

[Burp Suite] Session Handling Rules などの解説



abend@number3to4

※2015 年に作成した個人的な資料で、現在(2019 年 9 月時点)では、インターフェースや名称などが一部変わっている箇所もありますが、適宜読み替えてください。また、一部仕様がかわっている可能性もあるため、予めご了承ください。

1. セッション管理などの補助機能

Burp Suite には、セッション管理などを補助するための機能がいくつか用意されています。

- Macro
- Cookie jar
- Session Handling Rules

Macro は 1 つまたは複数のリクエストを Burp Suite に送信させるための仕組みです。ログインやショッピングサイトでの商品の購入処理などの複数のリクエストを Macro として登録することで、同様のリクエストを送信させ、ログインの試行や商品の購入処理を再現することが可能です。

Cookie jar は Macro などを利用して取得された Cookie をドメインおよび Cookie 名で管理します。取得された Cookie は、同一ドメインに対するリクエストへ自動的にセットされ、送信されます。Cookie jar で管理する Cookie は Set-Cookie のタイミングで更新されます。

Session Handling Rules は Macro や Cookie jar を制御するための仕組みです。Macro の実行や Cookie jar の更新範囲を指定して、セッションが無効となった場合に、ログインを試行することなど、条件に合わせたリクエストの送信が可能となります。

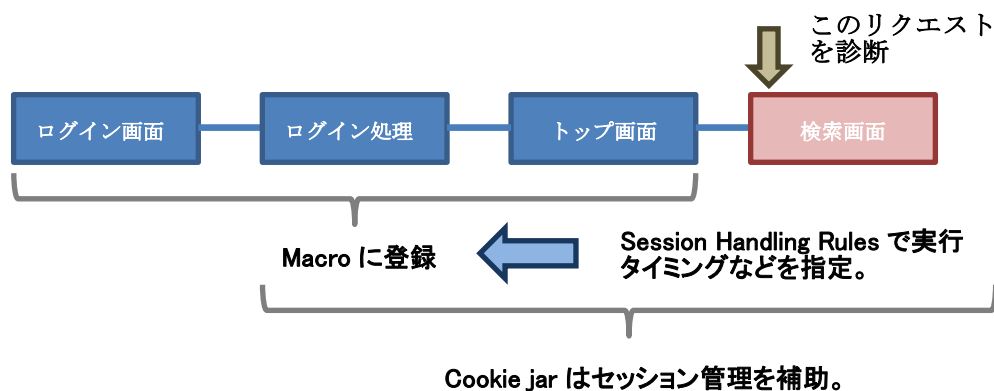


図 1

2. Macro

[Options]-[Session]-[Macro]で Macro の登録できます。Macro は1つまたは複数のリクエストを1つのセットとして登録して、登録されたリクエストを再送することができます。ログインなどの処理を再現させることができます。



図 2

[Add]を押下すると[Macro Recorder]が表示されます。ここで Macro の登録対象となるリクエストを選択します。選択可能なリクエストは[Proxy]-[HTTP history]に表示されている HTTP ログのみとなります。

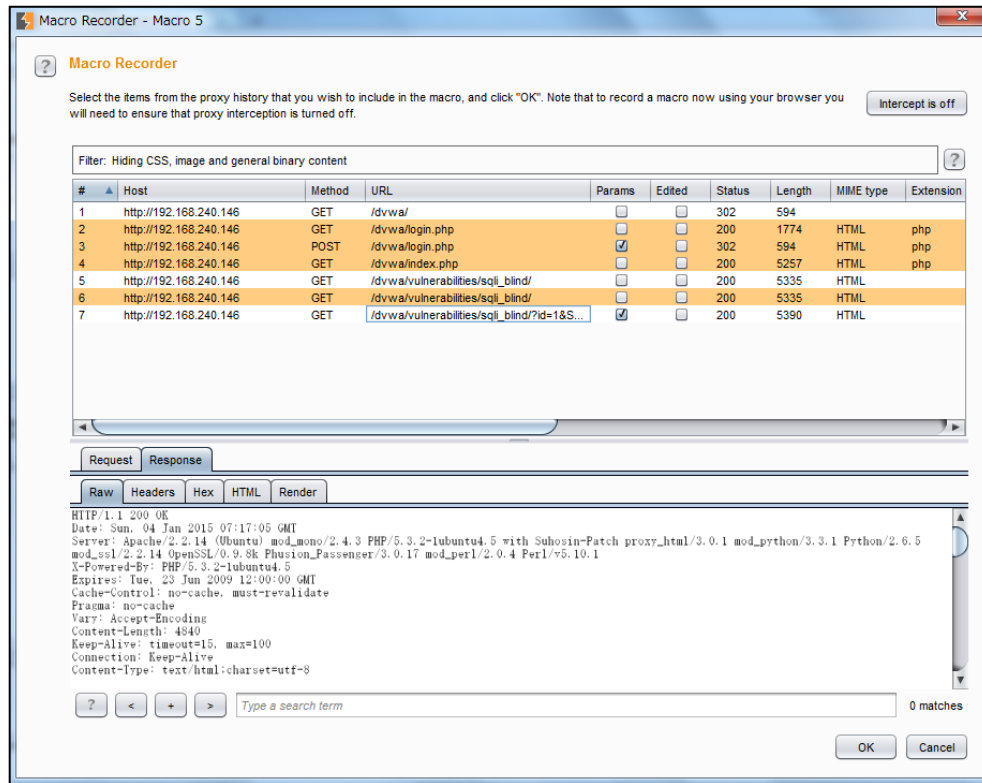


図 3

[Macro Editor]では、Macro として登録するリクエストに対してパラメータの追加や削除などの編集が可能です。下側のペインで直接リクエストを編集できます。ただ、パラメータの追加や削除を行った場合は、[Macro Editor]に再認識させるために[Re-analyze macro]を実行する必要があります。

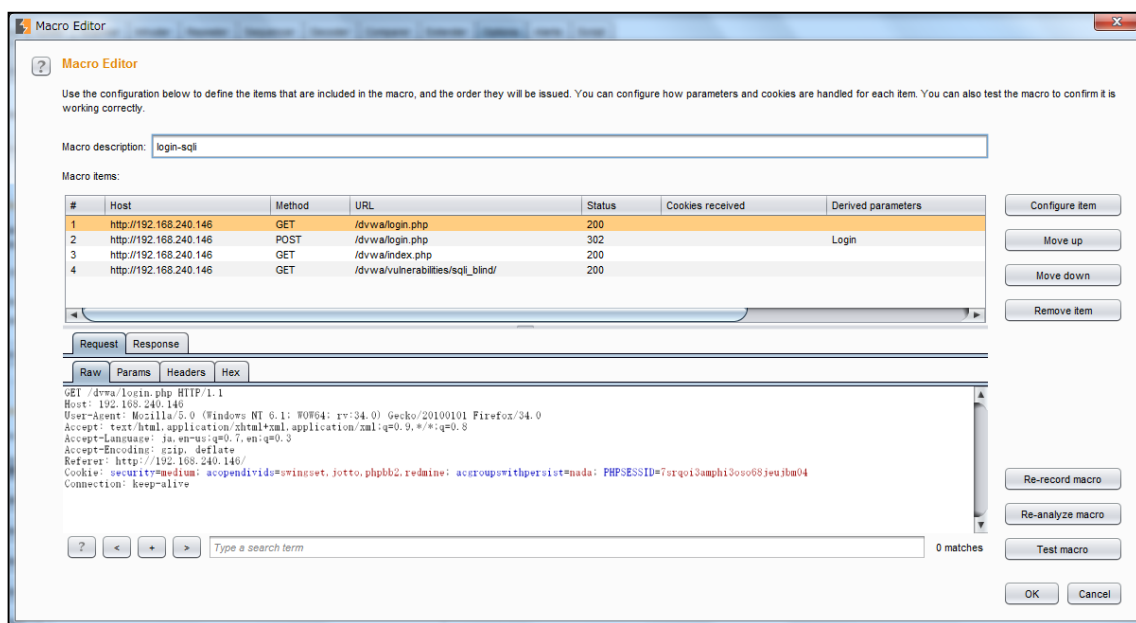


図 4

HTTP ログを選択し、[Configure item]を押下すると該当リクエストのパラメータや Cookie の設定できます。[Re-record macro]は、[Macro Recorder]が表示されて再度対象となる HTTP ログの再選択が可能となります。[Re-analyze macro]を実行すると、Macro にパラメータの追加・削除・変更が反映されます。

[Configure Macro Item]は、[Cookie handling]、[Parameter handling]、[Custom parameter locations in response]の 3 つのパートから構成されています。

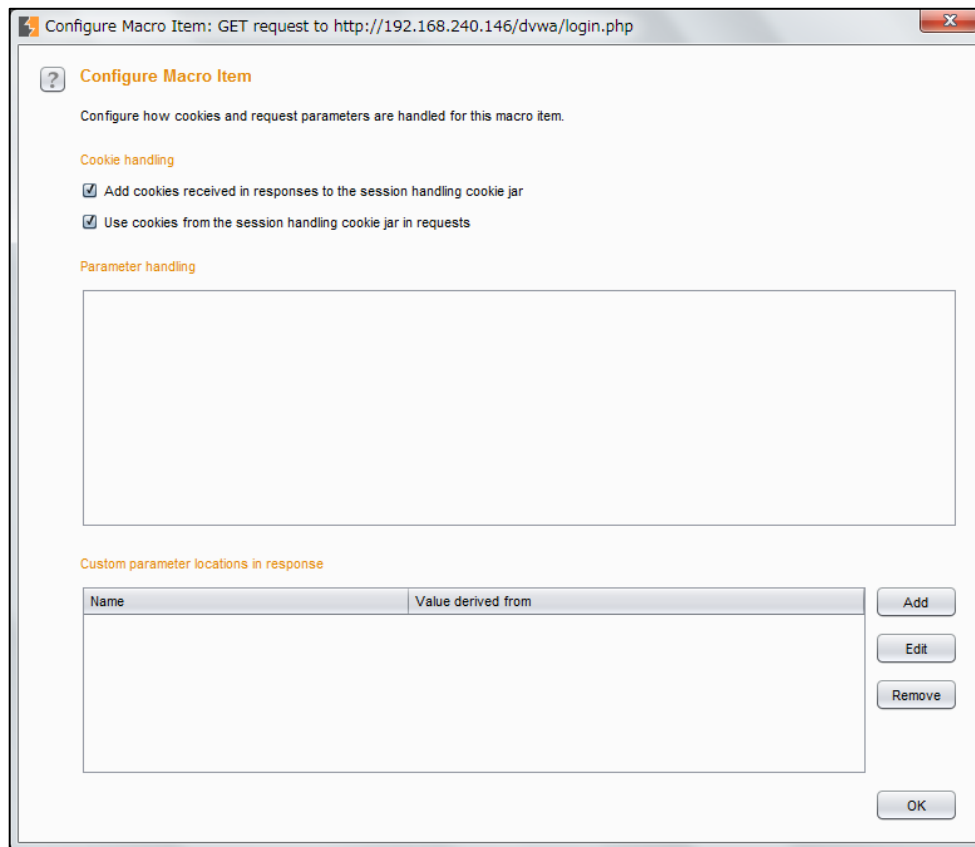


図 5

[Cookie handling]では、Cookie jar の処理を指定することができます。[Add cookies received in responses to the handling cookie jar]をチェックしている場合、レスポンスで Set-Cookie された場合に Cookie jar へ登録されます。既に登録済みの Cookie が存在している場合は、値が更新されます。[Use cookies from the session handling cookie jar in requests]をチェックしている場合、Cookie jar に登録されている Cookie がリクエストに反映されます。

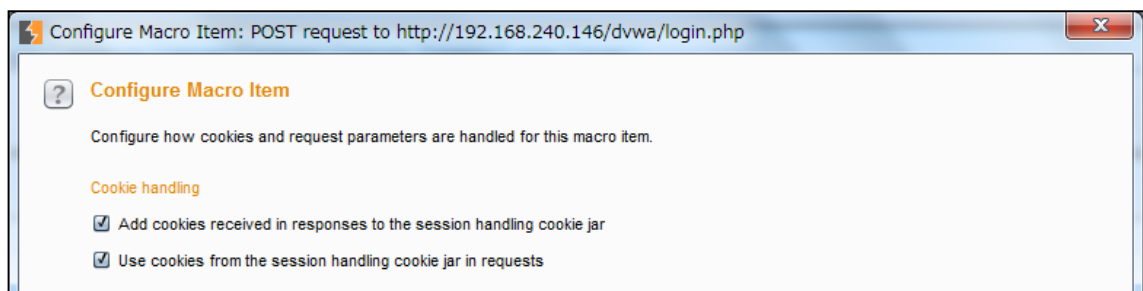


図 6

	[Add cookies received ...]を有効	[Add cookies received ...]を無効
[Use cookies from...]を有効	Cookie jar の Cookie をリクエストに反映し、レスポンスで Set-Cookie された場合は Cookie jar に反映させる。	Cookie jar の Cookie をリクエストに反映する（レスポンスに Set-Cookieがあっても Cookie jar は更新しない）。
[Use cookies from...]を無効	Cookie jar の Cookie をリクエストに反映せず（Macro Editor 選択時の Cookie でリクエストする）、レスポンスで Set-Cookie された場合は Cookie jar に反映させる。	Macro Editor 選択時の Cookie でリクエストする（レスポンスに Set-Cookieがあっても Cookie jar は更新しない）。

表 1

[Parameter handling]はリクエストに存在する GET および POST パラメータが一覧表示されます。[Use preset value]は固定値として設定します。[Drive from prior response]はレスポンスから値を設定することが可能で、ワントタイムトークンなど値が可変であるパラメータに有効です。

The screenshot shows a window titled "Parameter handling". Inside, there are three rows of parameter configuration:

- username**: A dropdown menu is set to "Use preset value", and a text box next to it contains the value "admin".
- password**: A dropdown menu is set to "Use preset value", and a text box next to it contains the value "admin".
- Login**: A dropdown menu is set to "Derive from prior response", and a second dropdown menu next to it is set to "Response 1".

