is evaluated as hacking attack.</p>	

¹⁶ We use Firefox CookieCuller Plug-In for cookie deletion tasks. You can close/open your browser instead.

¹⁷ Add & Edit Cookies - Cookies Editor: <http://addneditcookies.mozdev.org/>

No	Description of Test	Expected Result	Actual Result (Please fill in here your test results)
5	<p>a) Go to the following URL and authenticate correctly</p> <p>http://www.but.ch/mod_but/protected/index.html</p> <p>b) Open the <i>Add N Edit Cookie</i> Plug-in and remember the session value. Copy the session value into a temporary text file.</p> <p>c) Logout</p> <p>d) Close the browser or delete all cookies.</p> <p>e) Go to the following URL and authenticate correctly</p> <p>http://www.but.ch/mod_but/protected/index.html</p> <p>f) Use the <i>Add N Edit Cookie</i> Plug-in and replace the current session value with the session you have remembered previously in step b).</p>	<p>You are redirected to a page where you get informed that you are using a “history” session.</p> <p>A history session is a session used earlier in MOD_BUT. This behavior will not send a “hacking attempt” page to the client, as we did in step 5</p> <p>Used MOD_BUT sessions are held within a shared memory history store for a configurable duration. The current configuration keeps such history sessions for 8 hours.</p>	
6	<p>a) Go to the following URL and authenticate correctly</p> <p>http://www.but.ch/mod_but/protected/index.html</p> <p>b) Logout using the logout button.</p> <p>c) Press the browser's back-button several times. Vulnerable login services offer a “resend POST data again”. But because MOD_BUT uses redirects after successful authentication, MOD_BUT is not vulnerable to such attacks.</p>	<p>You should not see a message “resend POST data” at any time.</p> <p>It would mean that your login service is vulnerable to a “post data resend attack”.</p>	

2.3.3 Cookie Store

Goal	Session Handling
Preparation	Internet

MOD_BUT offers a session store, which holds back backend-session within the shared memory and hides them from the client. Backend sessions are referenced via the MOD_BUT session and are added automatically to the HTTP request headers between the reverse proxy and the backend application or are removed from the HTTP response headers when inserted by a backend application.

MOD_BUT supports “free” cookies. These cookies are not handled by MOD_BUT but transparently wired between the client and the backend system. Such cookies are not held within the shared memory segment. The demonstration application has two free cookies configured. They are named as *trustme* and *language*.

No	Description of Test	Expected Result	Actual Result (please fill in here your test results)
7	<p>a) Go to the following URL and authenticate correctly</p> <p>http://www.but.ch/webapp/but/SetCookie</p> <p>b) You should see a form where you can enter cookie names, values and path properties. Now you can play around with the SetCookie Servlet. This Servlet gives you the chance to set cookies in exactly the same manner as BEA WebLogic, IBM WebSphere or any other J2EE containers. We recommend trying it without a specific “expected result”. Just play around with the SetCookie Servlet. Go to the following Servlet, if you want to see all request headers sent from MOD_BUT to the backend application</p> <p>http://www.but.ch/webapp/but/EchoRequest</p>	N/A	

No	Description of Test	Expected Result	Actual Result (please fill in here your test results)
8	<p>After you have played with previous test case #7, we continue with specific test cases. They should be more understandable if you have previously tried test case #7!</p> <p>a) Go to the following URL and authenticate correctly http://www.but.ch/webapp/but/SetCookie</p> <p>b) Configure a cookie with Cookie name = Cool4You Cookie value = 12345 Path = /</p> <p>Click submit button</p> <p>c) Check your browser's stored cookies. (Use CookieCuller you have installed in test case #4).</p> <p>d) Go to the EchoRequest Servlet http://www.but.ch/webapp/but/EchoRequest</p>	<p>b) You should not see a <i>Cool4You</i> cookie within your browser</p> <p>c) The <i>Cool4You</i> cookie should be sent to the backend system. The cookie was held within the shared memory segment and automatically inserted to your d)-request header.</p>	
9	<p>a) Make sure you performed test case #8 and you have not deleted cookies afterwards.</p> <p>b) Go to the following URL and authenticate correctly http://www.but.ch/webapp/but/SetCookie</p> <p>c) Configure a cookie with Cookie name = Cool4You Cookie value = abcdefg Path = /</p> <p>Click submit button</p> <p>d) Check your browser's stored cookies. (Use CookieCuller you have installed in test case #4).</p> <p>d) Go to the EchoRequest Servlet http://www.but.ch/webapp/but/EchoRequest</p>	<p>c) You should not see a <i>Cool4You</i> cookie within your browser</p> <p>d) MOD_BUT has detected that you want to change (update) the cookie value from test case #8. The EchoRequest Servlet should show you an updated <i>Cool4You</i> cookie value. In this test case, the value should be abcdefg.</p>	

No	Description of Test	Expected Result	Actual Result (please fill in here your test results)
10	<p>a) Make sure you performed test case #9 and you have not deleted cookies afterwards.</p> <p>b) Go to the following URL and authenticate correctly</p> <p>http://www.but.ch/webapp/but/SetCookie</p> <p>c) Configure a first cookie with</p> <p>Cookie name = MyCookie Cookie value = foobar Path = /</p> <p>and a second cookie..</p> <p>Cookie name = Cool4You Cookie value = 7777777 Path = /</p> <p>Click the submit button.</p> <p>d) Check your browser's stored cookies. (Use CookieCuller you have installed in test case #4).</p> <p>e) Go to the EchoRequest Servlet</p> <p>http://www.but.ch/webapp/but/EchoRequest</p>	<p>d) You should still not see application cookies; only the MOD_BUT session should be visible.</p> <p>e) You should see a new cookie <i>MyCookie</i> and an updated <i>Cool4You</i> cookie.</p>	

No	Description of Test	Expected Result	Actual Result (please fill in here your test results)
11	<p>a) Make sure you performed test case #9 and you have not delete cookies afterwards.</p> <p>b) Go to the following URL and authenticate correctly</p> <p>http://www.but.ch/webapp/but/SetCookie</p> <p>c) Configure a first cookie with</p> <p>Cookie name = Cool4You Cookie value = 998899 Path = /</p> <p>and a second cookie..</p> <p>Cookie name = AnotherCookie Cookie value = qwertz Path = /</p> <p>Click the submit button.</p> <p>d) Check your browser's stored cookies. (Use CookieCuller you have installed in test case #4).</p> <p>e) Go to the EchoRequest Servlet</p> <p>http://www.but.ch/webapp/but/EchoRequest</p>	<p>d) You should still not see application cookies, only the MOD_BUT session</p> <p>e) You should see a new cookie <i>AnotherCookie</i> and an updated <i>Cool4You</i> cookie.</p> <p>This test is very similar to the test case #10. But we want to make sure MOD_BUT works independent from the set-cookie order. If you performed test case #10 and #11 with the expected result you have proof that MOD_BUT handles known and new cookies correctly. This is important, because if MOD_BUT receives a change request for a known cookie, everything needs to be newly assigned within the shared memory.</p>	

No	Description of Test	Expected Result	Actual Result (please fill in here your test results)
12	<p>Now it is time to start with a clean setup. Please remove all cookies or close and start again your browser.</p> <p>This test verifies the “free” cookies support. These are cookies which are not held within MOD_BUT but are transparently wired between the client and the backend system.</p> <p>a) Go to the following URL and authenticate correctly</p> <p>http://www.but.ch/webapp/but/SetCookie</p> <p>b) Configure a cookie with</p> <p>Cookie name = trustme Cookie value = 12345 Path = /</p> <p>Click the submit button</p> <p>c) Check your browser's stored cookies. (Use CookieCuller you have installed in test case #4).</p> <p>d) Go to the EchoRequest Servlet</p> <p>http://www.but.ch/webapp/but/EchoRequest</p>	<p>b) You should see the cookie <i>trustme</i> within your browser's cookie store. Test this via CookieCuller plug-in.</p> <p>This means, that MOD_BUT did not store the <i>trustme</i> cookie to your session store. Instead, it is sent to your browser.</p> <p>d) Check with the EchoRequest Servlet if your <i>trustme</i> cookie is wired to the backend system. The <i>trustme</i> cookie should be seen as output of the EchoRequest.</p>	
13	Find new test cases by yourself. Try combinations and remember MOD_BUT handles <i>trustme</i> and <i>language</i> cookie names as “free”.	N/A	
14	<p>How does MOD_BUT handle unexpected cookies? Download the <i>Add N Edit Cookie Plug-in</i>¹⁸ for Firefox, which enables the client to manipulate cookie attributes within the client's browser instance.</p> <p>Go to the demo application, authenticate correctly and set some cookies of your choice.</p> <p>Now add another cookie name with the Cookie Editor for Firefox and go to the EchoRequest Servlet afterwards. If you do so, MOD_BUT receives an unexpected cookie from the outside and shall not wire the unexpected cookie to the backend system.</p>	<p>Your self-defined “unexpected” cookie shall not be seen within the EchoRequest Servlet. Only the cookies declared to MOD_BUT shall be seen.</p> <p>These are:</p> <p>a) MOD_BUT session b) Optionally: <i>trustme</i> cookie c) Optionally: <i>language</i> cookie d) Optionally: your self defined cookies from the SetCookie Servlet</p>	

