Living on the Edge

Advanced ModSecurity to Save Your Ass

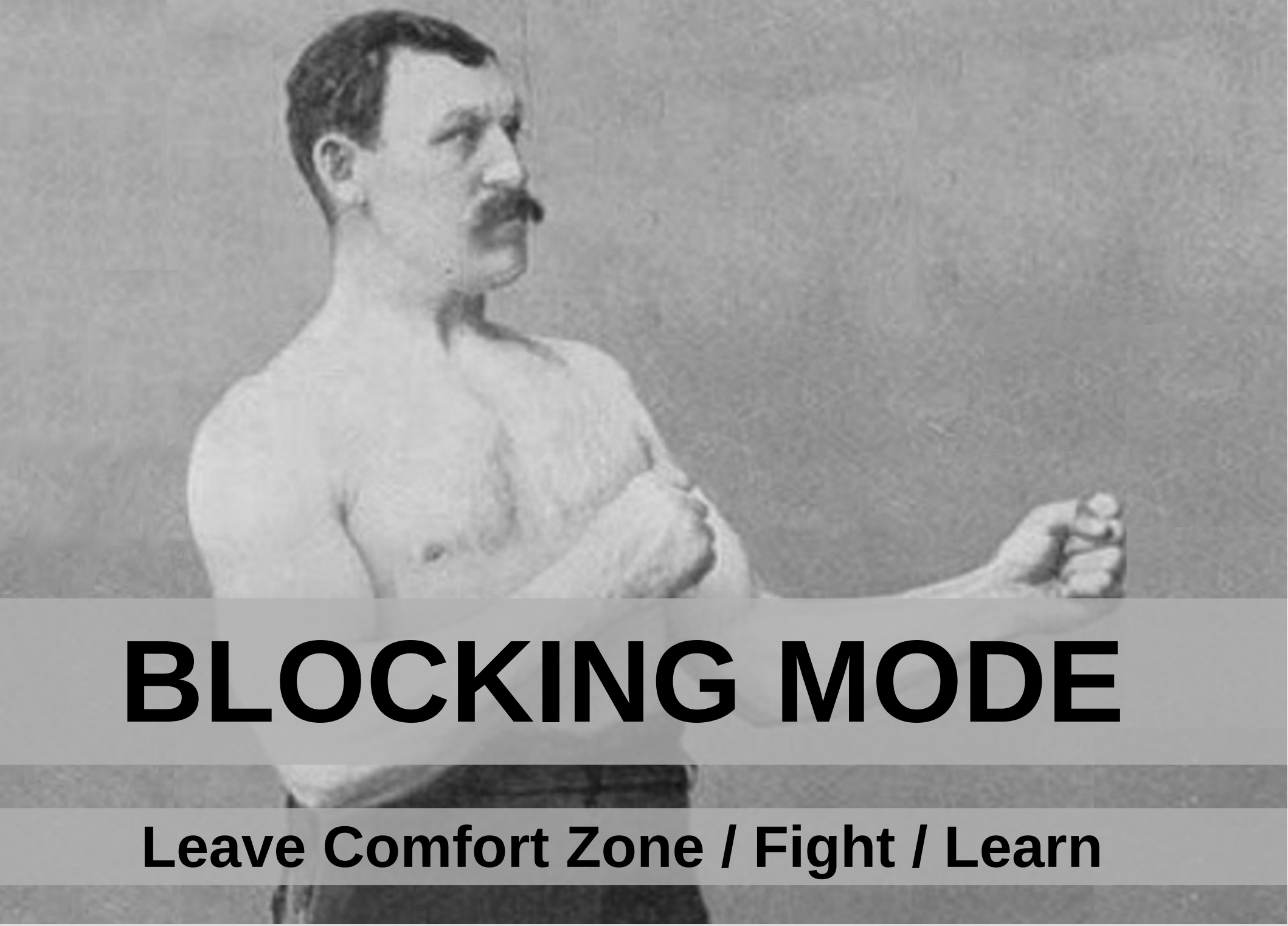
Christian Folini, netnea.com

OWASP Switzerland, 2014.11.12

VERY SLIM CHANCE

Asymmetric Game / Shoot Out

christian.folini@netnea.com | www.christian-folini.ch | @ChrFolini | +41 79 220 23 76



BLOCKING MODE

Leave Comfort Zone / Fight / Learn

christian.folini@netnea.com | www.christian-folini.ch | @ChrFolini | +41 79 220 23 76

