CHAPTER 17 Byte Pattern Recognition Engine

The Byte Pattern Recognition Engine (BPRE) is a new, protocol-independent engine to identify packets and payloads based on recognized byte patterns in packet payloads. BPRE is fully generic so you can use it to identify byte patterns in any TCP- or UDP-based packet. You can use BPRE to

- Define byte-pattern match criteria (up to 100 match definitions per appliance) to identify packets with specific messages such as response codes, error messages, parameter values, and transaction IDs.
- Capture timestamps, payload bytes, IPs, and ports for matching packets.
- Search and retrieve information about matching packets using a Web Services API.
- Use this information to identify captured packets to download and analyze using tools such as Packet Analyzer, Transaction Analyzer, and WireShark.

BPRE is especially useful for troubleshooting and analyzing custom applications that standard tools cannot decode.

Note: BPRE is included as a preview feature in this release. During this phase, customers can provide feedback while Riverbed evaluates whether to include this or similar functionality in future releases. Riverbed is especially interested in information about how customers use this feature and how important they find this feature relative to other features in AppResponse.

This section discusses the following:

Important Notes

Note the following:

- This feature is available on the following appliance models only:
  - 4200
  - 4300
  - 5000
  - 5100
  - 6000
- BPRE searches IPv4 packets only and ignores IPv6 packets.
- This feature is very resource-intensive. Riverbed recommends that you not enable BPRE at the same time as any other resource-intensive processes such as VoIP Monitoring, NetFlow Monitoring, or Shark Module. You can enable BPRE with Web Transaction Analysis but should do so only if you are using BPRE to analyze web traffic.
- Due to the increased load while BPRE is enabled, the appliance will experience lower overall performance.
- The Appliance Health Check insight checks for CPU and other resource consumption. It is good practice to run this insight both before and after you enable BPRE to estimate the load increase and to ensure that overall load on the appliance is within reasonable limits. For more information about the Appliance Health Check Insight, see "Required Practices for AppResponse Appliances".
- BPRE parses the TCP/UDP payload (data after the headers) only; it does not parse any other packet data.
- The appliance detects byte patterns in real time and stores the matching-packet information in a rolling buffer. The rolling buffer should be able to hold up to 24 hours of matching-packet data, but this window might be shorter if there are many matching packets.
- You can access byte pattern occurrence records through a Web Services API, which returns data in CSV format for optional post processing.
- BPRE supports the use of regular expressions to find and capture byte patterns of interest. However, these processes are highly CPU-intensive and should be used in a highly targeted manner to avoid excessive load on the appliance. For more information, see "Regular Expressions: Important Notes" on page 536.
- Some information about BPRE might have changed since this documentation was published. For updated information and guidance, log in to the Support site (https://login.riverbed.com/login_support.htm) and search for KB entry S27267.

Workflow Description

The following steps outline the general workflow:

1) Determine the following:

   a) The exact byte patterns that identify packets of interest—for example, “transaction-start” and “transaction-end” packets for a specific application.

   b) The exact position in the packet payload for each byte pattern, specified as an offset (0 is the first payload byte in the payload data). You can also choose to search the entire payload in each packet.

   c) The traffic flows (IPs and TCP/UDP ports) that you want BPRE to monitor for packets of interest.

2) Define the byte patterns and other search/capture criteria in the bpre-settings.xml file (see “BPRE Settings File” on page 534).

3) Upload the bpre-settings.xml file to the appliance (see “Upload bpre-settings.xml to Appliance” on page 539).

4) Start the BPRE data collection process (see “Start/Stop the BPRE Process” on page 540).

5) Wait for the appliance to collect information about each matching packet (see “Query BPRE Results with Web Service URLs” on page 540).

   The appliance saves the following information for each matching packet: source IP, destination IP, source port, destination port, byte pattern ID, and observation time stamp. (BPRE can also save byte patterns of interest, but this process is highly CPU-intensive and should be used only in highly targeted cases.)

6) Query the appliance for information about matching packets using the Web Services REST API (see “BPRE Output Format” on page 542).

BPRE Settings File

The following example file is configured to find packets for a custom application XYZ that uses TCP port 40. BPRE saves the IPs, ports, and timestamps for all matching packets within a specific IP range. The following packets are flagged as matching:

0) “Request” packets—`XYZ/REQ` in payload bytes 0-6

1) “Error Timeout” packets—`XYZ/706` in payload bytes 0-6

2) “Error Forbidden” packets—`XYZ/713` in payload bytes 0-6,
AND `Forbidden:` is anywhere in the payload.