¹⁸ Add & Edit Cookies - Cookies Editor: <http://addneditcookies.mozdev.org/>

2.3.4 Service Authorization

Goal	Test the service authorization. If a user authenticates at mod_but, he is authorized to access all backend sessions. By sending a special cookie (this is normally done by the login application), the user is allowed to request a sub-set of all backend applications but denied of requesting all backend-systems.
Preparation	Internet

No	Description of Test	Expected Result	Actual Result (please fill in here your test results)
15	<p>a) Delete all cookies from your browser.</p> <p>b) Go to the following URL and authenticate correctly</p> <p>http://www.but.ch/webapp/but/SetCookie</p> <p>c) Configure two cookies:</p> <p>First Cookie name = LOGON Cookie value = ok</p> <p>Second Cookie name = MOD_BUT_SERVICE_LIST Cookie value = (^/webapp/but/SetCookie)</p> <p>Click the submit button. The cookie above re-configures the default service authorization and you are only allowed requesting the above SetCookie servlet.</p> <p>d) Go to the EchoRequest Servlet</p> <p>http://www.but.ch/webapp/but/EchoRequest</p> <p>You should not be allowed to see something. Do not click on the "renew" button here.</p> <p>e) Click again on the SetCookie Servlet</p> <p>http://www.but.ch/webapp/but/SetCookie</p> <p>You should be allowed for the above SetCookie URI, because it is part of the regexp expression.</p>	<p>d) You are not allowed to see the EchoRequest Servlet any more. The only URL for which you are authorized includes /webapp/but/SetCookie</p> <p>e) You are allowed to see the SetCookie Servlet (because it is an authorized url now)</p>	

2.3.5 Backend Cookie Domains

Goal	Test session store, if cookies in the backend use the name cookie name (e.g. jsessionid)
Preparation	Internet

MOD_BUT was not working properly for a client, because /webmail and /freesms are both using a cookie name "jsessionid" as session identifier.

<http://somehost/webmail>

<http://somehost/freesms>

Once authenticated at both backend applications concurrently, one jsessionid cookie value was overwriting the other's jsessionid value all the time while using the applications.

This problem was solved by the introduction of the so-called **MOD_BUT_LOCATION_ID**. It will enable MOD_BUT using backend cookies with the same name properly.

Example httpd.conf for the test cases below:

```
<Location /webapp/one/>
    ProxyPass                http://127.0.0.1:9090/
    ProxyPassReverse         http://127.0.0.1/webapp/one/
    MOD_BUT_LOGON_REQUIRED   On
    MOD_BUT_LOCATION_ID      1
</Location>

<Location /webapp/two/>
    ProxyPass                http://127.0.0.1:8080/
    ProxyPassReverse         http://127.0.0.1/webapp/two/
    MOD_BUT_LOGON_REQUIRED   On
    MOD_BUT_LOCATION_ID      2
</Location>
```

As you can see above, the url /webapp/one uses a different MOD_BUT_LOCATION_ID than /webapp/two. If MOD_BUT_LOCATION_ID is not set, it will internally set to 0.

Page: 24 Date: Nov 21, 2006

No	Description of Test	Expected Result	Actual Result (please fill in here your test results)
17	<p>Please perform test case 16, but without closing and opening your browser. Keep your session information.</p> <p>a) Please go to the following SetCookie servlet http://www.but.ch/webapp/two/SetCookie</p> <p>Please note: the URL above uses /webapp/two/</p> <p>b) Please set a cookie CookieName = jsessionid Cookievalue = two-two</p> <p>c) Please click on the URL below (/webapp/two) http://www.but.ch/webapp/two/EchoRequest</p> <p>d) Please click on the URL below (/webapp/one) http://www.but.ch/webapp/one/EchoRequest</p>	<p>b) You have configured a second cookie with the cookienname = jsessionid, but for another URL.</p> <p>c) You should see jsessionid=two-two. (because it was set for the URL /webapp/one)</p> <p>d) You should see jsessionid=one-one (because it was set for the URL /webapp/one)</p> <p>If this test case works, you should be convinced, that MOD_BUT can handle identical cookie names concurrently, if MOD_BUT_LOCATION_ID is different for the URL's in question. Groups of applications can share the same LOCATION_ID.</p>	

2.3.6 Authentication Strength

Goal	Testing SSO setup a) If user not authenticated and he requests a username/password protected page b) If user is username/password authenticated, an he requests a strong auth protected page
Preparation	Internet

This test case introduces anonymous access, username/password access and strong authenticated user access.

Assume we have a portal, certain url's are accessible anonymously. Certain pages are protected by username and password combinations and the last class of pages is only accessible for strong authenticated users.

What happens, if the user requests the username/password protected page and later, during its web session, he is requesting another ULR for which strong authentication is configured. In such a scenario, the user needs to login again (step-up) from username/password level to strong authentication level. These levels are referred as MOD_BUT_AUTH_STRENGTH.

Authentication Level	MOD_BUT_AUTH_STRENGTH
Anonymous access	0
Username/Password access	1
Strong authenticated users	2

The minimum required MOD_BUT_AUTH_STRENGTH is configurable via httpd.conf per LOCATION base. After successful authentication, the login app needs to tell MOD_BUT, if the user is now authenticated with username/password or with a strong authentication scheme. This information is stored to the users shared memory session.

For any further request, MOD_BUT will check, if the users AUTH_STRENGTH level enables him for requesting the desired URL. If the per httpd.conf configured AUTH_STRENGTH is higher than the sessions AUTH_STRENGTH, the user must re-authenticate by the higher authentication scheme.

The following configuration directive is used for this test case:

```
<Location /webapp/one>
    ProxyPass http://172.1.200.55:35080/but/
    ProxyPassReverse http://172.1.200.55:35080/one/
    MOD_BUT_LOGON_REQUIRED On
    MOD_BUT_AUTH_STRENGTH 1
    MOD_BUT_LOCATION_ID 1
</Location>

<Location /webapp/strong>
    ProxyPass http://172.1.200.55:35080/but/
    ProxyPassReverse http://172.1.200.55:35080/strong/
    MOD_BUT_LOGON_REQUIRED On
    MOD_BUT_AUTH_STRENGTH 2
    MOD_BUT_LOCATION_ID 5
</Location>
```

No	Description of Test	Expected Result	Actual Result (please fill in here your test results)
18	<p>a) Close all your cookies from the browser. Please use the following URL and authenticate using the correct username/password values</p> <p>http://www.but.ch/webapp/one/SetCookie</p> <p>b) Please click on the URL below again (this URL requires AUTH_LEVEL=2 (and authenticate using the correct username/password values)</p> <p>http://www.but.ch/webapp/strong/EchoRequest</p> <p>c) Please click on the URL below again (this URL requires AUTH_LEVEL=2</p> <p>http://www.but.ch/webapp/strong/EchoRequest</p>	<p>a) You have sufficient AUTH_LEVEL for this URL (because username/password level is required)</p> <p>b) You have insufficient AUTH_LEVEL for this URL. You need to step-up to strong AUTH_LEVEL.</p> <p>c) You have sufficient AUTH_LEVEL privileges for accessing the /webapp/strong URL.</p>	

Remarks:

The step-up login authenticates with username/password too – but think open and just imagine, a username/password/secuID form would have been shown. More important; you need to understand, the login service will inform MOD_BUT via http response headers about the current AUTH_LEVEL and the servlet used in b) is just returning a cookie with MOD_BUT_AUTH_STRENGTH=2.

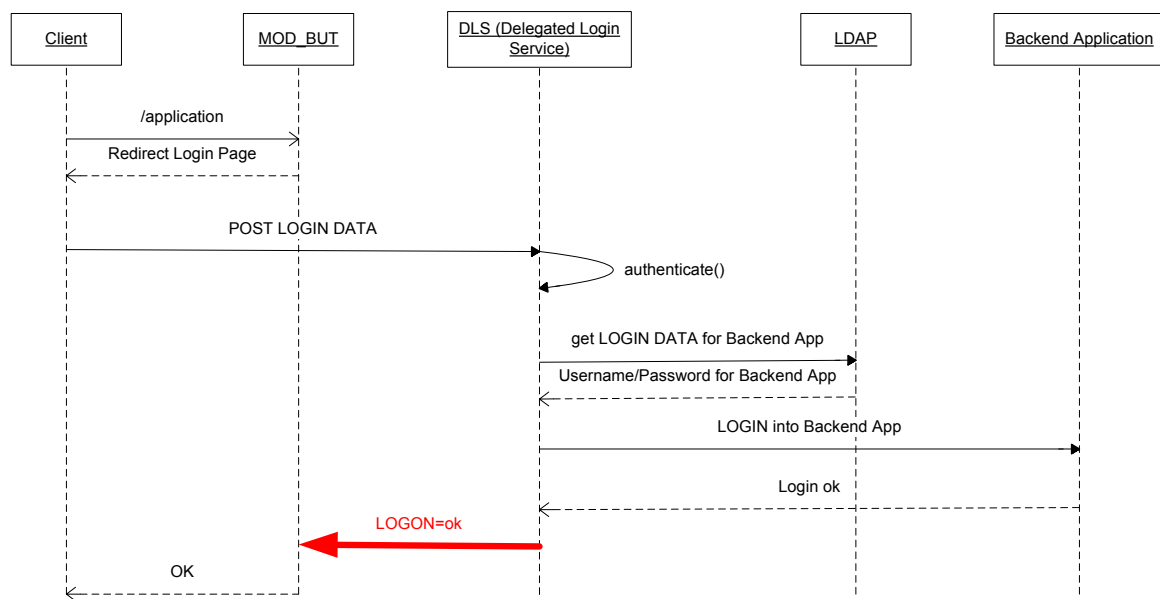
One can write his/her own login servlet, checking against a valid client certificate or any other strong scheme and if the user can prove its identity, it will only inform MOD_BUT via MOD_BUT_AUTH_STRENGTH.