図 7

[Parameter handling]は、GET パラメータなのか POST パラメータなのか画面上では判断がつかず、同じ名前のパラメータが存在していた場合はパラメータの値で区別する必要があります。設定内容を確認は、[Test macro]を実施しリクエスト中の値を確認する必要があります。

[Custom parameter locations in response]で追加したいパラメータの設定ができます。

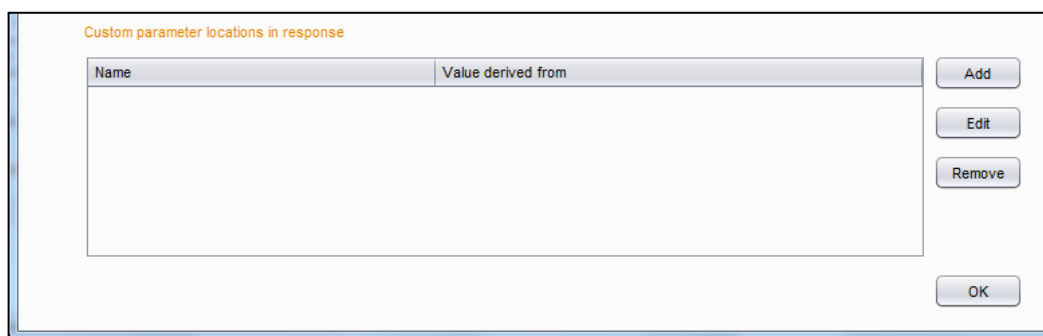


図 8

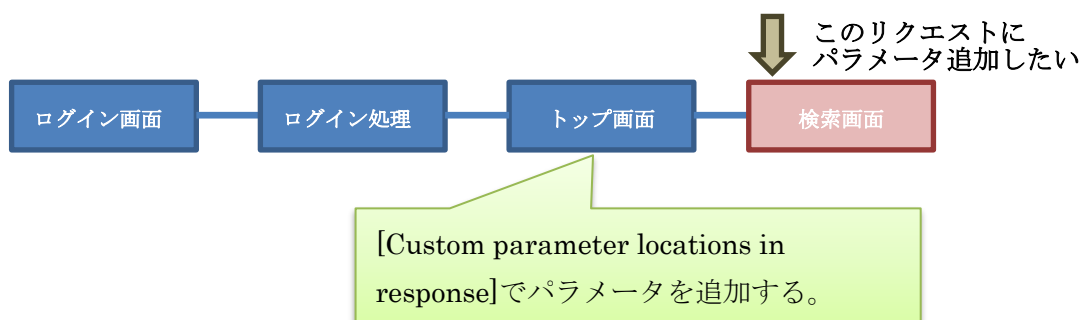


図 9

検索画面のリクエストにパラメータを追加する場合、トップ画面でパラメータ名や値を指定します。トップ画面で指定するパラメータ名と検索画面でリクエストに追加するパラメータ名は一致する必要があります。一致しない場合、値の設定は行われません。

図 9 の検索画面へのリクエストに GET パラメータ「`addition_param`」というパラメータを追加し、値はトップ画面の `title` タグ内の文字列をセットする場合の手順は、大きく 2 つあります。

- ① 追加したリクエストにパラメータを認識させる。
- ② 追加するパラメータの値を取得する。

- ① 追加したリクエストにパラメータを認識させる。

[Macro editor]で図 9 の検索画面の HTTP ログを選択し、[Request]タブから直接リクエストを修正します。検索画面には GET パラメータが存在していないため、URL「`http://192.168.240.146/dvwa/vulnerabilities/sqli_blind/`」の末尾に「`?addition_param=`」を追加します。



图 11

8 / 42

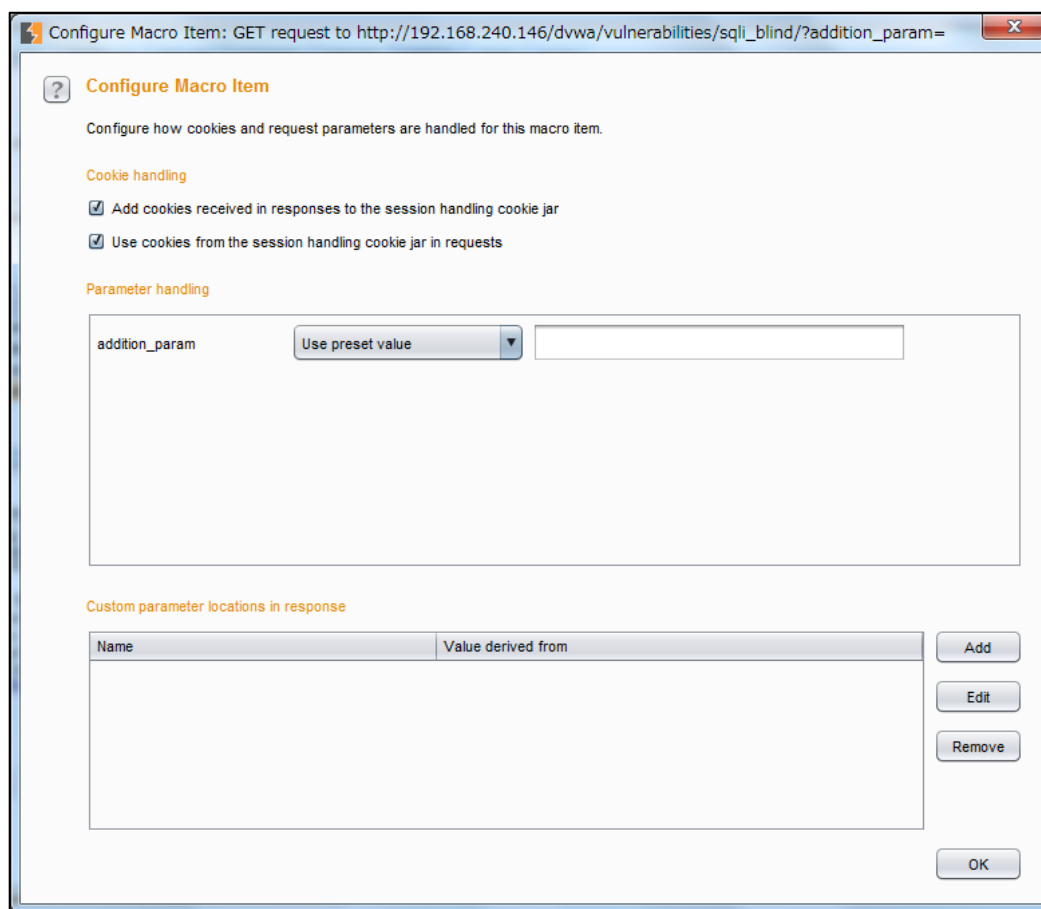


図 12

[Derive from prior response]を選択し、トップ画面を指す[Response 3]を指定します。
[Response X]の X は、[Macro editor]の[#]の番号を指しており、「ログイン処理」のレスポンスから値を取得したい場合は、[Response 2]を選択する必要があります。

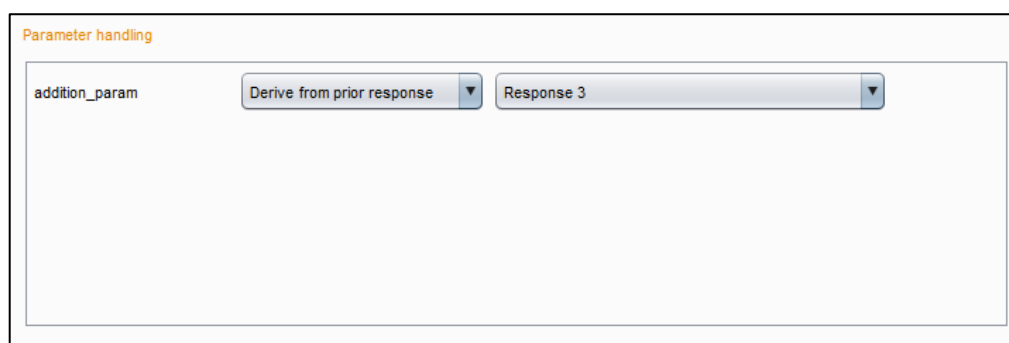
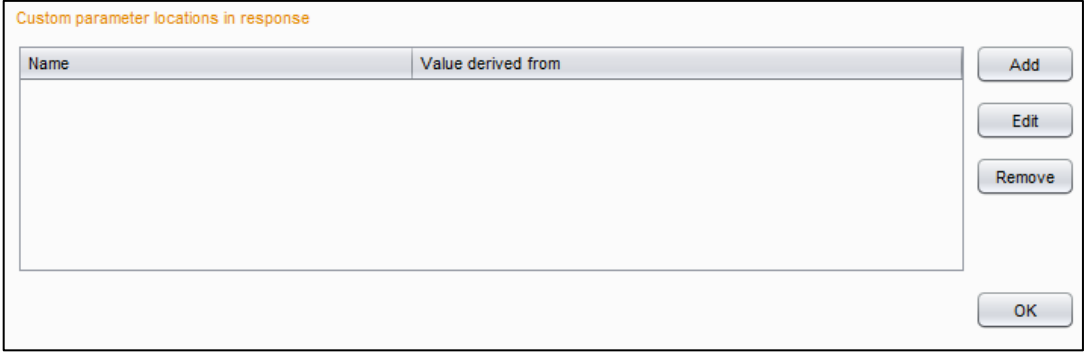


図 13

- ② 追加するパラメータの値を取得する。

トップ画面に該当する HTTP ログを選択し、[Configure item]を実行します。[Custom parameter locations in reponse]でパラメータ「addition_param」を作成します。



Name	Value derived from
------	--------------------

Add
Edit
Remove
OK

図 14

[Parameter name]に「addition_param」を入力します。パラメータ「addition_param」の値は、[Define start and end]を選択している場合、下の pane に表示されているレスポンスの title タグ内の文字列を選択すると自動的に値の取得ルールを設定してくれます。[Extract from regex group]を選択した場合は、正規表現で値を指定することも可能です。

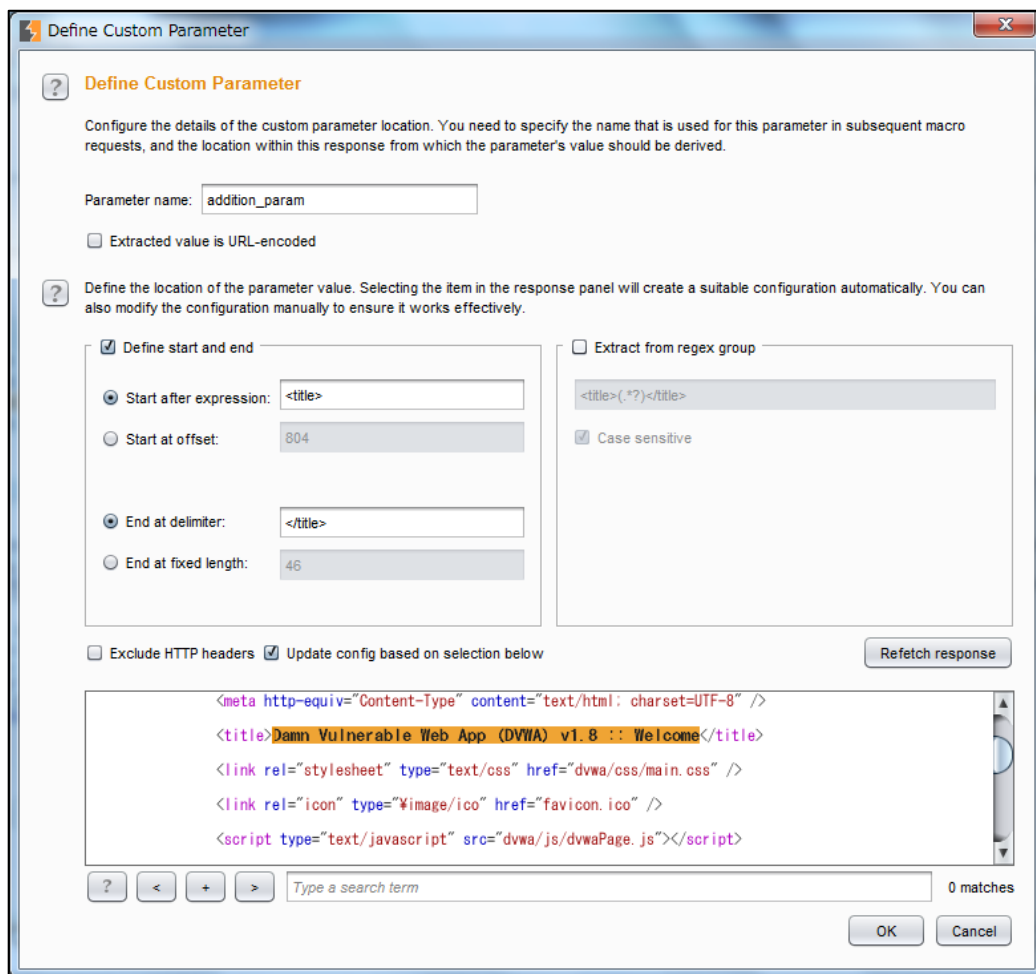


図 15

これでトップ画面の `title` タグ内の文字列をパラメータ「`addtion_param`」にセットして
 くれるようになります。設定がうまくできているか確認するために、[Macro editor]-[Test
 macro]を実行します。検索画面に該当する HTTP ログの[Derived parameters]に追加した
 「`addtion_param`」が出力されており、下の pane のリクエストでパラメータ
 「`addtion_param`」の値が `title` タグ内の文字列になっていることがわかります。値がセッ
 トされていない場合は、トップ画面および検索画面で指定するパラメータ名が誤っている
 か、追加するパラメータの値が正常に取得できていない可能性があります。

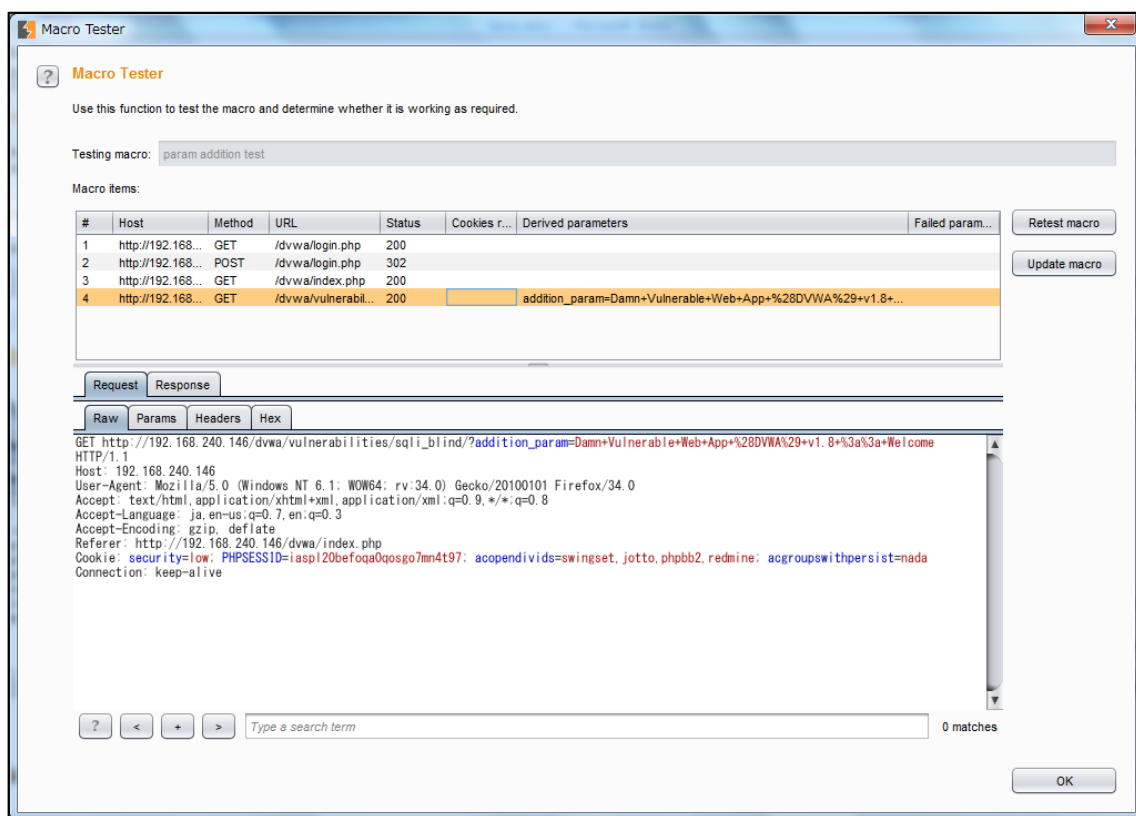


図 16

3. Cookie jar

Cookie jar は、Burp Suite でセッション管理を補助する機能で、Burp Suite を介してアクセスしたサイトの Cookie を共有する仕組みになっています。具体的には以下の 3 つの動作をします。