For “Error Forbidden” packets, BPRE also captures the entire error message—“`Forbidden:`” plus the next 10 bytes.

Figure 1 BPRE Settings File Format

```
<?xml version="1.0" encoding="UTF-8" ?>
<settings version="2" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="bpre-settings-schema-v2.xsd">
     <search-definitions>
          <search-definition id="0" key="searchdef1">
               <ip-addresses>
                    <ip-address range-begin="192.168.0.0" range-end="192.168.0.128"/>
               </ip-addresses>
               <ports>
                    <port range-begin="40" range-end="40" type="tcp"/>
               </ports>
               <matches>
                    <match id="0" range-begin="0" range-end="0"
                         pattern-format="ascii" pattern-name="Request">
                         <pattern><![CDATA[XYZ/REQ]]></pattern>
                    </match>
                    <match id="1" range-begin="0" range-end="0"
                         pattern-format="ascii" pattern-name="Error Timeout">
                         <pattern><![CDATA[XYZ/706]]></pattern>
                    </match>
                    <match id="2" range-begin="0" range-end="0"
                         pattern-format="ascii" pattern-name="Error Forbidden">
                         <capture-definition>
                              <capture-regex group="1">
                                   <![CDATA[(Forbidden:.{10})]]>
                              </capture-regex>
                         </capture-definition>
                         <pattern><![CDATA[XYZ/713]]></pattern>
                         <regex-pattern>
                              <![CDATA[Forbidden:]]>
                         </regex-pattern>“
                    </match>
               </matches>
          </search-definition>
     </search-definitions>
</settings>
```

An XML data file includes one or more **`<search-definitions>`**. Each defines the set of traffic to search and the byte pattern(s) to search for within this traffic.

- **`<ip-addresses>`** — Search packets to or from IPs within these ranges.

- **`<port>`** — Search packets transferred on ports within these ranges.
- **`<matches>`** — One or more matches used to identify matching packets. Each **`<match>`** includes
  - A unique **`id`** and **`pattern-name`**
  - The **`<pattern>`** to search and the **`format`** (**`ascii`** or **`hex`**).
  - The range of bytes (**`range-begin`** and **`range-end`**) within each packet payload to search for the `<pattern>`.
  - (*Optional*) You can also specify a secondary **`<regex-pattern>`** that is applied to the entire payload. If a packet meets all the criteria for the `<pattern>` (first test), apply the `<regex-pattern>` to the entire payload (second test). The packet is flagged as matching only if the payload meets both tests.
  - (*Optional*) If you want to capture relevant bytes from matching packets, you can specify a **`<capture-definition>`** to save a specific sequence of bytes in the BPRE rolling buffer. All bytes that match the `<capture-regex>` sequence are saved in the "REGEXBYTES" field for each matching packet.

BPRE processes each packet as follows:

1) *Check IP/port range*—If packet is not in IP or port range, no match. Proceed to next packet.

2) *Check pattern*—If the `pattern` is not found within the byte range, no match. Proceed to next packet.

3) *Check regex pattern*—If the match includes a regex pattern, check the entire payload. If the regex pattern is not found, no match. Proceed to next packet.

   If all three tests pass, the packet matches. Save the IP, port, and timestamp information in the BPRE rolling buffer.

4) *Check capture definition*—If the match includes a capture regex, check the entire payload for this pattern. If the pattern is found, save the matching bytes to the BPRE rolling buffer.

Regular Expressions: Important Notes

If you want to use regular expressions in BPRE, note the following:

- This feature requires a working knowledge of regular expressions, which is outside the scope of this documentation.
- BPRE uses the TRE library for matching regular expressions, so any expressions you define must be compliant with TRE.
- Using regular expressions to search for and capture byte patterns is highly CPU-intensive. Unlike the `<pattern>` option, which searches a range of bytes, the `<regex-pattern>` and `<capture-definition>` options search the entire payload of each matching packet.

  Because regex processing is so CPU-intensive, you should follow these best practices:

  – In general, the `<regex-pattern>` option is preferable to `<pattern>` only if you cannot predict the location of the byte pattern in the payload. If the pattern is in a fixed location, use `<pattern>` and set `range-begin` and `range-end` to the payload address of the first byte in the pattern.

  – Use `<regex-pattern>` or `<capture-regex>` only on highly targeted traffic flows of interest. This means that the IP and port ranges in the `<search-definition>` parent element should be defined as narrowly as possible.

  – If a match includes a regular expression, define the `<pattern>` element as specifically as possible, so that BPRE applies the regular expression only to packets that are truly relevant to your analysis.