2.3.7 Single Sign On

Goal	Testing DLS (delegated login service)
Preparation	Internet

The DLS is referred as delegated login service, an automatic login workbench which enables Single Sign On. If successfully authenticated at MOD_BUT, the DLS will silently authenticate users into the user's backend applications and set backend session cookies into the user's session store.

By using this technique, one is not urged for multiple authentications. The DLS will do it invisible in the backend.



LOGON=ok

The LOGON=ok message, sent by the DLS to MOD_BUT informs MOD_BUT about an successfully authenticated user. Additionally, the DLS can (should) send additional information to MOD_BUT. Especially the reached AUTH_LEVEL and SERVICE_LEVEL authorization is required by MOD_BUT.

A full message sent from the DLS to MOD_BUT shall look like the following HTTP Response Message:

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 bvalue= backend cookie value bclearance = backend LOCATION_ID clearance

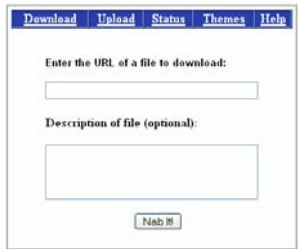
```

HTTP/1.x 200 OK
Date: Wed, 22 Nov 2006 07:49:20 GMT
Server: Apache
Last-Modified: Wed, 22 Nov 2006 07:41:13 GMT
Etag: "425b-133b3840"
Accept-Ranges: bytes
Content-Length: 16987
Cache-Control: max-age=60
Expires: Wed, 22 Nov 2006 07:50:20 GMT
Content-Type: application/xml
X-Cache: MISS from horus.csnc.ch
Proxy-Connection: keep-alive

Set-Cookie: LOGON=ok
Set-Cookie: MOD_BUT_AUTH_STRENGTH=2
Set-Cookie: MOD_BUT_SERVICE_LIST=(^/backendapp(.*))
Set-Cookie: MOD_BUT_BACKEND_SESSION=bname=JSESSIONID;bvalue=1234;bclearance=6;
Set-Cookie: MOD_BUT_USERNAME=<the Identity of the authenticated user>

```

This report shall teach you in using MOD_BUT. Therefore, the demonstration page does not include a proper DLS component. Instead, you have authenticate into the demonstration backend application without MOD_BUT (In the normal concept, this should be done by the DLS) and parallel, you are requesting the same application via MOD_BUT. Once you are successfully authenticated via browser directly (without MOD_BUT), you can plant the important cookies via Set-Cookie servlet of the demonstration page to your MOD_BUT session shared memory segment. By doing so, you are authenticated via MOD_BUT too, if you have plant all cookies correctly. This is what the next test case teaches.

No	Description of Test	Expected Result	Actual Result (please fill in here your test results)
19	<p>a) Please visit the URL below. This URL is NOT MOD_BUT protected. Use admin as username and password.</p> <p>http://80.254.178.109/filenabber/nabit</p> <p>b) Please remember the cookies from a). You can find out the cookie names and values via Firefox Live Http Header plug-in or via wireshark sniffing (because the demo uses http)</p> <p>c) Clear all cookies from a) and b). Ideally, close your browser and reopen it again to make sure, everything is clean.</p> <p>d) Now it is time to test the application via MOD_BUT. Please click on the following URL. The MOD_BUT login screen appears. Please authenticate at MOD_BUT, but not into filenabber!</p> <p>http://www.but.ch/filenabber/</p> <p>e) Visit the Set-Cookie servlet (in Firefox CTRL-T). This servlet simulates the DLS in this demonstration. You should now inject the "correct" cookies via SetCookie servlet into MOD_BUT.</p> <p>http://www.but.ch/webapp/but/SetCookie</p> <p>Please set the correct cookie values: Set-Cookie1: LOGON ok</p> <p>Set-Cookie2: MOD_BUT_BACKEND_SESSION bname=JSESSIONID;bvalue=<value from task b>;bclearance=6;</p> <p>(please enter the ";", otherwise it will not work!)</p> <p>f) Click to the following URL or refresh the first tab in Firefox, where the login screen for filenabber is.</p> <p>http://www.but.ch/filenabber/</p>	<p>a) You should be logged-in successfully, if you see something like:</p>  <p>b) Remember the filenabber session somewhere (notepad, kwrite or similar). The session name should be JSESSIONID.</p> <p>c) Clear session</p> <p>d) You should stop at the login screen of filenabber. Do not login into filenabber!</p> <p>e) After you have submit the SetCookie servlet, all filenabber cookies are set into your sessions shared memory store.</p> <p>f) If everything works as expected, you are now authenticated in filenabber too.</p>	

If you have more than one MOD_BUT protected backend applications (not provided by the demonstration page), multiple appearance of MOD_BUT_BACKEND_SESSIONS is allowed.

Cookie Name	Cookie Value
LOGON	Ok
MOD_BUT_BACKEND_SESSION	bname=<name>; bvalue=<value>; bclearance=<int>;
MOD_BUT_BACKEND_SESSION	bname=<name>; bvalue=<value>; bclearance=<int>;

Note:

MOD_BUT_BACKEND_SESSION does only have a special meaning to MOD_BUT, if the LOGON=ok cookie is found. Otherwise, MOD_BUT will handle the cookie as "normal" cookie.

2.3.8 Important URLs

There are several test cases you can perform against MOD_BUT. Use the information below for further test cases you like to perform.

- http://www.but.ch/mod_but/
- http://www.but.ch/mod_but/protected/
- <http://www.but.ch/logout/>
- <http://www.but.ch/webapp/but/EchoRequest>
- <http://www.but.ch/webapp/but/SetCookie>

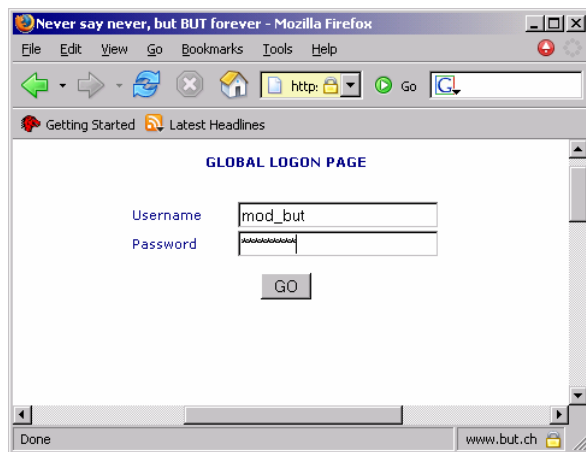
Public accessible (anonymous access)

Authentication required

Session termination URL

Authentication required

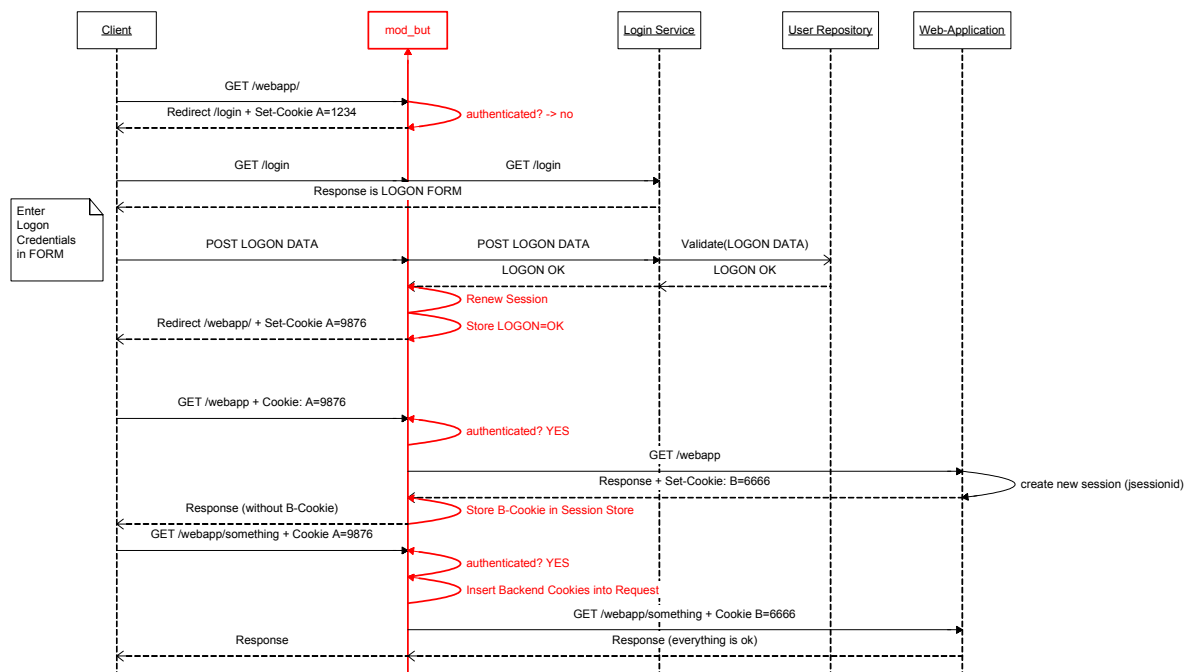
Authentication required



2.4 UML Sequence Diagram

The sequence diagram below introduces the initial login phase and backend system initialization phase.