- ① 保存されている Cookie でドメインが一致する場合は送信する。
- ② Set-Cookie された Cookie が存在する場合、該当 Cookie の値を更新する。
- ③ Set-Cookie された Cookie が存在していない場合、その Cookie を登録する。

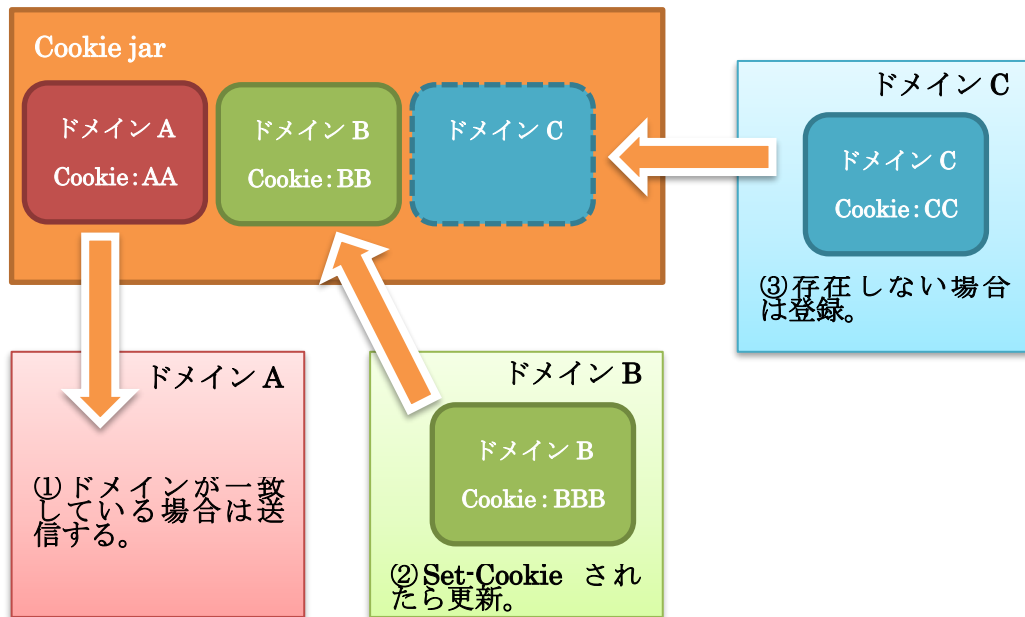


図 17

Cookie jar への更新を行う機能を指定することが可能です。[Options]-[Sessions]で更新を許可する機能を選択します。デフォルトでは、[Proxy]と[Spider]のみが有効になっており、この 2 つで Cookie の追加および更新が行われた場合に Cookie jar へ反映されます。

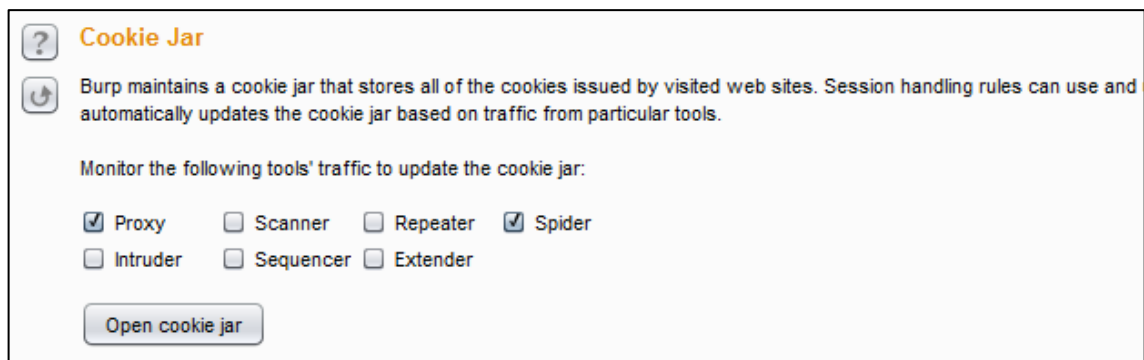


図 18

[Open cookie jar]を押下すると、Cookie jar に登録されている Cookie の一覧が表示されます。

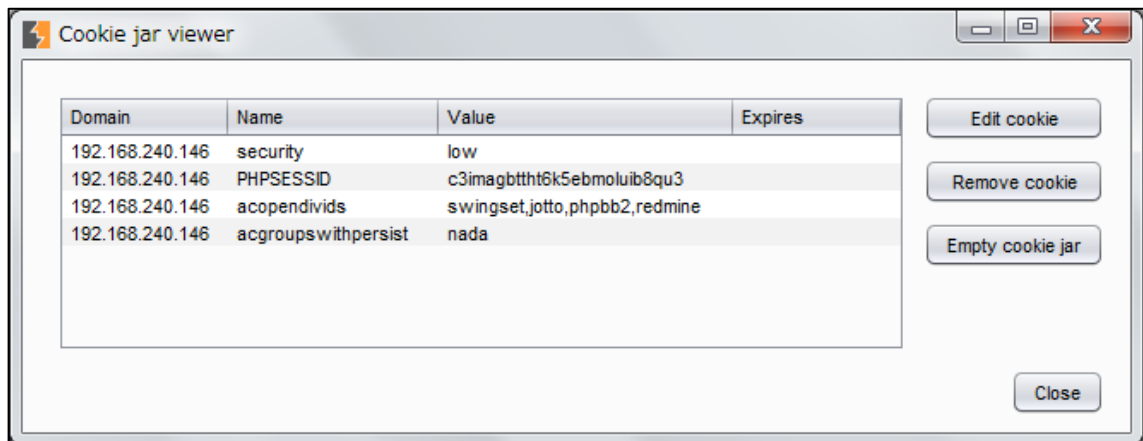


図 19

[Edit cookie]で登録されている Cookie の編集が可能です。編集可能な項目は、[Domain]、[Name]、[Value]です。任意の Cookie を選択し[Remove cookie]を実行すると、選択された Cookie のみ破棄することができます。[Empty cookie jar]を実行すると登録されているすべての Cookie が破棄されます。

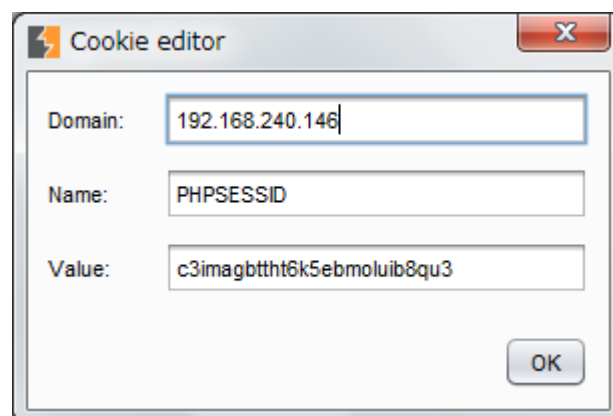


図 20

遷移が自動的に行えるようになるなど、Cookie jar は非常に有意義な機能ですが、大きな欠陥もあります。Cookie jar が、Cookie を識別する項目はドメイン名と Cookie 名だけしかないため、パスが異なる Cookie やプロトコルの違い（HTTP や HTTPS）が判別できません。

Cookie を HTTP、HTTPS と /、/sample でそれぞれ発行する図 21 のプログラム（値およびパスを表中の値に変更）を配置し、Cookie jar でどのように識別されるか試しました。

```
<?php
setcookie("cookie","値",0,"パス","",0,0)
?>
```

図 21

setcookie 関数の仕様は以下です。

```
bool setcookie ( string $name [, string $value [, int $expire = 0 [, string $path [, string $domain [, bool $secure = false [, bool $httponly = false ]]]]] )
```

name : クッキーの名前

value : クッキーの値

expire : クッキーの有効期限

path : サーバー上での、クッキーを有効としたいパス

domain : クッキーが有効なドメイン

secure : セキュア属性の有無

httponly : HttpOnly 属性の有無

図 22

パス/プロトコル	http	https
/	Cookie の値 : http_root	Cookie の値 : https_root
/sample	Cookie の値 : http_sample	Cookie の値 : https_sample

表 2

以下の順序でアクセスした際の Cookie jar に登録された Cookie が、図 23 から図 26 になります。

- ① プロトコルが http、パスが/にアクセス (図 23)
- ② プロトコルが https、パスが/にアクセス (図 24 図 19)
- ③ プロトコルが https、パスが/sample にアクセス (図 25)
- ④ プロトコルが http、パスが/sample にアクセス (図 26)

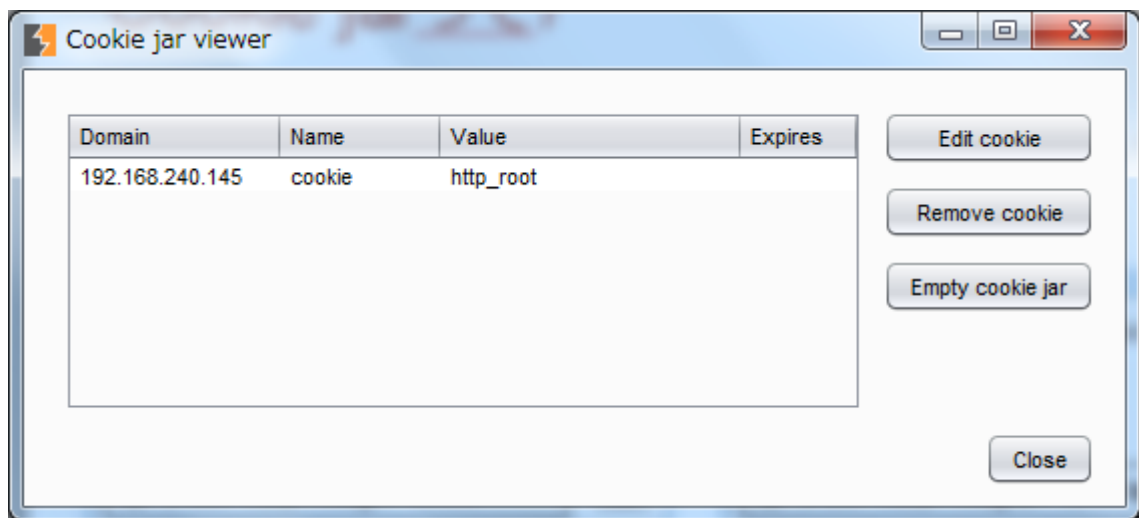


图 23

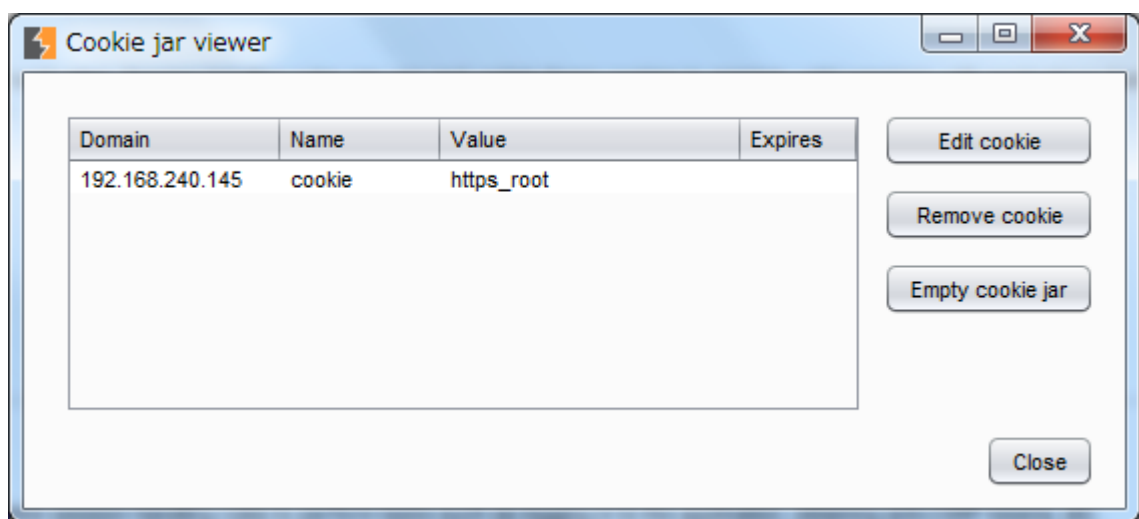


图 24

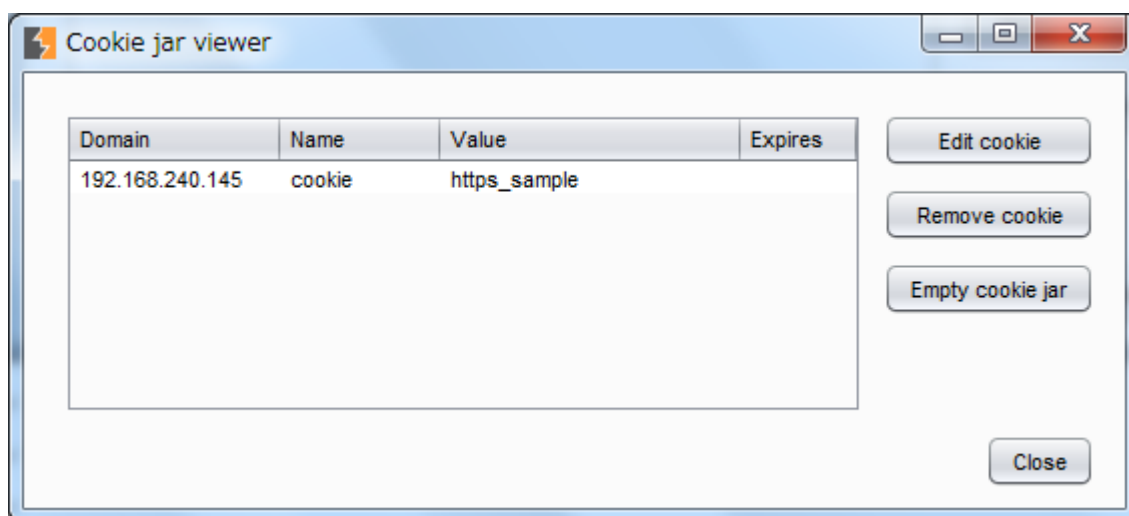


図 25

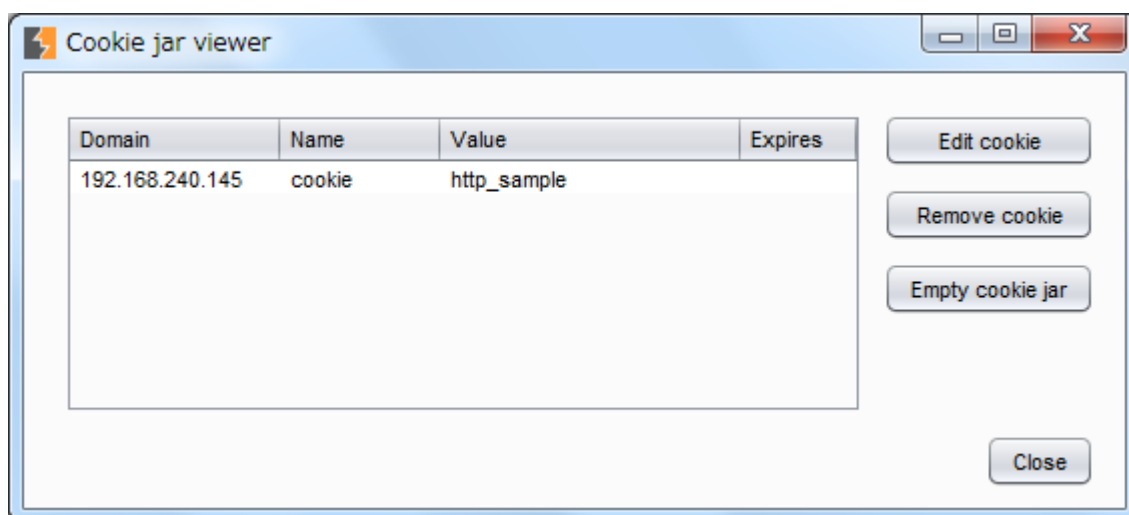


図 26

すべて同じ Cookie として Cookie jar が認識しているために、Cookie の値は「http_root」→「https_root」→「https_sample」→「http_sample」と変化していますが、それぞれの Cookie が追加されていないことがわかります。

