



everRun[®]

SNMP Guide

**Release 6.0
October 2010**

MARATHON

The Application Availability Experts™

NOTICE

Marathon Technologies Corporation reserves the right to make improvements to this document and the product it describes at any time and without further notice.

COPYRIGHT

© Marathon Technologies Corporation 2008, 2009-2010. All rights reserved.

This document is copyrighted, and all rights are reserved. No part of the document or the products it describes may be reproduced by any means or in any form, without prior consent in writing from Marathon Technologies Corporation. Printed in the U.S.A.

Marathon everRun products are protected by one or more of the following patents:

U.S. Patent Numbers: 5,600,784; 5,615,403; 5,787,485; 5,790,397; 5,896,523; 5,956,474; 5,983,371; 6,038,685; 6,205,565; 6,279,119; 6,473,869; 6,728,898, 7,373,545.

European Patent Numbers: EP0731945; EP0974912; EP0986784; EP0993633; EP1000397; EP1000404; EP1029267; EP1496434; GB2392536; Japanese Patent Numbers: 3679412; 4166939; 4264136; 4472995. Other patents pending.

Marathon Assured Availability, ComputeThru, SplitSite, everRun, and the Marathon logos are registered trademarks or trademarks of Marathon Technologies Corporation. Copyright © 2008, 2009-2010 Marathon Technologies Corporation. All rights reserved.

SOFTWARE COPYRIGHT NOTICE

The software described in this document is covered by the following copyright:

© Marathon Technologies Corporation. 2008, 2009-2010.

GNU PUBLIC LICENSE AND LESSER GNU PUBLIC LICENSE

The product makes use of software covered under the GNU Public License (GPL) and the Lesser GNU Public License (LGPL). For a written copy of these licenses, see `/opt/everRun/current_everRun/licenses` on any XenServer host on which everRun VM is installed. Marathon will make source components available upon request, via email to support@marathontechnologies.com, as required under the terms of the GPL.

TRADEMARK NOTICE

Microsoft® and Windows® are registered trademarks of Microsoft Corporation in the United States and/or other countries; Xen® and Citrix® are registered trademarks and XenServer™ and XenCenter™ are trademarks of Citrix Systems, Inc.; Java® is a registered trademark of Sun Microsystems; Linux® is a registered trademark of Linus Torvalds. Intel® is a registered trademark of Intel Corporation.

All other brands and product names are trademarks of their respective companies or organizations.

SOFTWARE REVISION

The revision of the software that this document supports is Release 6.0.

Marathon Technologies Corporation
295 Foster Street, Littleton, MA 01460
(978) 489.1100 or (888) 682.1142
www.marathontechnologies.com

Contents



Chapter 1 — Introduction to everRun SNMP

Overview	1
Functionality	1
Operation	2
The everRun SNMP Extension Agent	2
The everRun SNMP MIB	3
Activating SNMP	3
For More Information	3
Documents	3
Standards	3
Contents of this Guide	4

Chapter 2 — Setup and Configuration

Activating and Configuring SNMP	5
Installing the Windows SNMP Service	7
Enabling the everRun SNMP Agent	11
Disabling the everRun SNMP Agent	11
Configuring SNMP to Send Traps to Remote Clients	12
Enabling SNMP on Remote Clients	14
Copying the everRun MIB File to a Remote Client	14
Locating the SNMP Files	14
Copying the SNMP MIB File to the Remote Client	15
Enabling the SNMP Service on a XenServer Host	15
Using Third-Party Software to Monitor Host Connectivity	16

Chapter 3 — Management Information Base — MIBv1

Overview	17
Layout of the everRun MIB Space	18
MIB Naming Tree	18
Marathon MIB Subtree	19
everRun MIBv1 Subtree	20
Manageable Objects	21
Component States	23
everRun PVM Parameters	25
everRun PVM Operating System Parameters	26
Mirrored Disk Parameters	27
Network Parameters	30
everRun PVM Hosts Parameters	30
NICs Parameters	31
Disks Parameters	32
Link Adapter Parameters	32
Compute Instance Parameters	33
XenServer Host Information	34
everRun IP Isolation Address Information	35
everRun PVM Availability Link Parameters	35
A-Link Path Parameters	36
everRun PVM Quorum Service Parameters	36
Preferred and Alternate Quorum Service Parameters	38
For More Information on everRun Traps	38

Chapter 4 — everRun SNMP Extension Agent Traps

About the everRun SNMP Extension Agent	39
About everRun Traps (Event Notifications)	39
Variables Common to All Traps	45

Chapter 5 — Windows Event Log Entries

Summary of Windows Event Log Messages	49
---	----

SNMP Glossary

SNMP Glossary	55
---------------------	----

Introduction to everRun SNMP

1



This chapter describes the main components of the everRun Simple Network Management Protocol (SNMP) implementation and provides a brief overview of system functionality.

Overview

everRun SNMP enables you to monitor the health of each everRun protected virtual machine (PVM) and to receive notice of state changes within a PVM.

This chapter provides a brief overview of everRun SNMP. For more detail on Windows® SNMP and MIB design, see the documents listed in “For More Information” on page 3.

Functionality

In the everRun SNMP implementation, an everRun SNMP extension agent and an SNMP-compliant MIB (Management Information Base) work together to provide the following functionality:

- The everRun SNMP extension agent generates SNMP traps (system alerts — also known as event notifications) and reports them to remote clients. The agent runs as a Windows Server 2003 or 2008 SNMP extension agent on the PVM.
- The MIB and agent provide read-only access to the following data:
 - Software revision information
 - PVM configuration information and operating system state
 - Per-host PVM configuration information and state
 - I/O device configuration information and states
 - Disk mirror copy statistics

- Availability link (A-link) states
- Quorum service configuration information and state
- Pool isolation IP address information and state, as well as the connectivity of the pool master host

The everRun SNMP agent generates traps for significant state changes of the PVM and the PVM operating system on each host, as well as for the state changes of the A-links, the preferred and alternate quorum servers, the I/O devices, pool isolation IP address and the connectivity of the XenServer pool master host. Typically, traps indicate the current state, the state prior to the current state, and the time and date when the state change occurred.

When monitoring the everRun PVM remotely, the system administrator uses a MIB browser or other third-party application to read MIB data and to receive traps from the everRun SNMP extension agent.

Operation

Figure 1 shows a simple overview of everRun SNMP operation.

