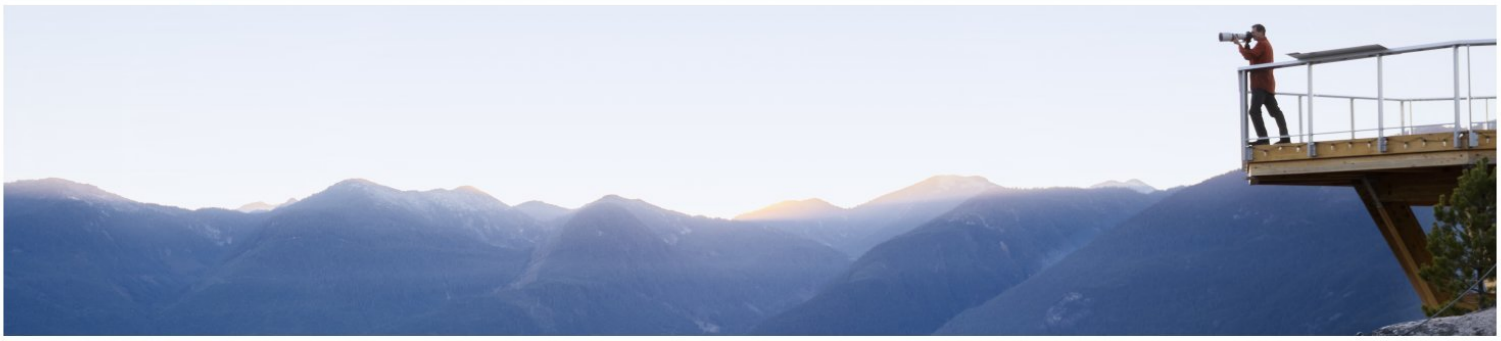
Hewlett Packard Enterprise

Real User Monitor

Version 9.40, Released August 2017

Real User Monitor Deployment Planning Guide

Published August 2017



Legal Notices

Warranty

The only warranties for Hewlett Packard Enterprise products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HPE shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

Restricted Rights Legend

Confidential computer software. Valid license from HPE required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright Notice

© Copyright 2017 Hewlett Packard Enterprise Development LP

Trademark Notices

Adobe® and Acrobat® are trademarks of Adobe Systems Incorporated.

AMD and the AMD Arrow symbol are trademarks of Advanced Micro Devices, Inc.

Google™ and Google Maps™ are trademarks of Google Inc.

Intel®, Itanium®, Pentium®, and Intel® Xeon® are trademarks of Intel Corporation in the U.S. and other countries.

iPod is a trademark of Apple Computer, Inc.

Java is a registered trademark of Oracle and/or its affiliates.

Microsoft®, Windows®, Windows NT®, Windows® XP, and Windows Vista® are U.S. registered trademarks of Microsoft Corporation.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates.

UNIX® is a registered trademark of The Open Group.

Documentation Updates

The title page of this document contains the following identifying information:

- Software Version number, which indicates the software version.
- Document Release Date, which changes each time the document is updated.
- Software Release Date, which indicates the release date of this version of the software.

To check for recent updates or to verify that you are using the most recent edition of a document, go to: https://softwaresupport.hpe.com/group/softwaresupport/search-result?keyword=.

This site requires an HPE Passport account. If you do not have one, click the **Create an account** button on the HPE Passport Sign in page.

Support

Visit the HPE Software Support website at: **https://softwaresupport.hpe.com**

This website provides contact information and details about the products, services, and support that HPE Software offers.

HPE Software Support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the support website to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Download software patches
- Manage support contracts
- Look up HPE support contacts
- Review information about available services
- Enter into discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HPE Passport user and sign in. Many also require a support contract. To register for an HPE Passport ID, go to **https://softwaresupport.hpe.com** and click **Register**.

To find more information about access levels, go to:
https://softwaresupport.hpe.com/web/softwaresupport/access-levels

HPE Software Integrations and Solutions

Visit the Integrations and Solutions Catalog at **https://softwaresupport.hpe.com/km/KM01702731** to explore how the products in the HPE Software catalog work together, exchange information, and solve business needs.

Contents

Chapter 1: Introduction 5
Chapter 2: Implementation Strategies 6
Data Collection Methods 6
Choosing the Data Collection Method 11
Chapter 3: RUM Components and Placement Overview 12
RUM Components 12
Communication Channels 14
Components Placement 15
Common Deployment Scenarios 17
Ports 18
Chapter 4: Splitting Monitored Traffic by Engines/Probes 20
RUM Sniffer Probe Load Balancing 20
RUM Client Monitor Probe Load Balancing 21
Chapter 5: Using RUM Sniffer Probes 23
Supported Monitored Protocols 23
RUM For Citrix Deployment 24
Traffic Mirroring Techniques using RUM Sniffer Probe 25
SSL Decryption of Monitored Traffic 37
Monitor Servers or End Users 40
Send Documentation Feedback 41

Chapter 1: Introduction

This document provides the information necessary for designing and planning the deployment of the RUM solution. For system requirements and installation instructions, see the Real User Monitor Installation and Upgrade Guide.

This document includes:

- Deployment considerations and common implementation strategies
- Client side monitoring deployment planning—specifically, mobile planning
- Network-specific hardware and software requirements and network settings
- Instructions regarding implementation in different types of environments, such as:
  - Clustering
  - DMZ isolation
  - Virtualization
  - Network port mirroring/tapping

Chapter 2: Implementation Strategies

This section describes the various implementation strategies and how to choose the strategy best for your implementation and includes the following topics:

- "Data Collection Methods" below
- "Choosing the Data Collection Method" on page 11

Data Collection Methods

This section describes the different methods by which the RUM Probe can obtain monitored data.

It includes the following topics:

- "RUM Data Collection Methods" below
- "Data Collection Using a Network Tap or Switch Configuration" on the next page
- "Sniffing Using the RUM Server Collector" on page 8
- "RUM Client Monitor Probe" on page 9

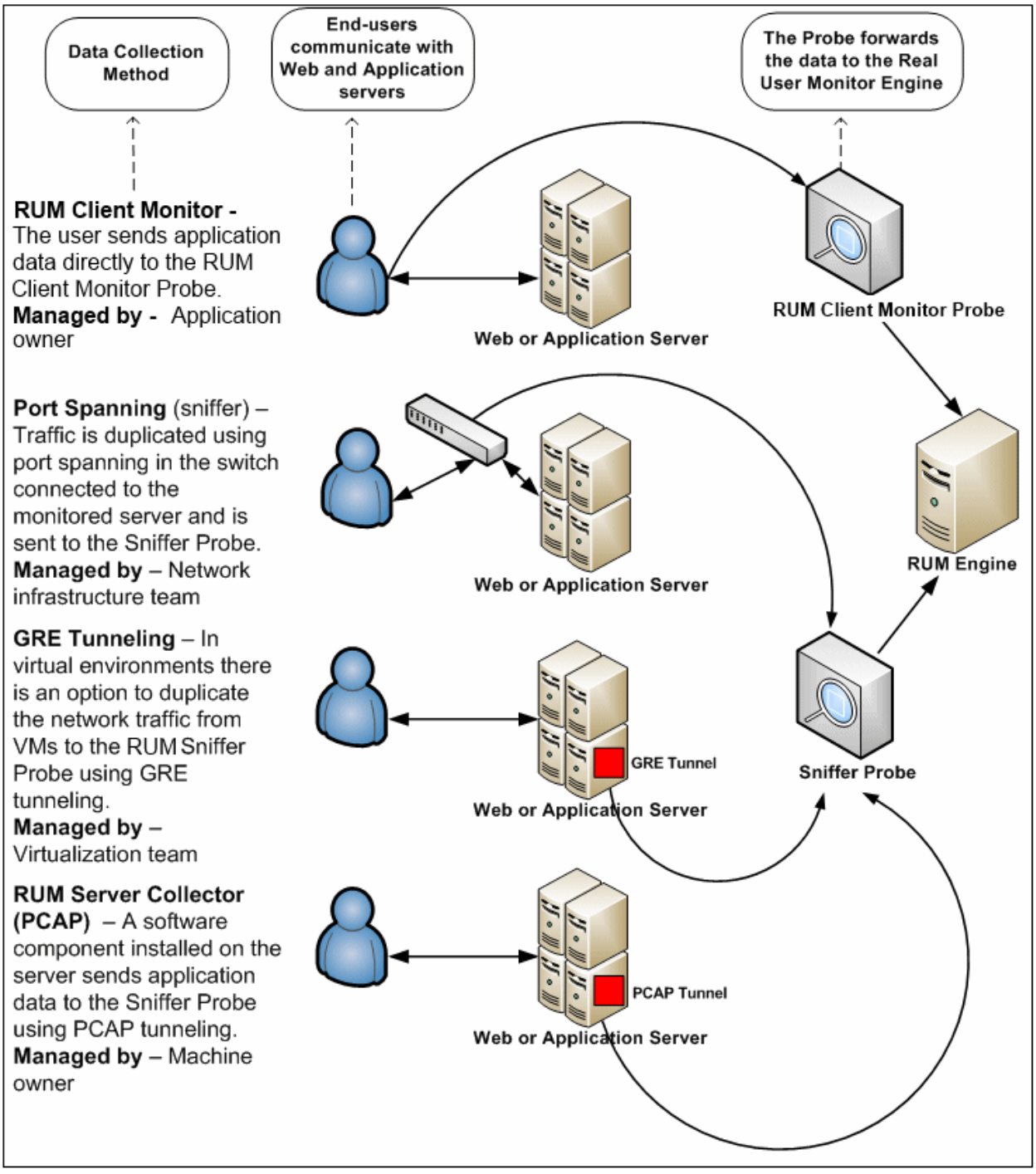
RUM Data Collection Methods

There are a number of ways by which the RUM Probe can obtain data for monitored applications. The available monitoring solutions depend on the type of RUM Probe you use:

- **Sniffer Probe** data collection methods:
  - **Network tap or switch configuration.** For details, see "Data Collection Using a Network Tap or Switch Configuration" on the next page.
  - **RUM Server Collector.** For details, see "Sniffing Using the RUM Server Collector" on page 8.
  - **VMware.** For details, see "Duplicating Traffic for RUM with VMware" on page 30.
- **RUM Client Monitor Probe.** For details, see "RUM Client Monitor Probe" on page 9.

For details on installing the RUM Probe, refer to the Real User Monitor Installation and Upgrade Guide.

The following diagram illustrates the data flow for different RUM Probes and their data collection methods:

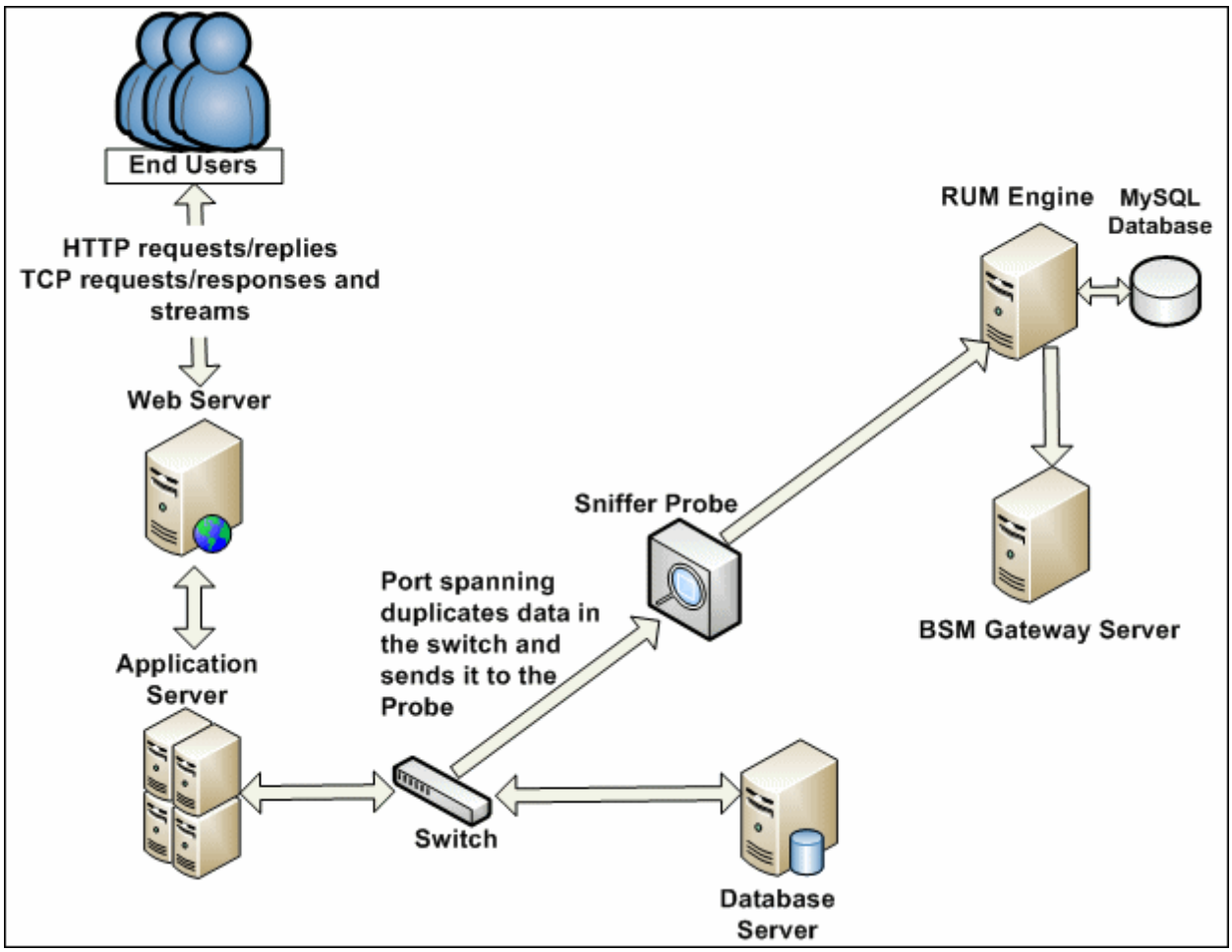


Data Collection Using a Network Tap or Switch Configuration

The Sniffer Probe is a non-intrusive, passive listening device that is subject to the same traffic the server receives. It is plugged into a network tap that is connected to a monitored server. As end-user traffic passes through the tap, the probe listens to requests and responses sent to and from the server. In this way, data is tracked all the way from the end-user's IP address to the server handling the request.