FALSE POSITIVES

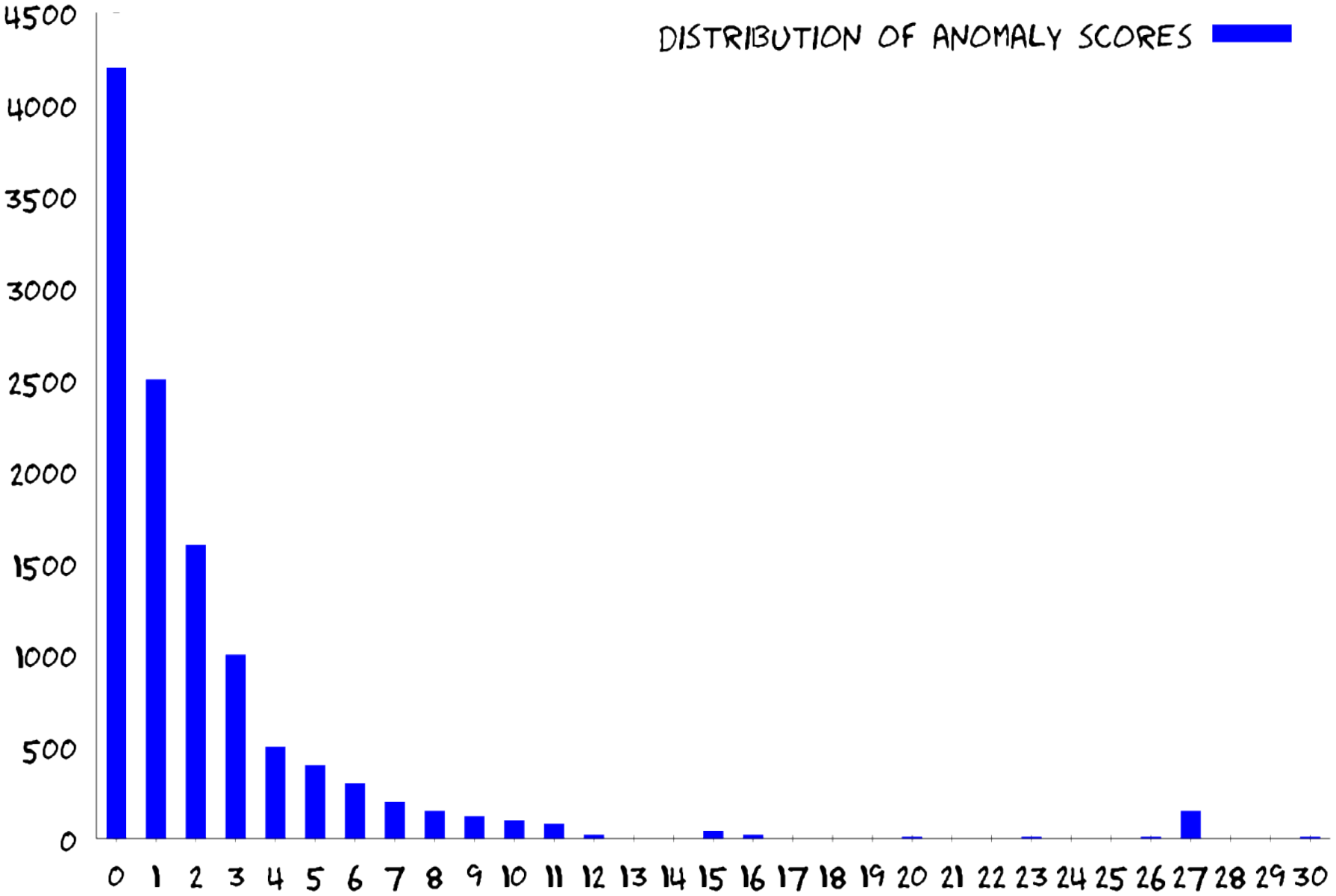
Log / Divide and Rule / Educated Limits

christian.folini@netnea.com | www.christian-folini.ch | @ChrFolini | +41 79 220 23 76