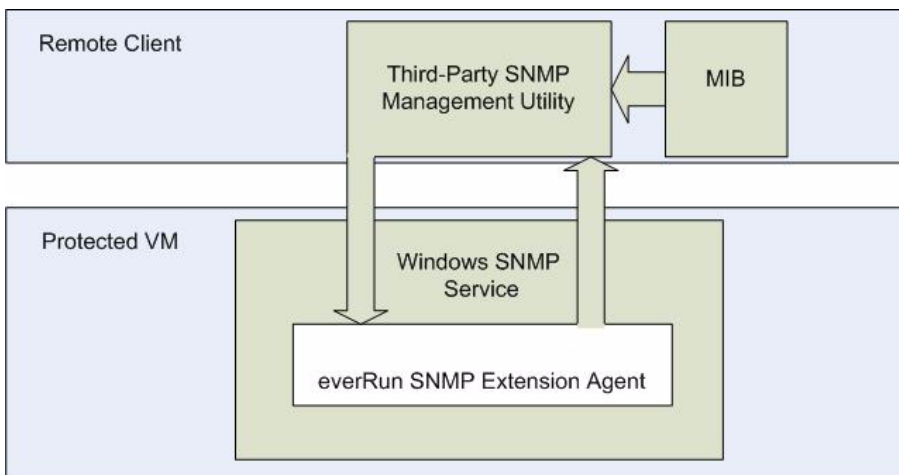


Figure 1 The everRun SNMP extension agent employs the Windows SNMP service to provide PVM event notification, delivered to a third-party SNMP management application running on remote clients.

The everRun SNMP Extension Agent

The everRun SNMP extension agent is a user-mode DLL (dynamic-link library) that runs on an everRun PVM. As shown in the figure, the everRun agent operates in the context of the Windows SNMP service.

The agent accesses everRun management data, then exports the data — through the Windows SNMP API — to remote clients. Each remote client must be running an SNMP management application that formats the management data for viewing. A number of suitable SNMP utilities — based on Windows, Linux, or other operating systems — are available.

When system transitions occur, the everRun agent sends alerts (traps) up to the remote client. The remote application can also use SNMP GET operations to request information from the everRun agent.

The everRun SNMP MIB

The agent uses the everRun SNMP MIB to describe the exported data and the SNMP traps generated. The MIB contains the definitions for both the manageable data and the SNMP traps generated by the agent during everRun server state transitions.

The everRun SNMP extension agent supports SNMP protocol version 1 (SNMPv1), which is backward-compatible with all subsequent SNMP versions.

Activating SNMP

SNMP functionality is integrated with the everRun software; however, you must configure and enable SNMP on each protected VM you want to monitor, on any client computer(s) where you want to receive SNMP alerts, and on the XenServer hosts in your pool. Chapter 2, “Setup and Configuration,” contains an overview table of these tasks, with links to specific sections that describe the configuration details.

For More Information

To learn more about Windows SNMP and MIB design, consult the following resources.

Documents

Two documents were used for the design of the everRun MIB and the SNMP extension agent. Both provide useful information regarding SNMP and its implementation on Windows.

- Murray, James D. *Windows NT SNMP* (O'Reilly Media, January 1998).
- Perkins, David T., and Evan McGinnis. *Understanding SNMP MIBs* (Prentice Hall, December 1996).

Standards

The following standards describe the Simple Network Management Protocol:

- *RFC1155, Structure and Identification of Management Information for TCP/IP-*

based Internets (May 1990).

- *RFC1157, A Simple Network Management Protocol (SNMP)* (May 1990).
- *RFC1902, Structure of Management Information for Version 2 of the Simple Network Management Protocol (SNMPv2)* (January 1996).
- *RFC1905, Protocol Operation for Version 2 of the Simple Network Management Protocol (SNMPv2)* (January 1996).

Contents of this Guide

The remaining chapters of this guide contain the following information:

- Chapter 2, Setup and Configuration — Covers all aspects of the everRun SNMP activation and configuration: installing the Windows SNMP service, enabling everRun SNMP, enabling SNMP on the XenServer hosts in your pool, and configuring one or more remote management servers to receive SNMP data.
- Chapter 3, Management Information Base (MIBv1) — Describes the layout of the everRun MIB tree and provides detailed lists of the manageable data objects.
- Chapter 4, SNMP Agent and Event Notifications — Describes the operation of the everRun SNMP extension agent and summarizes the traps (event notifications).
- Chapter 5, everRun Event Log Entries — Summarizes the messages that the everRun SNMP agent writes to the Windows event log.
- SNMP Glossary — Defines some basic SNMP terms.

The everRun SNMP extension agent allows monitoring on a per-PVM basis. This chapter contains instructions for configuring the software required to implement the everRun SNMP reporting service.

Activating and Configuring SNMP

Three software applications are required for the everRun SNMP implementation:

- The **Windows SNMP service** should be installed on each PVM that you want to monitor.
- The **everRun extension agent software** must be enabled on each PVM you want to monitor.
- A **third-party SNMP management utility of your choice** should be installed on each remote client that will receive SNMB trap information from everRun PVMs.

NOTE: The SNMP management utility may run on Windows or Linux operating systems. Marathon does not recommend any specific software for this role. However, the Marathon Customer Support Knowledge Base includes a white paper that illustrates one available SNMP management utility that has been tested with everRun SNMP. For information about this utility, go to the Marathon customer portal, click the **Knowledge** button at the top of the interface, click **Knowledge Topics** at left, enable the **Title Only** button, and type **SNMP Tool** in the Search field.

In addition, several configuration tasks are required to complete the implementation:

- In order to obtain information from the everRun PVMs, you must copy the **everRun MIB file** to each remote client, then load it into the third-party management utility.

- In order to **monitor connectivity of your XenServer hosts** using SNMP management tools, you must enable and start SNMP daemon on each XenServer host that contains everRun-protected VMs and then configure your third-party management utility to receive the alerts.

The following checklist summarizes all configuration tasks needed for everRun SNMP reporting. The list is organized according to location — what to do on the PVM, on the XenServer host, and on the remote client. To learn about a task, locate it in the Task column, then follow the link to the appropriate instructions under Where to Find Information.

Location: Each protected VM you want to monitor			
✓	Task	Purpose	Where to Learn More
	Install and enable the Windows SNMP service.	To support operation of everRun SNMP on the protected VM.	“Installing the Windows SNMP Service” on page 7.
	Enable everRun SNMP.	To activate the everRun SNMP extension agent.	“Enabling the everRun SNMP Agent” on page 11.
	Configure SNMP to send traps to remote client(s).	To specify remote client(s) that will receive trap information.	“Configuring SNMP to Send Traps to Remote Clients” on page 12.
	Copy the MIB file from the Marathon install directory to a known location on each remote client that is running a third-party management utility.	To load a copy of the MIB file into the third-party management utility on each remote client.	“Copying the everRun MIB File to a Remote Client” on page 14. For information on loading the MIB file onto a client computer, see “Install and configure third-party management utility” in the next section of this checklist.