Note: The configuration in a switch is usually called a mirror or span port, depending on the switch vendor.

The following diagram illustrates the flow for data collection using a network tap or switch configuration:



Sniffing Using the RUM Server Collector

Note: The RUM Server Collector poses a significant impact on network performance and a slight impact on the server's CPU. Therefore, use of the RUM Server Collector is limited to networks with very low throughput and is mostly recommended for proof of concept (POC) purposes. For example, you can use the RUM Server Collector if the network traffic is not high (150 Mbps for windows and 250 Mbps for Linux) and if you are aware of the implications as noted above.

When it is not possible to use a network tap or port spanning, you can install the RUM Server Collector on a monitored server so that the server sends packets directly to the probe (that is, the probe receives packets directly from the monitored agent). The probe then processes the packets and forwards data to APM in the regular manner.

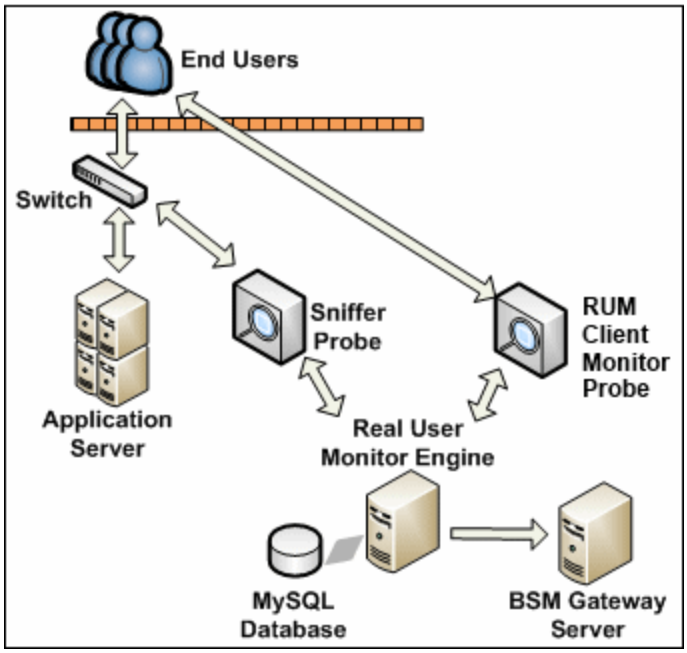
The benefit of this is that you only have to be the machine owner of the server on which you install the RUM Server Collector, and are not dependent on the infrastructure team. However, this method does require you to install a software component on the server that runs your application, that is more than just a plug-in to the application.

Note: The recommended data collection method is to use a network tap or port spanning when possible as this eliminates the need to configure and maintain the monitored servers for data collection.

RUM Client Monitor Probe

The RUM Client Monitor Probe collects user experience data for your users. Unlike the RUM Sniffer Probe that collects traffic by monitoring network packets using traffic duplication, the RUM Client Monitor Probe receives data for monitored web or mobile applications directly from the client (end user).

The following diagram shows the conceptual difference between the RUM Sniffer Probe and the RUM Client Monitor Probe:



You use different solutions for monitoring web and mobile applications with the RUM Client Monitor Probe:

- The RUM client monitor solution enables you to monitor web applications through an end user's Internet browser. For details, see "Using the RUM Client Monitor Solution to Monitor Web Applications" below.
- The RUM mobile solution enables you to monitor mobile applications through an app on an end user's mobile device. For details, see "Using the RUM Mobile Solution to Monitor Mobile Applications" on the next page.

Note: As the RUM Client Monitor Probe monitors traffic from the client side, reported data in EUM reports is displayed for domains instead of running software elements.

Using the RUM Client Monitor Solution to Monitor Web Applications

The RUM client monitor solution enables you to monitor web applications through a user's Internet browser and sends the collected data from the browser directly to the RUM Client Monitor Probe. The advantage of monitoring traffic on the client side instead of the server side, is that the metrics are more accurate as the traffic includes data for the following:

- Proxies
- Content Delivery Networks (CDN)
- External sources (other servers than the one providing the HTML that provide external content such as images)

Note: Data about failures, however, is not included as only successful pages are reported back to the client.

You enable the RUM client monitor solution by installing a JavaScript snippet in the specific HTML pages you want to monitor. This snippet is responsible for collecting performance data on the client machine and sending the collected data to a specific RUM Client Monitor Probe machine. For details, see "Installing the JavaScript Snippet" in the Real User Monitor Installation and Upgrade Guide.

Supported Browsers

The RUM client monitor solution supports the following Internet browsers:

- Internet Explorer
- Google Chrome
- Firefox
- Safari
- Opera

Using the RUM Mobile Solution to Monitor Mobile Applications

The RUM mobile solution enables you to monitor mobile applications through apps on a user's mobile device and sends the collected data from the app directly to the RUM Client Monitor Probe. The advantages of monitoring traffic on the client side instead of the server side are:

- The user experience is measured including the latency of the mobile network.
- Data is broken down by operating system, device, connection, and application version.

This section includes the following topics:

- "Supported Operating Systems" below
- "Getting Started with the RUM Client Monitor Probe" below
- "Using the RUM Mobile Solution to Monitor Mobile Applications" above

Supported Operating Systems

The RUM mobile solution supports the following operating systems:

- Android 2.3 and later
- iOS 5 and later

Getting Started with the RUM Client Monitor Probe

To use the RUM mobile solution with the RUM Client Monitor Probe, you must:

1. Install the RUM Client Monitor Probe. For details, see "Installing the RUM Client Monitor Probe" in the Real User Monitor Installation and Upgrade Guide.
2. Instrument the mobile app. For details, see "Instrumenting Mobile Apps for Android" and "Instrumenting iOS Apps" in the Real User Monitor Installation and Upgrade Guide.

   Note: For more information on monitoring hybrid apps, see "Monitoring Hybrid Applications" in the Real User Monitor Installation and Upgrade Guide.

3. In APM, configure the mobile application whose pages you want to monitor. For details, see "Using the RUM Mobile Solution to Monitor Mobile Applications" above.

Choosing the Data Collection Method

In most cases, you must use only one type of probe to monitor your data - RUM Sniffer Probe or RUM Client Monitor Probe.

You should use RUM Sniffer Probes when monitoring back end traffic or any protocol which is not regular HTTP. The RUM Client Monitor Probe is easier to deploy because you do not need the security team to provide private keys for SSL decryption or for the network team to configure traffic duplication.

You can use the RUM Sniffer Probe or RUM Client Monitor Probe or a combination of the two when monitoring front end traffic .

The following are the advantages of each type of probe:

- **RUM Client Monitor Probe** - Data collection is performed on the client side, so all data is monitored including data which does not arrive at the customer data center (such as CDN (Content Delivery Network) or other data).
- **RUM Sniffer Probe** - Only data that arrives at the customer data center is monitored. However, this data provides a better breakdown of the network problems and also enables advanced features such as session replay which is not available when using the RUM Client Monitor Probe.

HPE recommends using a combination of both probes.

Chapter 3: RUM Components and Placement Overview

This chapter describes the different components in RUM, their roles, the communication between them and the placement consideration for each one.

This chapter includes the following topics:

- "RUM Components" below
- "Communication Channels" on page 14
- "Components Placement" on page 15
- "Common Deployment Scenarios" on page 17

RUM Components

The RUM solution is comprised of the following components:

- **RUM Probes**

  There are two kinds of probes:

  - **RUM Sniffer Probe** – This probe is responsible for sniffing and basic analysis of the traffic. For more information, see "Components Placement" on page 15.

  - **RUM Client Monitor Probe** – This probe, also known as the Client Probe, is responsible for client monitoring data. Client monitoring data is collected from web and mobile applications that are instrumented to send performance data measured at the client end. For more information, see "Components Placement" on page 15.

  Both types of probes receive their configuration from the RUM Engine and collected data is pulled by the Engine. One RUM Engine may manage several probes (of both types), but each probe is connected to only one RUM Engine. The number of probes to use, and where to place them, depends on the network settings, probe type (Client Monitor or Sniffer), desired monitoring metrics, and amount of traffic.

  Note: In this document, wherever *RUM Probe* is mentioned, it refers to both probe types. Otherwise, the specific probe type will be mentioned.

- **RUM Engine**

  The RUM Engine collects the information from one or more RUM Probes, performs additional processing, and stores raw data in its database. Aggregations are reported to APM. When APM reports require access to raw data, APM accesses the RUM Engine HTTP-REST APIs, which retrieve data from the RUM Database. The RUM Engine receives its configuration from the APM server and sends it to the relevant RUM Probes. A single APM deployment can manage several RUM Engines, but each RUM Engine only reports to a single APM system. Each application configured in APM to be monitored by RUM can be monitored by several RUM Engines and RUM Probes.

- **RUM Server Collector**

The RUM Server Collector is an optional component installed on a monitored server. It duplicates the server traffic to a Sniffer Probe.

> **Note:** The RUM Server Collector poses a significant impact on network performance and a slight impact on the server's CPU. Therefore, use of the RUM Server Collector is limited to networks with very low throughput and is mostly recommended for proof of concept (POC) purposes. For example, you can use the RUM Server Collector if the network traffic is not high (150 Mbps for windows and 250 Mbps for Linux) and if you are aware of the implications as noted above.

- **RUM Database**

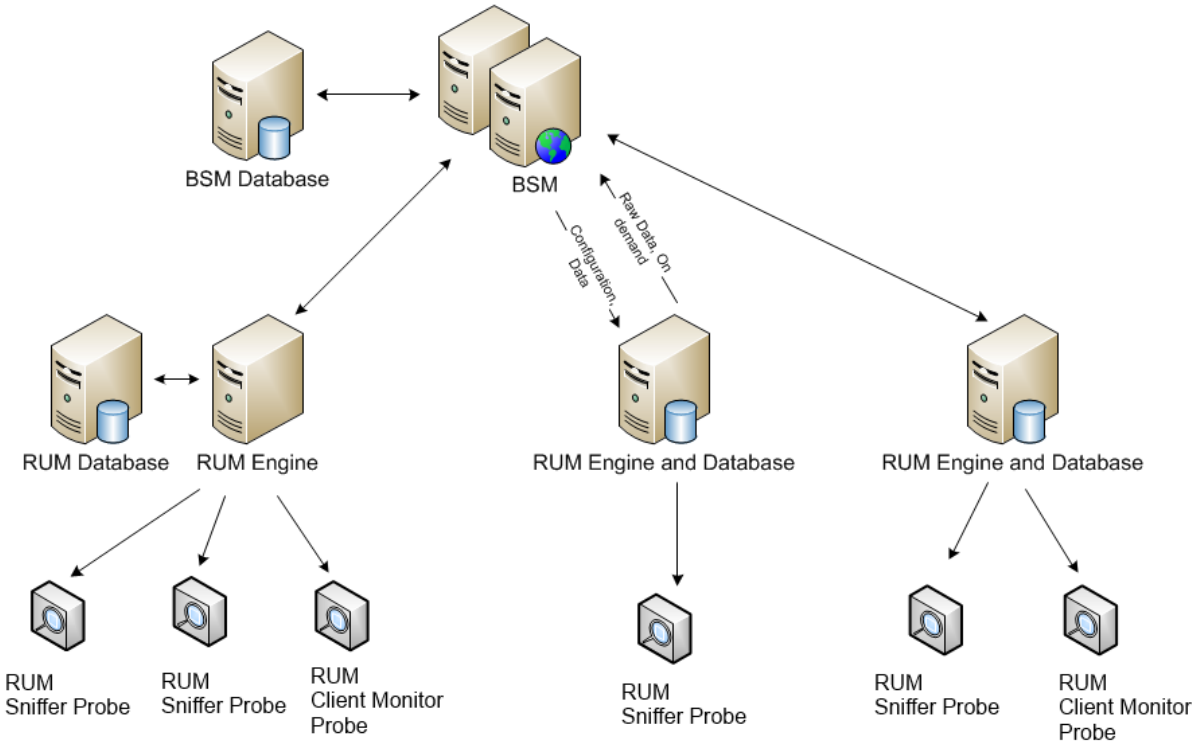
  The RUM Database is used for saving raw data for short term usage. The RUM Database is MySQL, which comes embedded in the RUM installation. The RUM Database can be installed on the same server as the RUM Engine (although it is highly recommended to use different hard disks), or on a separate server.