  – The `<regex-pattern>` search and the `<capture-regex>` search/capture are separate processes, each of which adds to the overall CPU load. If possible, define your match to include only one of these options rather than both.

  – If you want to find packets based on multiple match criteria (such as packets that have three patterns in three different locations in the payload), the following workflow is preferable to using the `<regex-pattern>` option:

  1) Specify a search definition that includes a match for each pattern of interest in the payload. For example:

```
<match id="0" range-begin="0" range-end="0"
   pattern-format="ascii" pattern-name="GET">
   <pattern><![CDATA[GET ]]></pattern>
</match>

<match id="1" range-begin="18" range-end="18"
   pattern-format="ascii" pattern-name="User-Agent SXL 3.1">
   <pattern><![CDATA[SXL/3.1 ]]></pattern>
</match>

<match id="2" range-begin="60" range-end="60"
   pattern-format="ascii" pattern-name="Host http.00.s.dbot.x1.net">
   pattern><![CDATA[http.00.dbot.x1.net]></pattern>
</match>
```

  BPRE collects data for all packets that meet any match criteria in the settings file.

  2) Use boolean operators in the Web Service URL to find packets that pass multiple matches. For example, to find packets that match all three patterns in this example, the query string would look like this:

```
&query='matchid=0 and matchid=1 and matchid=2'
```

  For more information, see “Query BPRE Results with Web Service URLs” on page 540.

  This method is strongly recommended as an alternative to using `<regex-pattern>` to find packets based on multiple match criteria.

XML Reference

The following table describes the elements and attributes in the "BPRE Settings File Format".

Table 1 BPRE XML Elements and Attributes

Element	Description and Attributes
`<search-definitions>`	Container for one or more `<search-definition>` elements.
`<search-definition>`	A search definition specifies the traffic in which to search (IP and port ranges) and one or more byte-pattern matches to search in. Required elements: • `id` = Integer ID for the search definition. This integer ID must be unique and appears in the "SEARCHDEFINITIONID"field of the "BPRE Output Format". • `key` = String ID for the match. This key string must be unique. If you want to find matching packets for a search definition, you can use either the ID or the key string for the "`searchdefinitionid`" parameter in the query string of the web service URL. • `<ip-addresses>` of the traffic flows to include in this search definition • `<ports>` of the packets st to search • `<match>` criteria to search for within each packet.
`<ip-addresses>`	BPRE searches only packets within an `<ip-address>` AND a `<port>` range defined in the search definition. A search definition can include multiple IP ranges.
`<ip-address>`	Search packets sent to or from IPs within this range. IPs must be in dotted, IPv4 notation such as "`10.33.120.106`". Subnet masks are not supported. Required attributes: • `range-begin` = IP range start (inclusive) • `range-end` = IP range end (inclusive)
`<ports>`	BPRE searches only packets within a defined `<ip-addresses>` AND a `<ports>` range in the search definition. A search definition can include multiple port ranges.
`<port>`	Search packets sent or received on TCP/UDP ports within this range (inclusive). Ports must be expressed as integers. Required attributes: • `range-begin` = port range start (inclusive) • `range-end` = port range end (inclusive) • `type` = `tcp` (default) or `udp`

Table 1 BPRE XML Elements and Attributes (Continued)

Element	Description and Attributes
`<matches>`	The set of byte-pattern match criteria to search for in traffic flows defined by the `<ip-addresses>` and `<ports>` filters.
`<match>`	A single BPRE match criterion. Required attributes: • `id` = Integer ID for the match. This ID must be unique within the parent `<search-definition>`. This ID appears in the "MATCHID" field for each matching packet in the Web Service output. • `range-begin` = The start of the byte range (inclusive) to search within the payload of each packet. **Note—**BPRE searches application payload data only; it does not search IP, transports, or other packet header data. Byte 0 is the first byte in the application payload, not the first byte in the packet. A match means that the *first byte in the byte pattern* was found within this range. • `range-end` = The end of the byte range (inclusive) to search within the payload of each packet. If `range-end="-1"` all payload bytes are searched. **Note—**To reduce unnecessary processing load, it is good practice to define the range of packets to search as narrowly as possible. For example, suppose you want to find packets with byte pattern "ABCD" and that this pattern is always located at 0-3 (the first four bytes in the packet payload). In this case, the first byte in the pattern must be at byte 0 for the packet to be flagged as matching. In this case, define the range as `range-begin=0 range-end=0`.