Location: Each remote client where you want to receive SNMP traps (alarms or event notification)			
✓	Task	Purpose	Where to Learn More
	Install and enable the Windows SNMP service.	To support operation of everRun SNMP on the remote client.	“Installing the Windows SNMP Service” on page 7.
	Install and configure a third-party management utility. During installation, load everRun MIB file when requested.	To set up management service to receive and interpret everRun SNMP trap reports and to associate SNMP services with the everRun MIB.	Consult the documentation for your third-party management utility.
	Optionally, configure the management utility to send email notices.	To provide email notification of SNMP events.	Consult the documentation for your third-party management utility.
	Configure the utility to monitor the state of XenServer hosts.	To provide SNMP notification of host state changes.	Consult the documentation for your third-party management utility.

Location: Each XenServer Host you want to monitor			
✓	Task	Purpose	Where to Learn More
	Enable the Linux-based SNMP daemon on each host	To monitor health of a XenServer host using SNMP management tools.	See “Enabling an SNMP Service on a XenServer Host” on page 15.

Installing the Windows SNMP Service

Before enabling the everRun SNMP extension agent, you must first prepare each protected VM by installing the Windows SNMP service on that PVM.

The Windows SNMP service is a Windows application that provides an underlying API for the everRun SNMP messaging system. Configuration of the Windows SNMP and SNMP trap services is slightly different for Windows Server 2003 and Windows Server 2008 platforms, as described below.

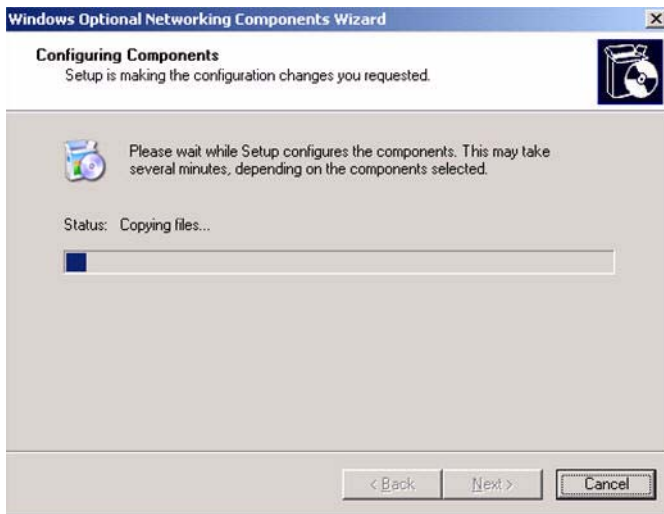
To install the SNMP service on a PVM running Windows Server 2003:

1. To open the Windows Network Connections application, from a PVM's classic Start menu choose **Start > Settings > Network Connections**.
2. From the menu bar, select **Advanced > Optional Networking Components**.

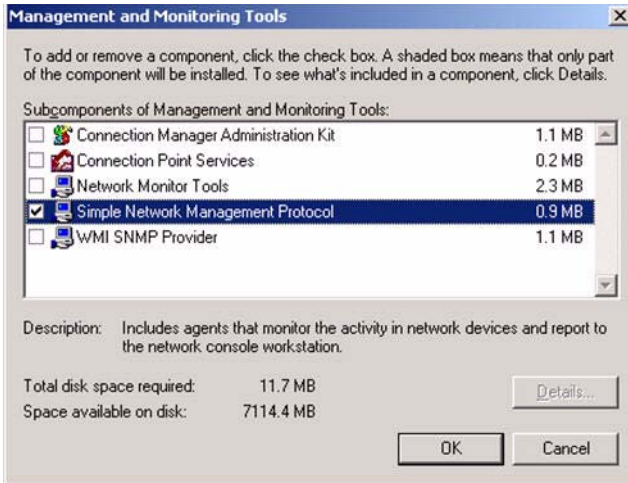
NOTE: If your PVM uses the newer Windows Start menu, choose **Start > Control Panel > Network and Internet Connections > Network Connections**. Open the **Advanced** menu and choose **Optional Networking Components**. Now you're ready for step 3 below.

3. From the Components list in the Windows Optional Networking Components wizard, select **Management and Monitoring Tools**, then click **Details**.

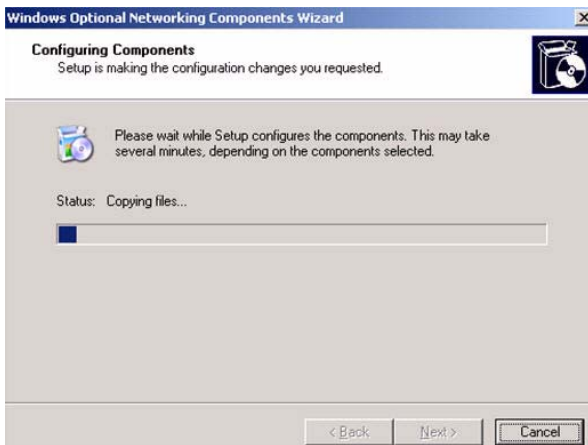
The Configuring Components dialog box appears.



4. Click **Next** to continue. The Managing and Monitoring Tools dialog box appears.



5. Select **Simple Network Management Protocol** and click **OK**, then click **Next** to open the Windows Optional Networking Components wizard.

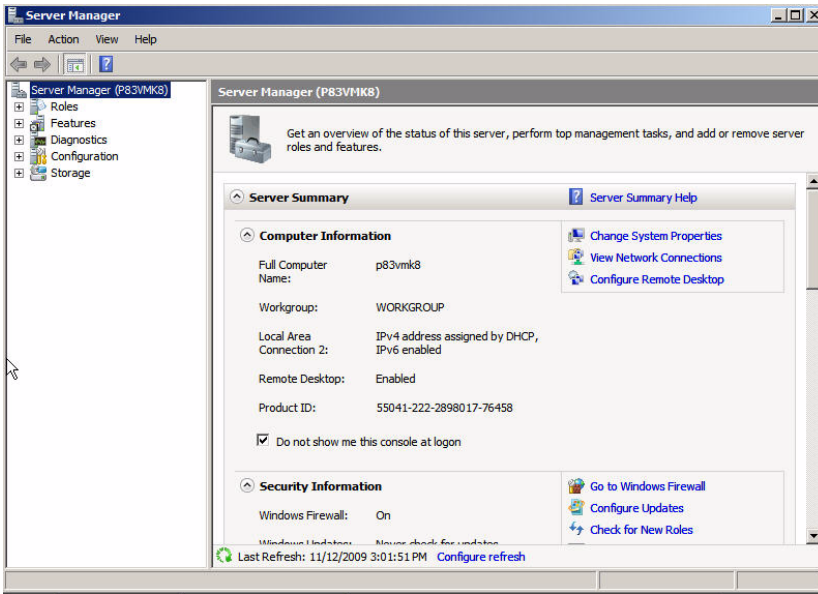


6. In most cases, you will be asked for a Windows installation kit. When asked, locate the self-extracting archive file **SNMPZP . EXE** on the Windows CD-ROM, move it to an empty directory on the protected VM, and execute it.