- **HPE Application Performance Management (APM)**

  HPE Application Performance Management (APM) provides a suite of monitoring products that offer a comprehensive methodology for monitoring and measuring IT services from a business perspective. APM tools enable you to identify problems, understand their business impact, and prioritize the triage and remediation process. Therefore, APM enables your IT organization to optimize the performance and availability of applications in production, and proactively resolve problems when they arise, thus assisting your organization to deliver more effective business results with lower IT costs.

RUM Component Relationship

The following diagram displays the relationship between the different RUM components.



Communication Channels

The following are the communication channels:

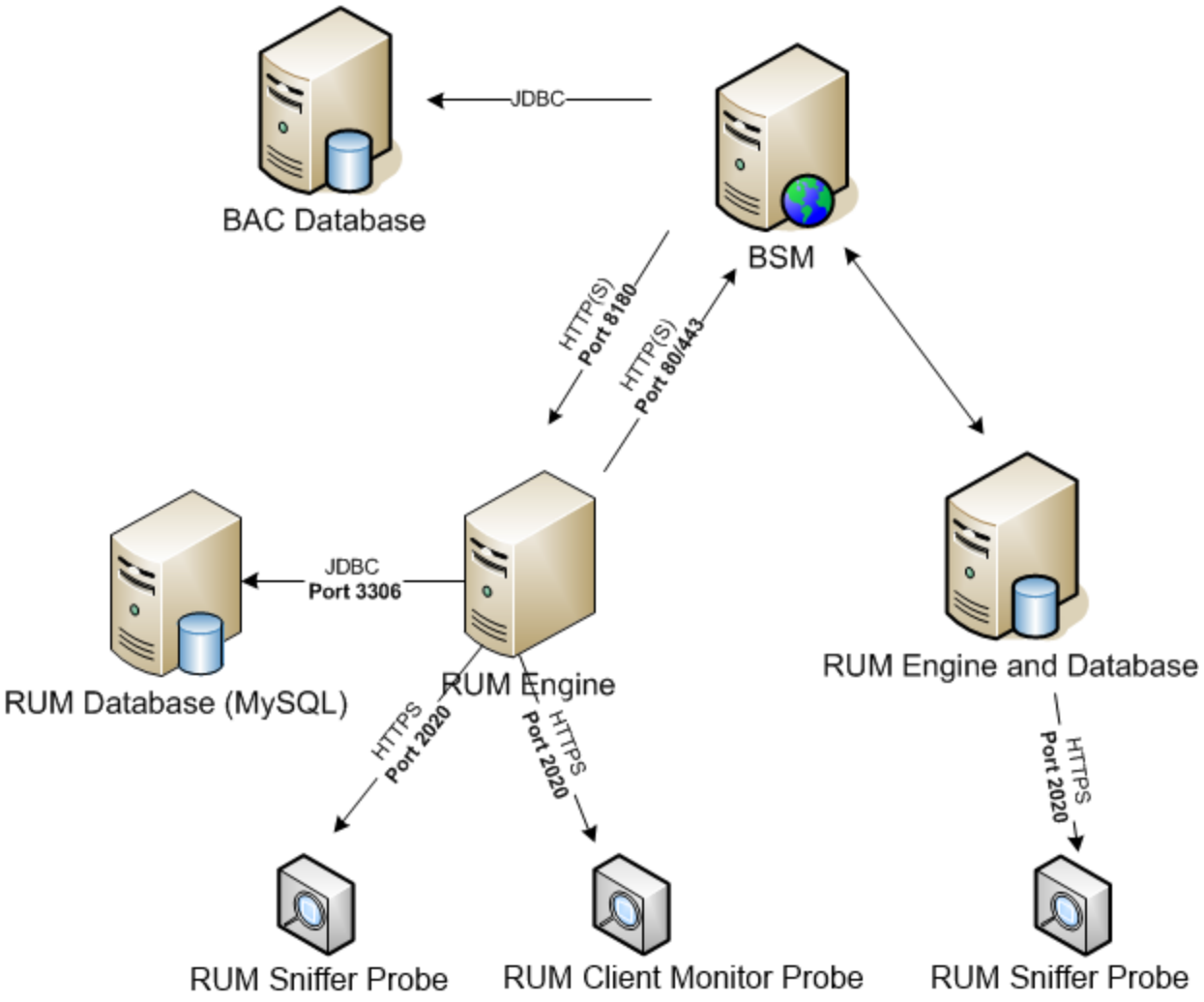
- **RUM ↔ APM:** Bi-directional communication. RUM gets its configuration from APM and sends data to APM. RUM is accessed on demand when there is a need to retrieve raw data.

  There is an option to configure a proxy for RUM in APM EUM Admin, so when APM accesses the RUM Engine to retrieve raw data, the RUM Engine is accessed via the proxy server. The only communication is between the RUM Engine and the APM Gateway server. RUM does not access the APM Processing server or the APM Database, and APM does not access the RUM Probe or the RUM Database directly.
- **RUM Engine → RUM Probe:** The communication is in one direction only, from the RUM Engine to the RUM Probe. This enables the placement of the RUM Probe in a less secure environment (such as DMZ), while keeping the engine in the internal LAN without the need to open another port in the firewall.

RUM uses the following communication channels:

Channel	Direction	Default Port	Default Protocol
Get configuration from APM, and send data to APM	RUM Engine -> APM Gateway	80	HTTP
Retrieve raw data from RUM	APM Gateway -> RUM Engine	8180	HTTP
Send configuration and get data from the RUM Probe	RUM Engine -> RUM Probe	2020	HTTPS
Store and retrieve raw data from the RUM Database	RUM Engine -> RUM Database	3306	JDBC
RUM Engine Administration	User -> RUM Engine	8180	HTTP

The following diagram maps the different communication channels:



Components Placement

This section includes the following topics:

- "RUM Sniffer Probe" below
- "RUM Client Monitor Probe " on the next page
- "RUM Engine and Database" on the next page
- "RUM Engine and RUM Probe" on page 17
- "RUM Engine and APM Gateway Server " on page 17
- "APM Server" on page 17

RUM Sniffer Probe

The RUM Sniffer Probe is responsible for sniffing traffic. Therefore, the main concern when deploying the RUM Sniffer Probe is the physical ability to perform the traffic sniffing. In a physical network, this means the

RUM Sniffer Probe has to be physically close to the network tap or switch from which the traffic is mirrored. Typically, a physical RUM Sniffer Probe has one NIC dedicated for communication with the RUM Engine and at least one other for traffic sniffing, so that the RUM Sniffer Probe does not need to communicate on the same network it monitors. On virtual networks, the RUM Sniffer Probe placement depends on the probe and VMware versions.

It is not uncommon to have to deploy different probes in different rooms, buildings, or data centers.

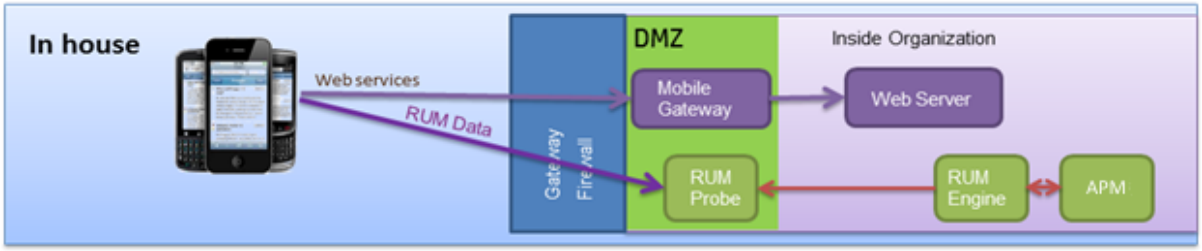
The other thing to consider concerning the RUM Sniffer Probe placement is capacity. On heavily loaded environments, it might be necessary to split the load between several RUM Sniffer Probes. For more details, see the RUM Sizing Guide.

RUM Client Monitor Probe

The RUM Client Monitor Probe accepts data from the instrumented applications through the Internet. Therefore, it must have an open port to the Internet. The RUM Client Monitor Probe can be installed either in the DMZ or in the cloud.

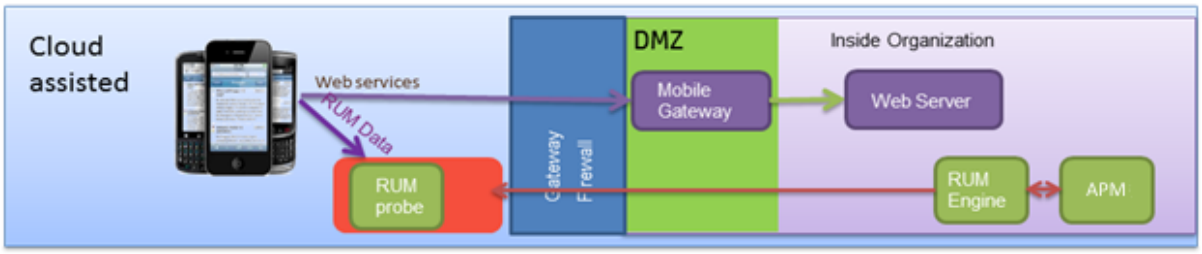
- **In-House Deployment**

  In an in-house deployment, the RUM Client Monitor Probe is installed in the DMZ behind the organization gateway and firewall. This enables you to apply gateway and firewall policies on the client-to-probe communication, but requires the firewall to allow client communication to enter. This is the recommended deployment.



- **Cloud-Assisted Deployment**

  If your security policy or other technical limitations prohibits client access to a RUM Client Monitor Probe in the DMZ, the Rum Client Monitor Probe can be located in an external network. (This is possible since engine-probe communication is always engine to probe and never vice versa.) In this case, it is strongly recommended to use HTTPS communication only for client to probe communication. Additional security considerations for this type of deployment are discussed in the RUM Hardening Guide.



RUM Engine and Database

The RUM Engine collects information from the RUM Probe and stores it in its database or sends it to the APM server. Therefore, it has to have access to all three components.

The RUM Database can reside on the same machine as the RUM Engine, but since I/O is a major bottleneck in RUM, it is recommended to deploy these components on different hard drives.

If the RUM Engine and RUM Database are located on two different servers, it is highly recommended to install them on the same LAN and ensure that there is good network communication between them.

While it is technically possible to share a single RUM Database between several RUM Engines, it is not recommended due to capacity reasons.

RUM Engine and RUM Probe

The communication between the RUM Engine and the RUM Probe is one-sided, from the RUM Engine to the RUM Probe, so it is possible for the RUM Probe to be located in a less secure environment than the RUM Engine and RUM Database. For example, while the RUM Probe sometimes has to reside in the DMZ in order to be physically close to the web server, it is recommended that the RUM Engine and RUM Database reside on the LAN for security reasons.

The throughput between the RUM Engine and the RUM Probe is relatively high, so it is recommended they are placed in such a way that they have a good network connection between them.

RUM Engine and APM Gateway Server

The throughput between the RUM Engine and the APM Gateway server is much less than the throughput between the RUM Engine and the RUM Probe. Therefore, if necessary, the RUM Engine can be placed on a geographically remote location and communicate with APM over the WAN. However, since the communication is two-sided, ports have to be opened in both directions.

APM Server

The APM Gateway and Processing servers are usually placed where they can be accessed by users.

It is also possible to connect RUM to APM in HPE SaaS. For details see "Common Deployment Scenarios" below.

Common Deployment Scenarios

This section includes the following topics:

- " Large Deployment" below
- "Virtualized Data Center" on the next page
- "Monitoring Distributed Data Centers" on the next page
- "HPE - SaaS" on the next page

Large Deployment

Since the RUM solution comprises three component types (the RUM Probe, the RUM Engine, and APM), it is easily scaled and is flexible for large deployments. You should consider the following issues for large deployments:

- **APM capacity.** For details, refer to the APM and RUM Sizing Guides.
- **Hardware utilization.** Many customers prefer to deploy strong, but fewer, servers instead of deploying a physical server for each component. Since RUM is a software solution, and not a hardware appliance solution, it is possible to use the following methods:
  - Deploy several of the RUM components on one HPE blade server. One blade server contains several physical computers, each one on a different PC card.
  - Use a DL38x server with several hard drives and deploy VMware ESX. Deploy each component on a different virtual machine.