- Client: Internet client using a browser (IE, Firefox, Mozilla, ...)
- MOD_BUT: Apache 2.x reverse proxy with enabled MOD_BUT
- Login Service: Tomcat login application (Servlet)
- User Repository: LDAP directory
- Web-Application: http://www.but.ch/mod_but/protected/index.html



MOD_BUT offers pre-authentication if configured. Only authenticated sessions are sent to the backend web-application server. MOD_BUT will check all client requests for the existence of a MOD_BUT session and does only process those client request having one. Once the client sends a MOD_BUT session and it is marked as “authenticated”, the request will be fed into mod_proxy for the final backend destination.

MOD_BUT keeps its sessions in its local shared memory segment¹⁹ (including authentication status, last requested URL, backend cookies etc). By this technique, backend sessions are hidden from the client.

The URL of the Login Service is configurable in MOD_BUT, globally or per directory level (Location directive in httpd.conf).

¹⁹ Some name the local shared memory as „Session Store“, „Cookie Store“ or “Cookie Bag”. This document refers to it as “Cookie Store”, because MOD_only handles cookie-based session handling.

2.5 Compatibility

MOD_BUT was designed to operate properly with the following standard Apache modules:

- `mod_rewrite`²⁰
- `mod_proxy`²¹
- `mod_ssl`²²

Additionally the following non-standard module is supported:

- `mod_security`²³
- `mod_replace`²⁴

MOD_BUT was successfully tested and compiled on:

- Solaris 8 / Sparc / gcc / Apache 2.0.55
- Solaris 10 / Sparc / gcc / Apache 2.2.0
- Linux 2.6.12 / Intel / gcc / Apache 2.0.55
- Microsoft Windows / VC++ / Apache 2.0.55

MOD_BUT is a new module. It was developed in Q1/Q2 2005. If you feel like sending bug reports or other comments, please this e-mail address: e1@but.ch.

2.6 Limitations

Limitation	Possible remedy; future enhancement
MOD_BUT integrates properly into a “real” reverse proxy configuration. In mixed-mode, where you deliver static content locally (at the reverse-proxy) and send other requests to backend systems, MOD_BUT will fail.	N/A
MOD_BUT does accept control cookies from any backend system.	Another improvement step can be just accepting authentication=ok messages from the login service.
MOD_BUT does not support URL-based access control. This is on the wish list too, because once you have been authenticated in MOD_BUT, you can reach all backend systems and URLs.	The improvement could be implemented by inserting a “role” to the LDAP directory and binding users to such roles. Once the user authenticates successfully, he or she has access to the URLs for that role.

²⁰ `mod_rewrite`: http://httpd.apache.org/docs/2.0/mod/mod_rewrite.html

²¹ `mod_proxy`: http://httpd.apache.org/docs/2.0/mod/mod_proxy.html

²² `mod_ssl`: http://httpd.apache.org/docs/2.0/mod/mod_ssl.html

²³ `mod_security`: <http://www.modsecurity.org/>

²⁴ `mod_replace`: <http://mod-replace.sourceforge.net/docs.html>

Limitation	Possible remedy; future enhancement
MOD_BUT does not filter requests, parameters and values.	Such functionality is usually implemented in commercial entry server software. We use mod_security instead, which is compatible with MOD_BUT.
MOD_BUT is not SSL session ID aware.	Commercial products are able to use the SSL session id as additional session handling mechanism. Potentially, the use of the SSL session id will be implemented in future versions of MOD_BUT.
MOD_BUT has not been improved for the use with client certificates.	This is potentially another update step.

2.7 Mailing List

If you want to keep track of MOD_BUT and its development, please subscribe to the following mailing list. Send an e-mail and use "subscribe" in the message subject.

Mailing List: mod_but@but.ch

2.8 Browsers with disabled Cookie Support

MOD_BUT requires the client to have cookie support enabled. **Otherwise, MOD_BUT does not work.** Therefore, MOD_BUT implements some test routines at the initialization phase. If a client denies the *Set-Cookie* header several times, the user will be redirected to an error page. This error page is configurable in httpd.conf and informs the user that he or she has to enable cookies while using the MOD_BUT protected site.

Make sure this error page is configured somewhere where MOD_BUT does not enforce the existence of a MOD_BUT session. Otherwise Apache will loop infinitely.

3 MOD_BUT Configuration Directives

3.1 List of Configuration Directives

MOD_BUT_ENABLED	Chapter 3.2	
MOD_BUT_CLIENT_REFUSES_COOKIES_URL	Chapter 3.3	
MOD_BUT_COOKIE_NAME	Chapter 3.4	
MOD_BUT_COOKIE_DOMAIN	Chapter 3.4	
MOD_BUT_COOKIE_PATH	Chapter 3.4	
MOD_BUT_COOKIE_EXPIRATION	Chapter 3.4	
MOD_BUT_COOKIE_SECURE	Chapter 3.4	
MOD_BUT_COOKIE_HTTPONLY	Chapter 3.4	
MOD_BUT_SESSION_FREE_URL	Chapter 3.5	
MOD_BUT_SESSION_TIMEOUT	Chapter 3.5	
MOD_BUT_SESSION_TIMEOUT_URL	Chapter 3.5	
MOD_BUT_SESSION_RENEW_URL	Chapter 3.5	
MOD_BUT_SESSION_INACTIVITY_TIMEOUT	Chapter 3.5	
MOD_BUT_SESSION_INACTIVITY_TIMEOUT_URL	Chapter 3.5	
MOD_BUT_SESSION_HACKING_ATTEMPT_URL	Chapter 3.5	
MOD_BUT_SESSION_TIMEOUT_HISTORY	Chapter 3.5	
MOD_BUT_SESSION_DESTROY	Chapter 3.5	
MOD_BUT_SESSION_DESTROY_URL	Chapter 3.5	
MOD_BUT_SESSION_STORE_FREE_COOKIES	Chapter 3.5	
MOD_BUT_ALL_SHM_SPACE_USED_URL	Chapter 3.6	
MOD_BUT_AUTHORIZATION_ENABLED	Chapter 3.7	
MOD_BUT_GLOBAL_LOGON_SERVER_URL	Chapter 3.7	
MOD_BUT_GLOBAL_LOGON_SERVER_URL_1	Chapter 3.7	New since Version 2.8
MOD_BUT_GLOBAL_LOGON_SERVER_URL_2	Chapter 3.7	New since Version 2.8
MOD_BUT_GLOBAL_LOGON_AUTH_COOKIE_NAME	Chapter 3.7	
MOD_BUT_GLOBAL_LOGON_AUTH_COOKIE_VALUE	Chapter 3.7	
MOD_BUT_AUTHORIZED_LOGON_URL	Chapter 3.7	New since Version 2.0
MOD_BUT_AUTH_STRENGTH	Chapter 3.10	New since Version 2.8
MOD_BUT_LOGON_SERVER_URL	Chapter 3.8	
MOD_BUT_LOGON_REQUIRED	Chapter 3.8	
MOD_BUT_SERVICE_LIST_ENABLED	Chapter 3.9	New since Version 2.0
MOD_BUT_SERVICE_LIST_COOKIE_NAME	Chapter 3.9	New since Version 2.0
MOD_BUT_SERVICE_LIST_COOKIE_VALUE	Chapter 3.9	New since Version 2.0
MOD_BUT_SERVICE_LIST_AUTH_ERROR_URL	Chapter 3.9	New since Version 2.0
MOD_BUT_LOCATION_ID	Chapter 3.11	New since Version 2.6

3.2 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 Default: Off

3.3 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 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**.