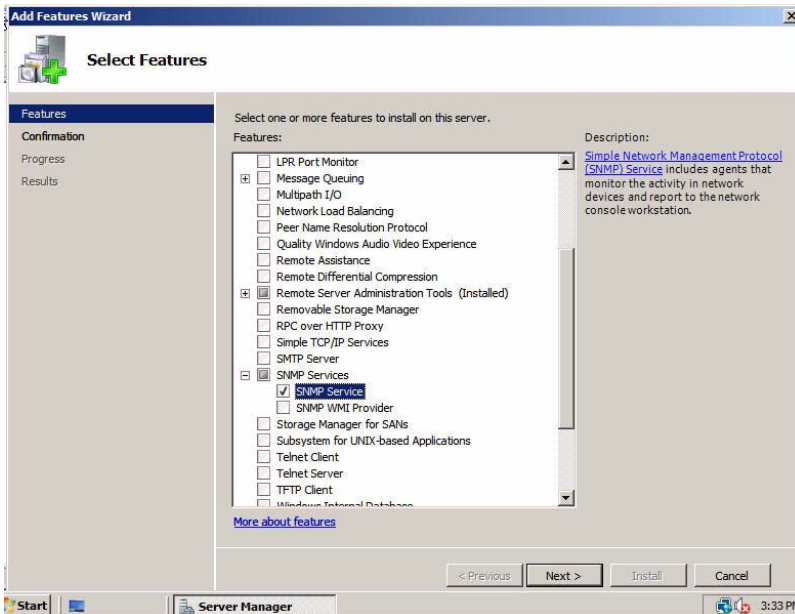
When the Windows SNMP service installation completes, both the SNMP service and the SNMP trap service are configured and started on the PVM.

To install the SNMP service on a PVM running Windows Server 2008:

1. On a PVM, open the Server Manager window: **Start > Server Manager**.



2. From the Server Manager, choose **Features > Add Features**, then click the plus mark to open the **SNMP Services** menu, and click to the left of **SNMP Service**.



3. To complete the installation, click **Next**, then click **Install**.

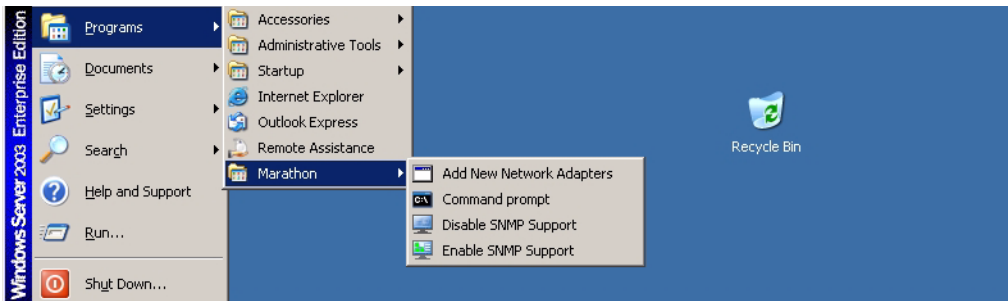
When the Windows SNMP service installation completes, both the SNMP service and the SNMP trap service are configured and started on the PVM.

Enabling the everRun SNMP Agent

You must also enable the everRun SNMP agent on each protected VM where you want the service to run. Marathon menu options installed in the PVM during SNMP setup provide the capability to enable or disable the SNMP. You simply click the Enable option to turn on SNMP reporting.

To enable the everRun Agent:

1. Choose **Start > Programs > Marathon > Enable SNMP support**.



2. When asked if you want to enable Marathon support for SNMP, click **Yes** to continue.
3. If the Windows SNMP service is running, you are asked to confirm that it must be restarted in order to enable the everRun SNMP agent, click **Yes** to continue.

Disabling the everRun SNMP Agent

Marathon menu options installed in the PVM during SNMP setup provide the capability to enable or disable the SNMP. You simply click the Disable option to turn off SNMP reporting.

To disable the everRun Agent:

1. If the everRun agent is installed at the default location, it can be disabled using the following command sequence:

Start > Programs > Marathon > Disable SNMP support

2. When asked if you want to disable Marathon support for SNM, click **Yes** to continue.
3. If the Windows SNMP service is running, you are asked to confirm that the Windows service must be restarted in order to disable the everRun SNMP agent, click **Yes** to continue.

Configuring SNMP to Send Traps to Remote Clients

This procedure assumes that you have installed the required third-party SNMP utility on each remote client you designate to receive SNMP traps.

NOTE: If any remote client is running an operating system other than Windows, a compatible SNMP utility must be installed to receive SNMP traps. In that case, refer to the appropriate operating system documentation for information about how to enable the receipt of SNMP traps. A note on the opening page of this chapter provides information about one Windows utility that has been tested with everRun SNMP.

At any time after the SNMP files are installed on a PVM, you can configure SNMP to send traps from the PVM to remote management systems that are running a third-party SNMP utility. The remote clients can be running Windows or any other operating system that supports the SNMP protocol. To enable the receipt of the traps, the SNMP trap service must also be running on each remote client that contains the third-party SNMP management utility.

This configuration has two parts:

- From the Control Panel of the everRun PVM you want to monitor, configure the remote trap destinations you want to use.
- On each remote client, enable the Windows SNMP trap service.