Virtualized Data Center

RUM Engines and RUM Probes can be deployed on virtual machines and they can also monitor virtual machine servers.

For details on mirroring traffic to the RUM Probe, see "Virtual Network" on page 29.

Monitoring Distributed Data Centers

Sometimes it is necessary to collect traffic from data centers in different geographical locations, which are connected over a WAN. Since the throughput between the RUM Engine and the RUM Probe is higher than the throughput between the RUM Engine and APM, it is recommended to deploy the RUM Engine close to the RUM Probe. The throughput between the RUM Engine and RUM Probe in a loaded system is less than 10 Mbps.

HPE - SaaS

In an HPE SaaS deployment, APM is placed in the HPE SaaS environment, while the RUM Engine and RUM Probe are deployed at the customer site. Since APM connects to the RUM Engine via port 8180, it is necessary to open port 8180 at the customer site. This is usually done by some kind of VPN, or other secure channel.

It is also possible to connect RUM to APM in HPE SaaS.

Ports

RUM requires the following ports to be opened for incoming traffic by the machine, gateway, proxy, and firewall:

5001, 5002, 8180, 8443, 8009, 1198, 1199, 4544, 4545, 8183, 22734, 5000, 3306, 2020

The machine, gateway, proxy, and firewall can be set to a different port number.

For the RUM Client Monitor Probe to get external monitoring reports, ports 8080 (HTTP) and 2021 (HTTPS) need to be opened in the firewall. The port numbers can be configured. Also, it is possible to allow only HTTP or HTTPS communication. See the RUM Hardening Guide 9.23 for more details.

The following table summarizes the usages of those ports:

Component	Process	Port	Description
Engine	Nanny	5001	Probe Web Interface
Engine	Nanny	5002	Probe Web Interface
Engine	HPRUM – JBOSS	8180	Web console
Engine	HPRUM – JBOSS	8443	SSL Web console
Engine	HPRUM – JBOSS	8009	Tomcat AJP connector
Engine	HPRUM – JBOSS	1198	JNDI Service
Engine	HPRUM – JBOSS	1199	JNDI JNP bootstrap
Engine	HPRUM – JBOSS	4544	RMI / JRMP invoker

Component	Process	Port	Description
Engine	HPRUM – JBOSS	4545	JBOSS invoker
Engine	HPRUM – JBOSS	8183	JBOSS web service
Engine	HPRUM – JBOSS	22734	JBOSS web service
Engine	HPRUM – JBOSS	5000	JBOSS web service
Engine	HPRUM – JBOSS	22736	JBOSS Unified Invoker
Engine	HPRUM – JBOSS	22737	JBOSS TS Recovery Manager
Engine	HPRUM – JBOSS	22738	JBOSS TS Status Manager
Engine	HPRUM – JBOSS	22739	JBOSS TS process ID
Engine	HPRUM – JBOSS	22740	JBOSS clear/non-SSL Remoting
Engine	HPRUM – JBOSS	22741	JBOSS SSL Remoting
Engine	HPRUM – JBOSS	22742	JBOSS HornetQ
Engine	HPRUM – JBOSS	22743	JBOSS HornetQ
Engine	HPRUM – JBOSS	22744	JBOSS HornetQ
Engine	HPRUM – JBOSS	22745	JBOSS legacy RMI/JRMP invoker
Database	MySQL	3306	DB PORT
Probe	Probe	2020	Probe Web Services

Chapter 4: Splitting Monitored Traffic by Engines/Probes

In the RUM solution, the RUM Probe tracks the user session.

When there is a need to split the traffic between several RUM Probes, it important that packets and reported client metrics from the same user session are always directed to the same RUM Probe.

RUM Sniffer Probe Load Balancing

Usually, a user session does not cross different applications, but in some scenarios it does.

In many monitored applications, there is a sticky session which means that once a user session is forwarded to a specific server, the session continues with the same server. In this case, it helps to split the traffic between RUM Probe (see "Sniffer Load Balancing using Servers/Client IP Filters" below).

Probe Load Balancing vs Engine Load Balancing

If one RUM Probe cannot handle all the traffic, there is a need to install several RUM Probes and spread the traffic among the probes.

In the RUM Sniffer Probe, there are four levels of filtering:

- There is an option to split the traffic before the traffic arrives at the RUM Sniffer Probe in the hardware level (Tap/Mirror Ports, etc.)
- According to TCP port – This filter is the most efficient and is done on the kernel level
- According to IP address – This filter is second in efficiency
- According to URL /applications

In the RUM Engine, the load balancing is performed differently in a performance perspective.

Load Balancing using Application Definition

Load balancing using application definitions is the most common and simplest way to perform load balancing. There is an option to define which RUM Engines/RUM Probes will monitor the application (in APM Admin), so the load is spread among the applications.

This method performs filtering according to URL /applications in the RUM Sniffer Probe if the applications are configured on the same TCP port. It performs filtering according to TCP port if the applications are configured to different ports. In most cases, performing filtering according to the TCP port it is sufficient and is the easiest method.

Sniffer Load Balancing using Servers/Client IP Filters

If one RUM Sniffer Probe cannot handle a specific application, there is an option to split the traffic in the server's filters:

- **Split traffic according to server IP/Port in the UI:** The filter is set per RUM Sniffer Probe using the RUM web console. In the application, click **Configuration > Probe Management** and click the **Servers Filters** button and add the servers.

- **Split traffic according to Client IP – manually:** There is a manual option to configure load balancing using the client range. However, this requires that on an IP protocol level, you can distinguish between different client IPs. If you deploy a Sniffer Probe behind the load balancer you will not be able to distinguish between different client IPs. Note that this feature does not work for applications with Tier Discovery enabled.

  It is necessary to edit the following file in the RUM Engine machine:

  Conf\configurationmanager\Beatbox_probe-name-or-ip_Const_Configuration.xml

  In the **collector** section, add the **cluster main-cluster** section as following:

```
[collector]

device all

[cluster main-cluster]

client_mask 255.255.255.255

server_mask 0.0.0.0

member_id 1 4

heartbeat_interval 0
```

  This definition will split the traffic based on the client IP address into **4** probes. This probe will be the **1**st out of 4 which will monitor every client IP address (which after hash and modulo 4, receives the value of 1-1=0).

Sniffer Load Balancing using Hardware (Taps/Mirror Ports Definition)

Various switch and tap venders provide load balancing capabilities.

In addition, some taps have a filtering option which provides the ability to perform the required load balancing based on the client and/or server. Such taps are called *filterable tap*. For details on filterable tabs, see http://en.wikipedia.org/wiki/Network_tap.

Some switches allow load balancing based on various traffic characteristics, such as VLAN ID. For details, check for the switch vendor's filtering capabilities. For example, http://www.cisco.com/en/US/products/hw/switches/ps708/products_tech_note09186a008015c612.shtml.

RUM Client Monitor Probe Load Balancing

Load Balancing Using Application Definition

As long as a Client Probe's capacity allows, it is recommended to use a single Probe for each defined application, while the same Probe can handle different applications. In this way, you can provide different Probe URLs when instrumenting different applications.

Load Balancing Between Probes

When multiple Probes are monitoring the same application, load balancing is required between the Probes in a way that ensures that all requests from same session are always directed to the same Probe.

If the load balancer is able to retrieve the client's IP address, you can perform load balancing according to the client's IP address. This is a standard load balancing option.

If the load balancer is unable to retrieve the client's IP address (for example, when it installed behind the proxy server) you can perform the load balancing according to the Session ID:

- With the RUM client monitor solution, session cookies can be used to ensure session stickiness. The cookie name is unique_id by default, but this name can be configured by setting another value to the cm_impl.sessionIdCookieName property in the snippet. (See the RUM Client Monitor client configuration documentation for more details.) Alternatively you can set the session cookie by Load Balancer.
- With the RUM Mobile solution, you need to configure load balancing according to the HTTP header X-RUM-SESSION-ID.

Configuring Client Monitor Probe Load Balancing

To keep a session persistent, RUM Client Monitor Probe load balancing requires you to configure a specific persistence profile and assign it to a virtual server.

For example, in F5 define the following:

1. In the BIG-IP Configuration Utility portal, create the following iRule which creates the persistence profile:

```
when HTTP_REQUEST {
   if { [HTTP::header exists "X-RUM-SESSION-ID"] } {
     persist uie [HTTP::header "X-RUM-SESSION-ID"]
   }
}
```

2. Assign this iRule to the RUM virtual server. The RUM virtual server is the server that receives the data from the client.

Configuring Health Monitoring

Create a health monitor to determine if a Client Monitor Probe is active.

For example, in F5 define the following:

1. In the Send String field of the Monitors page, enter:

```
GET /\r\n
```

2. In the Receive String field of the Monitors page, enter:

```
hello
```

Chapter 5: Using RUM Sniffer Probes

The following sections relate only to the RUM Sniffer Probe:

- "Supported Monitored Protocols" below
- "RUM For Citrix Deployment" on the next page
- "Traffic Mirroring Techniques using RUM Sniffer Probe" on page 25
- "SSL Decryption of Monitored Traffic" on page 37
- "Monitor Servers or End Users " on page 40

Supported Monitored Protocols

Supported and Unsupported Networks

- **Network Protocols**
  - RUM supports Ethernet only.
  - GRE (ERSPAN) is supported as of RUM version 9.2x.

    To monitor GRE (ERSPAN) traffic, it is necessary to edit the following file on the engine machine:

    conf\configurationmanager\BeatBox_default_const_coniguration.xml

    In the **collector** section, add `process_gre true`.
  - RUM can monitor VLAN Tags and MPLS traffic over Ethernet.

    When monitoring MPLS traffic, it is necessary to edit the following file on the engine machine:

    conf\configurationmanager\BeatBox_default_const_coniguration.xml

    In the **collector** section, add `use_bpf true`.
  - As of version 9.23, RUM can monitor fragmented IP packets.

    When monitoring fragmented IP layer traffic, it is necessary to edit the following file on the engine machine:

    conf\confiogurationmanager\Beatbox_default_const_configuration.xml

    In the **collector** section, add `support_ip_fragmentation true`.
- **Supported Protocols**

  RUM Sniffer Probe supports different protocols over TCP and UDP. The application protocol is configured in APM (EUM Admin) as part of the application definition.

  The list of supported RUM Sniffer Probe protocols is updated in almost every minor release and can be found in the RUM Administration Guide.

  On the client side, the RUM Client Monitor Probe monitors HTTP and HTTPS only.

  RUM supports TLS 1.0, 1.1, and 1.2 and SSL 2.0 and 3.0.

RUM For Citrix Deployment

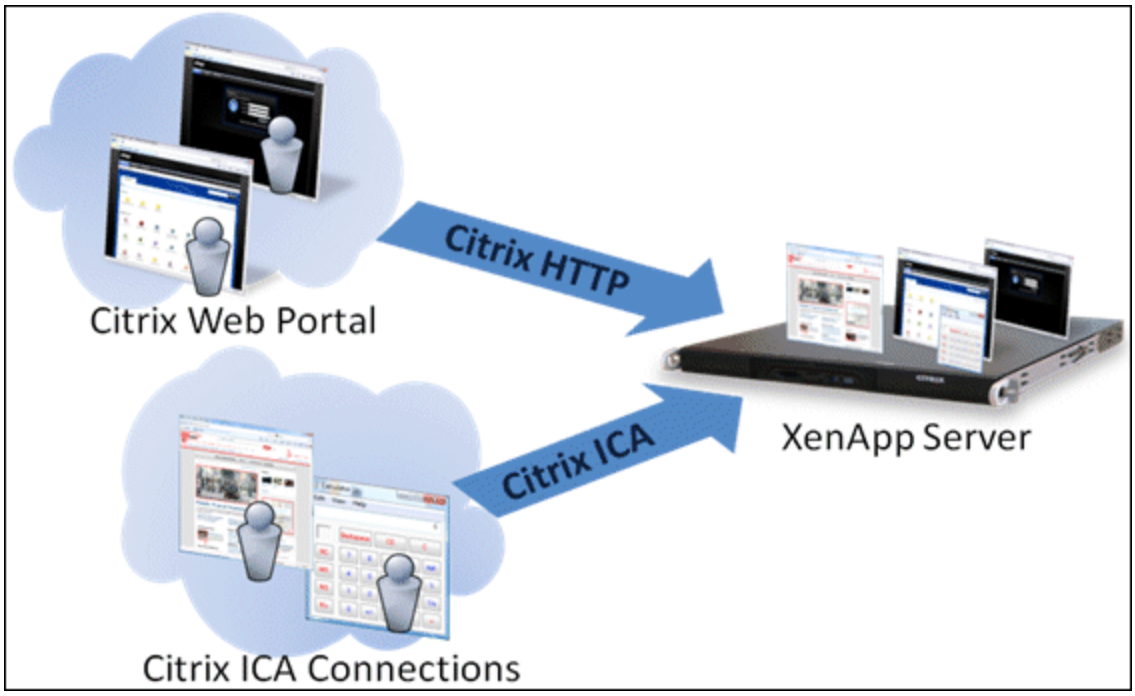
The section contains the following topics:

- "Monitoring Incoming Traffic to a XenApp Server" below
- "Monitoring Outgoing Traffic from a XenApp Server" below

Monitoring Incoming Traffic to a XenApp Server

End users can connect to a Citrix XenApp server via a Web portal or a direct ICA connection.