3.4 Cookie Settings

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

Context: virtual host

Key	Parameter	Description
MOD_BUT_COOKIE_NAME	String	Configure cookie name Default: MOD_BUT
MOD_BUT_COOKIE_DOMAIN	String	Configure cookie domain Default: not specified

Key	Parameter	Description
MOD_BUT_COOKIE_PATH	String	Configure cookie path Default: /
MOD_BUT_COOKIE_EXPIRATION	String	Configure cookie expiration date Default: not specified
MOD_BUT_COOKIE_SECURE	String	Configure cookie secure Default: secure
MOD_BUT_COOKIE_HTTPONLY	String	Configure HttpOnly (IE 6.0 SP1) Default: HttpOnly

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.

3.5 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 Default: not specified
MOD_BUT_SESSION_TIMEOUT	Integer	Configure max session time of MOD_BUT (elapsed time). Default: 3600 (seconds)
MOD_BUT_SESSION_TIMEOUT_URL	String	Error URL in case the session is timed out Default: not specified
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 Default: not specified

Key	Parameter	Description
MOD_BUT_SESSION_INACTIVITY_TIMEOUT	Integer	Configure inactivity timeout. This timeout is below the session timeout Default: 900 (seconds)
MOD_BUT_SESSION_INACTIVITY_TIMEOUT_URL	String	Error URL in case the inactivity timeout is reached Default: not specified
MOD_BUT_SESSION_HACKING_ATTEMPT_URL	String	Error URL in case the client is "guessing" sessions Default: not specified
MOD_BUT_SESSION_TIMEOUT_HISTORY	Integer	Configure how long MOD_BUT shall "remember" used session. I have configured this value to 24 hours Default:
MOD_BUT_SESSION_DESTROY	String	Regular expression: Logout pattern (session destroy) Default:
MOD_BUT_SESSION_DESTROY_URL	String	Error URL in case the client has logged out 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 Default: not specified

3.6 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 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) Default: 1000
MOD_BUT_SESSION_HISTORY_COUNT	Integer	Define number of history count (the history of used sessions) Default: 1000
MOD_BUT_COOKIESTORE_COUNT	Integer	Define cookie store count. This shared memory segment stores the backend cookies (sessions) Default: 5000

3.7 Global Authentication and Authorization

Context: Virtual Host

Key	Parameter	Description
MOD_BUT_AUTHORIZATION_ENABLED	On, Off	<p>If set to "On", MOD_BUT will test the MOD_BUT session for authentication. Only authenticated users are allowed requesting protected URL's</p> <p>If set to "Off", MOD_BUT will not enforce authentication to any URL.</p> <p>Default: off</p>
MOD_BUT_GLOBAL_LOGON_SERVER_URL	String	<p>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.</p> <p>Default: not specified</p>
MOD_BUT_GLOBAL_LOGON_SERVER_URL_1	String	<p>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.</p> <p>URL_1 is used for medium security strength (medium = 1)</p> <p>Default: not specified</p>

Key	Parameter	Description
MOD_BUT_GLOBAL_LOGON_SERVER_URL_2	String	<p>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.</p> <p>URL_1 is used for high security strength (medium = 2)</p> <p>Default: not specified</p>
MOD_BUT_GLOBAL_LOGON_AUTH_COOKIE_NAME	String	<p>Define cookie name of MOD_BUT, which changes the authentication status. This cookie is sent from a backend-system as response header</p> <p>Default: LOGON</p>
MOD_BUT_GLOBAL_LOGON_AUTH_COOKIE_VALUE	String	<p>Define cookie value of MOD_BUT, which changes the authentication status.</p> <p>Default: ok</p>
MOD_BUT_AUTHORIZED_LOGON_URL	String	<p>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-</p> <p>Default: (^/.*\$)</p>

3.8 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 Default: not specified
MOD_BUT_LOGON_REQUIRED	On, Off	If set to "On" within a Location directive, MOD_BUT will enforce an authenticated session If set to "Off", the URL within the Location is open for all and authentication is not enforced. Default: Off

3.9 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	String	Turns the service list authorization on and off. Default: Off
MOD_BUT_SERVICE_LIST_COOKIE_NAME	String	Name of the service list authorization Set-Cookie header. Default: 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 allows "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. 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. Default: /mod_but/error/authorization_error.html

3.10 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:

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

If you leave the MOD_BUT_AUTH_STRENGTH from the per-directory configuration directive, the default auth_strength=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) 0=anonymous 1=username/password auth 2=strong auth Default: 0

3.11 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 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.

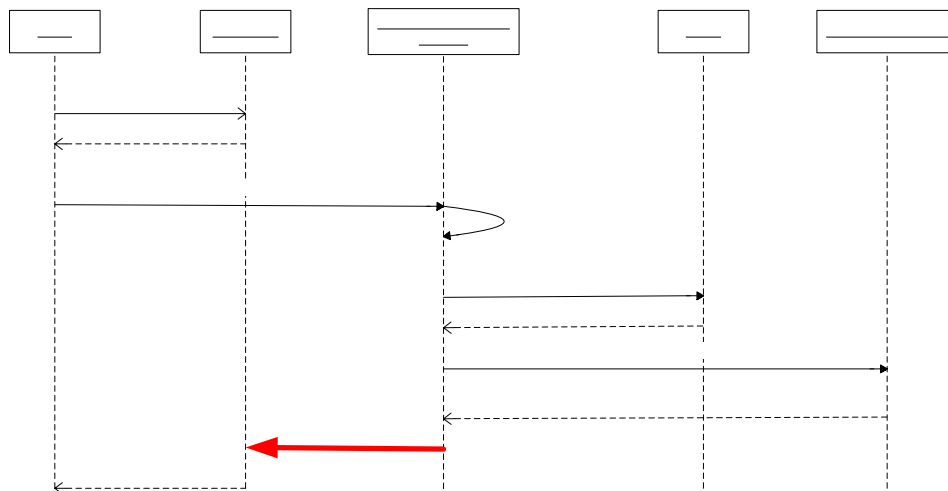
If you do not configure MOD_BUT_LOCATION_ID, the location will be configured in it’s default location_id = 0.

3.12 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 bvalue= backend cookie value 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.



4 Installing MOD_BUT

4.1 Introduction

This chapter introduces the steps we walked through while bringing the demo reverse proxy online. We start from the very beginning, where we compile our reverse-proxy from scratch. These instructions have been done on a Solaris 8.0 server.

4.2 Preparation

MOD_BUT uses standard and additional modules. If you want to follow these instructions, you need the following packages

- Newest apache 2.x source code
- Mod_replace source code
- MOD_BUT source code
- OpenSSL (recommended)

This setup guide goes through the following steps

Step	Link to Chapter	Description	Status
1	4.3	Disable TRACE within the Apache source code	Optional
2	4.4	Change Apache server banner in Apache source code	Optional
3	4.5	Enable mod_replace by adding mod_replace.c to the Apache source tree	Optional
4	4.6	Compile Apache without MOD_BUT	Required
5	4.7	Test your Apache without MOD_BUT	Required
6	4.8	Compiling MOD_BUT	Required
7	4.10	Sample configuration of http://www.but.ch/mod_but/	For your information
8	N/A	The login servlets are not part of mod_but. They are very simple servlets, which authenticates against the OpenLDAP directory. Once the user is authenticated, the login servlets sets the specific LOGON cookies and MOD_BUT will learn by these response headers whether the user is MOD_BUT-authenticated or not.	N/A ²⁵

²⁵ The source of the login servlets is not a secret. If required, we can send it to you as a sample login application. But please note, the login servlets is a "hack". It is not designed for production.

4.3 Disable “TRACE” in the Apache Source Code

Please note that disabling the HTTP method TRACE is not required for the properly running of MOD_BUT. But we urge you to disable TRACE either by removing the functionality from the code, or by mod_rewrite pattern for security reasons. We prefer the source code solution.

Edit \$APACHE_SRC/modules/http/http_protocols.c
remove the register_one_method(p, “TRACE”, M_TRACE); definition.