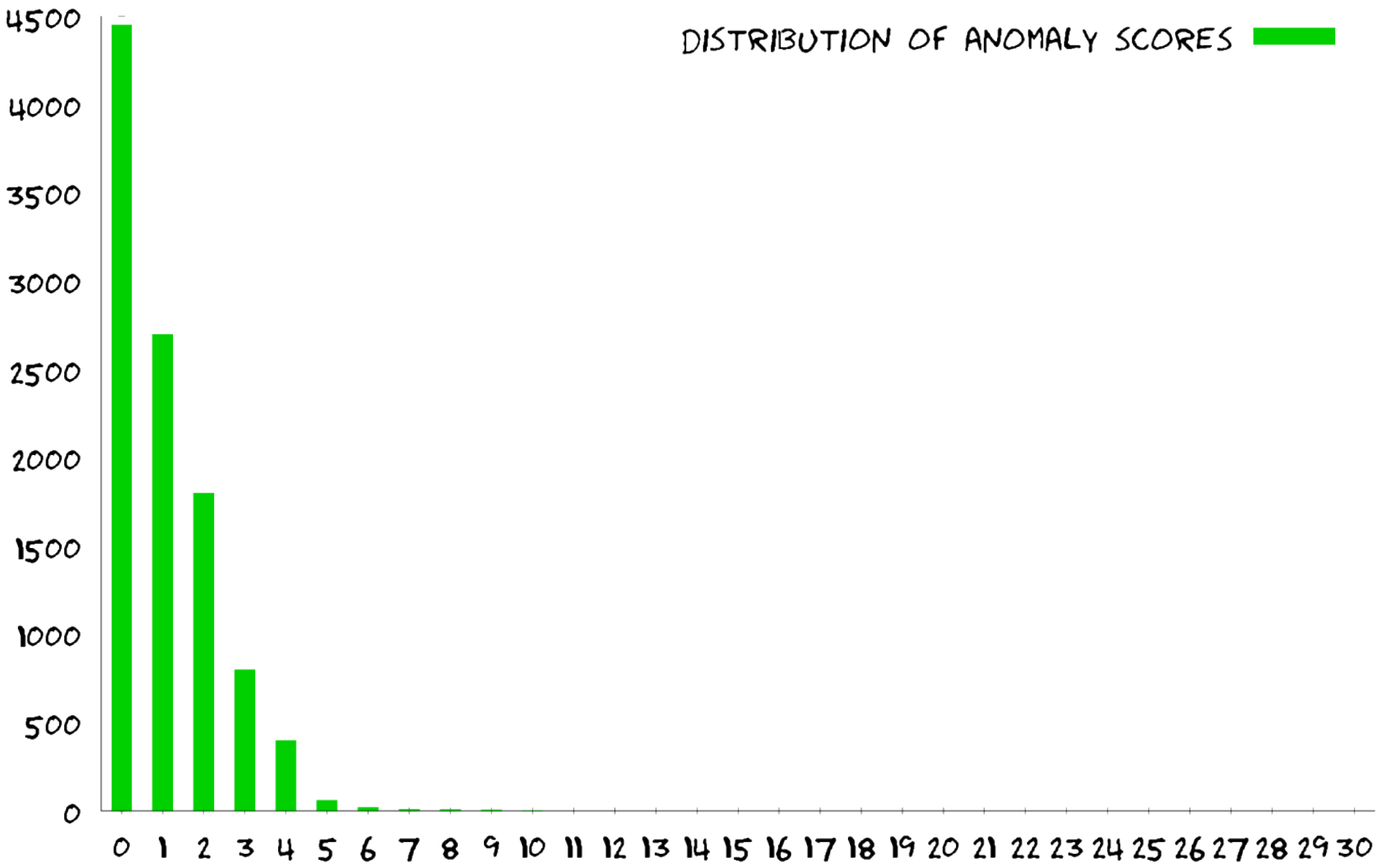
TAXONOMY

christian.folini@netnea.com | www.christian-folini.ch | @ChrFolini | +41 79 220 23 76

Access-Log of an Untuned ModSec Core Rules Installation



Access-Log of an Tuned ModSec Core Rules Installation



Example #0 – Remedy for a Session Fixation Vulnerability

Goal: Interferring with HTTP Request

Example #0 – Remedy for a Session Fixation Vulnerability

```
SecRule REQUEST_METHOD "^POST$" "id:10001,chain,pass,log, \
      msg:'Stripping Cookie header of login POST request'"
  SecRule REQUEST_FILENAME "@beginsWith  /do/login.action" \
      "setenv:stripsession=1"

RequestHeader unset "Cookie" env=stripsession
```

Example #0 – Remedy for a Session Fixation Vulnerability

```
SecRule REQUEST_METHOD "^POST$" "id:10001,chain,pass,log, \
      msg:'Stripping Cookie header of login POST request'"
  SecRule REQUEST_FILENAME "@beginsWith  /do/login.action" \
      "setenv:stripsession=1"

RequestHeader unset "Cookie" env=stripsession
```

Example #1 – Fix Stupid Client with an Authentication Cache

Goals:

Call External Script

Maintain a Session

Example #1 – Fix Stupid Client with an Authenication Cache (1 of 7)

```
# Login Cache: Session Initialization (assign request a session, new or existing)

SecRule REQUEST_HEADERS:Authorization "^Basic (.*)$" "id:10101,capture,pass,nolog"

SecRule TX:1 "^.*$" "id:10102,capture,t:sha1,t:hexencode,pass,log, \
    msg:'Auth Cache: Generating session key, initializing session (%{TX.0})',setsid:%{TX.0}"


# Login Cache: Perform Authentication Externally

SecRule SESSION:login "!^$" "id:10201,pass,log,skipAfter:LOGIN_CACHE_LOGIN_END, \
    msg:'Auth Cache: Login is cached. Skipping login.'"

SecAction "id:10202,pass,log,setvar:session.login=FAIL,expirevar:session.login=300, \
    msg:'Auth Cache: Performing login, initializing status as login failed',"

SecRule REQUEST_HEADERS:Authorization "^Basic (.*)$" "id:10203,capture,pass,nolog"

SecRule TX:1 "@inspectFile /apache/bin/login-connector.pl" "id:10204,pass,log, \
    msg:'Auth Cache: Login: Executing login via external script.',setvar:session.login=OK"

SecMarker LOGIN_CACHE_LOGIN_END


# Login Cache: Authenication Check

SecRule SESSION:login  "^OK$"   "id:10301,pass,log, \
    msg:'Auth Cache: Login success. Granting access.'"

SecRule SESSION:login  "!^OK$"  "id:10302,deny,status:403,log, \
    msg:'Auth Cache: Login failed. Denying access.'"
```

Example #1 – Fix Stupid Client with an Authenication Cache (2 of 7)

```
# Login Cache: Session Initialization (assign request a session, new or existing)

SecRule REQUEST_HEADERS:Authorization "^Basic (.*)$" "id:10101,capture,pass,nolog"

SecRule TX:1 "^.*$" "id:10102,capture,t:sha1,t:hexencode,pass,log, \
    msg:'Auth Cache: Generating session key, initializing session (%{TX.0})',setsid:%{TX.0}"


# Login Cache: Perform Authentication Externally

SecRule SESSION:login "!^$" "id:10201,pass,log,skipAfter:LOGIN_CACHE_LOGIN_END, \
    msg:'Auth Cache: Login is cached. Skipping login.'"

SecAction "id:10202,pass,log,setvar:session.login=FAIL,expirevar:session.login=300, \
    msg:'Auth Cache: Performing login, initializing status as login failed',"

SecRule REQUEST_HEADERS:Authorization "^Basic (.*)$" "id:10203,capture,pass,nolog"

SecRule TX:1 "@inspectFile /apache/bin/login-connector.pl" "id:10204,pass,log, \
    msg:'Auth Cache: Login: Executing login via external script.',setvar:session.login=OK"

SecMarker LOGIN_CACHE_LOGIN_END


# Login Cache: Authenication Check

SecRule SESSION:login  "^OK$"        "id:10301,pass,log, \
    msg:'Auth Cache: Login success. Granting access.'"

SecRule SESSION:login  "!^OK$"       "id:10302,deny,status:403,log, \
    msg:'Auth Cache: Login failed. Denying access.'"
```