The following diagram displays typical traffic between end users and a Citrix XenApp server:



In this scenario, the you open a Citrix Web portal and select one of the published application. This creates an ICA session. The application you select runs on the XenApp server in the ICA session. You run the application remotely.

Alternatively, you can create an ICA connection directly, without going through a Web portal.

To monitor this traffic with RUM 9.x or later, no agent installation is required and you just configure the application in APM using the **Citrix HTTP** and **Citrix ICA** application templates. For user interface details on creating RUM applications in APM, see "Real User Monitor Application Configuration Wizard" in the APM Application Administration Guide.

Monitoring Outgoing Traffic from a XenApp Server

When multiple users connect to the same XenApp server, requests sent from the XenApp server originate from the same client, regardless of the originating end user.

The following diagram displays multiple users connected to the same XenApp server, each running an instance of Internet Explorer to connect to a Web server. In this scenario, the traffic from the multiple users

are combined and all connections opened to the Web server originate from a single client, which is the XenApp server.

When monitoring the Web server with RUM, we recommend that you analyze each end user separately, rather than having a single client combining all the traffic. To do this, install the RUM VDI Agent on the XenApp server, regardless of the originating end user. In this scenario, the traffic from each user remains distinct.

Traffic Mirroring Techniques using RUM Sniffer Probe

In traffic mirroring, traffic is mirrored to the RUM Sniffer Probe. Each RUM Sniffer Probe gets both transmit (TX) and receive (RX) traffic—there is no option to send TX traffic to one RUM Sniffer Probe and RX traffic to another RUM Sniffer Probe .

The RUM Sniffer Probe does not limit the number of interfaces to which it can listen. Any limitation is based on hardware.

This section contains the following topics:

- "Physical Network" below
- "This section contains the following topics:" on page 29
- "Duplicating Traffic for RUM with VMware" on page 30
- "Blade Server" on page 33

Physical Network

In order for the RUM Sniffer Probe to perform its monitoring tasks, it needs to receive a duplicate stream of traffic to analyze. There are two methods to implement this:

- Network taps (preferred), see "Network Taps" on the next page
- Spanning/mirroring, see "Spanning/Mirroring" on page 28

Network Taps

Network taps are inline devices which act in a similar manner to a video or stereo splitter; you need to match the physical media. This means that if the network link you are trying to tap is copper, you will need a copper tap. If the network link is fiber, you will need a fiber tap.

Tapping is done physically, and hence has to deal with physical limitations. For example, copper taps are always active, which means they require power in order to not only produce copies of the traffic, but also to properly pass the original traffic through without interruption. This means that if you tap a copper link and the copper tap loses power, your actual network connection is severed momentarily until a relay closes within the

tap that restores the pass-through. The Ethernet link will flap for a few milliseconds, which might have an effect on the network, depending upon how it is configured.

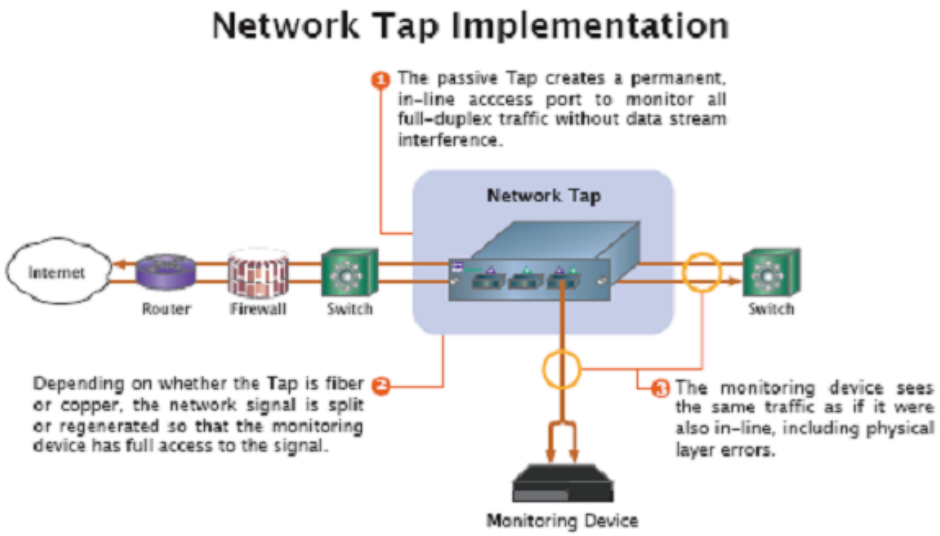
Fiber taps, however, are passive, which means that even when power is lost, they will pass the original traffic without loss. Only copies of the traffic are affected by loss of power to a fiber tap.

While this may seem to render copper taps too risky, in reality the enterprise grade commercial vendors (such as, Datacom and NetOptics) have designed their taps with redundant power supplies and special circuitry which almost eliminates this risk.

Network taps fall into several categories depending on their configuration:

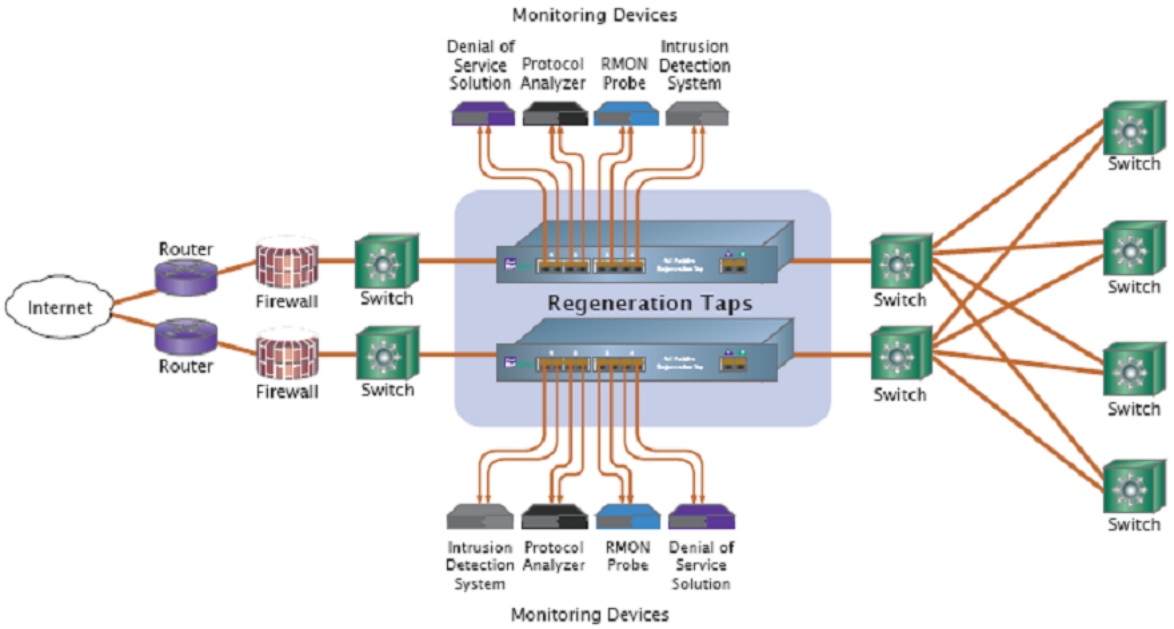
- **Network Taps**

  These are devices that typically have one set of inputs (transmit and receive) to one set of outputs (transmit and receive).



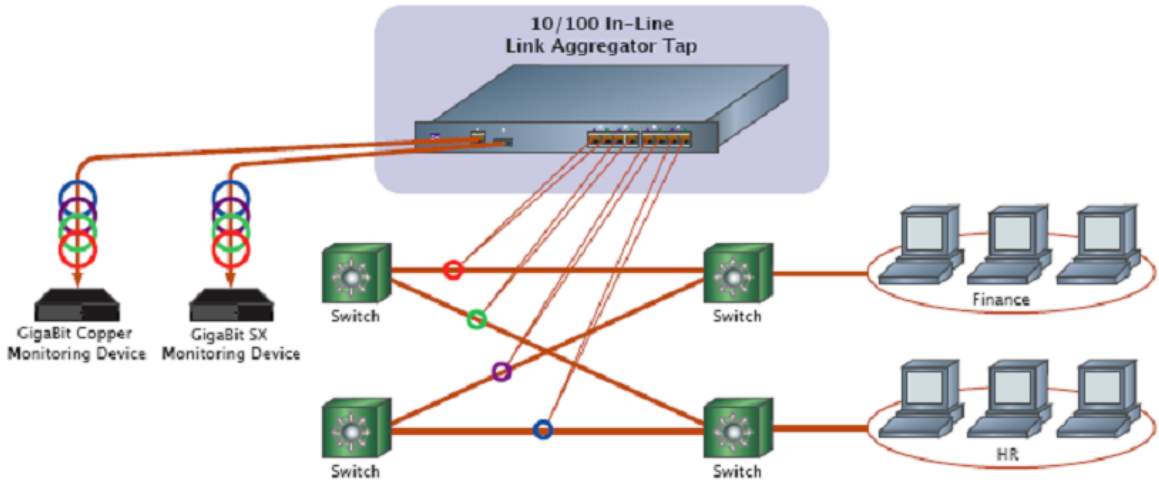
- **Regeneration Taps**

  These are devices that have a set of inputs (transmit and receive) and are able to send the same output to multiple transmit and receive ports to allow for multiple devices to evaluate the same traffic.



- **Aggregation Taps**

  These are devices that are able to take in multiple sources of traffic and combine that traffic so that it can be consumed by one monitoring device.



Tap Vendors

Two commonly used vendors in RUM implementations are:

- Datacom Systems – http://www.datacomsystems.com
- Ixia – https://www.ixiacom.com

These vendors provide network, aggregation, and regeneration taps.

Copper vs. Fiber Taps

In the case of fiber taps, the key internal components—fiber-optic splitters—do not require power, so they are not vulnerable to a power outage. Two key aspects of fiber taps are split ratio and light source. The splitter

divides the light signal into two streams, and the tap needs to make sure the network signal has enough strength to reach its destination.

The split ratio for fiber taps is determined by factors such as the device's transmitter strength and receiver sensitivity, net losses from cable connections, and length.

Because the goal is to maximize the signal retained in the network, the optimal split ratio is the highest. So if 70-to-30, 60-to-40, and 50-to-50 split ratios are viable, splitters with a 70-to-30 split ratio are optimal.

Splitters also need to support the light source used on the links. For example, Gigabit SX devices transmit data using 850-nm lasers, so Gigabit SX taps should have compatible splitters. This ensures accuracy in the insertion losses dictated by the chosen split ratio. Performance does not degrade from the laser light intensity, which could occur if splitters supporting lower-intensity LED transmission are used on these links.

Copper taps regenerate the transmitted network signal, instead of splitting it. Regeneration amplifies the signal to a level where it can be received by the other network devices and the monitoring device.

Regenerating the electrical signal takes place on a powered board. When power is available to the tap, the electrical signal passes through an open bypass circuit to the area of the board where regenerating and directing the signal occurs. Some newer copper taps feature fail-safe reserve power within the tap to maintain this availability.

If power is not available, the bypass circuit closes, so the transmitted signal passes directly to the receiving network device. The bypass circuit requires no external input, so copper taps remain passive.

Learn More about Taps

The vendors listed in "Tap Vendors" on the previous page have a large amount of useful information and their product pages are an excellent source of information.

Spanning/Mirroring

You implement spanning/mirroring by configuring the switch to duplicate a stream of traffic from one or more ports to other ports. You connect one of the sniffing NICs on the RUM Sniffer Probe to the physical port on the switch to which the duplicate stream is sent. You need to ensure that both receive (RX) and transmit (TX) streams are duplicated

Spanning/Mirroring vs. Taps Comparison

Factor	Tap	Span/Mirror
Quality and Performance	Always a clear signal and copy	Under high loads, switch may drop packets
Description	An *in stream* device that passively copies traffic to another source, like a stereo splitter	Uses the software in the network device to copy traffic from one or more ports to another
Type	Hardware device	Software configuration
Cost	Varies by vendor and type of device but usually low ($1K-$10K)	None. Implemented in software on the switch
Scalability	High. You can combine and link taps much like network hubs or switches	Limited. Each switch has a limited number that can be implemented

Factor	Tap	Span/Mirror
Flexibility	None. All traffic is duplicated by default on most taps. However taps with filtering capabilities do exist (e.g., Datacom System's Filtered SINGLEstream™—http://www.datacomsystems.com/).	May allow you to filter the traffic
Installation	Requires cabling and may also require switch downtime	On-the-fly, no downtime
Risk	Since taps are inline devices, tap failure may sever the network link. However, taps are available with redundant power supply and circuit design which virtually eliminate this risk.	None. Switch will drop packets under load

Virtual Network

This section contains the following topics:

- "VMware Solution" below
- "VMware Solution Concerns" on the next page

Note: Microsoft virtual environments are not supported.