Before

```
AP_DECLARE(void) ap_method_registry_init(apr_pool_t *p)
{
    methods_registry = apr_hash_make(p);
    apr_pool_cleanup_register(p, NULL,
                             ap_method_registry_destroy,
                             apr_pool_cleanup_null);

    /* put all the standard methods into the registry hash to ease the
       mapping operations between name and number */
    register_one_method(p, "GET", M_GET);
    register_one_method(p, "PUT", M_PUT);
    register_one_method(p, "POST", M_POST);
    register_one_method(p, "DELETE", M_DELETE);
    register_one_method(p, "CONNECT", M_CONNECT);
    register_one_method(p, "OPTIONS", M_OPTIONS);
    register_one_method(p, "TRACE", M_TRACE);
    register_one_method(p, "PATCH", M_PATCH);
    register_one_method(p, "PROPFIND", M_PROPFIND);
    register_one_method(p, "PROPPATCH", M_PROPPATCH);
    register_one_method(p, "MKCOL", M_MKCOL);
    register_one_method(p, "COPY", M_COPY);
    register_one_method(p, "MOVE", M_MOVE);
    register_one_method(p, "LOCK", M_LOCK);
    register_one_method(p, "UNLOCK", M_UNLOCK);
    register_one_method(p, "VERSION-CONTROL", M_VERSION_CONTROL);
    register_one_method(p, "CHECKOUT", M_CHECKOUT);
    register_one_method(p, "UNCHECKOUT", M_UNCHECKOUT);
    register_one_method(p, "CHECKIN", M_CHECKIN);
    register_one_method(p, "UPDATE", M_UPDATE);
    register_one_method(p, "LABEL", M_LABEL);
    register_one_method(p, "REPORT", M_REPORT);
    register_one_method(p, "MKWORKSPACE", M_MKWORKSPACE);
    register_one_method(p, "MKACTIVITY", M_MKACTIVITY);
    register_one_method(p, "BASELINE-CONTROL", M_BASELINE_CONTROL);
    register_one_method(p, "MERGE", M_MERGE);
}
```

After

```
AP_DECLARE(void) ap_method_registry_init(apr_pool_t *p)
{
    methods_registry = apr_hash_make(p);
    apr_pool_cleanup_register(p, NULL,
                             ap_method_registry_destroy,
                             apr_pool_cleanup_null);

    /* put all the standard methods into the registry hash to ease the
       mapping operations between name and number */
    register_one_method(p, "GET", M_GET);
    register_one_method(p, "PUT", M_PUT);
}
```

```
register_one_method(p, "POST", M_POST);
register_one_method(p, "DELETE", M_DELETE);
register_one_method(p, "CONNECT", M_CONNECT);
register_one_method(p, "OPTIONS", M_OPTIONS);
register_one_method(p, "PATCH", M_PATCH);
register_one_method(p, "PROPFIND", M_PROPFIND);
register_one_method(p, "PROPPATCH", M_PROPPATCH);
register_one_method(p, "MKCOL", M_MKCOL);
register_one_method(p, "COPY", M_COPY);
register_one_method(p, "MOVE", M_MOVE);
register_one_method(p, "LOCK", M_LOCK);
register_one_method(p, "UNLOCK", M_UNLOCK);
register_one_method(p, "VERSION-CONTROL", M_VERSION_CONTROL);
register_one_method(p, "CHECKOUT", M_CHECKOUT);
register_one_method(p, "UNCHECKOUT", M_UNCHECKOUT);
register_one_method(p, "CHECKIN", M_CHECKIN);
register_one_method(p, "UPDATE", M_UPDATE);
register_one_method(p, "LABEL", M_LABEL);
register_one_method(p, "REPORT", M_REPORT);
register_one_method(p, "MKWORKSPACE", M_MKWORKSPACE);
register_one_method(p, "MKACTIVITY", M_MKACTIVITY);
register_one_method(p, "BASELINE-CONTROL", M_BASELINE_CONTROL);
register_one_method(p, "MERGE", M_MERGE);
}
```

4.4 Hide Banner Info in the Apache Source Code

Please note that hiding Apache server banners is not required for the properly running of MOD_BUT but is advisable for security reasons. You could follow the instructions below, accept the Apache banners or hide them with mod_security.

Edit \$APACHE_SRC/include/ap_release.h

Before modification (Original):

```
#define AP_SERVER_BASEVENDOR "Apache Software Foundation"
#define AP_SERVER_BASEPRODUCT "Apache"
#define AP_SERVER_MAJORVERSION "2"
#define AP_SERVER_MINORVERSION "0"
#define AP_SERVER_PATCHLEVEL "$"
#define AP_SERVER_MINORREVISION AP_SERVER_MAJORVERSION "." AP_SERVER_MINORVERSION
#define AP_SERVER_BASEREVISION AP_SERVER_MINORREVISION "." AP_SERVER_PATCHLEVEL
#define AP_SERVER_BASEVERSION AP_SERVER_BASEPRODUCT "/" AP_SERVER_BASEREVISION
#define AP_SERVER_VERSION AP_SERVER_BASEVERSION
```

After modification:

```
#define AP_SERVER_BASEVENDOR "BUT SOLUTIONS"
#define AP_SERVER_BASEPRODUCT "Microsoft IIS"
#define AP_SERVER_MAJORVERSION "2077"
#define AP_SERVER_MINORVERSION "8"
#define AP_SERVER_PATCHLEVEL "03"
#define AP_SERVER_MINORREVISION AP_SERVER_MAJORVERSION "." AP_SERVER_MINORVERSION
#define AP_SERVER_BASEREVISION AP_SERVER_MINORREVISION "." AP_SERVER_PATCHLEVEL
#define AP_SERVER_BASEVERSION AP_SERVER_BASEPRODUCT "/" AP_SERVER_BASEREVISION
#define AP_SERVER_VERSION AP_SERVER_BASEVERSION
```

4.5 Enable mod_replace

There is no better module designed to alter HTTP response bodies than mod_replace. It is a proprietary module (like MOD_BUT) but very powerful and fast.

You can use MOD_BUT without mod_replace. But if you want to alter URL directives from your backend system, mod_replace is a good choice.

```
cp mod_replace.c $APACHE_SRC/modules/filters/
```

Edit \$APACHE_SRC/modules/filter/config.m4

Before modification (Original)

```
APACHE_MODULE(ext_filter, external filter module, , , most)
APACHE_MODULE(include, Server Side Includes, , , yes)
```

After modification:

```
APACHE_MODULE(ext_filter, external filter module, , , most)
APACHE_MODULE(replace, replace filter module, , , most)
```

GLÄRNISCHSTR. 7
POSTFACH 1671
CH-8640 RAPPERSWIL

Tel. +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch www.csnc.ch

```
APACHE_MODULE(include, Server Side Includes, , , yes)
```

Run **autoconf** in \$APACHE_SRC directory. Afterwards you will “see” the `--config-replace` switch of the `./configure` script. You can test proper configuration of `mod_replace` by

```
./configure --help | grep replace
```

4.6 Compile your Reverse proxy (without MOD_BUT)

```
cd $APACHE_SRC
autoconf

./configure --prefix=/opt/applic/reverse_proxy/ \
--enable-so \
--enable-ssl \
--enable-rewrite \
--enable-replace \
--enable-proxy \
--enable-headers \
--enable-expire \
--enable-auth-digest \
--enable-unique-id \
--enable-log-forensic
```

Note: run `autoconf` after editing `config.m4` (see one step before). Otherwise, `mod_replace` will not compile.

4.7 Install and Test

Now do the following

```
make
make install
```

Test your `httpd.conf` by

```
/opt/applic/reverse_proxy/bin/apachectl -t
```

Start your Apache server manually

```
/opt/applic/reverse_proxy/bin/apachectl start
```

If you have problems here, please use the standard Apache *readme* and *docs*. You will find more useful information at <http://httpd.apache.org/>

4.8 Compilation MOD_BUT with Unix

The MOD_BUT package consists of the following components

```
el@hakka:~/apache$ ls -al
-rw-r--r--  1 hobo csnc   516 2006-02-06 15:25 Makefile
-rw-r--r--  1 hobo csnc  6155 2006-02-03 13:33 mod_but_access.c
-rw-r--r--  1 hobo csnc  4305 2006-02-03 13:33 mod_but_authorization.c
-rw-r--r--  1 hobo csnc 39039 2006-02-07 09:55 mod_but.c
-rw-r--r--  1 hobo csnc  9673 2006-02-07 09:58 mod_but_config.c
-rw-r--r--  1 hobo csnc  8308 2006-02-03 13:32 mod_but_cookiestore.c
-rw-r--r--  1 hobo csnc   180 2006-02-02 11:20 mod_but.dep
-rw-r--r--  1 hobo csnc  5369 2006-02-02 11:20 mod_but.dsp
-rw-r--r--  1 hobo csnc 15106 2006-02-07 09:56 mod_but.h
-rw-r--r--  1 hobo csnc 10528 2006-02-02 11:20 mod_but.mak
-rw-r--r--  1 hobo csnc 11645 2006-02-07 10:24 mod_but_output_filter.c
-rw-r--r--  1 hobo csnc   986 2006-02-02 11:20 mod_but.rc
-rw-r--r--  1 hobo csnc  5691 2006-02-03 15:24 mod_but_request_filter.c
-rw-r--r--  1 hobo csnc 15795 2006-02-03 13:32 mod_but_session.c
-rw-r--r--  1 hobo csnc 17420 2006-02-03 13:31 mod_but_shm.c
```

Makefile

Please consult the documentations at <http://apache.org>, if you need further explanation of the APXS flags.

-c = compile
-i = install in \$APACHE_SRC/modules
-a = configure LOAD directive into httpd.conf

```
#####
#
# mod_but Makefile V1.0 by BUT
#
#####

APXS=/opt/applic/httpd-2.0.55/bin/apxs
APXSFLAGS=-c -i -a -Wc,-O8 -Wc,-Wall
SRC=mod_but.c mod_but_access.c mod_but_request_filter.c mod_but_output_filter.c
mod_but_config.c mod_but_session.c mod_but_shm.c mod_but_cookiestore.c
mod_but_authorization.c

all: mod_but

mod_but: $(SRC)
        $(APXS) $(APXSFLAGS) $(SRC)

clean:
        rm -r *.la *.slo *.o *.lo .libs
```

You can adapt the Makefile to meet your needs. Please configure APXS and APXSFLAGS.