この仕様により、同じドメインでパスが異なる同名の Cookie でセッション管理しているサイトの場合、Macro を用いた遷移が正常に行えない可能性があります。例として、シングルサイオンしてパスごとに各サービスを提供しているようなサイトなどがあげられます。各サービスで同じ Cookie 名を発行している場合、Cookie jar は1つの Cookie としてしか認識しないため、誤動作する可能性があります。

4. Session Handling Rules について

[Options]-[Session]-[Session Handling Rules]は、セッション管理方法や実行範囲などを設定可能です。Macro を活用することで、リクエストの送信を自動化し、手動で操作するリクエストを最小限にすることが可能です。また、レスポンスの内容を評価し、リクエストを送信することができるため、何らかの理由でセッションが破棄された場合に自動で認証処理を行い、有効なセッションを取得し診断を再開することも可能です。

[Session Handling Rules]で[Add]を実行することで Session Handling Rule を追加できます。[Edit]は作成した Session Handling Rule を編集することができます。[Remove]は作成した Session Handling Rule を削除します。[Duplicate]は作成された Session Handling Rule の複製をつくります。[Up]、[Down]で Session Handling Rule の評価順を変更することができます。[Session Handling Rules]は有効化されている Session Handling Rule を上から順番に評価します。

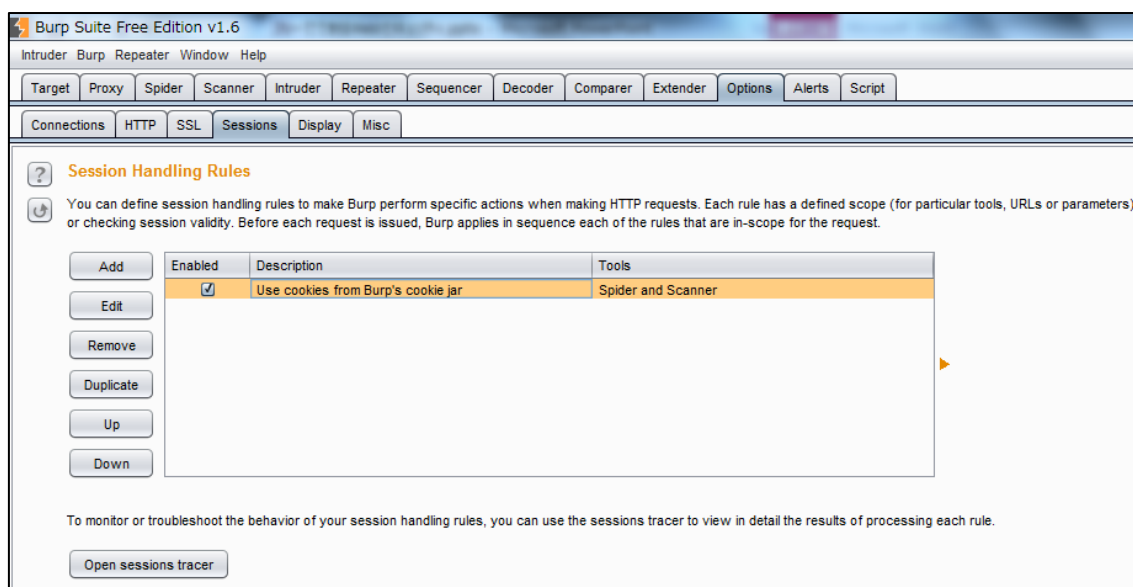


図 27

[Session handling rule editor]で動作のパターンおよび範囲を指定することで制御します。[Defaults]タブでは、[Rule description]に Session Handling Rule の説明を入力することができます。改行の入力も可能ですが、[Session Handling Rules]に一覧表示され際には改行が除去された文字列が表示されます。

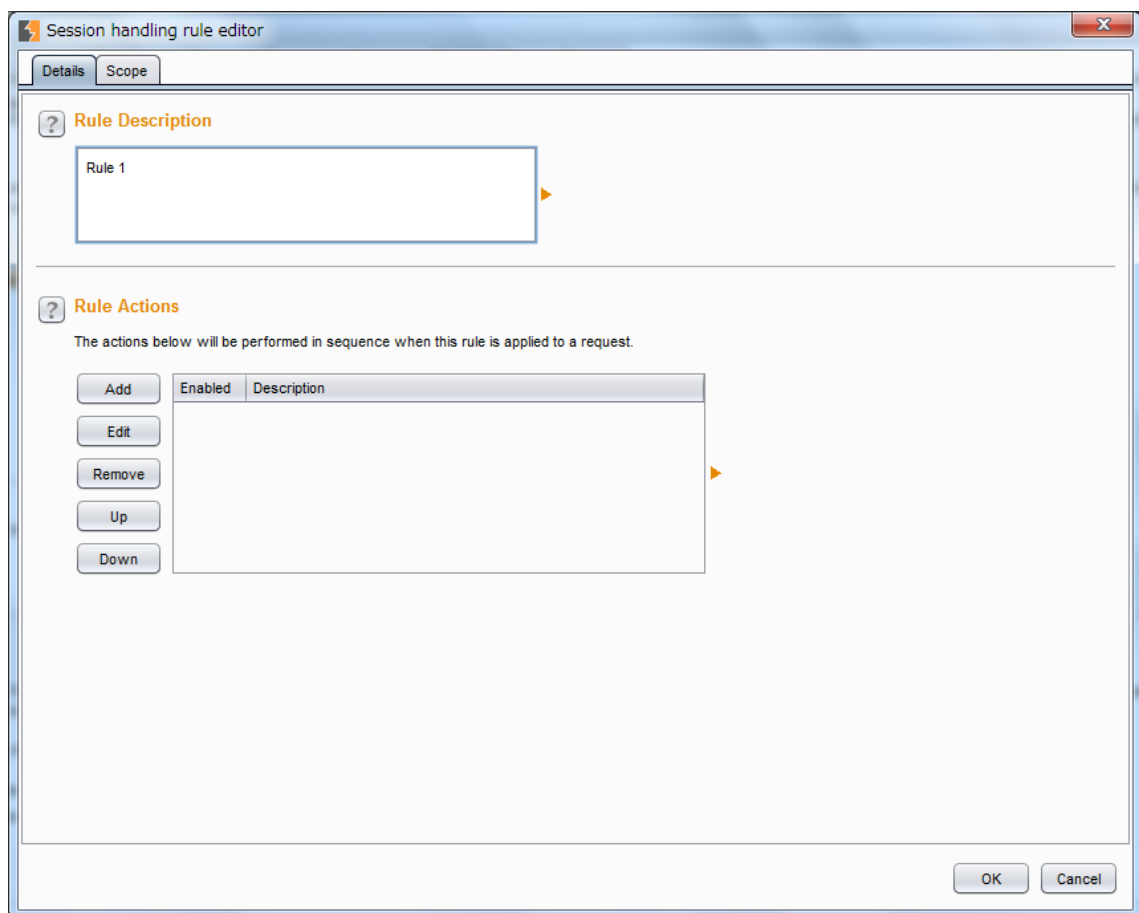


図 28

[Rule Actions]で動作パターンを設定します。[Rule Actions]で複数の動作パターンを設定することが可能で、上から順番に評価されます。設定できる動作パターンは以下があります。

- Use cookies from the session handling cookie jar
- Set a specific cookie or parameter value
- Check session is valid
- Prompt for in-browser session recovery
- Run a macro
- Run a post-request macro
- Invoke a Burp extension

[Rule Actions]で設定した動作パターンを処理した上で、Current Request が送信されます。

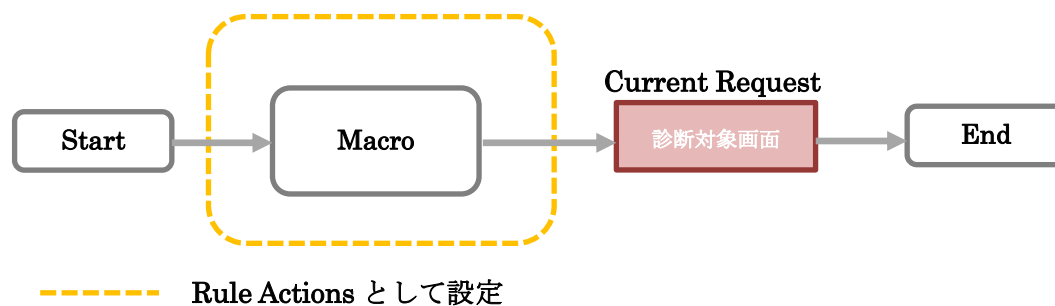


図 29

[Scope]は、[Tools scope]、[URL scope]、[Parameter scope]の3つから構成されています。[Tools scope]、[URL scope]、[Parameter scope]のそれぞれで指定する条件に合致した場合に Session Handling Rule が実行されます。

[Tools scope]は Burp Suite でどの機能に対して有効化するかを選択します。1つまたは複数選択することが可能です。何も選択しない場合、[Rule Actions]で設定した動作パターンは動作しません。

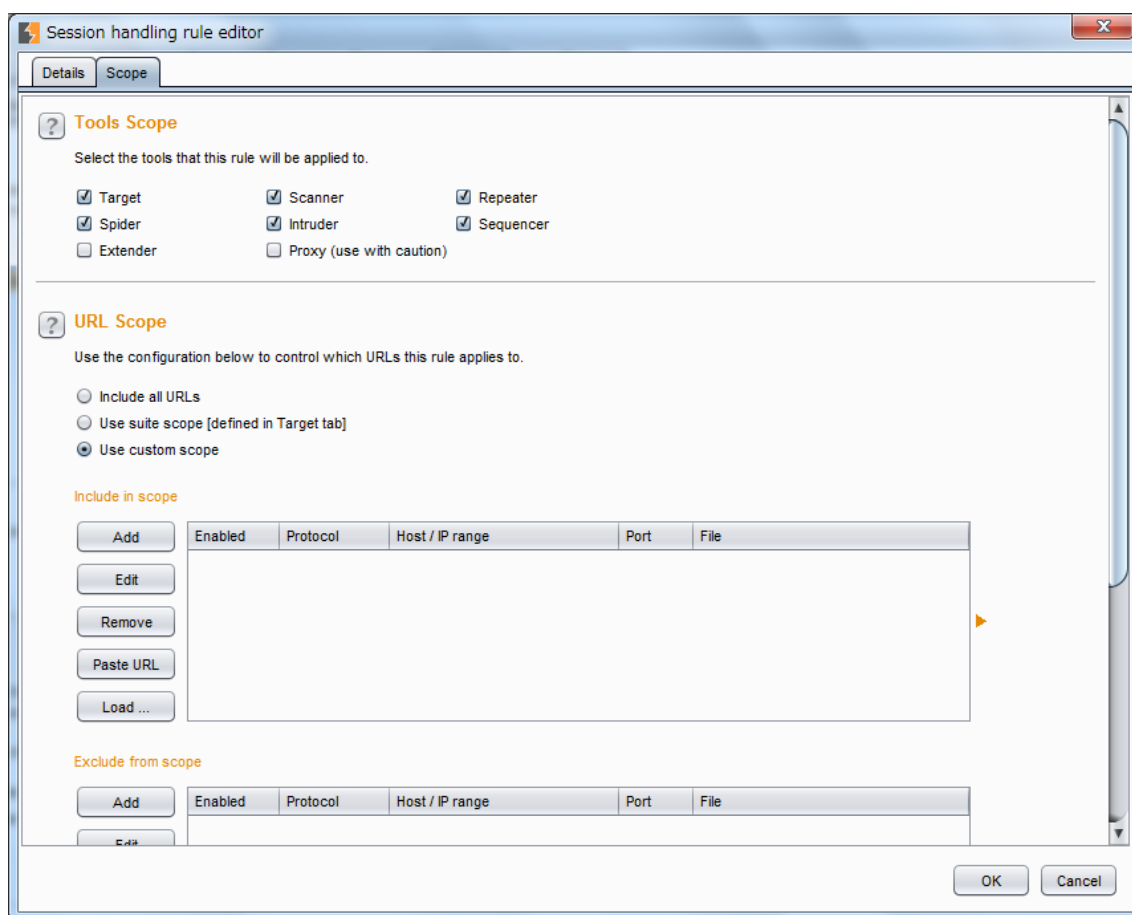


図 30

[URL scope]は[Include all URLs]、[Use suite scope]、[Use custom scope]のいずれかを選択する必要があります。初期値は[Use custom scope]になっています。

[Use custom scope] を指定する場合、[Include in scope] および[Exclude in scope] を入力します。[Include in scope]は、動作の対象とするホストや URL などを入力します。[Exclude in scope]は、動作の対象から除外するホストや URL などを入力します。[Include in scope]を指定しない場合、すべてのリクエストが該当しないため、Session Handling Rule は動作しません。

? URL Scope

Use the configuration below to control which URLs this rule applies to.

☐ Include all URLs
☐ Use suite scope [defined in Target tab]
☒ Use custom scope

Include in scope

Add

Edit

Remove

Paste URL

Load ...

Enabled	Protocol	Host / IP range	Port	File
---------	----------	-----------------	------	------

Exclude from scope

Add

Edit

Remove

Paste URL

Load ...