Example #1 – Fix Stupid Client with an Authenication Cache (3 of 7)

```
# Login Cache: Session Initialization (assign request a session, new or existing)

SecRule REQUEST_HEADERS:Authorization "^Basic (.*)$" "id:10101,capture,pass,nolog"

SecRule TX:1 "^.*$" "id:10102,capture,t:sha1,t:hexencode,pass,log, \
    msg:'Auth Cache: Generating session key, initializing session (%{TX.0})',setsid:%{TX.0}"


# Login Cache: Perform Authentication Externally

SecRule SESSION:login "!^$" "id:10201,pass,log,skipAfter:LOGIN_CACHE_LOGIN_END, \
    msg:'Auth Cache: Login is cached. Skipping login.'"

SecAction "id:10202,pass,log,setvar:session.login=FAIL,expirevar:session.login=300, \
    msg:'Auth Cache: Performing login, initializing status as login failed',"

SecRule REQUEST_HEADERS:Authorization "^Basic (.*)$" "id:10203,capture,pass,nolog"

SecRule TX:1 "@inspectFile /apache/bin/login-connector.pl" "id:10204,pass,log, \
    msg:'Auth Cache: Login: Executing login via external script.',setvar:session.login=OK"

SecMarker LOGIN_CACHE_LOGIN_END


# Login Cache: Authenication Check

SecRule SESSION:login  "^OK$"        "id:10301,pass,log, \
    msg:'Auth Cache: Login success. Granting access.'"

SecRule SESSION:login  "!^OK$"       "id:10302,deny,status:403,log, \
    msg:'Auth Cache: Login failed. Denying access.'"
```

Example #1 – Fix Stupid Client with an Authenication Cache (4 of 7)

```
# Login Cache: Session Initialization (assign request a session, new or existing)

SecRule REQUEST_HEADERS:Authorization "^Basic (.*)$" "id:10101,capture,pass,nolog"

SecRule TX:1 "^.*$" "id:10102,capture,t:sha1,t:hexencode,pass,log, \
    msg:'Auth Cache: Generating session key, initializing session (%{TX.0})',setsid:%{TX.0}"


# Login Cache: Perform Authentication Externally

SecRule SESSION:login "!^$" "id:10201,pass,log,skipAfter:LOGIN_CACHE_LOGIN_END, \
    msg:'Auth Cache: Login is cached. Skipping login.'"

SecAction "id:10202,pass,log,setvar:session.login=FAIL,expirevar:session.login=300, \
    msg:'Auth Cache: Performing login, initializing status as login failed',"

SecRule REQUEST_HEADERS:Authorization "^Basic (.*)$" "id:10203,capture,pass,nolog"

SecRule TX:1 "@inspectFile /apache/bin/login-connector.pl" "id:10204,pass,log, \
    msg:'Auth Cache: Login: Executing login via external script.',setvar:session.login=OK"

SecMarker LOGIN_CACHE_LOGIN_END


# Login Cache: Authenication Check

SecRule SESSION:login  "^OK$"        "id:10301,pass,log, \
    msg:'Auth Cache: Login success. Granting access.'"

SecRule SESSION:login  "!^OK$"       "id:10302,deny,status:403,log, \
    msg:'Auth Cache: Login failed. Denying access.'"
```

Example #1 – Fix Stupid Client with an Authenication Cache (5 of 7)

```
#!/usr/bin/perl -w

use strict;

my $login_state = "FAIL";

my ($BASE64LOGIN) = shift @ARGV;

# Smart Code Performing Authentication

...

if ( $login_state eq "OK" ) {
    print "0 login OK\n";
} else {
    print "1 login FAIL\n";
}

1;
```

Example #1 – Fix Stupid Client with an Authenication Cache (6 of 7)

```
# Login Cache: Session Initialization (assign request a session, new or existing)

SecRule REQUEST_HEADERS:Authorization "^Basic (.*)$" "id:10101,capture,pass,nolog"

SecRule TX:1 "^.*$" "id:10102,capture,t:sha1,t:hexencode,pass,log, \
    msg:'Auth Cache: Generating session key, initializing session (%{TX.0})',setsid:%{TX.0}"


# Login Cache: Perform Authentication Externally

SecRule SESSION:login "!^$" "id:10201,pass,log,skipAfter:LOGIN_CACHE_LOGIN_END, \
    msg:'Auth Cache: Login is cached. Skipping login.'"

SecAction "id:10202,pass,log,setvar:session.login=FAIL,expirevar:session.login=300, \
    msg:'Auth Cache: Performing login, initializing status as login failed',"

SecRule REQUEST_HEADERS:Authorization "^Basic (.*)$" "id:10203,capture,pass,nolog"

SecRule TX:1 "@inspectFile /apache/bin/login-connector.pl" "id:10204,pass,log, \
    msg:'Auth Cache: Login: Executing login via external script.',setvar:session.login=OK"

SecMarker LOGIN_CACHE_LOGIN_END


# Login Cache: Authenication Check

SecRule SESSION:login  "^OK$"        "id:10301,pass,log, \
    msg:'Auth Cache: Login success. Granting access.'"

SecRule SESSION:login  "!^OK$"       "id:10302,deny,status:403,log, \
    msg:'Auth Cache: Login failed. Denying access.'"
```