VMware Solution

There are several challenges when trying to monitor virtual environments. The main issue is how to copy the traffic from a virtual environment to the RUM Probe.

In non-virtual environments, there is an option to use a span/mirror port on the switch, or to use a network tap in order to duplicate the traffic to the RUM Probe. If there is such an option, use the mirror port on a regular switch or network tap. It is better than using the VM infrastructure.

In some virtual environments you cannot use these options.

In that case, there are several solutions:

- On VMware version 3.X and 4.X, there is an option to install the RUM Probe on a VM on the same ESX as the application, and configure the application to work in promiscuous mode.

  The problem with such a concept is the vMotion's feature. This enables VMs to be relocated from one ESX to another. When the VM is relocated, the RUM Probe is no longer on the same ESX and therefore does not receive traffic.

  In order to cope with this issue, there is an option in vCenter to create a link between two VMs and to configure them to move together between ESXs. Customers may have a problem with such a configuration as it may have implications on how vSphere handles the environment.
- You can use Cisco Nexus 1000 virtual switches (which can be downloaded from the VMware site). These switches have an option to configure the ERSPAN port which uses GRE tunneling to send packets to the RUM Probe IP.
- ESX version 5.1 supports ERSPAN by default.
- ESX version 5 has an option to configure a generic mirror port which should work even if there is vMotion between ESXs.
- There is an option to use third party hardware in order to duplicate traffic. There are several companies which provide taps for VMware, such as NetOptics and vssmonitoring. If the customer already has such a solution, the RUM Probe can connect to them.

VMware Solution Concerns

- **Security**

  Configuring promiscuous mode on a VMware machine allows other VMs to listen to the traffic. It is strongly recommended to allow only VMs to receive traffic duplication.

  For details, see the VMware knowledge base: http://kb.vmware.com/selfservice/microsites/search.do?language=en_US&cmd=displayKC&externalId=1000880

- **vMotion**

  One of the features supplied by VMware is vMotion. vMotion provides the ability to *Move running virtual machines from one physical server to another with no impact to end users*. A full description can be found in: http://www.vmware.com/products/vmotion/overview.html.

  The vMotion feature can detrimentally affect traffic monitoring on ESX versions lower than 5.x.

Duplicating Traffic for RUM with VMware

There are various considerations and solutions for duplicating traffic for RUM when the RUM Sniffer Probe is installed on a VMware platform.

VMware can use two different kinds of switches—a regular switch, and a DV (distributed) switch which is part of VMware's Enterprise Plus solution (for more information, refer to VMware's vSphere web page (http://www.vmware.com/vmwarestore/vsphere_purchaseoptions.html).

Working with VMware networks involves defining a virtual switch, which is equivalent to a regular switch. On each virtual machine, you can configure one or more virtual ports, which are equivalent to regular network cards. You then connect each such network card to the virtual switch. For more information, refer to the VMware Virtual Networking Concepts guide (http://www.vmware.com/files/pdf/virtual_networking_concepts.pdf).

There are several options for configuring port mirroring for a APM Sniffer Probe with VMware ESX. For details, see "Configuring Packet Duplication" on the next page.

Differences Between Virtual and Physical Environments

There are several differences between physical and virtual environments when duplicating network traffic to the RUM Sniffer Probe:

- In a physical network, there is an option to use network taps or switch port mirroring, but in a virtual environment these features are only partially available.
- In physical environments, the monitored application is tied to a physical machine which RUM then monitors to retrieve the application's network traffic.
- In virtual environments, the Vmotion feature enables an application to jump between virtual machines. If traffic is duplicated for a specific machine, when an application jumps to another machine the probe will not be able to monitor the traffic.
- In a physical network, different applications regularly use different machines and there is an option to duplicate traffic at a machine level (switches, network taps, and so forth). In a virtual environment, different VMs are regularly installed on the same physical machine.

Security

When you configure the promiscuous mode for a VMware machine, it enables other machines to listen to the traffic. For better security, it is recommended that you only allow specifically required machines to receive traffic duplication.

For more information, refer to Capturing virtual switch traffic with tcpdump and other utilities in the VMware knowledge base.

Configuring Packet Duplication

The solutions for duplicating traffic to the RUM Sniffer Probe differ depending on whether Vmotion is disabled or enabled. See the following:

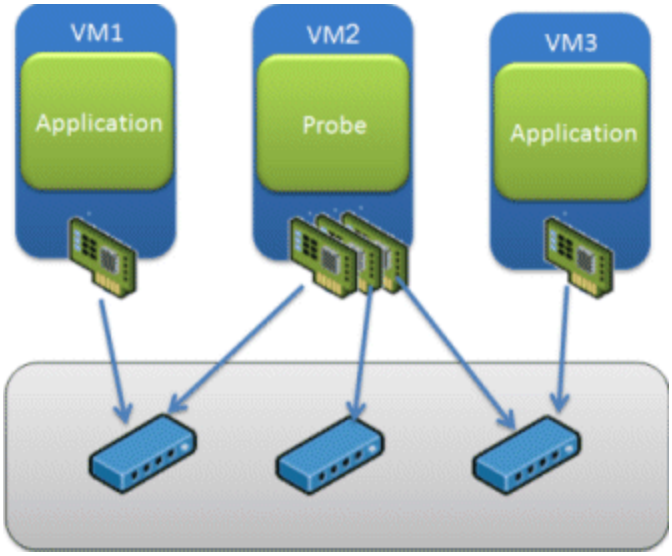
- "Monitoring Traffic when Vmotion is Disabled" below
- "Monitoring Traffic when Vmotion is Enabled" on the next page
- "Packet Duplication Using GRE Tunnel" on the next page

Monitoring Traffic when Vmotion is Disabled

It is easier to duplicate traffic when Vmotion is disabled, although in most deployments vMotion is enabled.

- **Monitoring Traffic when the Probe is Installed on the Same ESX**

  There is an option to deploy the RUM Sniffer Probe on the same ESX as the monitored application. This is illustrated for the virtual machine VM2 in the example below:



  There are two entities in VMware—a virtual switch and portgroups. By default, a guest operating a system's virtual network adapter only receives frames specific for that adapter. Placing the guest adapter in promiscuous mode causes it to detect all frames passed through the virtual switch that are allowed under the VLAN policy for the associated portgroup.

  To configure a portgroup or virtual switch for promiscuous mode using the Virtual Infrastructure Client:

  a. Select the ESX Server host and click the **Configuration** tab.
  b. Click **Properties** next to the virtual switch or portgroup.
  c. To allow promiscuous mode on the virtual switch or portgroup, select the name of the virtual switch or portgroup and click **Edit**.
  d. Click the **Security** tab.
  e. From the Promiscuous Mode drop-down menu, select **Accept**.

  Since there is an option to configure several virtual adapters for each virtual machine, and since each virtual adapter can be connected to another virtual switch, there is an option to configure the probe to get the traffic from any virtual switch on the ESX.

For more information, refer to Configuring promiscuous mode on a virtual switch or portgroup in the VMware knowledge base.

- **Monitoring Traffic when the Probe is Installed on a Different ESX**

  For details, see "Packet Duplication Using GRE Tunnel" below.

Monitoring Traffic when Vmotion is Enabled

If your environment is configured to use the Vmotion feature, you must verify that the RUM Sniffer Probe will get continuous traffic from the monitored application. You can use one of the following solutions:

- **Use GRE Tunneling**

  For details, see "Packet Duplication Using GRE Tunnel" below

- **Use Virtual Dedicated Taps**

  Another option is to use virtual taps, but this requires the addition of third party equipment. One of the recommended third parties is the NetOptic Phantom solution.

- **Use RUM Server Collector**

  Deploy the RUM Server Collector to run on the same VM as the monitored application.

  Note: The RUM Server Collector poses a significant impact on network performance and a slight impact on the server's CPU. Therefore, use of the RUM Server Collector is limited to networks with very low throughput and is mostly recommended for proof of concept (POC) purposes. For example, you can use the RUM Server Collector if the network traffic is not high (150 Mbps for windows and 250 Mbps for Linux) and if you are aware of the implications as noted above.

- **Install the RUM Probe on the Same ESX as the Monitored Application**

  Configure the RUM Probe to run on the same ESX as the monitored application. There is an option in VMware to keep two VMs together on the same ESX to improve efficiency. This option can also be used for keeping the probe and monitored application on the same ESX.

  To keep two VMs together:

  a. Right-click the DRS cluster that contains the VMs you want to keep together.
  b. Select **Edit Settings**.
  c. Under VMware DRS click **Rules**.
  d. To create a rule click **Add**.
  e. In the dialog box that opens, enter a name for the rule.
  f. Select the rule type **Separate virtual machines/Keep virtual machines together**.
  g. Click **Add** to select the virtual machines to which the rule applies. Accept the selection by clicking **OK** twice.
  h. Click **OK** to save and exit the settings.

  For further details, see http://www.vmware.com/pdf/vmware_drs_wp.pdf.

Packet Duplication Using GRE Tunnel

When the probe is deployed on a different ESX than the monitored application, it is necessary to send all the packets from the monitored application to the RUM Sniffer Probe.

In ESX 4.x there is no option to duplicate the traffic with the regular VMware virtual switch. Instead, there is an option to use Cisco Nexus switches to mirror the traffic between different ESXs.

The feature is called ERSPAN. For a detailed description of how to configure ERSPAN, refer to the Cisco documentation at http://www.cisco.com/en/US/docs/switches/datacenter/nexus1000/sw/4_0_4_s_v_1_3/system_management/configuration/guide/n1000v_system_9span.html.

RUM 9.23 supports GRE tunneling, which is required for ERSPAN.

In ESX 5.x there is an option to define a mirror port between two ESXs. For details, see http://blogs.vmware.com/networking/2011/08/vsphere-5-new-networking-features-port-mirroring.html.

Note: For VMware 5.1, use the web interface for the port mirror definition (this feature does not exist in the vSphere client).

References

The following references can provide additional, useful information.

- VMware mirroring configuration (see the book Mastering VMware vSphere 4 - (http://books.google.co.il/books?id=F_8qs4lPLpwC&pg=PA207&lpg=PA207&dq=vmware+promiscuous+configure&source=bl&ots=GmuIJOwSnv&sig=fIe0UCKHHPm4GRmBFj4PsLr4wQY&hl=en&ei=st49TJzMOMe2ngf0s_jdDg&sa=X&oi=book_result&ct=result&resnum=7&ved=0CDEQ6AEwBg#v=onepage&q&f=true)
- Defining promiscuous mode in VMware (http://communities.vmware.com/message/371562)

Blade Server

A blade server is a stripped down server computer with a modular design, optimized to minimize the use of physical space and energy. Each blade computer is placed on a card which is placed in an enclosure.

The enclosure (or chassis) performs many of the non-core computing services found in most computers, such as power supply and networking.

Blades Networking

Usually, all blade servers in an enclosure have one embedded switch. HPE blades regularly use procurve switches.

Blade servers may be used by the RUM Probe or by monitored applications.

- The monitored servers are blade servers.

  This is a common option. It is necessary to configure a mirror port for all relevant blade servers in the enclosure, duplicate the traffic to a specified port (a port is an entry point on the network cable), and duplicate the traffic from the enclosure to the RUM Probe.

- The RUM Probe is installed on a blade server.

  This option is more complicated since not all switches have the option to duplicate traffic to the blade.

  In this case, you must use Pass-Thru. Pass-Thru is hardware that helps to connect directly to the blade server network card, providing an option to duplicate the traffic directly to the RUM Probe.