Enabled	Protocol	Host / IP range	Port	File
---------	----------	-----------------	------	------

図 31

初期設定時に[Include in scope]を何も入力していない場合、図 32 の警告が表示されます。ただし、入力した上で無効化している（チェックを外している）場合、警告は表示されないのです、注意が必要です。



図 32

[Include all URLs]を選択した場合、すべての URL が対象となります。[Use suite scope]は[Target]タブで指定した値を使用し、制御します。

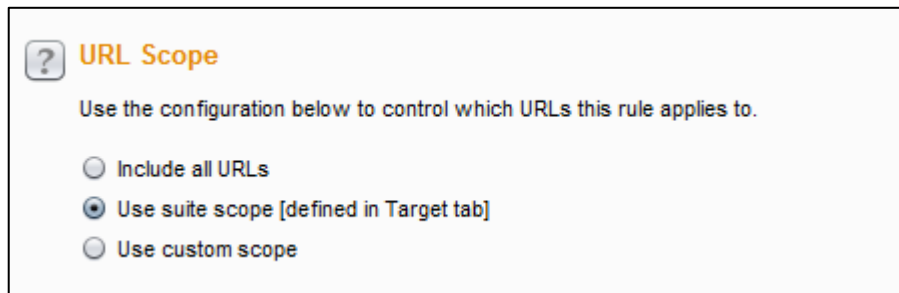


図 33

[Parameter Scope]は、指定するパラメータ名が含まれている場合に Session Handling Rule を動作させるか制御することができます。

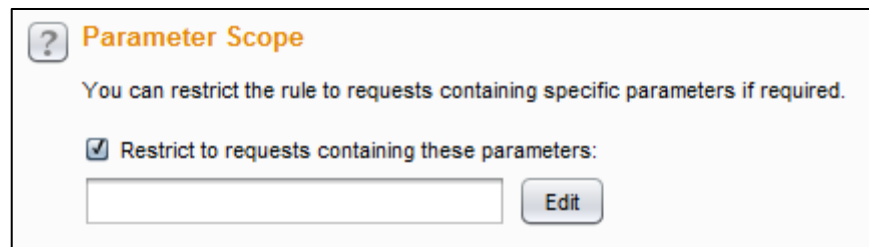


図 34

テキストボックスにパラメータ名を入力し、[Add]を実行すると対象となるパラメータ名が追加されます。[Paste]は、コピーした文字列をパラメータ名として登録できます。[Load]は、改行区切りのパラメーター一覧を読み込ませることもできます。[Remove]は該当するパラメータを選択し実行すると削除されます。[Clear]は、登録されているすべてのパラメータが削除されます。[Restrict to requests containing these parameters]を有効にし、図 35 でパラメータ名を何も指定しない場合、すべてのリクエストで有効であると評価されます。

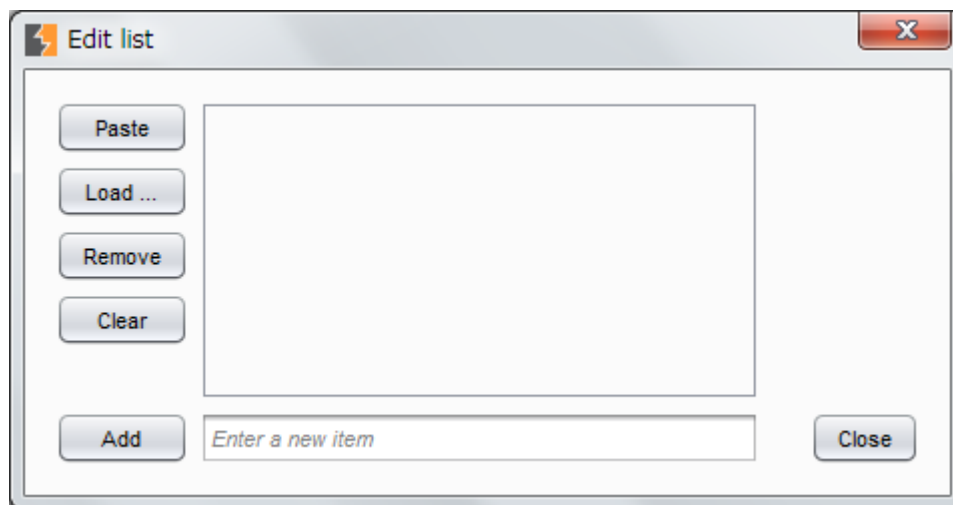


図 35

ここからは、[Rule Actions]で指定可能な動作パターンについて解説します。

[Set a specific cookie or parameter value]は、パラメータ名と値を指定することで指定されたパラメータ名の値を変更することが可能です。

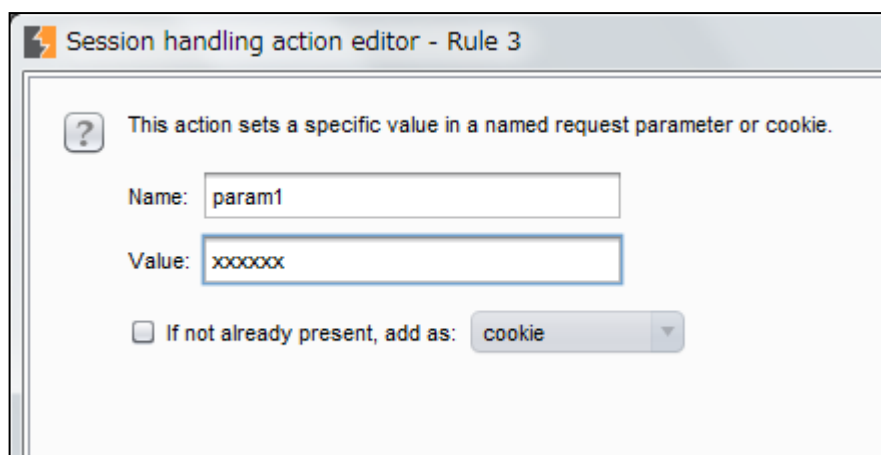


図 36

[If not already ...]を有効にすると、リクエスト中に指定したパラメータが存在しない場合に[URL parameter]、[body parameter]、[cookie]で指定した箇所にパラメータが追加されます。[If not already ...]で[body parameter]を GET リクエストで指定するとメソッドは GET のままですが、強制的にリクエストボディが追加されます。

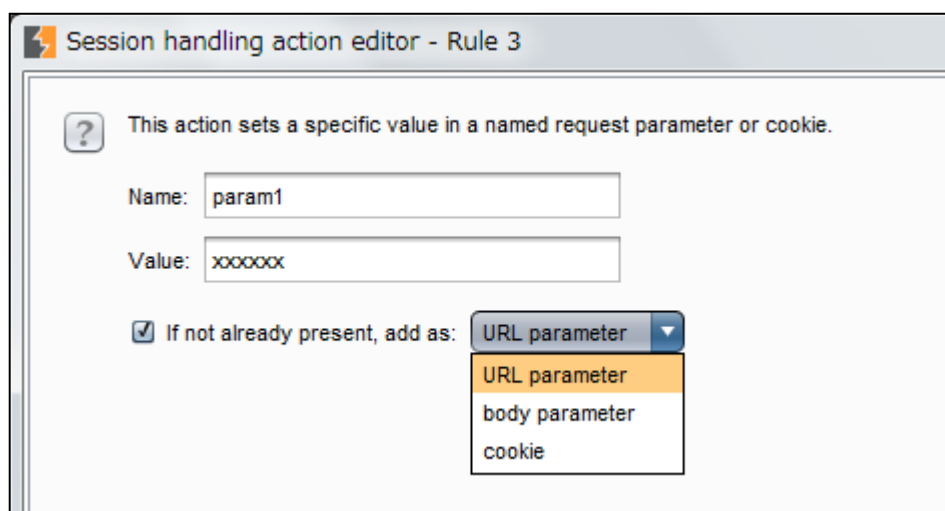


図 37

[Run macro]は、設定した 1 つまたは複数のリクエストを送信することが可能です。

図 38 の遷移するアプリケーションで「カート参照画面」を診断しようとした場合に、「ログイン画面」から「トップ画面」までの遷移を Macro として設定することで、Burp Suite が自動的にリクエストを送信します。そのため、「カート参照画面」のみのリクエストを送信すればよく、効率的な診断を実施することが可能となります。

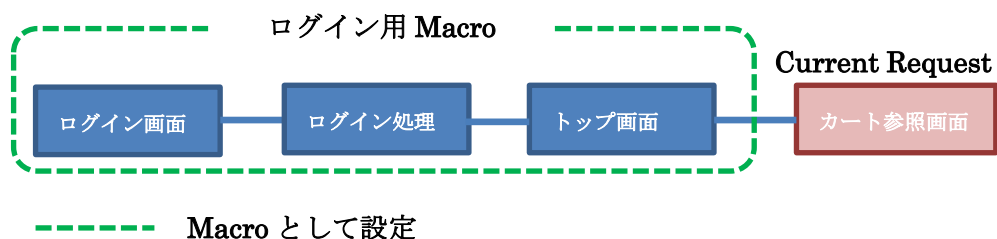


図 38

ログイン画面からトップ画面までを「ログイン用 Macro」として登録します。[Tools scope]などの実施条件が合致した場合、[Repeater]や[Intruder]などでログイン用 Macro は自動的にリクエストを送信し、Current Request に対する診断が実施できます。

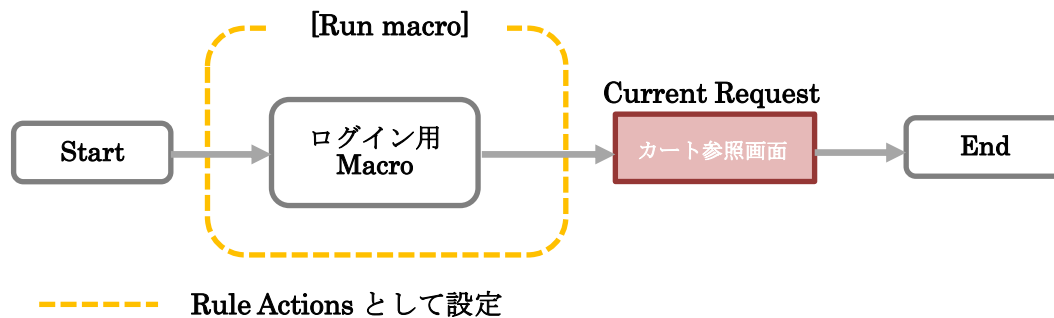


図 39

[Check session is valid]は、セッションの状況（有効なのか無効なのか）に応じて、アクションの実施など設定することができます。

ログインしていないと正常に「カート参照」画面へ遷移できないアプリケーションで「カート参照」画面を診断する場合、図 38 のように「ログイン」画面から「トップ画面」までを 1 つの Macro として登録し、[Run macro]を用いて実行するという方法もありますが、「カート参照」画面へ 1 つのリクエストを送信するために「ログイン」画面から「トップ」画面の 3 画面を常にアクセスする必要があります。そのために、診断対象への総アクセス数は 4 倍になってしまい、負荷をかけてしまう可能性があります。

[Check session is valid]は、「トップ」画面にアクセスし、セッションが有効か確認します。セッションが有効な場合は、「カート参照」画面へ診断を行います。セッションが無効の場合、ログイン処理を実行し、「カート参照」画面へ診断を行います。[Check session is valid]はセッションが有効な場合、「トップ」画面と「カート参照」画面の 2 つのアクセスのみとなるため、[Run macro]と比較すると診断対象への総アクセスは少なくなります。セッションが無効な場合にのみ、ログイン処理を実行しないため総アクセスが少なく適切な診断を実行できます。

図 40 のように「トップ」画面をセッション確認用 Macro とし、「ログイン」画面、「ログイン処理」画面をセッション有効化用 Macro として登録します。

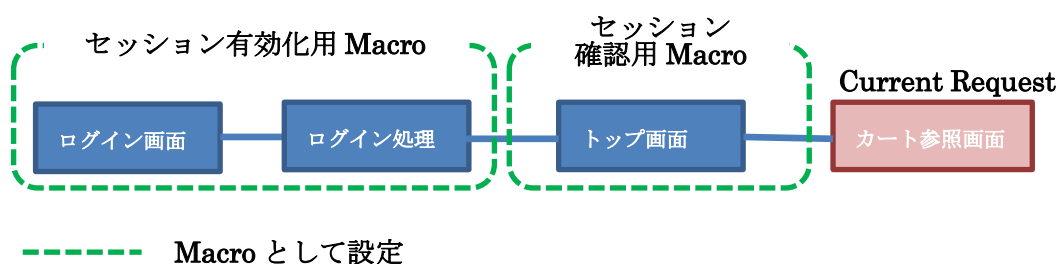


図 40

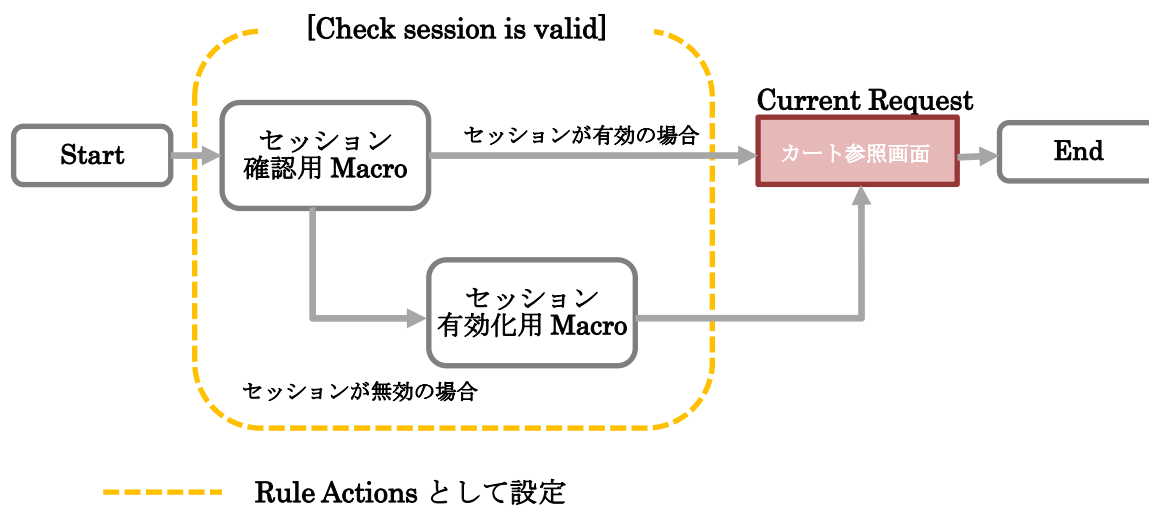


図 41

[Check session is valid]は、以下の 3 つのパートから構成されています。

- [Make request(s) to validate session]
- [Inspect response to determine session validity]
- [Define behaviour dependent on session validity]

[Make request(s) to validate session]は、セッション状況の確認を行うためのリクエストをどのタイミングで生成するか指定します。[Issue current request]または[Run macro]のいずれかを選択します。

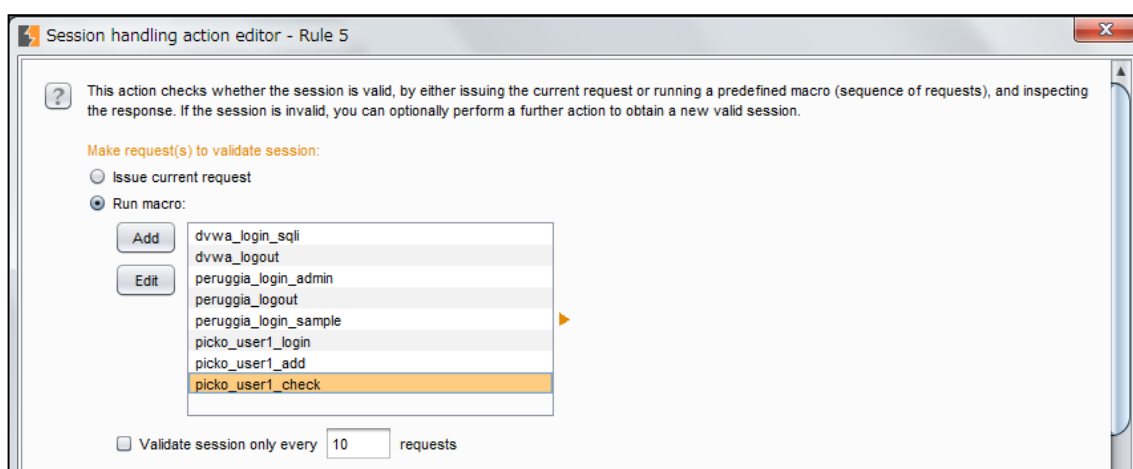


図 42

[Issue current request]を選択した場合、リクエストごとにセッションの状況を確認し、セッションを有効にするための Macro 実行などの後続処理を実施します。セッションが無効にならない限り、「カート参照」画面のみのリクエストしか発行しないため、図 41 の設定よりもリクエスト数が少なく、最小限のリクエストで診断が可能です。