Example #1 – Fix Stupid Client with an Authenication Cache (7 of 7)

```
# Login Cache: Session Initialization (assign request a session, new or existing)

SecRule REQUEST_HEADERS:Authorization "^Basic (.*)$" "id:10101,capture,pass,nolog"

SecRule TX:1 "^.*$" "id:10102,capture,t:sha1,t:hexencode,pass,log, \
    msg:'Auth Cache: Generating session key, initializing session (%{TX.0})',setsid:%{TX.0}"


# Login Cache: Perform Authentication Externally

SecRule SESSION:login "!^$" "id:10201,pass,log,skipAfter:LOGIN_CACHE_LOGIN_END, \
    msg:'Auth Cache: Login is cached. Skipping login.'"

SecAction "id:10202,pass,log,setvar:session.login=FAIL,expirevar:session.login=300, \
    msg:'Auth Cache: Performing login, initializing status as login failed',"

SecRule REQUEST_HEADERS:Authorization "^Basic (.*)$" "id:10203,capture,pass,nolog"

SecRule TX:1 "@inspectFile /apache/bin/login-connector.pl" "id:10204,pass,log, \
    msg:'Auth Cache: Login: Executing login via external script.',setvar:session.login=OK"

SecMarker LOGIN_CACHE_LOGIN_END


# Login Cache: Authenication Check

SecRule SESSION:login  "^OK$"        "id:10301,pass,log, \
    msg:'Auth Cache: Login success. Granting access.'"

SecRule SESSION:login  "!^OK$"       "id:10302,deny,status:403,log, \
    msg:'Auth Cache: Login failed. Denying access.'"
```

Example #2 – Fix an Authentication Bypass Vulnerability

Goals:

Maintain Session in Parallel to Authenticated Session of Application

Scan HTTP Response Body for Clues about Authorisation and Add to Session

Grant Access to Resources Based on Session Data

Example #2 – Fix an Authentication Bypass Vulnerability (1 of 7)

```
# Auth Bypass: Serverside Session Creation
SecRule RESPONSE_HEADERS:Set-Cookie "(?i:(jsessionid).*?=([^\s].*?)\;\s?)" "phase:5,id:11100,t:none,pass,log,capture, \
  msg:'Auth-Bypass: Appl created session. Initializing parallel session.',setsid:%{TX.2},setvar:session.initok=1"

# Auth Bypass: Session Initialization (assign request an existing session)
SecRule REQUEST_FILENAME "@beginsWith /login.do" "id:11200,nolog,pass,skipAfter:END_AUTH_BYPASS_SESSION_INIT, \
  msg:'Auth-Bypass: Login request aimed to obtain serverside jsessionid. Skipping session check.'"

SecRule REQUEST_COOKIES:JSESSIONID "!^$" "id:11201,t:none,pass,setsid:%{REQUEST_COOKIES.JSESSIONID}, \
  msg:'Auth-Bypass: Initializing session based on application cookie jsessionid'"

SecRule &SESSION:INITOK "!@eq 1" "id:11202,t:none,deny,status:403, \
  msg:'Auth-Bypass: Failed check for appl creation of session. Cookie missing? Client session fixation? Denying access.'"

SecMarker END_AUTH_BYPASS_SESSION_INIT

# Auth Bypass: Extracting csv files of session out of html responses
SecRule REQUEST_URI "!^/display-csv-link.do" "phase:4,id:11300,pass,log,skip:1, \
  msg:'Auth Bypass: Uninteresting URI. Skipping the entering of csv file to the session'"

SecRule RESPONSE_BODY "downloads/(.*?.csv)" "phase:4,id:11301,capture,setvar:session.files='%{session.files};%{TX.1}' \
  msg:'Auth-Bypass: Adding csv file to session (%{TX.1})',pass,log"

# Auth Bypass: Checking authentication / authorisation of file download
SecRule REQUEST_URI "!^(/downloads/.*csv)$" "id:11400,pass,log,skipAfter:END_AUTH_BYPASS_CHECK, \
  msg:'Auth-Bypass: Uninteresting URI. Skipping download check.'"

SecAction "id:11401,pass,log,msg:'Auth-Bypass: Perform auth check. Files in session: %{session.files}'"

SecRule REQUEST_URI "^/downloads/(.*?.csv)$" "id:11402,pass,log,chain,capture,skipAfter:END_AUTH_BYPASS_CHECK, \
  msg:'Auth-Bypass: Auth check OK. Requested file is part files displayed to client during of session (%{TX.1})'"
  SecRule SESSION:FILES "%{TX.1}" ""

SecAction "id:11403,deny,status:403,log, \
  msg:'Auth-Bypass: Bypass attempt detected. Requested file NOT part of session files. Denying access'"

SecMarker END_AUTH_BYPASS_CHECK
```

Example #2 – Fix an Authentication Bypass Vulnerability (2 of 7)

```
# Auth Bypass: Serverside Session Creation

SecRule RESPONSE_HEADERS:Set-Cookie "(?i:(jsessionid).*?=([^\s].*?)\;\s?)" "phase:5,id:11100,t:none,pass,log,capture, \
  msg:'Auth-Bypass: Appl created session. Initializing parallel session.',setsid:%{TX.2},setvar:session.initok=1"

# Auth Bypass: Session Initialization (assign request an existing session)

SecRule REQUEST_FILENAME "@beginsWith /login.do" "id:11200,nolog,pass,skipAfter:END_AUTH_BYPASS_SESSION_INIT, \
  msg:'Auth-Bypass: Login request aimed to obtain serverside jsessionid. Skipping session check.'"

SecRule REQUEST_COOKIES:JSESSIONID "!^$" "id:11201,t:none,pass,setsid:%{REQUEST_COOKIES.JSESSIONID}, \
  msg:'Auth-Bypass: Initializing session based on application cookie jsessionid'"

SecRule &SESSION:INITOK "!@eq 1" "id:11202,t:none,deny,status:403, \
  msg:'Auth-Bypass: Failed check for appl creation of session. Cookie missing? Client session fixation? Denying access.'"

SecMarker END_AUTH_BYPASS_SESSION_INIT

# Auth Bypass: Extracting csv files of session out of html responses

SecRule REQUEST_URI "!^/display-csv-link.do" "phase:4,id:11300,pass,log,skip:1, \
  msg:'Auth Bypass: Uninteresting URI. Skipping the entering of csv file to the session'"

SecRule RESPONSE_BODY "downloads/(.*?.csv)" "phase:4,id:11301,capture,setvar:session.files='%{session.files};%{TX.1}' \
  msg:'Auth-Bypass: Adding csv file to session (%{TX.1})',pass,log"

# Auth Bypass: Checking authentication / authorisation of file download

SecRule REQUEST_URI "!^(/downloads/.*csv)$" "id:11400,pass,log,skipAfter:END_AUTH_BYPASS_CHECK, \
  msg:'Auth-Bypass: Uninteresting URI. Skipping download check.'"

SecAction "id:11401,pass,log,msg:'Auth-Bypass: Perform auth check. Files in session: %{session.files}'"

SecRule REQUEST_URI "^/downloads/(.*?.csv)$" "id:11402,pass,log,chain,capture,skipAfter:END_AUTH_BYPASS_CHECK, \
  msg:'Auth-Bypass: Auth check OK. Requested file is part files displayed to client during of session (%{TX.1})'"
  SecRule SESSION:FILES "%{TX.1}" ""

SecAction "id:11403,deny,status:403,log, \
  msg:'Auth-Bypass: Bypass attempt detected. Requested file NOT part of session files. Denying access'"

SecMarker END_AUTH_BYPASS_CHECK
```