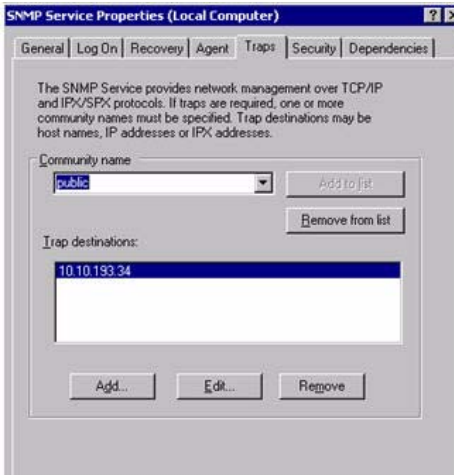
To configure trap destinations from the PVM:

1. From the Control Panel of the protected VM, select **Administrative Tools > Computer Management**.

Alternatively, you can right-click **My Computer**, typically on the PVM Desktop, and then select **Manage** to access the **Computer Management** window.

2. Expand the **Services and Applications** folder.
3. Select **Services**, then scroll down and double-click **SNMP Service**.
The SNMP Service Properties dialog box displays.
4. Click the **Traps** tab.

This tab provides a field where you can type the host name or IP address of remote clients you designate to receive trap reports. You can specify as many remote clients as you need.



5. Use the Add, Edit, and Remove buttons as needed to create a list of all remote clients you require.
6. When the list is complete, click the **Security** tab, where you will set security to accept public community names.



7. In the Security tab:
 - Make sure the Community name field displays the setting **public** (using all lowercase). If not, click **Add**, type **public**, and click **Add** again.
 - Click to enable the **Accept SNMP packets from any host** radio button.
8. Click **OK** to finish supplying the traps and their security settings.

Enabling SNMP on Remote Clients

You must also enable the Windows SNMP service on each client you designate to receive SNMP traps. Using the Windows documentation for your operating system, check to see if the Simple Network Management Protocol Service (SNMP) service is running. Depending on the operating system of your client computer, you can access the SNMP service via the Control Panel > Add/Remove Programs > Add/Remove Windows Components in the left panel, or Control Panel > Programs and Features in the left panel.

To enable SNMP trap service on each remote client:

1. To start the service:
 - Select **Services**, then double-click **SNMP Trap Service**.
 - In the SNMP Service Properties (Local Computer) dialog box, click the **Start** button under Service Status to start the SNMP trap service.
2. Adjust the settings so that the startup of the trap service is automatic.

Refer to Microsoft's SNMP documentation if you need specific information for your operating system.

Copying the everRun MIB File to a Remote Client

The final step in setting up everRun SNMP is to provide a copy of the everRun MIB file on each remote client that is running the third-party management software. The third-party software needs this MIB file to identify everRun traps.

Locating the SNMP MIB File

In a default installation, all everRun SNMP files are installed on the PVM under the Marathon install directory. The MIB file, **MarathonEverRunPVM.MIB**, is an SMIV1 file that defines all manageable objects and SNMP traps. On a 32-bit Windows server the location is **\Program files\Marathon**; on a 64-bit Windows server the location is **\Program files (x86)\Marathon**.

Copying the SNMP MIB File to the Remote Client

You can simply copy and paste the everRun MIB file from the PVM you are monitoring to each remote client where you want to receive reports.

To copy the MIB file to a remote client:

1. Browse to the Marathon install directory on your PVM, locate the file **MarathonEverRunPVM.MIB**, and make a copy of the file.
2. Paste a copy of the MIB file at a known location on the client that contains your third-party management software.

Later, when you install the third-party management software on your remote client, you will be asked to specify and load this MIB file.

Enabling an SNMP Service on a XenServer Host

In order to **monitor connectivity of your XenServer hosts** using SNMP management tools, you can enable and start an SNMP daemon on each XenServer host that contains everRun-protected VMs and configure your third-party management utility to receive alerts from those hosts.

This section describes how to enable the SNMP service on a XenServer host to monitor each pool host.

To enable the XenServer SNMP service on a host:

A Linux-based SNMP daemon already exists in on each XenServer host; however, it is disabled by default. To enable the SNMP service on the hosts in the pool, connect to each host console and perform the following steps:

1. In the file `/etc/sysconfig/iptables`, locate the line that reads:

```
-A RH-Firewall-1-INPUT -p tcp -m tcp --dport 631 -j ACCEPT
```
2. Add the following line below it, then save the file:

```
-A RH-Firewall-1-INPUT -p udp -m udp --dport 161 -j ACCEPT
```
3. Restart the IPTables service in Dom0 of the XenServer host:

```
service iptables restart
```
4. Configure the SNMP daemon to run on startup:

```
chkconfig snmpd on
```
5. Start up the SNMP daemon:

```
service snmpd start
```

Using Third-Party Software to Monitor Host Connectivity

After you have enabled the SNMP agent in Dom0 on each host, the remote (third-party) network monitoring application you set up to receive SNMP information can perform periodic “pings” to make sure the hosts you have identified are up and running.

NOTE: The Marathon Customer Support Knowledge Base includes a white paper that illustrates one available third-party management utility that has been tested with everRun SNMP. That white paper describes how to identify the hosts you want to monitor using ping keepalive information.

For information about this utility, go to the Marathon customer portal, click the **Knowledge** button at the top of the interface, click **Knowledge Topics** at left, enable the **Title Only** button, and type **SNMP Tool** in the Search field.

Management Information Base — MIBv1

3



The everRun SNMP Management Information Base (MIB) database provides definitions for the everRun server management data used with the everRun SNMP extension agent.

Overview

The everRun SNMP extension agent runs on the Windows server in a designated everRun PVM. The everRun agent gathers PVM management data described by the MIB and exports it to remote management clients via the SNMP protocol. Typically, when managing the everRun PVM remotely, the system administrator uses a MIB browser or other third-party application to read MIB data and to receive traps from the everRun SNMP extension agent.

The everRun MIB is a “read-only” database. The data can be viewed using any MIB browser that supports SMIv1 or SMIv2 and has access to the everRun MIB.

The agent also supports generation of the SNMP traps, as described in Chapter 4 of this document. As long as the Windows operating system is active in the PVM, all traps generated by the everRun agent originate there.

When the Windows server in the everRun PVM goes down, the agent is no longer present to generate traps. In the event of an unexpected loss of the operating system, a third-party SNMP client normally generates appropriate alerts when it loses contact with a server (PVM) that it is monitoring. In this way, the user receives notice of the loss of the Windows PVM server — even when the agent running on it is not available to generate traps.

Layout of the everRun MIB Space

The following sections describe the layout of the Marathon Technologies MIB space, as well as the named objects and traps associated with the everRun product.

Both the format of the MIB space and the naming of the objects and traps are based on descriptions and recommendations found in *Understanding SNMP MIBs*, by David T. Perkins and Evan McGinnis (Prentice Hall, December 1996). This book is a recognized standard for MIB implementation.