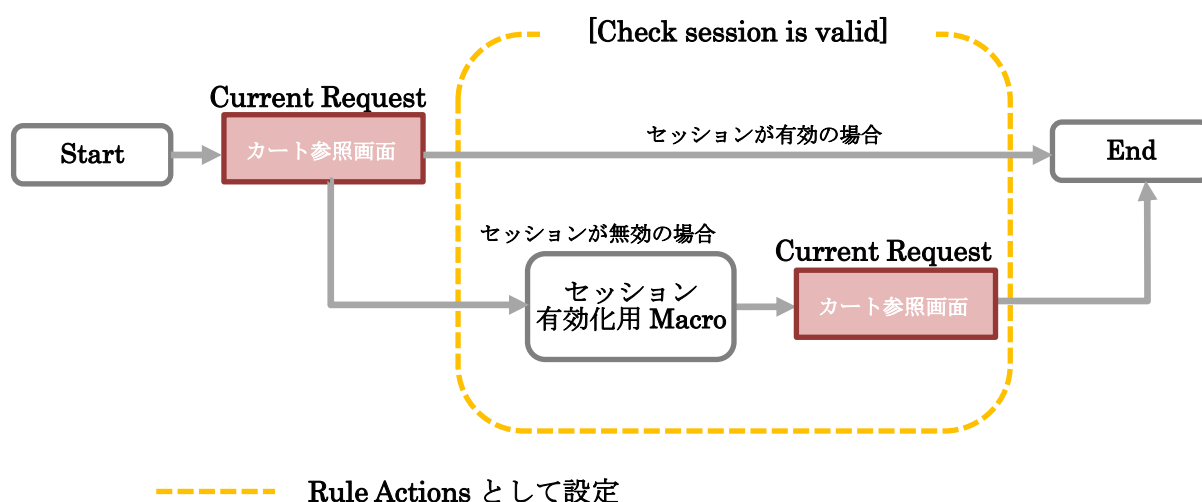


図 43

[Run macro]を選択した場合、セッション確認用 Macro にあたる Macro を指定します。[Validate session only every XX requests]を有効にすると、セッション確認用 Macro を何回ごとに実施するか指定できます。1 以下（負の数を含む）を指定すると毎回セッション確認用 Macro が実行されます。

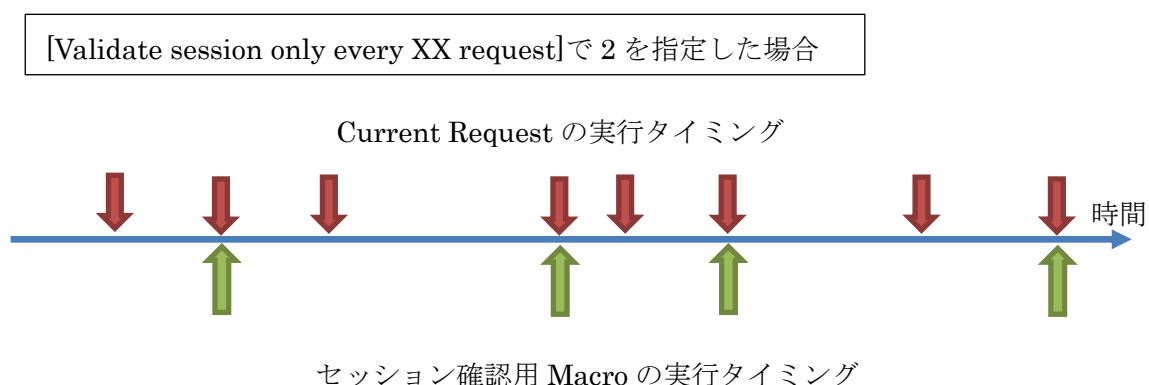
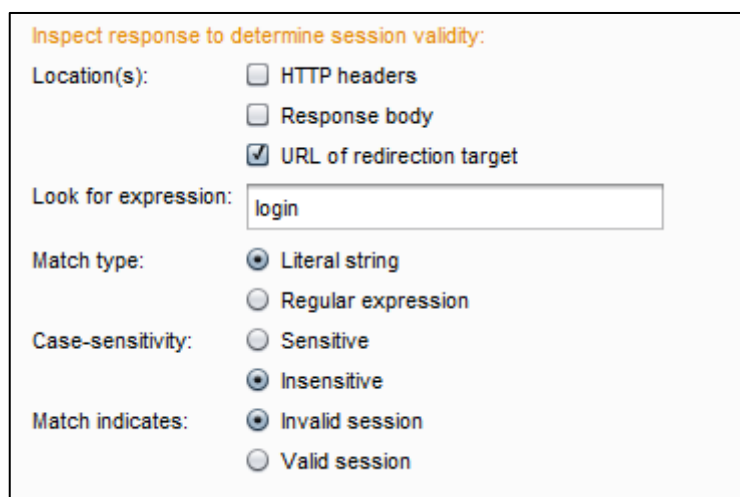


図 44

[Inspect response to determine session validity]は、セッションの有効性の確認方法について設定します。どの文字列が出力されていたら、セッションを有効または無効とする

のかを指定します。

設定項目は、[Locations(s)]、[Look for expression]、[Match type]、[Case-sensitivity]、[Match indicates]があります。[Locations(s)]は、[Look for expression]で指定する文字列がどこに出力されるか設定します。[Match type]は文字列マッチングの際に正規表現を使うかどうかを設定します。[Regular expression]を選択すると正規表現として評価されます。[Case-sensitivity]は、[Look for expression]で入力した文字列の大文字・小文字の区別をするか設定します。[Match indicates]は、[Locations(s)]、[Look for expression]などで設定した条件に合致した場合にセッションが無効(Invalid session)か有効か(Valid session)を設定します。



Inspect response to determine session validity:

Location(s): ☐ HTTP headers
☐ Response body
☒ URL of redirection target

Look for expression:

Match type: ☒ Literal string
☐ Regular expression

Case-sensitivity: ☐ Sensitive
☒ Insensitive

Match indicates: ☒ Invalid session
☐ Valid session

図 45

まだ記載途中！！設定例は生ログなどを用いてパターン化する。

```
HTTP/1.1 302 Found
Date: Sun, 08 Feb 2015 13:28:55 GMT
Server: Apache/2.2.14 (Ubuntu) mod_mono/2.4.3 PHP/5.3.2-1ubuntu4.5 with Suhosin-Patch proxy_html/3.0.1 mod_python/3.3.1 Python/2.6.5 mod_ssl/2.2.14 OpenSSL/0.9.8k Phusion_Passenger/3.0.17 mod_perl/2.0.4 Perl/v5.10.1
X-Powered-By: PHP/5.3.2-1ubuntu4.5
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Location: ../../login.php
Vary: Accept-Encoding
Content-Length: 0
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Content-Type: text/html
```

[Define behaviour dependent on session validity]はセッション状況に応じて、セッション有効化用 Macro の実行などの処理する内容を設定します。

[If session is valid,don't process any further rules or actions for this request]を有効にしている場合、セッションが有効な場合に他の動作（Run Macro など）を行いません。セッションが無効な場合は、設定されている[Check session is valid]以外の処理（Run macro]など）は実行されます。

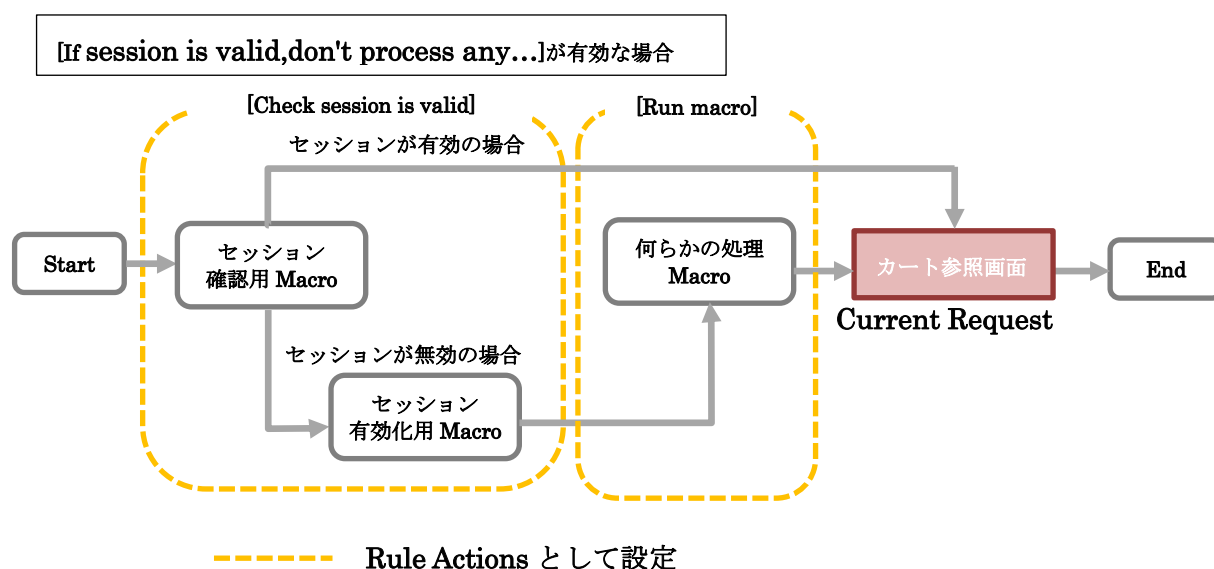


図 46

図 46 のように[Check session is valid]の後に[Run macro]を実施する設定の場合、[If session is valid,don't process any further rules or actions for this request]およびセッションが有効だと[Run macro]として登録されている[run macro:something operation]は実行されません。

Rule Actions

The actions below will be performed in sequence when this rule is applied to a request.

Enabled	Description
<input checked="" type="checkbox"/>	Check session is valid
<input checked="" type="checkbox"/>	run macro: something operation

Buttons: Add, Edit, Remove, Up, Down

図 47

[If session is invalid,perform the action below]を有効にしている場合、セッションが無効時に指定する処理を実行されます。指定できる処理は、[Run a macro]か[Prompt for in-browser session recovery]のどちらかになります。

[Run a macro]を選択した場合、登録されている Macro から選択します。

Define behaviour dependent on session validity:

☐ If session is valid, don't process any further rules or actions for this request

☒ If session is invalid, perform the action below:

Run a macro

Select macro:

login
login check
XSS
logout
something operation

Buttons: Add, Edit

図 48

[Prompt for in-browser session recovery]を選択した場合、ブラウザを用いて有効なセッションの取得を促されます。[OK]を押下すると、後続処理が実施されます。

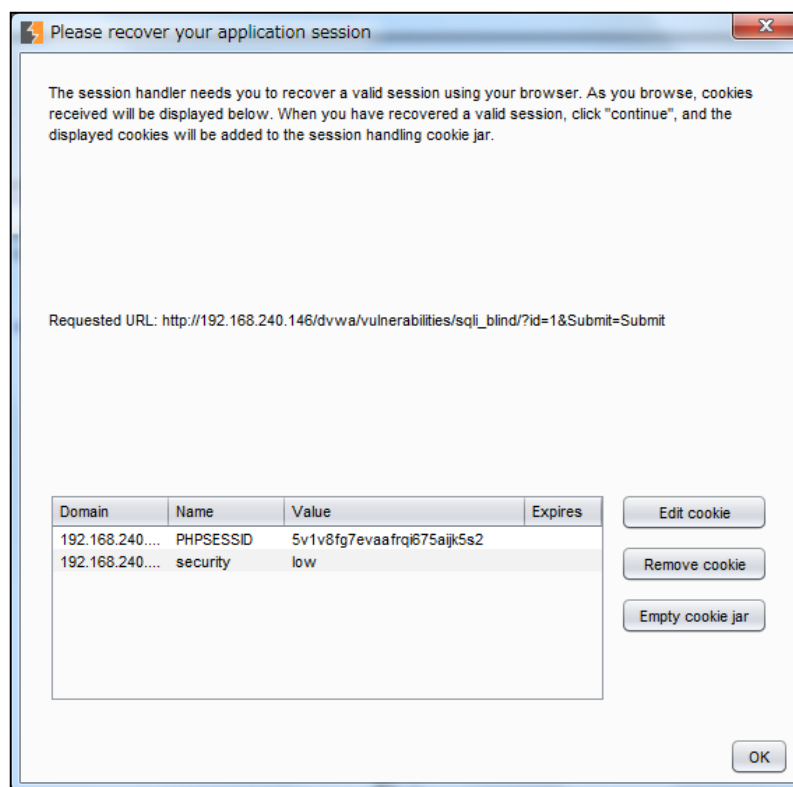


図 49

[Update current request with parameters matched from final macro response]は、Macro の最後のレスポンスからパラメータを引き継ぐかどうかを指定します。この設定は、Current Request に CSRF の対策としてワンタイムトークンが用いられている場合、自動的に引き継がれるため、診断が行えます。

[Update current request with cookies from session handling cookie jar]は、Cookie jar の更新を行うかどうかを指定します。

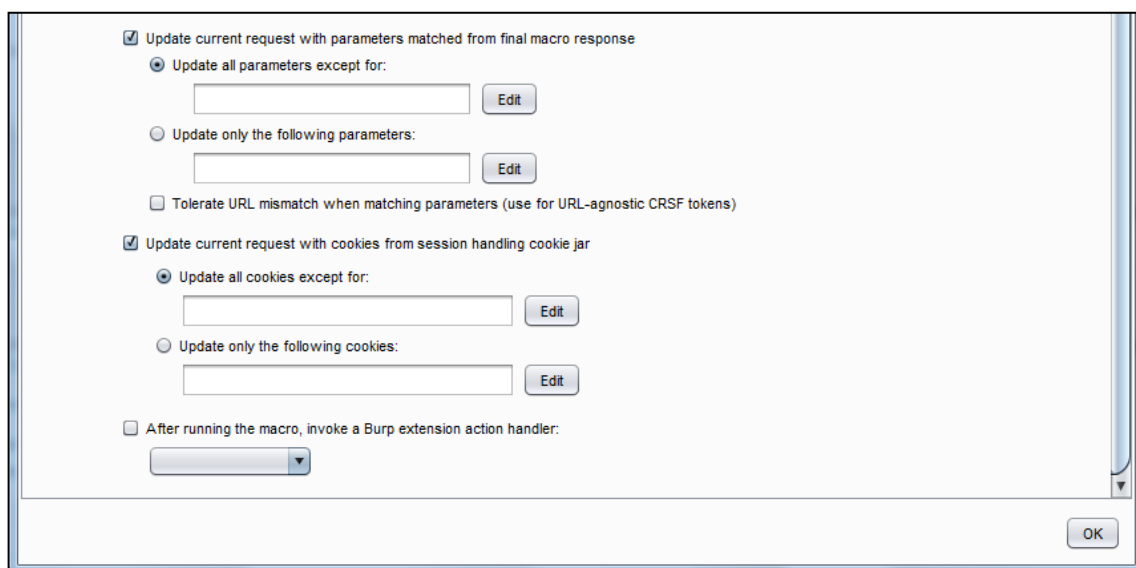


図 50

[After running the macro, invoke a Burp extension action handler]は、[Extender]-[Extensions]で追加した extension を指定することが可能です。extension で変更された Current Request を送信します。