Example #2 – Fix an Authentication Bypass Vulnerability (3 of 7)

```
# Auth Bypass: Serverside Session Creation
SecRule RESPONSE_HEADERS:Set-Cookie "(?i:(jsessionid).*?=([^\s].*?)\;\s?)" "phase:5,id:11100,t:none,pass,log,capture, \
  msg:'Auth-Bypass: Appl created session. Initializing parallel session.',setsid:%{TX.2},setvar:session.initok=1"

# Auth Bypass: Session Initialization (assign request an existing session)
SecRule REQUEST_FILENAME "@beginsWith /login.do" "id:11200,nolog,pass,skipAfter:END_AUTH_BYPASS_SESSION_INIT, \
  msg:'Auth-Bypass: Login request aimed to obtain serverside jsessionid. Skipping session check.'"

SecRule REQUEST_COOKIES:JSESSIONID "!^$" "id:11201,t:none,pass,setsid:%{REQUEST_COOKIES.JSESSIONID}, \
  msg:'Auth-Bypass: Initializing session based on application cookie jsessionid'"

SecRule &SESSION:INITOK "!@eq 1" "id:11202,t:none,deny,status:403, \
  msg:'Auth-Bypass: Failed check for appl creation of session. Cookie missing? Client session fixation? Denying access.'"

SecMarker END_AUTH_BYPASS_SESSION_INIT

# Auth Bypass: Extracting csv files of session out of html responses
SecRule REQUEST_URI "!^/display-csv-link.do" "phase:4,id:11300,pass,log,skip:1, \
  msg:'Auth Bypass: Uninteresting URI. Skipping the entering of csv file to the session'"

SecRule RESPONSE_BODY "downloads/(.*?.csv)" "phase:4,id:11301,capture,setvar:session.files='%{session.files};%{TX.1}' \
  msg:'Auth-Bypass: Adding csv file to session (%{TX.1})',pass,log"

# Auth Bypass: Checking authentication / authorisation of file download
SecRule REQUEST_URI "!^(/downloads/.*csv)$" "id:11400,pass,log,skipAfter:END_AUTH_BYPASS_CHECK, \
  msg:'Auth-Bypass: Uninteresting URI. Skipping download check.'"

SecAction "id:11401,pass,log,msg:'Auth-Bypass: Perform auth check. Files in session: %{session.files}'"

SecRule REQUEST_URI "^/downloads/(.*?.csv)$" "id:11402,pass,log,chain,capture,skipAfter:END_AUTH_BYPASS_CHECK, \
  msg:'Auth-Bypass: Auth check OK. Requested file is part files displayed to client during of session (%{TX.1})'"
  SecRule SESSION:FILES "%{TX.1}" ""

SecAction "id:11403,deny,status:403,log, \
  msg:'Auth-Bypass: Bypass attempt detected. Requested file NOT part of session files. Denying access'"

SecMarker END_AUTH_BYPASS_CHECK
```

