Wallarm FAST Test policies

Wallarm Test Policy

Wallarm Test Policy is what governs generation of FAST tests from baselines. The policy is a string comprised of a number of parameters each of each is described below.
Within Wallarm FAST proxy, the policy is follows -- set-policy directive.
Within the test/request each policy is expressed with an http header X-Wallarm-Test-Policy.

X-Wallarm-Test-Policy

Each request may contain multiple X-Wallarm-Test-Policy headers. Each such header is interpreted separately and results in a separate generated Test set.

X-Wallarm-Test-Policy header includes up to four parameters, each with its own internal structure.

```
  X-Wallarm-Test-Policy: insertion=val1:prop1,val2; ...;
```

The following parameters are supported:

1. type
2. insertion
3. payloads
4. criteria

Not all types parameters are applicable to all types. Here is a quick reference

types	xss, sqli, rce, ptrav	fuzzer
insertion	Yes	Yes
payloads	N/A	Yes
criteria	N/A	Yes

1. type

This parameters lists comma-separated types of the tests that should be generated based on the baseline.

```
X-Wallarm-Test-Policy: type=sqli,rce,xss,ptrav,xxe,fuzzer;
```

The following types are supported: *xss, sqli, rce, ptrav, fuzzer*

If *type* parameter is omitted, all possible types of tests will be generated. Negative logic is also supported by prefixing the name of type with an exclamation point.

```
X-Wallarm-Test-Policy: type=!fuzzer; ...
```

2. insertion

Insertion parameter is an abbreviation from “Insertion Points”. It describes where in the request the fuzzing will be applied to generate security tests.

This parameter may be specified by a value or via a regular expression in ruby language format (validity of the regular expression can be validated here http://rubular.com/).

If this parameter is omitted, all possible tests are generated. Within this parameter, include and exclude directives.

include

For which insertion points tests should be generated.

exclude

For which insertion points tests should *not* be generated.

Example:

```
X-Wallarm-Test-Policy: .....;
insertion=include:GET_(.*),POST_(.*),exclude:GET_sessid_value;
```

3. payloads

This parameter determines how payloads of the original baseline request are modified to create tests. This parameter is only applicable to *fuzzer* test type.

The following directives are supported:

all

Apply all the possible payload modifications

replace_all [N]

Replace the payload with N bytes generated by the fuzzer

prepend [N]

Prepend the payload with N bytes generated by the fuzzer

append [N]

Append N bytes generated by the fuzzer to the payload

replace M [N]

Replace the first M bytes of the payload with N bytes generated by the fuzzer

If M is a negative, than replace from the end of string. Similar to: http://ruby-doc.org/core-2.2.0/String.html#method-i-slice

replace_rand [N]

Replace arbitrary segment of N bytes with N bytes generated by the fuzzer

insert_rand [N]

Insert N bytes generated by the fuzzer into an arbitrary position within the payloads (the payload becomes N bytes longer) .

Example:

```
  X-Wallarm-Test-Policy: payloads=replace_all, 'append 10'; ...
```

Default value (if the parameter is not specified): replace_all 16.

If N is omitted in any of the directive, it will default to the value of 16.

4. criteria

This parameters is required to specify the policy for generating tests of the types *fuzzer*. It describes the test success criteria, specifically what is considered to be an anomaly and when the test execution should be halted. The directives here as as follows:

criteria[extended]

If the result matches the condition, it is considered an anomaly (the test fails)

status

length

time

length_diff

time_diff

dom_diff

regexp

Example:

```
X-Wallarm-Test-Policy: criteria[extended]=status: 408, !400; <- status
408 is an anomaly, status 400 is not an anomaly

X-Wallarm-Test-Policy: criteria[extended]=length: >=70; <- any response
with the length >=70 is an anomaly

X-Wallarm-Test-Policy: criteria[extended]=time: >=1; <- a response that
took more 1 second is considered an anomaly

X-Wallarm-Test-Policy: criteria[extended]=length_diff: >100; <- consider
it an anomaly if the length of app response to the test request differs
from the length of the response to the baseline request by >100 байт

X-Wallarm-Test-Policy: criteria[extended]=time_diff: >5; <-, consider it
an anomaly if the response time to the test request differs from the
response to the baseline request by >5 seconds
```

```
X-Wallarm-Test-Policy: criteria[extended]=dom_diff: >40; <- consider it
an anomaly if the difference between DOM elements in the test request
and in the baseline request is >40

X-Wallarm-Test-Policy: criteria[extended]=regexp: 'xxx'; <- consider it
an anomaly if the baseline satisfies a regular expression
```

criteria[excluded]

What to exclude from the anomalies

status

length

time

length_diff

time_diff

dom_diff

regexp

Example:

```
X-Wallarm-Test-Policy: criteria[excluded]=status: 408; <- any response
with status 408 is not an anomaly

X-Wallarm-Test-Policy: criteria[excluded]=length: >=70; <- any response
with the length >=70 is not an anomaly

X-Wallarm-Test-Policy: criteria[excluded]=time: >=1; <- a response that
took 1 or more second is not considered an anomaly

X-Wallarm-Test-Policy: criteria[excluded]=length_diff: <100;<- do not
consider it an anomaly if the length of app response to the test request
```

```
differs from the length of the response to the baseline request by >100
байт

X-Wallarm-Test-Policy: criteria[excluded]=time_diff: <5; <-, do not
consider it an anomaly if the response time to the test request differs
from the response to the baseline request by >5 seconds

X-Wallarm-Test-Policy: criteria[excluded]=dom_diff: <40; <- do not
consider it an anomaly if the difference between DOM elements in the
test request and in the baseline request is >40



X-Wallarm-Test-Policy: criteria[extended]=regexp: 'xxx'; <- do not
consider it an anomaly if the baseline satisfies a regular expression
```

criteria[hardstop]

Under what conditions test execution halts:

status

length

time

length_diff

time_diff

dom_diff

regexp

anomalies

timeout_errors

Example:

```
X-Wallarm-Test-Policy: criteria[hardstop]=status: 408; <- halt the Test run if a response with 408 status is received.

X-Wallarm-Test-Policy: criteria[hardstop]=length: >=70; <-  halt the Test run if a response’ length is >=70

X-Wallarm-Test-Policy: criteria[hardstop]=time: >=1; <- halt the Test run if a response time >=1 seconds

...

X-Wallarm-Test-Policy: criteria[hardstop]=anomalies: >=100; <- halt the Test run if found over 100 anomalies

X-Wallarm-Test-Policy: criteria[hardstop]=timeout_errors: >=10; <- halt the Test run if a the server responded with a timeout at least 10 times
```