MIB Naming Tree

Figure 2 shows the position of the Marathon Technologies vendor-specific domain in the standard MIB naming tree. All information specific to Marathon Technologies and its products appears below the **MarathonTechnologies** entry in the MIB naming tree.

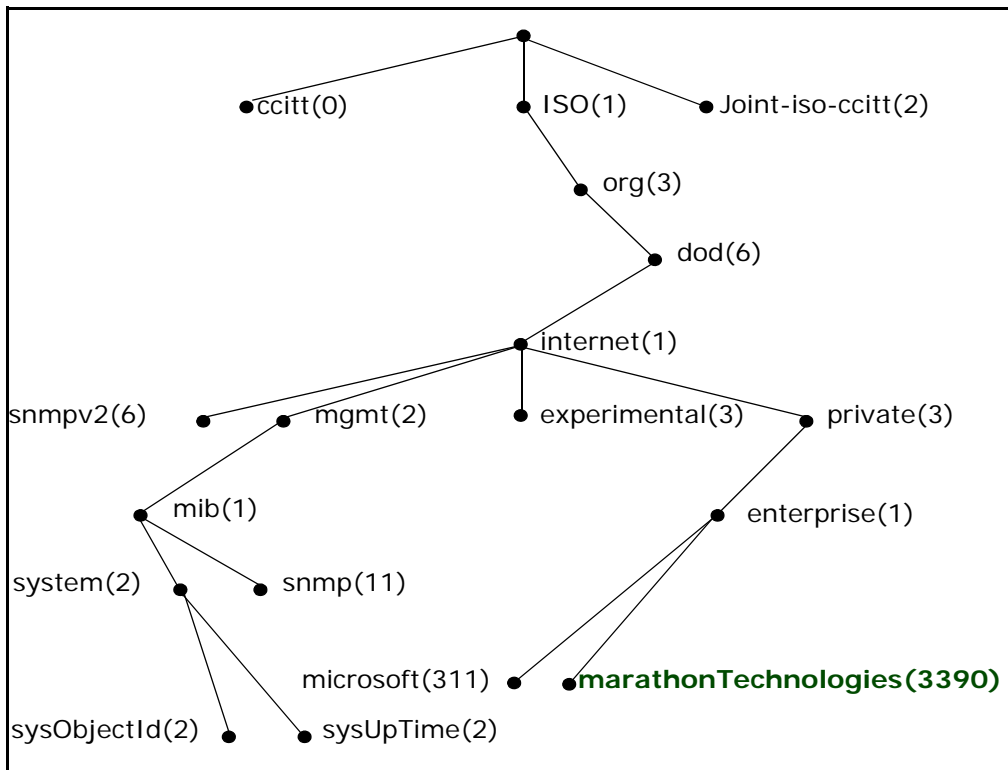


Figure 2 MIB Naming Tree

Marathon MIB Subtree

The Marathon Technologies subtree of the MIB naming tree is further divided into five categories, as shown in [Figure 3](#). Only the **mtcExperimental** and **mtcProducts** categories are used at this time. The others are reserved for future use.

A separate MIB module exists for each entry under **mtcProducts**. A separate MIB module exists for each Marathon product under **mtcExperimentalProducts** or **mtcProducts**. The common nodes shown below are defined in all Marathon MIB modules.

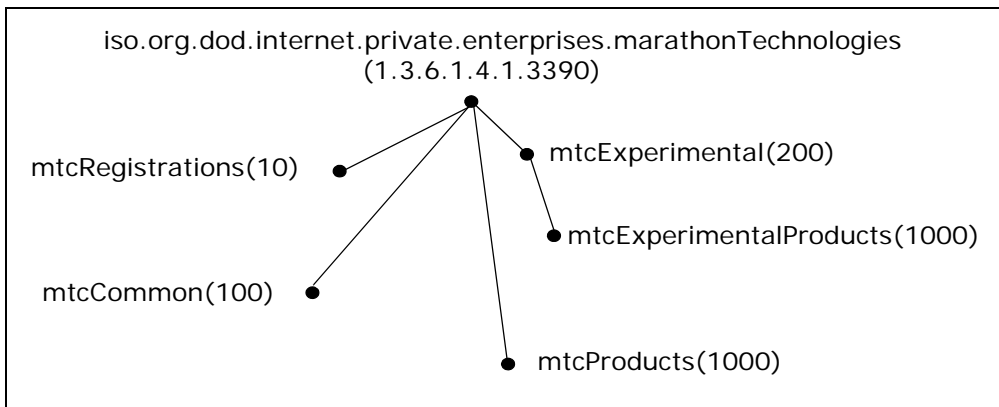


Figure 3 Marathon subtree of the MIB naming tree

Each of these categories is described in [Table 1](#).

Table 1 everRun MIBv1 subtree nodes

Node Name	Child No.	Description
mtcRegistrations	10	Reserved for registration of Marathon-specific modules, products, etc.
mtcCommon	100	Reserved for objects and traps used by multiple Marathon products.
mtcExperimental	200	Used for experimental product releases. The objects and traps associated with the everRun MIBv1 tree are listed in this subtree during in-house and beta testing.

Table 1 everRun MIBv1 subtree nodes

Node Name	Child No.	Description
mtcProducts	1000	Reserved for objects and traps for released and supported MIBs associated with Marathon products. The released everRunMIBv1 tree is located here.
mtcExperimental Products	200.1000	An intermediate node under mtcExperimental , under which experimental products are registered with the intent to move them to the mtcProducts subtree later. Note that the child number of this node matches that of mtcProducts .

everRun MIBv1 Subtree

The position of the MIB for the everRun product is shown in [Figure 4](#), starting with node **mtceverRunPVMMIBv1**. This MIB is implemented in the module **mtceverRunPVM**. All manageable objects and traps associated with the everrun product exist in a subtree under the **mtceverRunPVMMIBv1** node.

