

BML (BRITE Modelling Language)

Tutorial

Advanced Network Technologies Division

NIST

Last changed: 5/25/2011 3:35:00 PM

Contents

| | |
|--|---|
| What is BML? | 3 |
| Concepts | 4 |
| Contexts, Namespaces and Identifiers | 4 |
| Elements | 4 |
| Quick Start..... | 6 |
| A Simple Testcase | 6 |
| Bibliography | 9 |

What is BML?

BML is the BRITE Modelling Language. It is an XML-based language to model test cases to be run by BRITE (BGPSEC/RPKI Interoperability Test & Evaluation). This document assumes, that you are confident with the BRITE Webinterface and know the basic concepts of BRITE testcases. Otherwise, please refer to the BRITE User Tutorial (NIST Advanced Network Technologies Division (ANTD), 2011).

Concepts

Contexts, Namespaces and Identifiers

One BML file can refer to another BML file (with the `include` element), thus building a graph of BML files. A BRITE context is a tree of xml files, that together provide all necessary elements for a test to run. The root of this BRITE context tree needs to provide at least one `test` element. A test is able to run, if all references are resolved and no cycle is detected (tree requirement). Each BML file defines a namespace. This namespace is made available locally with the `include` element. An identifier denotes an element in the local or a remote namespace (Example: `@fil_conf:cfg_small:rtr_as50` for remote namespace `fil_conf`, element `<configuration id="cfg_small"><router id="rtr_as50">`).

Elements

With BML, you can define the following elements used by BRITE

- **<brite>**
The `brite` element is the top-level element. It is used to define the BRITE XML namespace (<http://www.antd.nist.gov/brite>, Schema Definition File: <http://brite.antd.nist.gov/brite.xsd>)
- **<configuration>**
`Configuration` elements are used to configure a set of protocol endpoints. One `configuration` element is used for a specific test. Endpoints define two types of services. First type are the services, which are instantiated by BRITE, second type are the services, which need to be configured on the side of the IUT. The BRITE Webinterface will use the content of the configuration element in order to generate configuration instructions and monitor the endpoints, which are instantiated by BRITE for the connection from the *Implementation Under Test* (IUT).
- **<include>**
`Include` elements reference to other files. They make the content of the referenced file available in the local file.
- **<alias>**
The `alias` element is used to substitute long identifiers to short handy abbreviations.
- **<whitelist>**
A `whitelist` element defines synthesized RPKI/RTR protocol PDUs. Each PDU can be sent out to the IUT by `traffic` elements or be used to write `goal` elements, that define expected `traffic` to be receive by BRITE during a `testrun`.

- **<bgp>**
Analogously to `whitelist` elements, `bgp` elements define PDUs for BGP-4 and BGPSEC traffic.
- **<test>**
`Test` elements define a testcase, to be evaluated by BRITE. The content of this element is used to create a description, to be shown by the Webinterface as well as `traffic` elements and `expectation` element. These both elements define the scenario of the test case.

Quick Start

A Simple Testcase

This example testcase shows a minimal way to define a useful test. It is written in one testfile