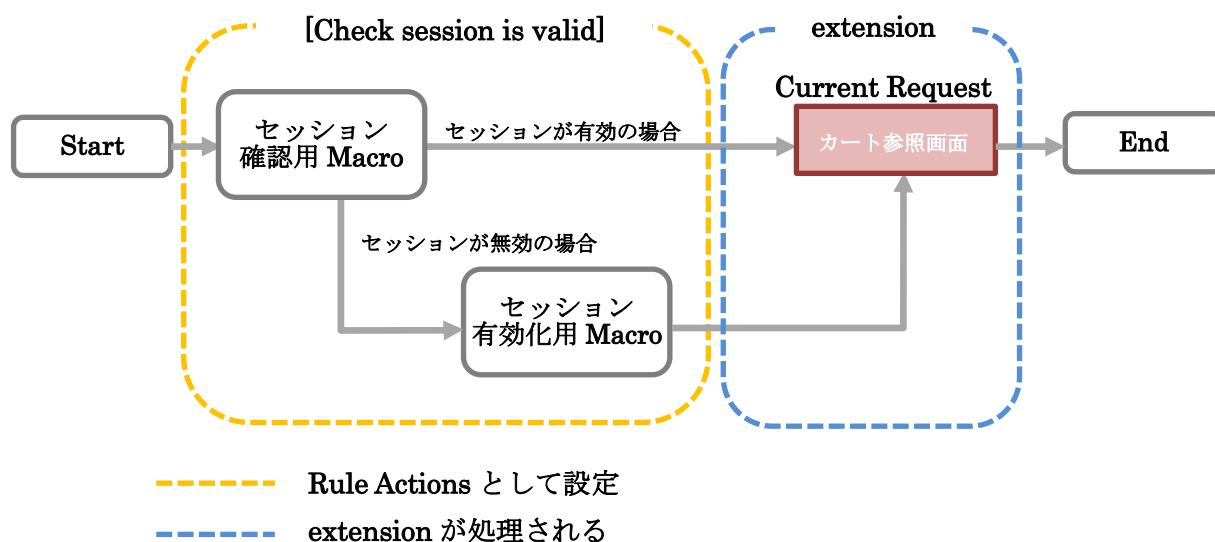


図 51

[Prompt for in-browser session recovery]は、ブラウザを用いてセッションを手動で設定することができます。設定は、どの Cookie の更新を受け入れるのか、または除外するのかを指定します。

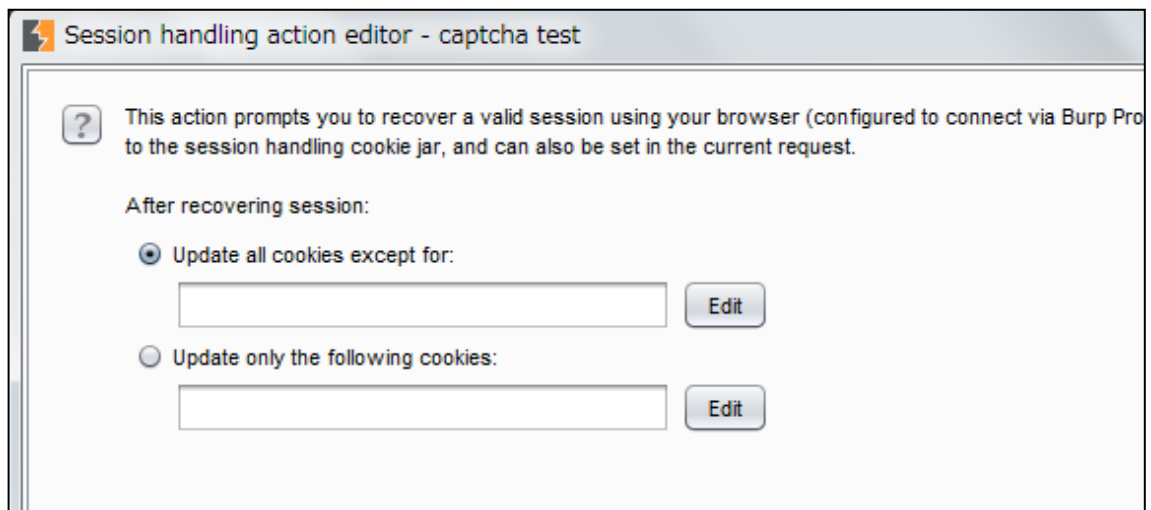


図 52

[Prompt for in-browser session recovery]は、CAPTCHA などのツールでは自動的に遷移ができない箇所を手動で遷移し、それ以降は Macro などに遷移をさせることで、手動での実施範囲が少なくなります。

図 53 のように自動的に遷移をできない箇所を[Prompt for in-browser session recovery]とし、それ以降の自動的に遷移が可能な箇所を[Run macro]で登録することが可能です(図 54)。

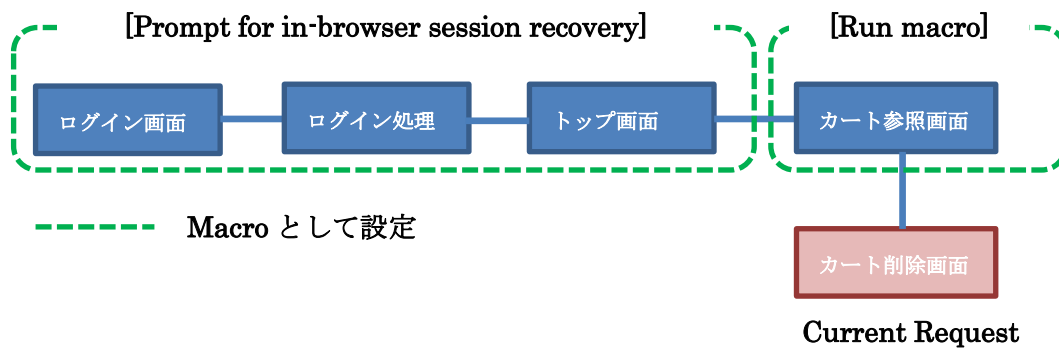


図 53

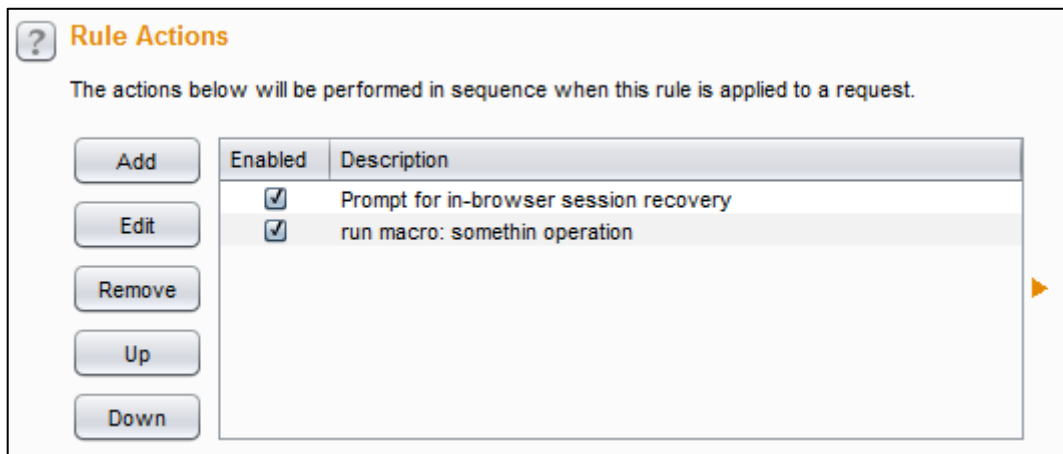


図 54

[Prompt for in-browser session recovery]で Cookie の待ち受けになると[Please recover your application session]が表示され、Set-Cookie を監視します。

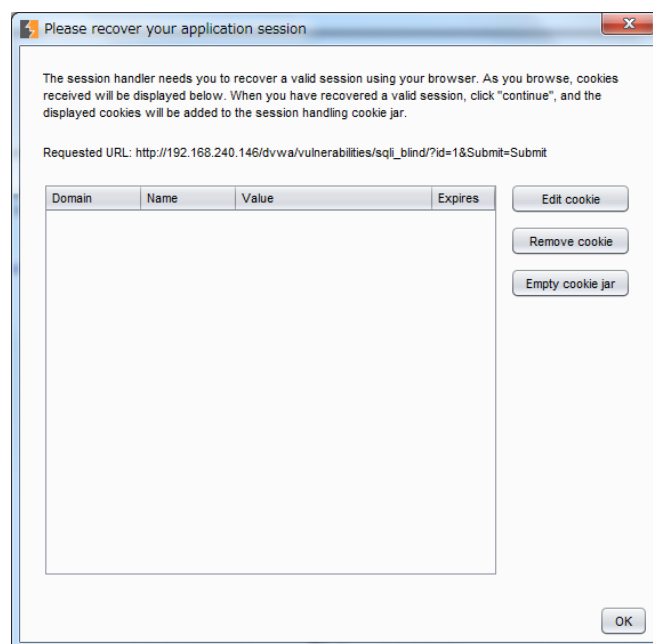


図 55

Proxy として Burp Suite を設定したブラウザで該当のアプリケーションへアクセスし、手動で遷移を行います。有効なセッションが取得できた時点で[OK]を実行すると Burp Suite は取得された Cookie を用いて後続処理を行います。