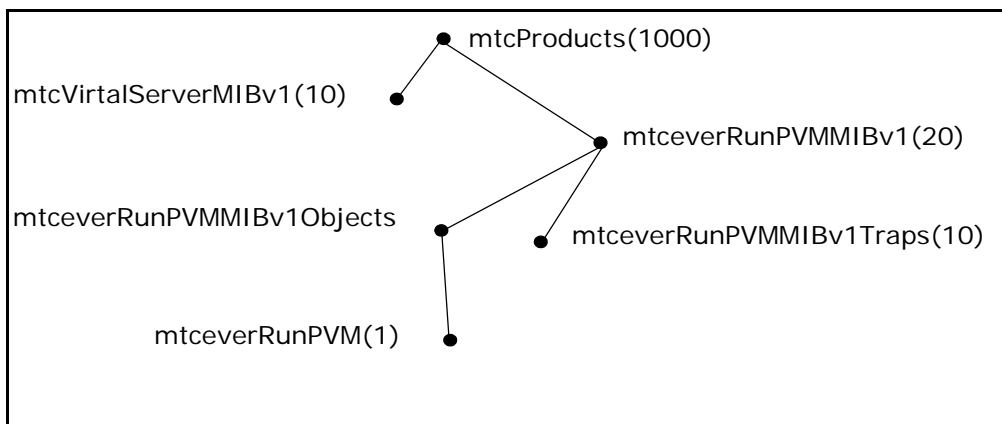


Figure 4 everRun MIBv1 subtree of the MIB naming tree

Each of these nodes is described in [Table 2](#).

Table 2 everRun MIBv1 subtree nodes

Trap Name	Child No.	Event Transitions
mtcVirtualServerMIBv1	10	All Marathon Endurance products exist under this mode. This subtree is not addressed in this document.
mtceverRunPVMMIBv1	20	All everRun PVM products exist under this mode.
mtceverRunMIBv1Objects	20.1	All manageable objects in the everRun PVM product exist under this node.
mtceverRunMIBv1Traps	20.10	All traps in the everRun PVM product exist under this node. In SNMPv1, traps are not assigned object IDs (OIDs).
mtceverRunPVM	20.1.1	This is the top-level node for all everRun PVM manageable objects.

Manageable Objects

The top level object under the **mtceverRunPVMMIBv1Objects** node is **mtceverRunPVM**, which represents the PVM as a whole. Objects under the PVM are divided into four categories:

1. The PVM Operating System object (**mtceverRunPVMOperatingSystem**) — Contains states and parameters for the PVM operating system itself, as well as the states and parameters for all logical devices seen by the PVM O/S.
2. The PVM Hosts object (**mtceverRunPVMHosts**)— Contains the states and parameters of the PVM hosts, with a subnode for each of its two hosts. This object also contains states and parameters for the compute instance on that host (Host1 or Host2) and the PVM logical devices on that host, including NICS, redirected disks, and A-Link adapters used by the host.
3. The Availability Links object (**mtceverRunPVMALink**) — Contains the states and parameters of the availability links and the A-link paths between the hosts.
4. The Quorum Service object (**mtceverRunPVMQuorumService**) — Contains the states and parameters of the quorum service on the PVM, including the states of the preferred and alternate quorum servers for each host.

The MIB subtree for the everRun MIBv1 manageable objects is shown in [Figure 5](#).

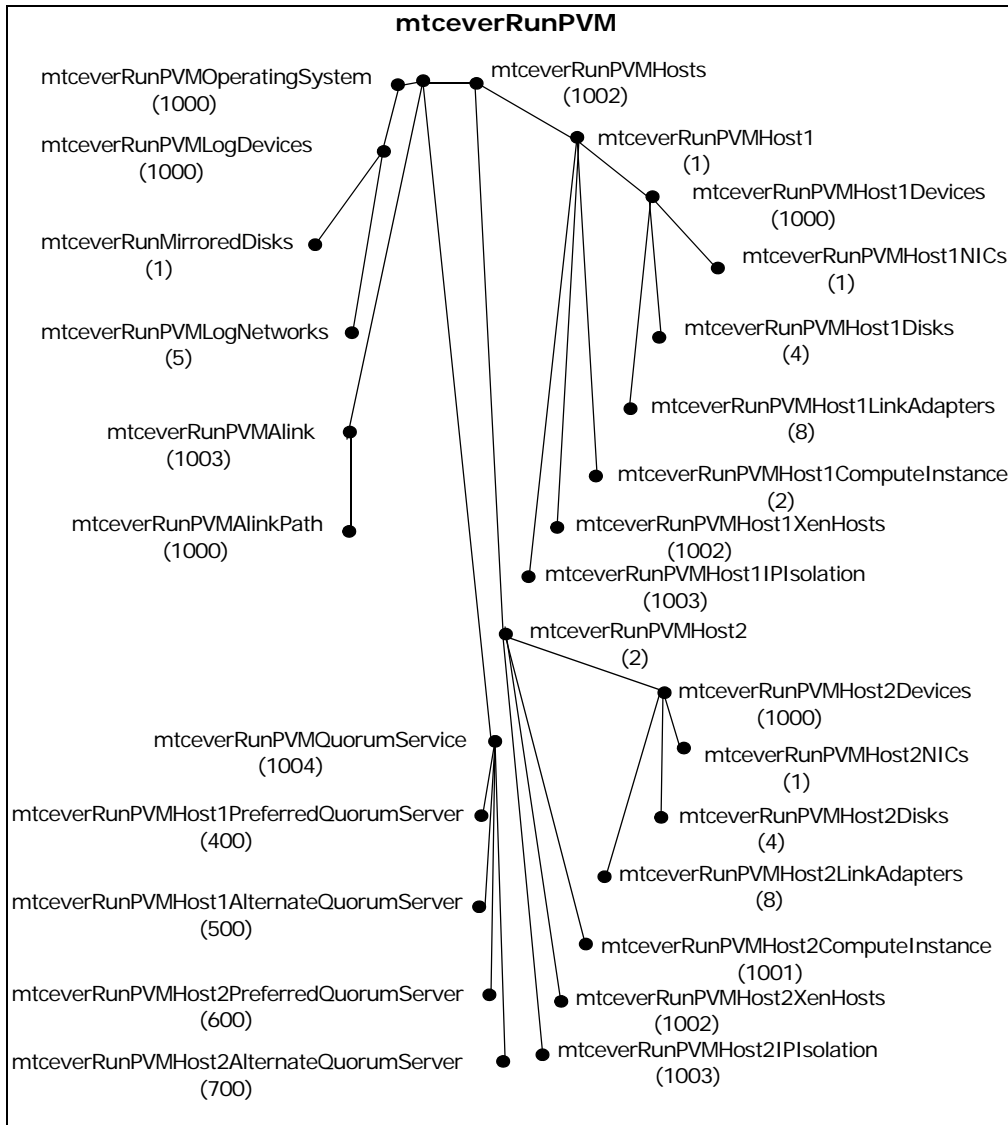


Figure 5 everRun objects subtree of MIB naming tree

The top node of the everRun MIBv1, **mtceverRunPVM**, contains state and configuration information for the PVM as a whole. It contains the managed objects described in “everRun PVM Parameters” on page 25. It further contains four subnodes:

- The **mtceverRunPVMOperatingSystem** node contains state and configuration information for the PVM operating system. A subnode under this node,

mtceverRunPVMLogDevices, includes separate subnodes for mirrored disks and logical networks. These represent the logical devices configured to the PVM operating system by the user. All of these nodes have managed objects associated with them, as described in “everRun PVM Operating System Parameters” on page 26.