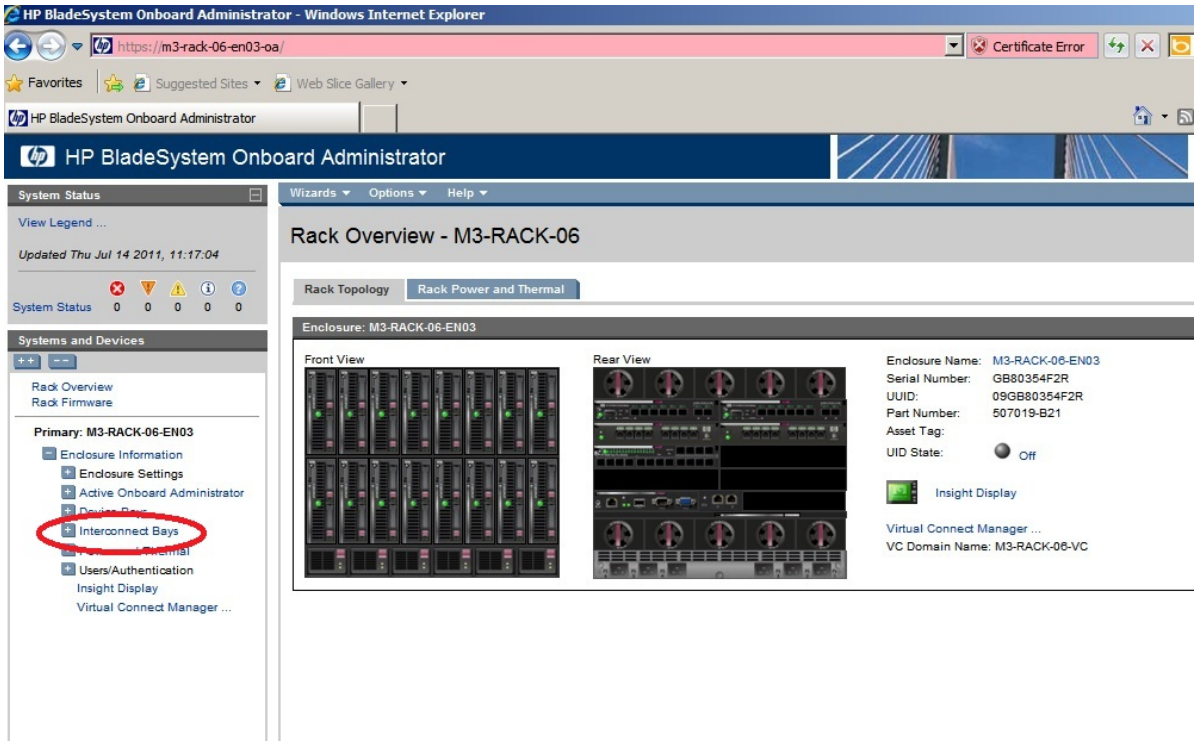
  We have encountered problems in the past with several Pass-Thru models that dropped packets, therefore we recommend using https://www.hpe.com/us/en/product-catalog/servers/server-adapters.hits-12.html for fiber network cards.

  It is necessary to use Pass-Thru even when the RUM Probe is installed on the same enclosure as the monitored servers.

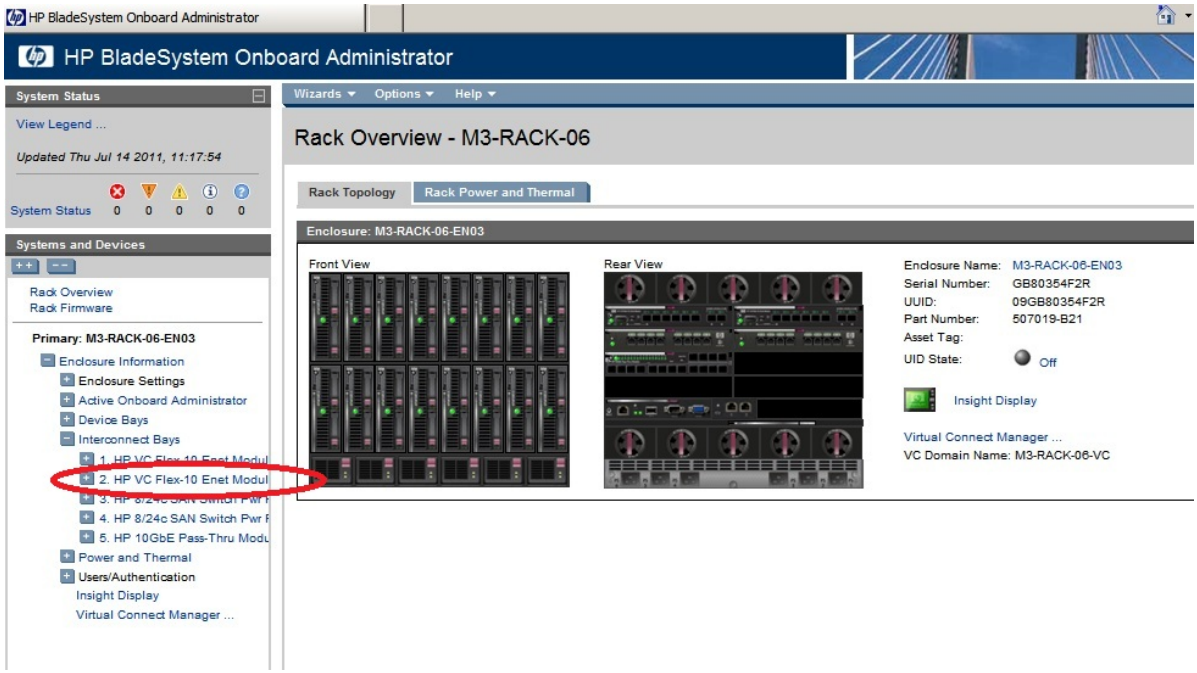
Sample Configuration

The following provides instructions for a sample configuration.

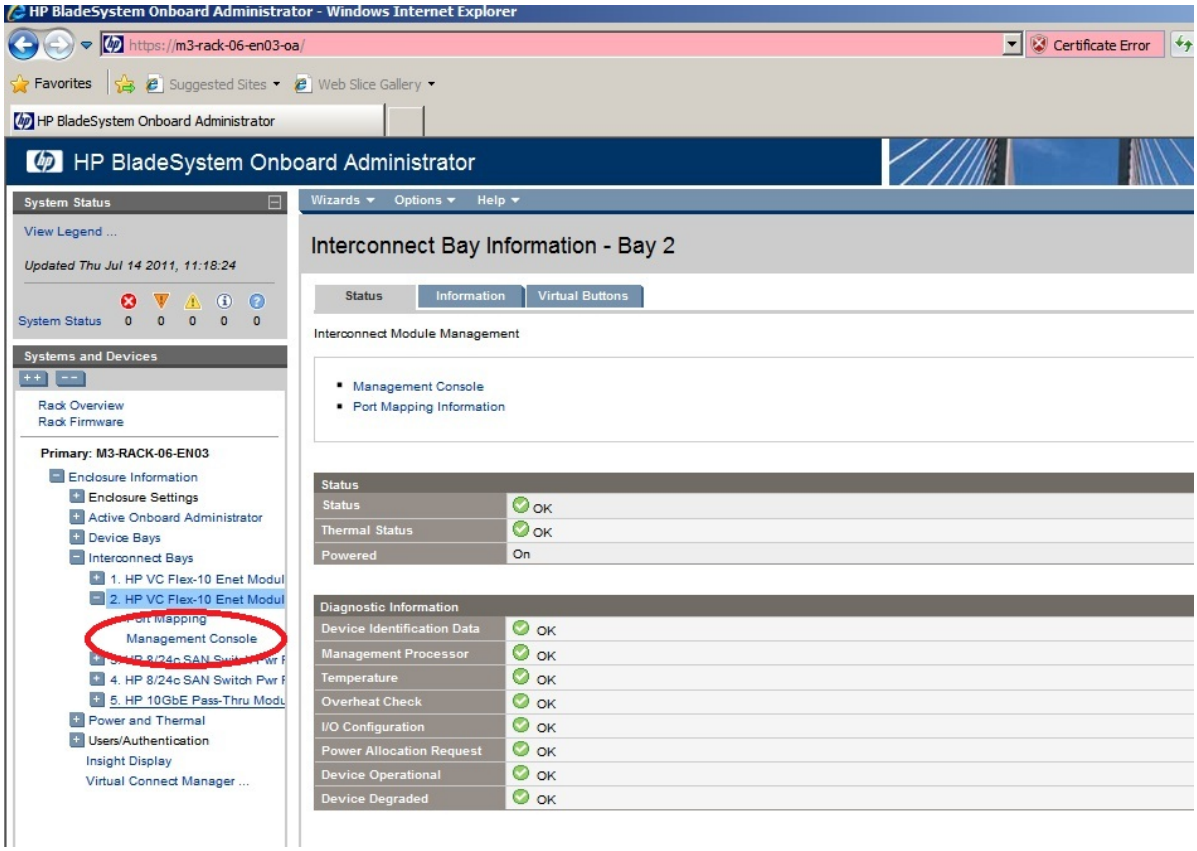
1. Log into the rack where the monitored servers are installed.
2. Click **Interconnect Bays**.



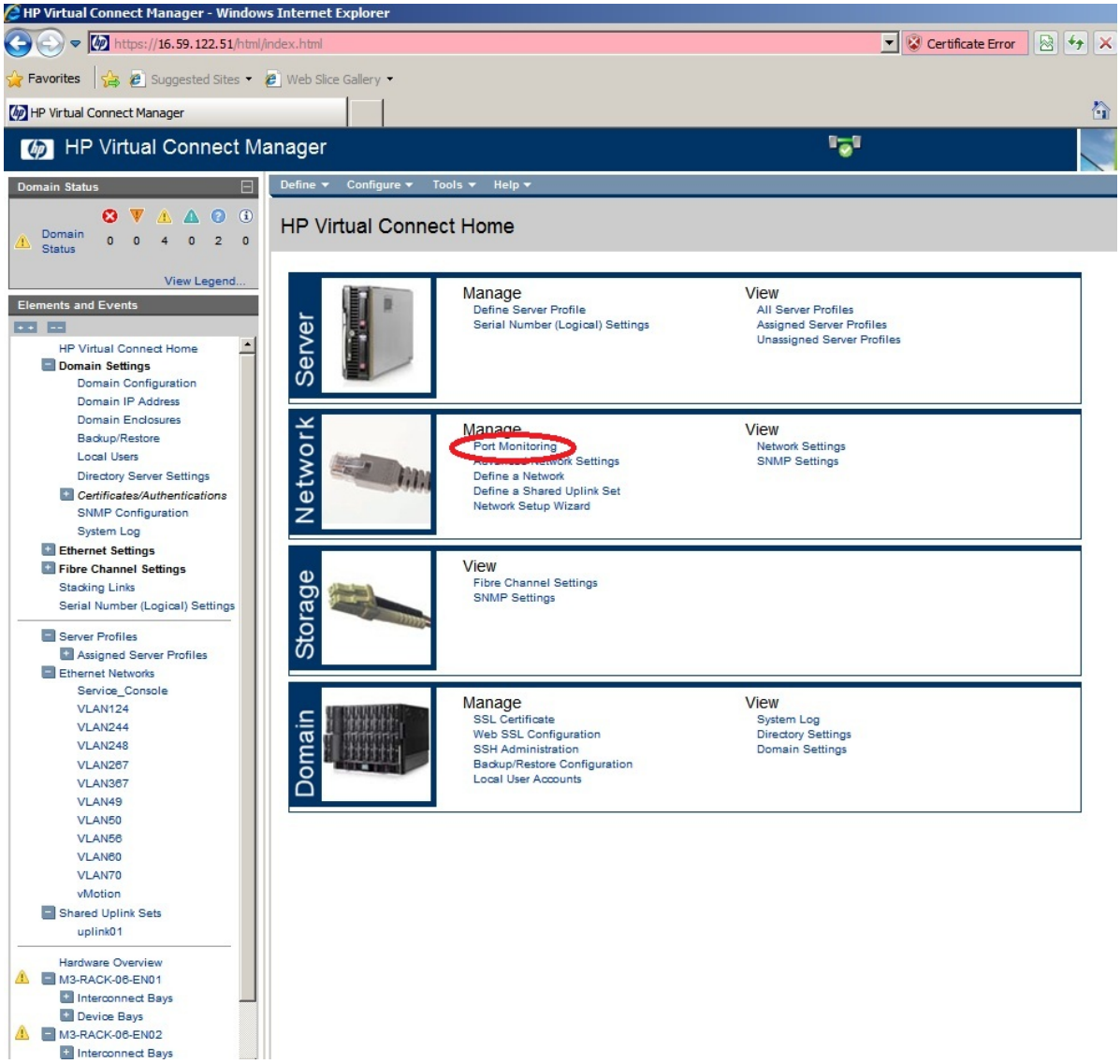
3. Select the device in the bay to which you are connected (in this example: enc03-bay02-port-X1).