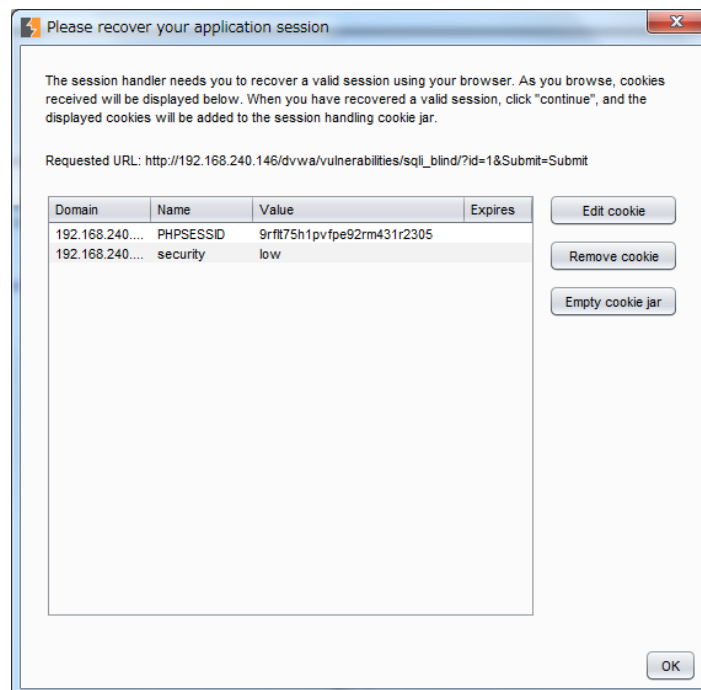


図 56

誤って他のサイトの Cookie が含まれてしまった場合でも、Domain の異なるサイトへは Cookie は送信しないため、気にする必要はありません。

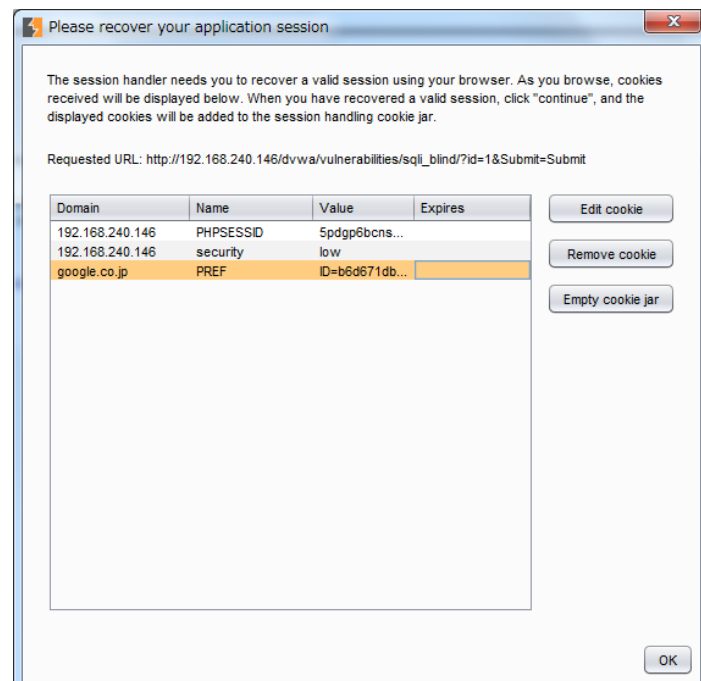


図 57

[Run a post-request macro]は、Current Request の後に指定した Macro を実行します。

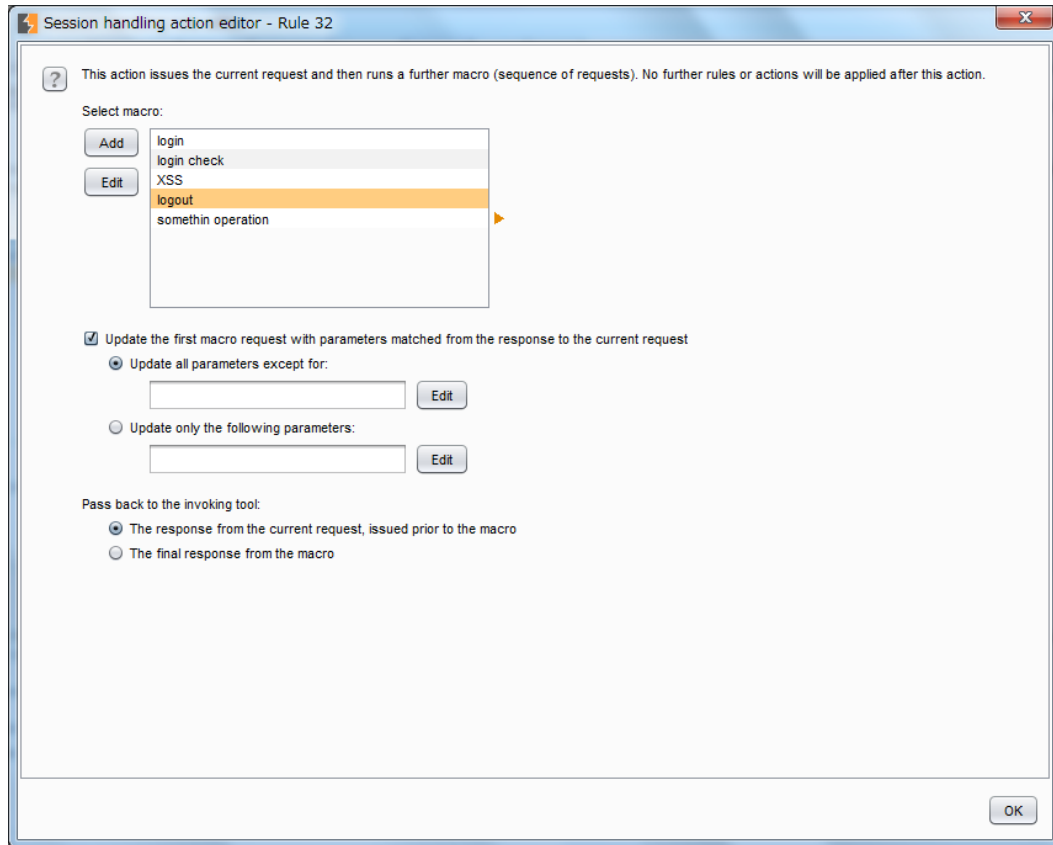


図 58

[Run a post-request macro]は、Current Request の後にログアウト処理を指定することなどが想定されます。

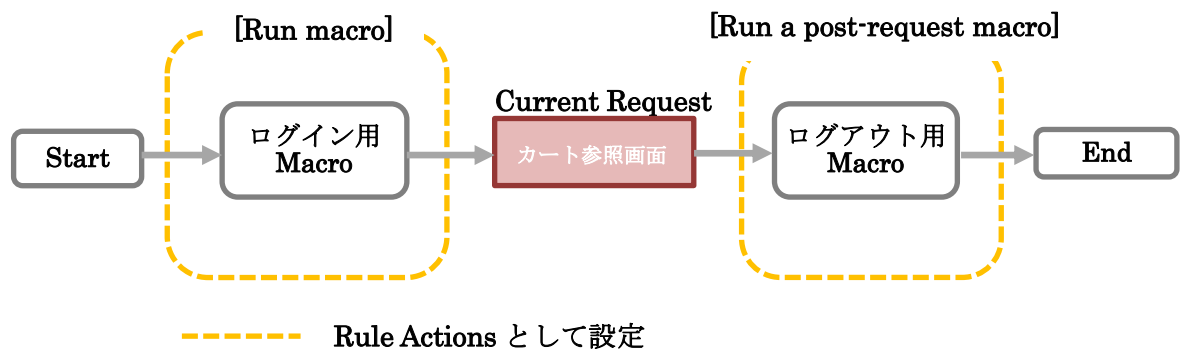


図 59

[Run macro]などを用いてログイン処理を行い、診断対象となる個所へのリクエスト後にログアウト処理を[Run a post-request macro]で指定します。ツールなどで自動的に認証機能のあるアプリケーションを診断する場合、診断対象箇所へのリクエスト後にログアウト処理を実施することが好ましいと考えています。理由として、ログアウト処理を実施しない場合、Web アプリケーションのセッションがそのまま残り続けることになり、診断による多数アクセスで診断対象のリソースを大量に使用してしまう可能性が高まるためです。ただ、ログアウトの実装が不完全だと意味をなしません。たとえば、ログアウトボタンが押下された時にログイン前の画面を表示するだけでサーバ上に保持されているセッションを破棄していない場合などがあげられます。

[Invoke a Burp extension]は、[Extender]-[Extensions]-[Burp Extensions]で登録した extension を指定することが可能です。

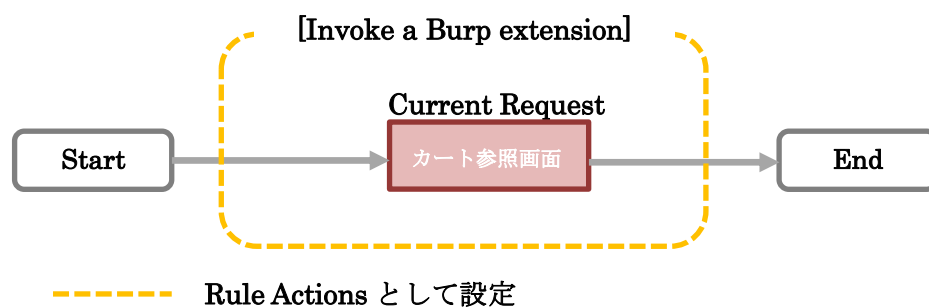


図 60

Extension は、Java、Python、Ruby で作成することが可能です。API は、[Extender]-[APIs]または以下の URL を参照してください。

API <http://portswigger.net/burp/extender/api/burp/package-summary.html>

Extension の作成に参考になっているサイトは以下です。

- Portswigger サポートサイト <https://support.portswigger.net/>
- Portswigger Extension サンプル <http://portswigger.net/burp/extender/>
- BURPEXTENSIONS.COM <http://www.burpextensions.com/>

Burp には、Session Handling Rules の動作状況をモニターする機能が用意されています。[Options]-[Sessions]-[Session Handling Rules]-[Open sessions tracer]を押下すると、Session Handling Rules で設定した各ルールがどのように評価されているか確認することができます。

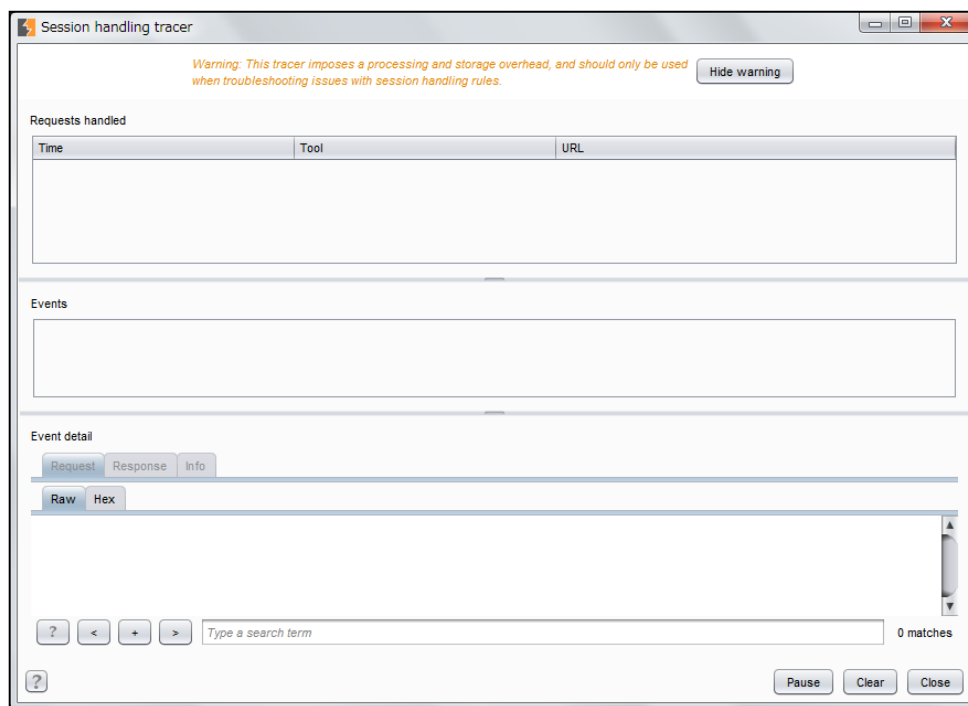


図 61

[Session handling tracer]がどのように動作するのか、以下の遷移をサンプルに説明します。ログイン用 Macro、セッション用 Macro を以下のリクエストとして設定します。

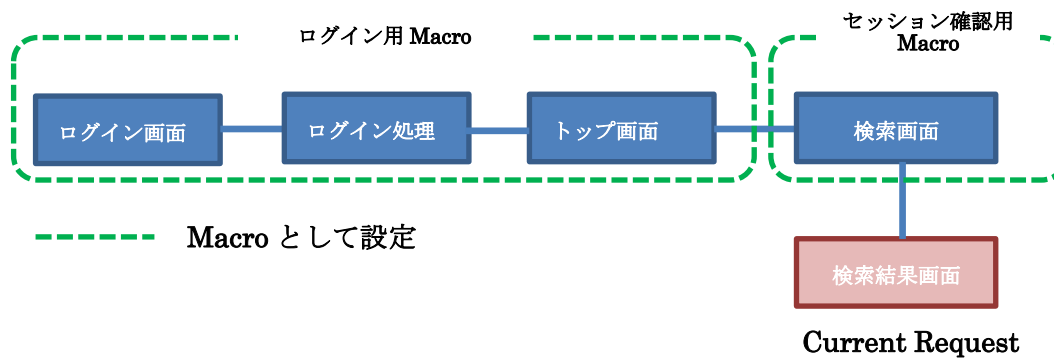


図 62

Session Handling Rules で[Check session is valid]を選択し、以下のルールにて設定します。

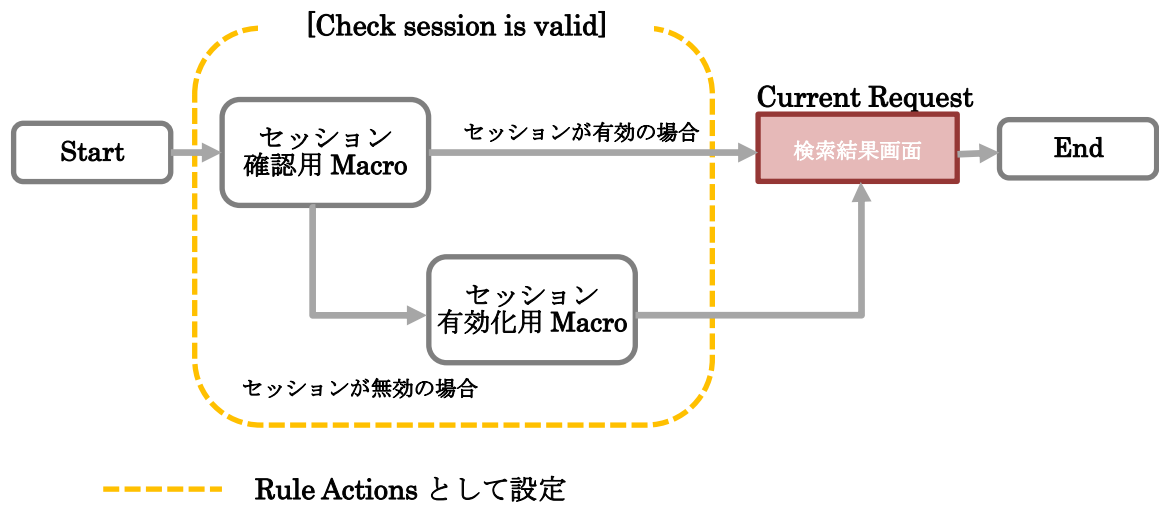


図 63

[Check session is valid]は以下の設定にしています。

- [Make request(s) ...]-[Run macro]でセッション確認用 Macro を指定。
- [Inspect response ...]は、該当アプリケーションがセッション無効な場合に login.php にリダイレクトされるため、[Location(s)]で[URL of redirection target]に login.php と記載。
- [If session is invalid,perform ...]-[Run a macro]でログイン用 Macro を指定。

[Session handling tracer]-[Events]では評価および動作結果が表示され、以下はセッションが無効な場合の結果です。黄色はリクエストが発行されたことを意味します。

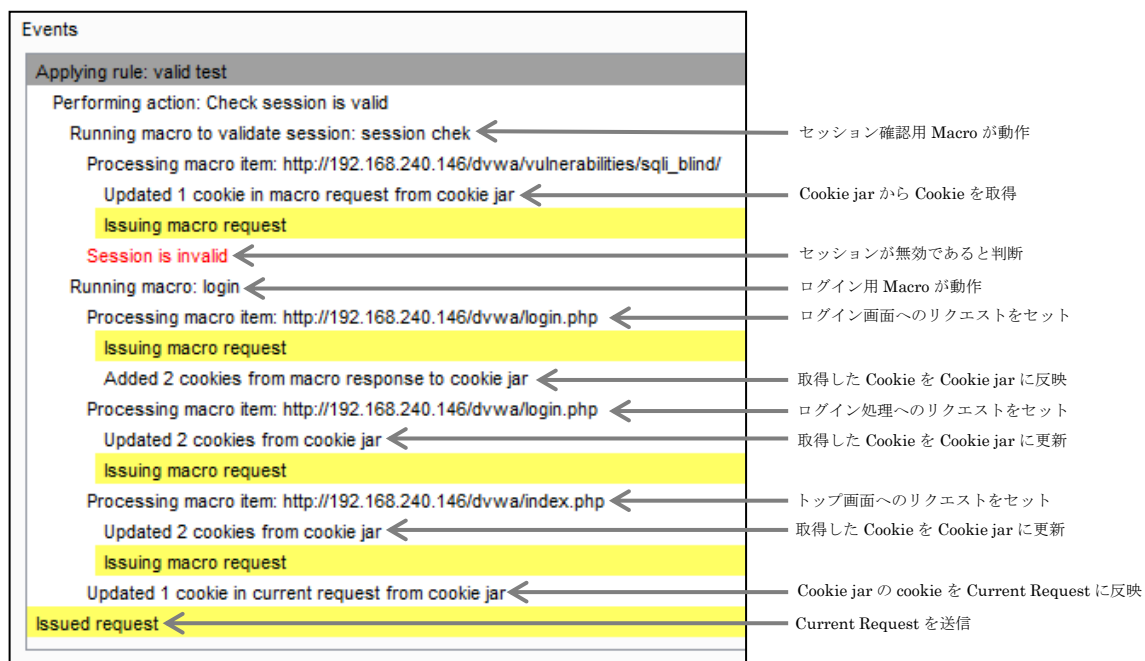


図 64

セッションが有効な場合は、セッション確認用 Macro を実施後、Current Request を送信しています。

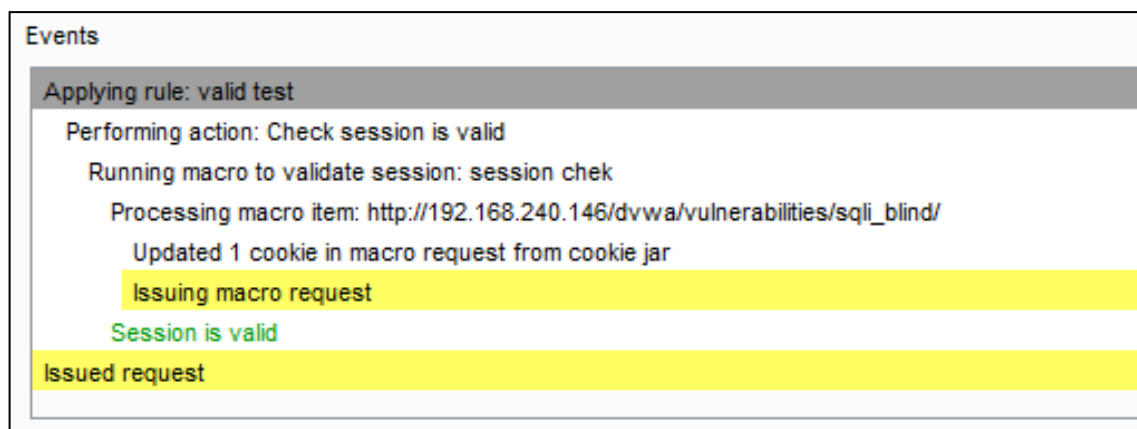


図 65

[Session handling tracer]-[Events]でリクエストを選択すると[Event detail]に該当リクエストおよびレスポンスが参照できます。

Event detail

Request

Response

Info

Raw

Headers

Hex

HTTP/1.1 302 Found
Date: Tue, 20 Jan 2015 10:50:32 GMT
Server: Apache/2.2.14 (Ubuntu) mod_mono/2.4.3 PHP/5.3.2-1ubuntu4.5 with Suhosin-Patch pro
X-Powered-By: PHP/5.3.2-1ubuntu4.5
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Location: ../../login.php
Vary: Accept-Encoding
Content-Length: 0
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Content-Type: text/html

?<+>

Type a search term

图 66