Example #2 – Fix an Authentication Bypass Vulnerability (4 of 7)

```
# Auth Bypass: Serverside Session Creation

SecRule RESPONSE_HEADERS:Set-Cookie "(?i:(jsessionid).*?=([^\s].*?)\;\s?)" "phase:5,id:11100,t:none,pass,log,capture, \
  msg:'Auth-Bypass: Appl created session. Initializing parallel session.',setsid:%{TX.2},setvar:session.initok=1"

# Auth Bypass: Session Initialization (assign request an existing session)

SecRule REQUEST_FILENAME "@beginsWith /login.do" "id:11200,nolog,pass,skipAfter:END_AUTH_BYPASS_SESSION_INIT, \
  msg:'Auth-Bypass: Login request aimed to obtain serverside jsessionid. Skipping session check.'"

SecRule REQUEST_COOKIES:JSESSIONID "!^$" "id:11201,t:none,pass,setsid:%{REQUEST_COOKIES.JSESSIONID}, \
  msg:'Auth-Bypass: Initializing session based on application cookie jsessionid'"

SecRule &SESSION:INITOK "!@eq 1" "id:11202,t:none,deny,status:403, \
  msg:'Auth-Bypass: Failed check for appl creation of session. Cookie missing? Client session fixation? Denying access.'"

SecMarker END_AUTH_BYPASS_SESSION_INIT

# Auth Bypass: Extracting csv files of session out of html responses

SecRule REQUEST_URI "!^/display-csv-link.do" "phase:4,id:11300,pass,log,skip:1, \
  msg:'Auth Bypass: Uninteresting URI. Skipping the entering of csv file to the session'"

SecRule RESPONSE_BODY "downloads/(.*?.csv)" "phase:4,id:11301,capture,setvar:session.files='%{session.files};%{TX.1}' \
  msg:'Auth-Bypass: Adding csv file to session (%{TX.1})',pass,log"

# Auth Bypass: Checking authentication / authorisation of file download

SecRule REQUEST_URI "!^(/downloads/.*csv)$" "id:11400,pass,log,skipAfter:END_AUTH_BYPASS_CHECK, \
  msg:'Auth-Bypass: Uninteresting URI. Skipping download check.'"

SecAction "id:11401,pass,log,msg:'Auth-Bypass: Perform auth check. Files in session: %{session.files}'"

SecRule REQUEST_URI "^/downloads/(.*?.csv)$" "id:11402,pass,log,chain,capture,skipAfter:END_AUTH_BYPASS_CHECK, \
  msg:'Auth-Bypass: Auth check OK. Requested file is part files displayed to client during of session (%{TX.1})'"
  SecRule SESSION:FILES "%{TX.1}" ""

SecAction "id:11403,deny,status:403,log, \
  msg:'Auth-Bypass: Bypass attempt detected. Requested file NOT part of session files. Denying access'"

SecMarker END_AUTH_BYPASS_CHECK
```

Example #2 – Fix an Authentication Bypass Vulnerability (5 of 7)

```
# Auth Bypass: Serverside Session Creation

SecRule RESPONSE_HEADERS:Set-Cookie "(?i:(jsessionid).*?=([^\s].*?)\;\s?)" "phase:5,id:11100,t:none,pass,log,capture, \
  msg:'Auth-Bypass: Appl created session. Initializing parallel session.',setsid:%{TX.2},setvar:session.initok=1"

# Auth Bypass: Session Initialization (assign request an existing session)

SecRule REQUEST_FILENAME "@beginsWith /login.do" "id:11200,nolog,pass,skipAfter:END_AUTH_BYPASS_SESSION_INIT, \
  msg:'Auth-Bypass: Login request aimed to obtain serverside jsessionid. Skipping session check.'"

SecRule REQUEST_COOKIES:JSESSIONID "!^$" "id:11201,t:none,pass,setsid:%{REQUEST_COOKIES.JSESSIONID}, \
  msg:'Auth-Bypass: Initializing session based on application cookie jsessionid'"

SecRule &SESSION:INITOK "!@eq 1" "id:11202,t:none,deny,status:403, \
  msg:'Auth-Bypass: Failed check for appl creation of session. Cookie missing? Client session fixation? Denying access.'"

SecMarker END_AUTH_BYPASS_SESSION_INIT

# Auth Bypass: Extracting csv files of session out of html responses
SecRule REQUEST_URI "!^/display-csv-link.do" "phase:4,id:11300,pass,log,skip:1, \
  msg:'Auth Bypass: Uninteresting URI. Skipping the entering of csv file to the session'"

SecRule RESPONSE_BODY "downloads/(.*?.csv)" "phase:4,id:11301,capture,setvar:session.files='%{session.files};%{TX.1}' \
  msg:'Auth-Bypass: Adding csv file to session (%{TX.1})',pass,log"

# Auth Bypass: Checking authentication / authorisation of file download

SecRule REQUEST_URI "!^(/downloads/.*csv)$" "id:11400,pass,log,skipAfter:END_AUTH_BYPASS_CHECK, \
  msg:'Auth-Bypass: Uninteresting URI. Skipping download check.'"

SecAction "id:11401,pass,log,msg:'Auth-Bypass: Perform auth check. Files in session: %{session.files}'"

SecRule REQUEST_URI "^/downloads/(.*?.csv)$" "id:11402,pass,log,chain,capture,skipAfter:END_AUTH_BYPASS_CHECK, \
  msg:'Auth-Bypass: Auth check OK. Requested file is part files displayed to client during of session (%{TX.1})'"
  SecRule SESSION:FILES "%{TX.1}" ""

SecAction "id:11403,deny,status:403,log, \
  msg:'Auth-Bypass: Bypass attempt detected. Requested file NOT part of session files. Denying access'"

SecMarker END_AUTH_BYPASS_CHECK
```