4. Click **Management Console** and log in to the flex-10 device.

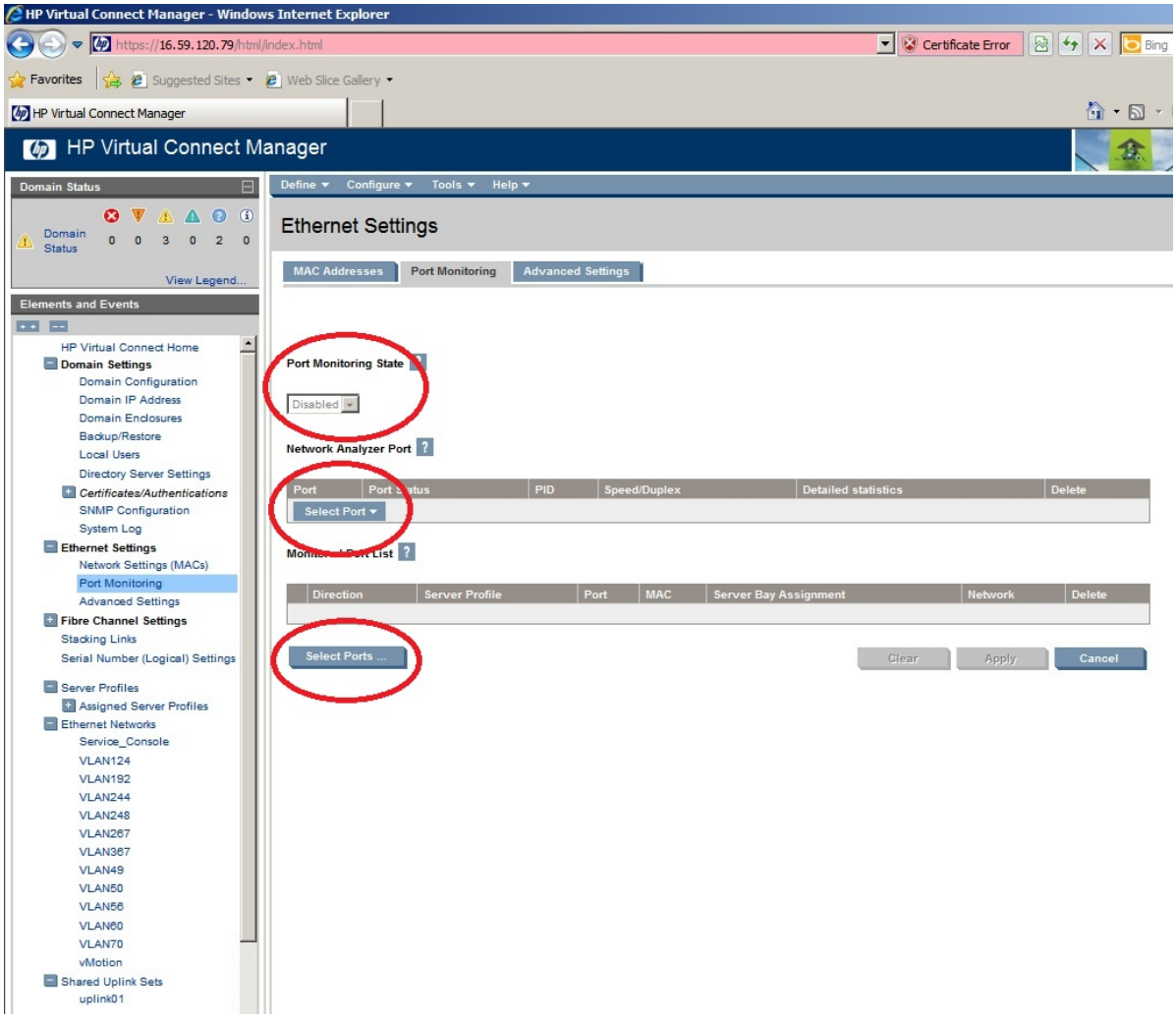


5. In the main page under **Network > Manage**, click **Port Monitoring**.



6. Under **Port Managing State**, select **Enable**.

7. Under **Network Analyzer Port**, select **Add Port**.



SSL Decryption of Monitored Traffic

For security reasons, applications may encrypt their traffic. Decryption is generally done in the network transport layer via the SSL and TLS encryption protocols.

Sniffer applications (like the RUM Probe) which capture the encrypted traffic, can decrypt the traffic only if:

- It is based on RSA algorithms (http://en.wikipedia.org/wiki/RSA_(algorithm))
- Private keys are uploaded to the RUM Probe
- There was no packet loss on the monitored SSL traffic. (Since SSL packet decryption relies on previous SSL packet information, the slightest packet loss may prevent decrypting the entire session.)

The following sections describe the steps to configure the RUM Sniffer Probe to extract SSL communication. The RUM Client Monitor Probe collects the data on the client side before it gets encrypted and therefore does not require this mechanism. (The client to RUM Client Monitor Probe should always use HTTPS, but for this channel the required keys are already installed. See the RUM Hardening Guide for details.)

This section includes the following topics:

- "Private Keys" below
- "HSM" below
- "Common Problems/Limitations With SSL" on the next page

Private Keys

Private keys are one of the most effective ways to secure data in an organization. They are used to encrypt the traffic sent to the end user. Anyone who has the private key can access this data. Unauthorized access to this data could cause a lot of damage, so private keys are well protected. Customers should not send private keys by mail or in any unsecured way.

The owner of the private keys is the organization's security administrator. The security administrator's responsibility is to upload the keys to RUM. It is very important to contact the security administrator during the early phase of deployment. RUM cannot decrypt the traffic without the keys.

The keys are generally protected by passwords. You need the relevant passwords in order to upload the keys.

RUM stores the keys in a secure way on the RUM Probe. For further information, refer to the RUM Hardening Guide.

HSM

Some customers do not store private keys in software keystores and use a hardware solution instead.

There are several solutions to store and manage SSL keys:

- Hardware cards (such as nCipher) store the private key on a secured hardware card on the server (sometimes for security concerns and sometimes in order to save CPU time).
- Central HSM (such as nCipher netHSM) performs the SSL decryption for all the web-servers on a remote machine.
- SSL Offloaders, also known as SSL decryptors (such as http://www.f5.com/glossary/ssl-offloading.html), decrypt the traffic before passing it to the web servers.
- There are also customers who use special appliances for decrypting SSL traffic for probes such as monitoring or IDS systems.

nCipher /netHSM

Customers who use these solutions usually have security administrators who know how to configure web servers to use these cards.

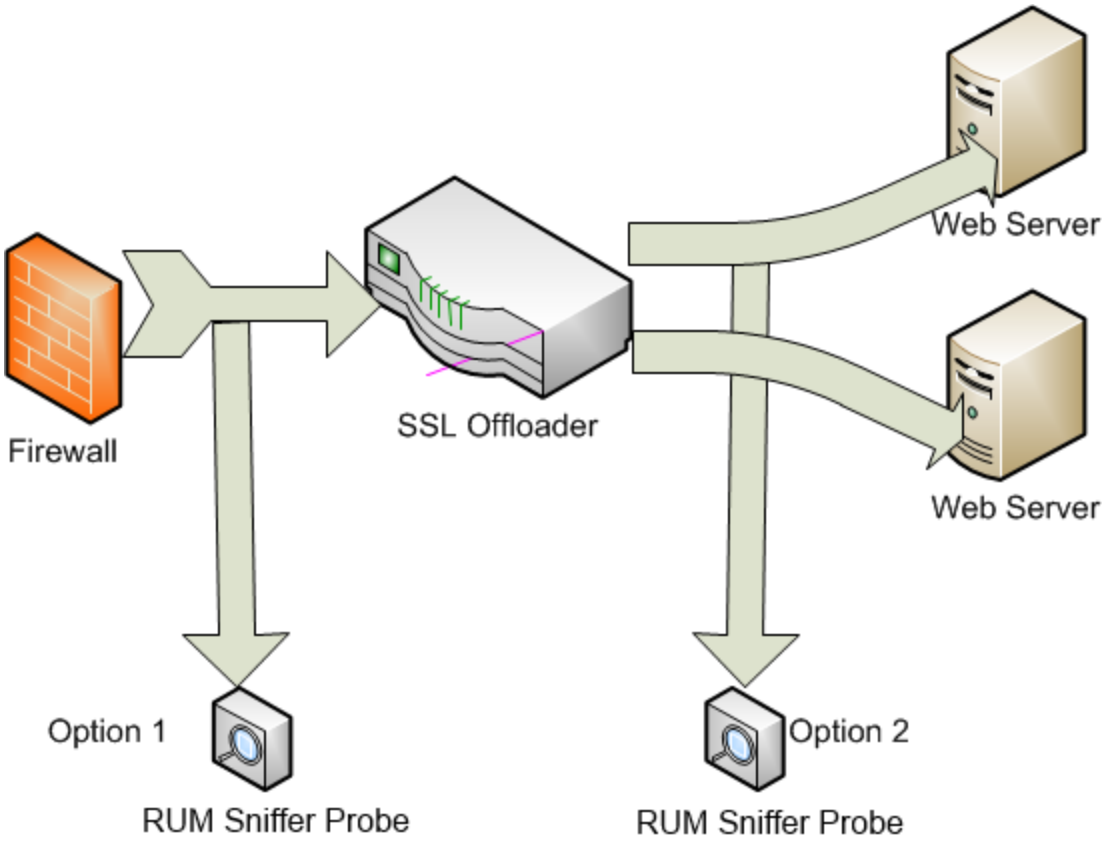
The RUM Probe uses the openSSL library, which supports communication with several types of hardware cards. The RUM Probe uses the hardware card in the same way as the web server. For further details, refer to the RUM Administration Guide.

SSL Offloader

When a customer uses an SSL Offloader, the traffic before the SSL Offloader is SSL, but the traffic between the SSL Offloader and web servers is regular HTTP.

There are several options for deployment in such cases:

- RUM monitors the traffic before the SSL Offloader
- RUM monitors the traffic after the SSL Offloader
- RUM monitors the traffic both before and after the SSL Offloader



There are pros and cons to each method which are the same as for load balancers, but there are also some issues specific to SSL:

- **Deployment** – If the RUM Probe is deployed between the SSL Offloader and web servers, there is no need for the private keys/security administrator in order to deploy RUM.
- **Correctness** – When monitoring SSL, there should be no packet loss in the mirror port. Each loss is crucial in SSL, so if RUM monitors the SSL traffic it is recommended to install a network tap.
- **Completeness** – If the RUM Probe is deployed between the SSL Offloader and web servers, the HTTPS metrics will be missing.

Common Problems/Limitations With SSL

The following are common problems and limitations with SSL.

Technical Limitations

When monitoring SSL traffic, there are several technical limitations:

- **Packet loss in Mirror Port** - When duplicating the network traffic to the probe there may be a drop in Mirror Ports. In the case of a single packet loss in the TCP connection, the RUM Probe will not be able to decrypt the entire TCP connection in which the packet loss occurred. The problem may be worst for some network devices or using a connection pool and all the SSL traffic is sent in a few TCP connections.

  It is recommended to require a network tap while monitoring SSL traffic (see "Network Taps" on page 26) in order to avoid a drop in Mirror Ports, although we have encountered environments using Mirror Ports without any packet loss
- **Encryption Method Diffie–Hellman** - RUM is a sniffer based monitor. Technically, sniffer based solutions can decrypt only ciphers which use RSA algorithms for key exchange/agreement. Therefore, RUM does not support traffic encrypted using Diffie-Hellman. In order to decrypt all traffic (from all

browsers), you need to limit the server to use only the supported cipher suites (based on RSA key exchange/agreement).

The problem is indicated per probe in a RUM web-console.

To view the problem, click **Configuration > Probe Management** and click the **SSL** button. The table displays a column for **Decryption Failed (unsupported algorithms)**

How to resolve this issue:

- There is an option to configure the web-servers to accept only RSA based algorithms. (The security administrator is needed for this.)
- Place the RUM Probe after the load balancer if possible.
- Ignore browsers which do not use RSA algorithms.

Common Problems

Obtaining the correct private keys is a problem in SSL monitoring.

The most common issues are:

- The web-server certificate is supplied instead of the web-server private keys.
- The wrong private keys are supplied.
- Private keys are replaced on the web-servers and the keys are not uploaded to RUM.

Monitor Servers or End Users

When deploying RUM, one of the important considerations is where to deploy the RUM Sniffer Probe.

If the RUM Sniffer Probe listens to traffic before the load balancer, it will monitor the end user IP addresses but will not monitor the server IP addresses. If the RUM Sniffer Probe listens to traffic after the load balancer, it will monitor only server IP addresses, but will not monitor the traffic of the end users.

The recommended deployment of RUM for multi-tiered applications is to monitor all the tiers. This concept was introduced in version 9.0. However, monitoring all tiers requires more resources.

If there are capacity problems and there is no option to monitor all tiers, the recommended deployment is to listen to traffic between the load balancer and web servers. There is an option to configure the load balancer to add the client IP in HTTP headers. The usual header is called **X-forwarded-For** (http://en.wikipedia.org/wiki/X-Forwarded-For) and is supported by default. There is also an option to configure another header name in a parameter called **forwarded_for_header** in the global section in the **Beatbox_Default_Const_Configuration.xml** file.

When **X-forwarded-for** is configured, the RUM Sniffer Probe reports both client and Server IPs for each page. The only missing data is the network time of the end user (latency, connect time,SSL time, etc.).

Send Documentation Feedback

If you have comments about this document, you can contact the documentation team by email. If an email client is configured on this system, click the link above and an email window opens with the following information in the subject line:

Feedback on Real User Monitor Deployment Planning Guide (Real User Monitor 9.40)

Just add your feedback to the email and click send.

If no email client is available, copy the information above to a new message in a web mail client, and send your feedback to docteam@hpe.com.

We appreciate your feedback!