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <brite xmlns="http://www.antd.nist.gov/brite"
3      xsi:schemaLocation="http://www.antd.nist.gov/brite brite.xsd"
4      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
5
6      <configuration id="cfg_example1">
7          <router id="rtr_myfeed" as="10" type="FEED" />
8          <router id="rtr_mycollector" as="20" type="COLLECTOR" />
9          <router id="rtr_myiut" as="100" type="IUT" />
10     </configuration>
11
12     <bgp id="bgp_example1">
13         <data id="dat_example1">
14             <path>{IUT} {R} {R} 3000 2000 1000</path>
15             <announce>
16                 <prefix>192.0.2.0/24</prefix>
17             </announce>
18         </data>
19     </bgp>
20
21     <test name="Example One" start="2000-01-01" id="tst_example1"
22         configuration=":cfg_example1">
23         <description>Example description</description>
24
25         <traffic id="tfc_example1" type="bgp"
26             sender=":cfg_example1:rtr_myfeed">
27             <action time="00:00:30">
28                 :bgp_example1:dat_example1
29             </action>
30         </traffic>
31
32         <expectations id="exp_example1">
33             <goal id="gol_check_bgp_update" type="bgp"
34                 label="Receive Update from IUT" active_from="00:00:30"
35                 active_to="00:01:30">
36                 <require>
37                     :bgp_example1:dat_bgp_example1%:cfg_example1:rtr_myfeed
38                 </require>
39                 <report>
40                     <success>Received the expected update</success>
41                     <failure>Expected update not received.</failure>
42                 </report>
43             </goal>
44             <report>
45                 <success>Test succeeded. IUT is doing fine!</success>
46                 <failure>Something is wrong!!!</failure>
47             </report>
48         </expectations>
49     </test>
50 </brite>

```

This testcase defines a network topology, where three BGP speakers interact. In lines 7-9, you can see these routers defined. Each router element has a different behavior, based on the `type`-Parameter. A router with `type="IUT"` is used, to preset parameters for the IUT. Line 9 configures the IUT to be AS100, the webinterface will use this information to inform the user to configure his IUT to this ASN and requests an IP address, for BRITEs neighbor configuration. The two entries with `type="FEED"` and `type="COLLECTOR"` are router-Instances, that BRITE runs. A FEED is used to send out BGP Updates to the IUT and a COLLECTOR is used to evaluate BGP Updates, that are received from the IUT.

Line 12-19 define actual synthesized BGP Updates. They can be used by `traffic` elements (lines 25ff.) to be sent out from a router, which is `type="FEED"` or be evaluated by a goal element (lines 33ff.) received at the router `type="COLLECTOR"`. Each `data` element below a `bgp` element defines an Update and. For the case of sending a BGP Update, the `{R}` text symbols in the `path` Attribute are substituted to the ASN of the sending router and the `{IUT}` text symbols are just stripped out. For the case of receiving an BGP Update, `{R}` is replaced by the ASN of the original sender of the Update and `{IUT}` is replaced by the ASN of the IUT.

In line 21, an a test is defined. The parameter `start` defines the virtual start date of the test. A test always starts at midnight (00:00) of the day, that is defined by this parameter. The `configuration` element makes first use of a reference to another element. By writing `configuration=":cfg_example1"`, we define, that this test uses the configuration, defined by the element `<configuration id="cfg_example1" />` of the local namespace. We will see in a later example on how to reference elements in foreign namespaces.

The `description` element (line 23) is used to describe the test in a user-friendly way. This field allows html-Code to be inserted by masking it in a `<![CDATA[]]>` element. The webinterface uses the content of this element to produce a description to the user, that wants to run the test. Use this element to explain, what this test does in detail. Explain, how the scenario looks like and what expectations you have written. Also point out, which assumptions you have made about the IUT for this test (i.e. "This test assumes, that the BGP best path selection prefers short over long paths and converges within 30 seconds."). Please consider adding an image to the description. It is recommended to use a template for the `description` element.

Lines 25-30 define a `traffic` element of a testcase. It is used to send out `data` elements. In this case, we are referencing a `data` element below a `bgp` element, which makes this a BGP Update. By setting the Parameter `sender=":cfg_example1:rtr_myfeed"` we say, that all BGP Updates below this `traffic` element are sent from the router defined in

`<configuration id="cfg_example1"><router id="rtr_myfeed"...>`. The action Element references the BGP Update in `<bgp id="bgp_example1"><data id="dat_example1">` and sets the sending time to 30 seconds after test start.

Lines 32-48 define the `expectations` Element, which is used to evaluate the received data (in this example BGP Updates). Each `goal` element defines constraints, that are used to match the received data. Therefore, each `goal` element keeps an evaluation state. The constraints can be based on other `goal` and `goal-set` elements evaluation state, the order of the received data, the endpoint which received the data, the time the data is received and the amount of data received. In our case, we require to see a BGP Update in the interval 30 – 90 seconds after `testrun` start (`active_from="00:00:30" active_to="00:01:30"`). The parameter `label` gives this goal a human readable name. Line 37 references the BGP Update, the same way a `traffic` element does plus the original sender (`:cfg_example1:rtr_myfeed`) of the BGP Update.

The need to define the original sender is the result of the reusability of the BGP Update for sending and receiving. While sending a BGP Update, the `{R}` symbols in the `AS_PATH` are replaced with the sender router ASN, we expect the receive an Update, that looks the same way, plus having the `{IUT}` symbols substituted by the ASN of the IUT.

Lines 39-42 define a report block for the goal, that is shown in the report. Lines 44-47 define a global report block, which is reported based on the global evaluation state of the test.

Bibliography

NIST Advanced Network Technologies Division (ANTD). (2011, 05). *BRITE User Tutorial*. Retrieved from BRITE: BGPSEC/RPKI Interop Test & Evaluation: <http://brite.antd.nist.gov/statics/documentation>