Configuration directive	Description
APXS	Path to Apache apxs binary. In our example: /opt/applic/reverse_proxy/bin/apxs

Configuration directive	Description
APXSFLAGS -c -I -a	-c "compile as DSO module" -i "install into modules dir of http" -a "add LoadModule into httpd.conf"
APXSFLAGS -I	Include directory for Openssl.

Note: OpenSSL is required for future features I have planned to integrate. I will use the SSL-ID as improvement for the cookie-based session handling mechanism. It's some kind of mixed mode between cookies and the SSL-ID.

For the current release of MOD_BUT, OpenSSL is not required.

make

```
e1@hakka:~/apache$ make
hobo@h00k:~/mod_but/src$ date
Tue Feb  7 14:31:53 CET 2006
hobo@h00k:~/mod_but/src$ make
/opt/applic/httpd-2.0.55/bin/apxs -c -i -a -Wc,-O8 -Wc,-Wall mod_but.c mod_but_access.c mod_but_request_filter.c
mod_but_output_filter.c mod_but_config.c mod_but_session.c mod_but_shm.c mod_but_cookiestore.c mod_but_authorization.c
/opt/applic/httpd-2.0.55/build/libtool --silent --mode=compile gcc -prefer-pic -DAP_HAVE_DESIGNATED_INITIALIZER -DLINUX=2
-D_REENTRANT -D_XOPEN_SOURCE=500 -D_BSD_SOURCE -D_SVID_SOURCE -D_GNU_SOURCE -g -O2 -pthread -I/opt/applic/httpd-
2.0.55/include -I/opt/applic/httpd-2.0.55/include -I/opt/applic/httpd-2.0.55/include -O8 -Wall -c -o mod_but.lo
mod_but.c && touch mod_but.slo
/opt/applic/httpd-2.0.55/build/libtool --silent --mode=compile gcc -prefer-pic -DAP_HAVE_DESIGNATED_INITIALIZER -DLINUX=2
-D_REENTRANT -D_XOPEN_SOURCE=500 -D_BSD_SOURCE -D_SVID_SOURCE -D_GNU_SOURCE -g -O2 -pthread -I/opt/applic/httpd-
2.0.55/include -I/opt/applic/httpd-2.0.55/include -I/opt/applic/httpd-2.0.55/include -O8 -Wall -c -o
mod_but_access.lo mod_but_access.c && touch mod_but_access.slo
/opt/applic/httpd-2.0.55/build/libtool --silent --mode=compile gcc -prefer-pic -DAP_HAVE_DESIGNATED_INITIALIZER -DLINUX=2
-D_REENTRANT -D_XOPEN_SOURCE=500 -D_BSD_SOURCE -D_SVID_SOURCE -D_GNU_SOURCE -g -O2 -pthread -I/opt/applic/httpd-
2.0.55/include -I/opt/applic/httpd-2.0.55/include -I/opt/applic/httpd-2.0.55/include -O8 -Wall -c -o
mod_but_request_filter.lo mod_but_request_filter.c && touch mod_but_request_filter.slo
/opt/applic/httpd-2.0.55/build/libtool --silent --mode=compile gcc -prefer-pic -DAP_HAVE_DESIGNATED_INITIALIZER -DLINUX=2
-D_REENTRANT -D_XOPEN_SOURCE=500 -D_BSD_SOURCE -D_SVID_SOURCE -D_GNU_SOURCE -g -O2 -pthread -I/opt/applic/httpd-
2.0.55/include -I/opt/applic/httpd-2.0.55/include -I/opt/applic/httpd-2.0.55/include -O8 -Wall -c -o
mod_but_output_filter.lo mod_but_output_filter.c && touch mod_but_output_filter.slo
/opt/applic/httpd-2.0.55/build/libtool --silent --mode=compile gcc -prefer-pic -DAP_HAVE_DESIGNATED_INITIALIZER -DLINUX=2
-D_REENTRANT -D_XOPEN_SOURCE=500 -D_BSD_SOURCE -D_SVID_SOURCE -D_GNU_SOURCE -g -O2 -pthread -I/opt/applic/httpd-
2.0.55/include -I/opt/applic/httpd-2.0.55/include -I/opt/applic/httpd-2.0.55/include -O8 -Wall -c -o
mod_but_config.lo mod_but_config.c && touch mod_but_config.slo
/opt/applic/httpd-2.0.55/build/libtool --silent --mode=compile gcc -prefer-pic -DAP_HAVE_DESIGNATED_INITIALIZER -DLINUX=2
-D_REENTRANT -D_XOPEN_SOURCE=500 -D_BSD_SOURCE -D_SVID_SOURCE -D_GNU_SOURCE -g -O2 -pthread -I/opt/applic/httpd-
2.0.55/include -I/opt/applic/httpd-2.0.55/include -I/opt/applic/httpd-2.0.55/include -O8 -Wall -c -o
mod_but_session.lo mod_but_session.c && touch mod_but_session.slo
/opt/applic/httpd-2.0.55/build/libtool --silent --mode=compile gcc -prefer-pic -DAP_HAVE_DESIGNATED_INITIALIZER -DLINUX=2
-D_REENTRANT -D_XOPEN_SOURCE=500 -D_BSD_SOURCE -D_SVID_SOURCE -D_GNU_SOURCE -g -O2 -pthread -I/opt/applic/httpd-
2.0.55/include -I/opt/applic/httpd-2.0.55/include -I/opt/applic/httpd-2.0.55/include -O8 -Wall -c -o mod_but_shm.lo
mod_but_shm.c && touch mod_but_shm.slo
/opt/applic/httpd-2.0.55/build/libtool --silent --mode=compile gcc -prefer-pic -DAP_HAVE_DESIGNATED_INITIALIZER -DLINUX=2
-D_REENTRANT -D_XOPEN_SOURCE=500 -D_BSD_SOURCE -D_SVID_SOURCE -D_GNU_SOURCE -g -O2 -pthread -I/opt/applic/httpd-
2.0.55/include -I/opt/applic/httpd-2.0.55/include -I/opt/applic/httpd-2.0.55/include -O8 -Wall -c -o
mod_but_cookiestore.lo mod_but_cookiestore.c && touch mod_but_cookiestore.slo
/opt/applic/httpd-2.0.55/build/libtool --silent --mode=compile gcc -prefer-pic -DAP_HAVE_DESIGNATED_INITIALIZER -DLINUX=2
-D_REENTRANT -D_XOPEN_SOURCE=500 -D_BSD_SOURCE -D_SVID_SOURCE -D_GNU_SOURCE -g -O2 -pthread -I/opt/applic/httpd-
2.0.55/include -I/opt/applic/httpd-2.0.55/include -I/opt/applic/httpd-2.0.55/include -O8 -Wall -c -o
mod_but_authorization.lo mod_but_authorization.c && touch mod_but_authorization.slo
/opt/applic/httpd-2.0.55/build/libtool --silent --mode=link gcc -o mod_but.la -rpath /opt/applic/httpd-2.0.55/modules -
module -avoid-version mod_but_authorization.lo mod_but_cookiestore.lo mod_but_shm.lo mod_but_session.lo
mod_but_config.lo mod_but_output_filter.lo mod_but_request_filter.lo mod_but_access.lo mod_but.lo
/opt/applic/httpd-2.0.55/build/instldso.sh SH_LIBTOOL='/opt/applic/httpd-2.0.55/build/libtool' mod_but.la
/opt/applic/httpd-2.0.55/modules
/opt/applic/httpd-2.0.55/build/libtool --mode=install cp mod_but.la /opt/applic/httpd-2.0.55/modules/
cp .libs/mod_but.so /opt/applic/httpd-2.0.55/modules/mod_but.so
cp .libs/mod_but.lai /opt/applic/httpd-2.0.55/modules/mod_but.la
cp .libs/mod_but.a /opt/applic/httpd-2.0.55/modules/mod_but.a
ranlib /opt/applic/httpd-2.0.55/modules/mod_but.a
chmod 644 /opt/applic/httpd-2.0.55/modules/mod_but.a
PATH="$PATH:/sbin" ldconfig -n /opt/applic/httpd-2.0.55/modules
```

Additionally, MOD_BUT has been successfully compiled under Sparc/Solaris8 and Sparc/Solaris10 using gcc.