Example #2 – Fix an Authentication Bypass Vulnerability (6 of 7)

```
# Auth Bypass: Serverside Session Creation

SecRule RESPONSE_HEADERS:Set-Cookie "(?i:(jsessionid).*?=([^\s].*?)\;\s?)" "phase:5,id:11100,t:none,pass,log,capture, \
  msg:'Auth-Bypass: Appl created session. Initializing parallel session.',setsid:%{TX.2},setvar:session.initok=1"

# Auth Bypass: Session Initialization (assign request an existing session)

SecRule REQUEST_FILENAME "@beginsWith /login.do" "id:11200,nolog,pass,skipAfter:END_AUTH_BYPASS_SESSION_INIT, \
  msg:'Auth-Bypass: Login request aimed to obtain serverside jsessionid. Skipping session check.'"

SecRule REQUEST_COOKIES:JSESSIONID "!^$" "id:11201,t:none,pass,setsid:%{REQUEST_COOKIES.JSESSIONID}, \
  msg:'Auth-Bypass: Initializing session based on application cookie jsessionid'"

SecRule &SESSION:INITOK "!@eq 1" "id:11202,t:none,deny,status:403, \
  msg:'Auth-Bypass: Failed check for appl creation of session. Cookie missing? Client session fixation? Denying access.'"

SecMarker END_AUTH_BYPASS_SESSION_INIT

# Auth Bypass: Extracting csv files of session out of html responses

SecRule REQUEST_URI "!^/display-csv-link.do" "phase:4,id:11300,pass,log,skip:1, \
  msg:'Auth Bypass: Uninteresting URI. Skipping the entering of csv file to the session'"

SecRule RESPONSE_BODY "downloads/(.*?.csv)" "phase:4,id:11301,capture,setvar:session.files='%{session.files};%{TX.1}' \
  msg:'Auth-Bypass: Adding csv file to session (%{TX.1})',pass,log"

# Auth Bypass: Checking authentication / authorisation of file download

SecRule REQUEST_URI "!^(/downloads/.*csv)$" "id:11400,pass,log,skipAfter:END_AUTH_BYPASS_CHECK, \
  msg:'Auth-Bypass: Uninteresting URI. Skipping download check.'"

SecAction "id:11401,pass,log,msg:'Auth-Bypass: Perform auth check. Files in session: %{session.files}'"

SecRule REQUEST_URI "^/downloads/(.*?.csv)$" "id:11402,pass,log,chain,capture,skipAfter:END_AUTH_BYPASS_CHECK, \
  msg:'Auth-Bypass: Auth check OK. Requested file is part files displayed to client during of session (%{TX.1})'"
  SecRule SESSION:FILES "%{TX.1}" ""

SecAction "id:11403,deny,status:403,log, \
  msg:'Auth-Bypass: Bypass attempt detected. Requested file NOT part of session files. Denying access'"

SecMarker END_AUTH_BYPASS_CHECK
```

Example #2 – Fix an Authentication Bypass Vulnerability (7 of 7)

```
# Auth Bypass: Serverside Session Creation
SecRule RESPONSE_HEADERS:Set-Cookie "(?i:(jsessionid).*?=([^\s].*?)\;\s?)" "phase:5,id:11100,t:none,pass,log,capture, \
  msg:'Auth-Bypass: Appl created session. Initializing parallel session.',setsid:%{TX.2},setvar:session.initok=1"

# Auth Bypass: Session Initialization (assign request an existing session)
SecRule REQUEST_FILENAME "@beginsWith /login.do" "id:11200,nolog,pass,skipAfter:END_AUTH_BYPASS_SESSION_INIT, \
  msg:'Auth-Bypass: Login request aimed to obtain serverside jsessionid. Skipping session check.'"

SecRule REQUEST_COOKIES:JSESSIONID "!^$" "id:11201,t:none,pass,setsid:%{REQUEST_COOKIES.JSESSIONID}, \
  msg:'Auth-Bypass: Initializing session based on application cookie jsessionid'"

SecRule &SESSION:INITOK "!@eq 1" "id:11202,t:none,deny,status:403, \
  msg:'Auth-Bypass: Failed check for appl creation of session. Cookie missing? Client session fixation? Denying access.'"

SecMarker END_AUTH_BYPASS_SESSION_INIT

# Auth Bypass: Extracting csv files of session out of html responses
SecRule REQUEST_URI "!^/display-csv-link.do" "phase:4,id:11300,pass,log,skip:1, \
  msg:'Auth Bypass: Uninteresting URI. Skipping the entering of csv file to the session'"

SecRule RESPONSE_BODY "downloads/(.*?.csv)" "phase:4,id:11301,capture,setvar:session.files='%{session.files};%{TX.1}' \
  msg:'Auth-Bypass: Adding csv file to session (%{TX.1})',pass,log"

# Auth Bypass: Checking authentication / authorisation of file download
SecRule REQUEST_URI "!^(/downloads/.*csv)$" "id:11400,pass,log,skipAfter:END_AUTH_BYPASS_CHECK, \
  msg:'Auth-Bypass: Uninteresting URI. Skipping download check.'"

SecAction "id:11401,pass,log,msg:'Auth-Bypass: Perform auth check. Files in session: %{session.files}'"

SecRule REQUEST_URI "^/downloads/(.*?.csv)$" "id:11402,pass,log,chain,capture,skipAfter:END_AUTH_BYPASS_CHECK, \
  msg:'Auth-Bypass: Auth check OK. Requested file is part files displayed to client during of session (%{TX.1})'"
  SecRule SESSION:FILES "%{TX.1}" ""

SecAction "id:11403,deny,status:403,log, \
  msg:'Auth-Bypass: Bypass attempt detected. Requested file NOT part of session files. Denying access'"

SecMarker END_AUTH_BYPASS_CHECK
```

Message from our CEO

```
SecRule YOUR_INTEREST ".*ModSec.*" "phase:1,id:1337,chain,log,\
    msg:'Found a guy with interest in ModSec.'"
    SecRule NEW_CHALLENGE_APPETITE "@ge 1" "mailto:folini@netnea.com"
```

You think defending is cool and
you would like to work with ModSecurity?

Then get in touch! Netnea is hiring!

Sources (all licensed via Creative Commons or in the public domain)

Chain: https://www.flickr.com/photos/antrophe/4566360507 (by James 2)

Overly Manly Man: Public Domain

Archive: https://www.flickr.com/photos/londonmatt/9420645961 (by Matt Brown)

Taxonomist: https://www.flickr.com/photos/11065470@N03/3691475990/ (by Nelson Stauffer)

Script modsec-positive-stat

https://github.com/Apache-Labor/labor/blob/master/labor-06

Dr. Christian Folini

christian.folini@netnea.com

www.christian-folini.ch

@ChrFolini

+41 79 220 23 76

OWASP Switzerland, 2014.11.12