Table 1 BPRE XML Elements and Attributes (Continued)

Element	Description and Attributes
`<capture-definition>`	(*Optional*) You can include one or more capture definitions for packets that meet a specific match. The captured bytes are included in the "REGEXBYTES" field of the "BPRE Output Format".

`<capture-regex>`	A regular expression used to identify the packets within each packet to capture.
	Note—The use of `<capture-regex>` is highly resource-intensive and should be used selectively on as few packets as possible. For more information, see "Regular Expressions: Important Notes" on page 536.
	Required attributes:
	• `group` = Quantifier of the regex group position within `CDATA`. Specify 1 to capture the entire regex. This option is useful if the regex consists of multiple groups delineated with parentheses.
	• `CDATA` = Regex that defines the bytes to capture
	Note—To specify a series of hexadecimal bytes, you must add the prefix **`\x`** to each byte, like this:
	`<capture-regex group="1">`
	`<![CDATA[(\x52\x65\x66\x65\x72\x65\x72\x3a.{20})]]>`
	`</capture-regex>`

`<pattern>`	The byte pattern to search for in each packet payload. The following definitions search for the byte patterns "`HTTP/1.1 200`" (ASCII) and "`x72\x65\x3c\x3a]`" (hex) respectively:
	`<pattern><![CDATA[HTTP/1.1 200]]></pattern>`
	`<pattern><![CDATA[x72\x65\x3c\x3a]]></pattern>`

`<regex-pattern>`	(*Optional*) You can specify a secondary match criterion as a regex pattern. **Note—**The use of `<regex-pattern>` is highly resource-intensive and should be used selectively on as few packets as possible. For more information, see "Regular Expressions: Important Notes" on page 536.

Upload bpre-settings.xml to Appliance

Note the following about BPRE settings:

- This preview version of BPRE does not include functionality for viewing the BPRE settings installed on an appliance. For this reason, you should retain a copy of the bpre-settings.xml file as a reference for the current BPRE settings on the appliance.
- When you upload a BPRE settings file, the new settings are applied to new packets only and do not affect historical BPRE data on the appliance.
- The following procedure describes how to upload the settings file from a Windows host with plink.exe (PuTTY Link) installed. To download this utility, search the internet for "plink putty download".

To upload the bpre-settings.xml file to the appliance, do the following:

1) Copy the Pattern Definition to an easy-to-find location on your local host (for example, `c:\bpre\bpre-pattern-def.xml`).

2) Open a command prompt on your local host and **cd** to the folder where plink.exe is installed.

3) Enter the following command:

```
plink -ssh <username>@appliancename -pw <password> BPRE-Config add < <path-and-file>:
```

For example:

```
plink -ssh admin@appliancename -pw admnpss BPRE-Config add < c:\bpre\bpre-settings.xml:
```

4) Check the CLI output. You must start the BPRE process if you see a message such as `status: BPRE doesn't appear to be running`. For more information, see "Start/Stop the BPRE Process".

The following illustrates an example session:

```
>cd c:\putty
>plink -ssh myadmin@172.17.5.71 -pw mypassword BPRE-Config add < c:\bpre-settings.xml
-- status: reading the BPRE configuration file from standard input
-- status: BPRE was successfully configured
-- status: BPRE was restarted
-- status: operation complete
```

Start/Stop the BPRE Process

To start or stop the BPRE process on an appliance, you need to log in to the appliance as admin using an SSH-enabled CLI program such as PuTTY. Then enter the following command:

```
BPRE-Config start
```

To stop BPRE, enter the following command:

```
BPRE-Config stop
```

Query BPRE Results with Web Service URLs

The BPRE web service has the following URL format:

```
https://appliance-name:8443/webservice/DataServiceServlet/type/bpre/csv?UserName=username&Passwor
d=password&start=start-time&end=end-time&query=query-string
```

A BPRE URL has the following user-specified fields:

- appliance-name
- username
- password
- start-time and end-time

  The start and end of the time window to search for BPRE data. The following time formats are supported:

  – UNIX time, in microseconds

Use an online conversion tool to generate UNIX time values. Make sure you enter values in microseconds (seconds x 10e6). Thus if you use a converter that generates UNIX times in seconds, add six (6) zeros to the number.