4.9 Compilation MOD_BUT with Microsoft Windows

Task	Description
Download Apache Source Code	http://httpd.apache.org/download.cgi
Create your compilation environment	http://httpd.apache.org/docs/2.0/platform/win_compiling.html
Create directory structure in original apache source tree	Create a mod_but directory in \$APACHE_SOURCE\modules
Copy mod_but files from mod_but.tar.gz to the apache source tree	Copy all data from mod_but.tar.gz -> mod_but directory into the Apache source tree
Configure mod_but in BaseAddr.ref	<p>Edit \$APACHE_SOURCE\modules\os\win32\BaseAddr.ref</p> <p>Configure a max size of 0x00300000</p> <p>Example: mod_but.so 0x6F82000 0x00300000</p> <p>Important!! Addition of 0x6F82000 + 0x00300000 MUST result into the value above the mod_but entry!</p>
Configure Makefile.win	<p>Edit \$APACHE_SOURCE\Makefile.win</p> <p>Configure the mod_but snippets below.</p>

Task	Description
	<pre> ... !IF EXIST("src\lib\openssl") cd modules\ssl \$(MAKE) \$(MAKEOPT) -f mod_ssl.mak CFG="mod_ssl - Win32 \$(LONG)" RECURSE=0 \$(CTARGET) .\\$(LONG)\mod_ssl.so cd ..\.. cd modules\mod_but \$(MAKE) \$(MAKEOPT) -f mod_but.mak CFG="mod_but - Win32 \$(LONG)" RECURSE=0 \$(CTARGET) cd ..\.. cd support \$(MAKE) \$(MAKEOPT) -f abs.mak CFG="abs - Win32 \$(LONG)" RECURSE=0 \$(CTARGET) cd .. !ENDIF !IF EXIST("src\lib\openssl") copy modules\ssl\\$(LONG)\mod_ssl.\$(src_so) "\$(inst_so)" <.y copy modules\mod_but\\$(LONG)\mod_but.\$(src_so) "\$(inst_so)" <.y \$(quiet)copy src\lib\openssl\\$(SSLBIN)\openssl.\$(src_exe) "\$(inst_exe)" <.y \$(quiet)copy src\lib\openssl\\$(SSLBIN)\libeay32.\$(src_dll) "\$(inst_dll)" <.y \$(quiet)copy src\lib\openssl\\$(SSLBIN)\ssleay32.\$(src_dll) "\$(inst_dll)" <.y copy support\\$(LONG)\abs.\$(src_exe) "\$(inst_exe)\ab.\$(src_exe)" <.y !ELSE copy support\\$(LONG)\ab.\$(src_exe) "\$(inst_exe)" <.y !ENDIF ... </pre>
Compilation	<p>Please use the following howto:</p> <p>http://httpd.apache.org/docs/2.0/platform/win_compiling.html</p> <p>Only release-build will work! nmake /f Makefile.win_apacher</p>

4.10 Sample Configuration MOD_BUT

The following configuration is used in http://www.but.ch/mod_but/. Please feel free to use it as an example:

```

#####
# Description:  Apache Configuration
# Author:      BUT
# Target:      VirtualHost www.but.ch on 80.254.178.109
# Date:        September 2005
# Contact:     el@but.ch
#####

<VirtualHost 80.254.178.109:80>

#####
#             GLOBAL CONFIG
#####

```



```

ServerAdmin      webmaster@but.ch
DocumentRoot     "/opt/applic/www/www.but.ch/htdocs"
ServerName       www.but.ch
#   AddDefaultCharset UTF-8

ErrorLog /var/virtmp/logs/www.but.ch/error_www.but.ch.log
CustomLog /var/virtmp/logs/www.but.ch/access_www.but.ch.log common
CustomLog /var/virtmp/logs/www.but.ch/combined_www.but.ch.log combined
CustomLog /var/virtmp/logs/www.but.ch/referer_www.but.ch.log referer

#####
#   GLOBAL CONFIG
#####
#ForensicLog /var/virtmp/logs/www.but.ch/www.but.ch.forensic.log

#####
#   MOD_PROXY
#####

ProxyVia      Off

#####
#   MOD_HEADER
#####

RequestHeader append SSL_CIPHER "%{SSL_CIPHER}e"
Header set Cache-Control no-cache
Header add Cache-Control no-store
Header set Pragma no-cache

#####
#   MOD_REWRITE
#####
RewriteEngine On
RewriteRule    ^mod_but$  mod_but/  [P]
RewriteRule    ^/(.*)    /root/$1    [P]

#####
#   MOD_SECURITY
#####
SecFilterEngine On
SecAuditEngine On
SecFilterScanPost On
SecFilterScanOutput On
SecFilterCheckUnicodeEncoding On
SecFilterCheckURLEncoding On
# SecFilterCheckCookieFormat On
SecFilterForceByteRange 0 255
#SecServerSignature "Secure But Webserver"
SecFilterDebugLog      logs/www.but.ch.modsec_debug.log
SecFilterDebugLevel    4
SecAuditLog            logs/www.but.ch.modsec_audit.log

Include            conf/mod_security.conf

#####
#   MOD_UNIQ
#####
RequestHeader append UNIQUE_ID "%{UNIQUE_ID}e"

#####
#   MOD_BUT

```

```
#####
MOD_BUT_ENABLED On
MOD_BUT_CLIENT_REFUSES_COOKIES_URL /mod_but/error/refused_cookies.html
MOD_BUT_COOKIE_NAME ElHTTP
MOD_BUT_COOKIE_DOMAIN but.ch
MOD_BUT_COOKIE_PATH /
# MOD_BUT_COOKIE_EXPIRATION "3-Jan-2006 00:00:01 GMT"
# MOD_BUT_COOKIE_SECURE secure
MOD_BUT_COOKIE_HTTPONLY HttpOnly

MOD_BUT_SESSION_FREE_URL " (^/$) | (^/robots.txt) | (.*/but.css) | (.*/gif) |
(^/_shared/) | (^/cgi-bin/.* ) | (^/mod_but/error/) | (^/root/error/) | (^/renew/) |
(^/favicon.ico) | (^/public/) | (^/index.*) "

MOD_BUT_SESSION_TIMEOUT 14400
MOD_BUT_SESSION_TIMEOUT_URL /mod_but/error/session_expired.html
MOD_BUT_SESSION_RENEW_URL " (^/renew/) "
MOD_BUT_SESSION_INACTIVITY_TIMEOUT 3600
MOD_BUT_SESSION_INACTIVITY_TIMEOUT_URL /mod_but/error/session_inactivity.html
MOD_BUT_SESSION_HACKING_ATTEMPT_URL /mod_but/error/session_invalid.html
MOD_BUT_SESSION_TIMEOUT_HISTORY 86400
MOD_BUT_ALL_SHM_SPACE_USED_URL /mod_but/error/session_shm_used.html
MOD_BUT_SESSION_DESTROY " (^/logout.*) "
MOD_BUT_SESSION_DESTROY_URL /mod_but/error/session_destroy.html
MOD_BUT_AUTHORIZATION_ENABLED On
MOD_BUT_GLOBAL_LOGON_SERVER_URL /mod_but/login.html
MOD_BUT_GLOBAL_LOGON_AUTH_COOKIE_NAME LOGON
MOD_BUT_GLOBAL_LOGON_AUTH_COOKIE_VALUE ok
MOD_BUT_SESSION_STORE_FREE_COOKIES " (^language=.* ) | (^trustme=.* ) "
MOD_BUT_SERVICE_LIST_ENABLED On
MOD_BUT_AUTHORIZED_LOGON_URL " (^/webapp/login/.* ) "

#####
# ROOT GOES TO BACKEND
#####

<Location /webapp/but>
    ProxyPass http://172.1.200.55:35080/but/
    ProxyPassReverse http://172.1.200.55:35080/but/
    MOD_BUT_LOGON_REQUIRED On
    MOD_BUT_AUTH_STRENGTH 1
</Location>

<Location /webapp/one>
    ProxyPass http://172.1.200.55:35080/but/
    ProxyPassReverse http://172.1.200.55:35080/one/
    MOD_BUT_LOGON_REQUIRED On
    MOD_BUT_AUTH_STRENGTH 1
    MOD_BUT_LOCATION_ID 1
</Location>

<Location /webapp/two>
    ProxyPass http://172.1.200.55:35080/but/
    ProxyPassReverse http://172.1.200.55:35080/two/
    MOD_BUT_LOGON_REQUIRED On
    MOD_BUT_AUTH_STRENGTH 1
    MOD_BUT_LOCATION_ID 2
</Location>

<Location /webapp/strong>
    ProxyPass http://172.1.200.55:35080/but/
    ProxyPassReverse http://172.1.200.55:35080/strong/
    MOD_BUT_LOGON_REQUIRED On
```

```
MOD_BUT_AUTH_STRENGTH 2
MOD_BUT_LOCATION_ID 5
</Location>

<Location /filenabber>
    ProxyPass http://80.254.178.109/filenabber/
    ProxyPassReverse http://172.1.200.55:35080/filenabber/
    MOD_BUT_LOGON_REQUIRED On
    MOD_BUT_AUTH_STRENGTH 1
    MOD_BUT_LOCATION_ID 6
</Location>

<Location /loginok>
    ProxyPass http://127.0.0.1:20060/mod_but/
    ProxyPassReverse http://www.but.ch/loginok/
    MOD_BUT_LOGON_REQUIRED On
</Location>

<Location /root>
    ProxyPass http://127.0.0.1:20060/
    ProxyPassReverse http://www.but.ch/root/
    MOD_BUT_LOGON_REQUIRED Off
</Location>

Include conf/error.conf

</VirtualHost>
```