Example: `start=1435968000000000&end=1438646400000000`

You can also use the following format to specify a time window of "the most recent x seconds" `start=last` *`x`*`&end=0`

Example (last 10 minutes, URL-encoded): `start=last%20`*`600`*`&end=0`

- *`%year-%month-%day %hours:%minutes:%seconds`*

  Example: `start=2015%-07%-04% 00:%00:%00&end=2015%-08%-04% 00:%00:%00`

- *`%year-%month-%day %hours:%minutes`*

  Example: `start=2015%-07%-04%00:%00&end=2015%-08%-04%00:%00`

- Query string

  IP, port, match, and search-definition parameters that define the matching packets to display in the CSV output:

  - `srcip`—Source IP
  - `dstip`—Destination IP
  - `srcport`—Source port (TCP/UDP)
  - `dstport`—Destination port (TCP/UDP)
  - `matchid`—Integer ID of a specific "`<match>`" defined in the bpre-settings.xml file
  - `searchdefinitionid`

    Integer ID or string key of a specific "`<search-definition>`" defined in the bpre-settings.xml file. This query returns all packets for all matches in the search definition.

  Note the following:

  - A query string can include any of the above parameters. You can also create compound conditions using any of the following operators:

    = (boolean)

    != (boolean)

    `and` (boolean)

    `or` (boolean)

    `>` (for start/end times, IPs, or ports)

    `<` (for start/end times, IPs, or ports)

  - The query string should be within single quotes.

    Here are some examples of compound queries:

    ```
    ...&query='matchid=9 and srcip=10.1.1.1'

    ...&query='srcip=172.16.28.1 or dstip=172.16.28.1'

    ...&query='(srcip=172.16.28.1 and dstip=157.166.255.18) or (srcport=22 and dstport = 22)'
    ```

  - It is good practice to use URL encoding for non-alphanumeric characters if you want to retrieve BPRE data using custom scripts or CLI-based programs such as wget.

    The following query strings show the previous examples with URL encoding:

```
...%26query%3D%27matchid%3D9%20and%20srcip%3D10.1.1.1%27

...%26query%3D%27srcip%3D172.16.28.1%20or%20dstip%3D172.16.28.1%27

...%26query%3D%27(srcip%3D172.16.28.1%20and%20dstip%3D157.166.255.18)%20or%20(srcport%3D
22%20and%20dstport%20%3D%2022)%27
```

BPRE Output Format

BPRE queries are in CSV format with the following fields for each matching packet:

Table 2 BPRE Output

Field	Description
TIMESTAMP	Timestamp of the matching packet in seconds.microseconds
MATCHID	Matching id attribute of the “`<match>`” tag from the xml input file.
OFFSET	Byte offset of the first matching byte from byte 0 of the packet payload.
SEARCHDEFINITIONID	Matching ID attribute of the “`<search-definition>`” tag from the XML input file
SOURCEIP	Source IP of the matching packet
SOURCEPORT	Source port of the matching packet
DESTIP	Destination IP of the matching packet
DESTPORT	Destination port of the matching packet
REGEXBYTES	If the “`<match>`” criteria for a packet includes a “`<capture-definition>`”, any captured bytes are added to the end of the row for that packet. **Note—**BPRE captures bytes in hex format. Use an online conversion tool if you need to convert the captured bytes to ASCII.

The following output is based on the example settings file shown in Figure 1 on page 534. This sample file defines three matches: a data request, a timeout error, and a generic error message with the byte pattern “Forbidden:” somewhere in the payload. If an “Error Forbidden” packet is found, BPRE captures this string and the following 10 bytes in hex format.

```
#TIMESTAMP,SEARCHDEFINITIONID,MATCHID,SOURCEIP,SOURCEPORT,DESTIP,DESTPORT,REGEXBYTES
1338150589.811739,0,0,172.16.11.176,58944,172.16.1.8,5
1338150589.812024,1,0,172.16.11.176,58944,172.16.1.8,5
1338150591.874367,1,0,172.16.11.176,53619,172.16.1.8,5
1338150591.874625,1,0,172.16.11.176,53619,172.16.1.8,5
1338150600.388042,0,0,172.16.1.24,29405,172.16.1.8,5
1338150600.388394,2,0,172.16.1.24,29405,172.16.1.8,5,466f7262696464656e3a4e6f7441757468727a640d0a
1338150600.428311,0,0,172.16.1.24,29406,172.16.1.8,5
1338150600.428563,1,0,172.16.1.24,29406,172.16.1.8,5
1338150600.432370,1,0,172.16.1.24,29407,172.16.1.8,5
1338150600.432620,1,0,172.16.1.24,29407,172.16.1.8,5
1338150600.436457,0,0,172.16.1.24,29408,172.16.1.8,5
1338150600.436704,2,0,172.16.1.24,29408,172.16.1.8,5,466f7262696464656e3a4c6963457870697265640d0a
```