- The **mtceverRunPVMHosts** node contains two subnodes — one for each host. Each host subnode contains additional subnodes that provide state and configuration information for the PVMs on the host. All of these nodes have managed objects associated with them, as described in “everRun PVM Hosts Parameters” on page 30.
- The **mtceverRunPVMALink** node contains state and configuration information for the availability links. It includes one subnode that provides state and configuration information for each open path through the Availability Link. All of these nodes have managed objects associated with them, as described in “Link Adapter Parameters” on page 32.
- The **mtceverRunPVMQuorumService** node contains configuration information for the quorum service for the PVM, if the service is enabled. It further contains four subnodes that indicate the reachability of the preferred and alternate quorum servers from each host. All of these nodes have managed objects associated with them, as described in “everRun IP Isolation Address Information” on page 35.

Component States

All components that have states associated with them use the same set of subnodes to describe the state. While the base node and the component-specific prefix are different for the each component, certain fields (name, offset from the base node, and description) are the same for every component. For brevity, [Table 3](#) describes each component's node structure, rather than repeat the entries in every table that describes manageable components.

Table 3 mtceverRunPVM Component State Nodes

Field Name	Base Offset	Description
State	0	The state of the everRun PVM. One of: <i>(1) failed</i> <i>(2) offline</i> <i>(3) disabled</i> <i>(4) degraded</i> <i>(5) transitioning</i> <i>(6) good</i> <i>(7) unknown</i> <i>(8) idle</i>
StateString	1	The state of the component in string form.
SubState	2	The substate of the component.
SubStateString	3	The substate of the component in string form.
Severity	4	The severity of the current component state. One of: <i>(1) error</i> <i>(2) unknown</i> <i>(3) warning</i> <i>(4) informational</i> <i>(5) good</i>
SeverityString	5	The severity of the current component state in string form.
PreviousState	6	The state of the component before the last transition. Same values as State row above.
PreviousStateString	7	The state of the component before the last transition in string form.
PreviousSubState	8	The substate of the component before the last transition.

Table 3 mtceverRunPVM Component State Nodes

Field Name	Base Offset	Description
PreviousSubStateString	9	The substate of the component before the last transition in string form.
PreviousSeverity	10	The severity of the previous component state. Same values as Severity row above
PreviousSeverityString	11	The severity of the previous component state in string form.
StateChangeReason	12	The reason for the last component state transition.
StateChangeReasonString	13	The reason for the last component state transition in string form.
StateChangeTime	14	The date and time of the last component state transition.

everRun PVM Parameters

The **mtceverRunPVM** node of the everRun MIBv1 contains objects that describe parameters associated with the operation of the entire everRun PVM. These objects, shown in [Table 4](#), are all scalar data types.

For brevity, the full variable name of each field string has been truncated in the following table. Each name begins with the term **mtceverRunPVM**; for example, the first field name is **mtceverRunPVMProductName**.

Table 4 mtceverRunPVM Scalar Entries

Field Name	Child No.	Description
ProductName	2	The everRun product name.
RevisionLevel	3	The revision of the everRun software that is installed on the PVM.
AgentSystemName	4	The host name of the system where the everRun extension agent is running.

Table 4 mtceverRunPVM Scalar Entries

Field Name	Child No.	Description
AgentLocation	5	The location where the everRun SNMP extension agent is running.
PrimaryHostId	6	The ID of the primary host for the everRun PVM.
DeviceAffinity	8	If non-zero, device affinity is enabled.
ProtectionLevel	9	The protection level associated with this everRun PVM (Level 2 or 3).
<i>States</i>	100 - 114	See Table 3 on page 24 .

everRun PVM Operating System Parameters

The **mtceverRunPVMOperatingSystem** node contains state and configuration information for the PVM operating system. These objects are all scalar data types ([Table 5](#)). For brevity, the **mtceverRunPVMOperatingSystem** prefix has been removed from each field name.

Table 5 mtceverRunPVMOperatingSystem Scalar Entries

Field Name	Child No.	Description
Name	1	The host name of the PVM operating system.
Version	2	The version of the PVM operating system.
SysDir	3	The system directory of the PVM operating system.
AutoBoot	4	Flag indicating whether the protected virtual machine will automatically boot.

Table 5 mtceverRunPVMOperatingSystem Scalar Entries

Field Name	Child No.	Description
AutoResynch	5	Flag indicating whether the Level 3 protected virtual machine will be automatically resynchronized.
<i>States</i>	100 - 114	See Table 3 on page 24 .
MigrationCapable	200	Status information indicating whether the PVM can currently be migrated online.
MigrationCompatibilityUnknown	201.1	If non-zero, migration compatibility is unknown.
MigrationCompatibilityMemorySizeMismatch	201.2	If non-zero, migration is not supported because the amount of memory available on the two hosts is not the same.
MigrationCompatibilityCpuCountMismatch	201.3	If non-zero, migration is not supported because the number of processors available on the two hosts is not the same.
MigrationCompatibilityCpuRevisionMismatch	201.4	If non-zero, migration is not supported because the CPU revisions of the two hosts are not compatible.
MigrationCompatibilityCpuFeatureMismatch	201.5	If non-zero, migration is not supported because the CPU features are not the same on the two hosts.

Mirrored Disk Parameters

The **mtceverRunPVMMirroredDisks** node of the everRun PVM MIBv1 contains objects that describe mirrored disks configured to the PVM operating system by the user. These objects are all scalar data types ([Table 6](#)). For brevity, the **mtceverRunPVMMirroredDisks** prefix has been removed from each field name.

Table 6 mtceverRunPVMMirroredDisks Scalar Entries

Field Name	Child No.	Description
Index	1	An index for the PVM mirrored disk name, state, and configuration table.
Name	2	The name of the PVM mirrored disk.
LogScsiId	5	The PVM mirrored disk SCSI ID.
IsBootDevice	6	This flag indicates if this disk is the PVM boot device.
SectorSize	7	Sector size of this PVM mirrored disk.
NumSectors	8	Number of sectors on the PVM mirrored disk.
SourceHost	9	The ID of the PVM host that was the source of the last mirror copy or the host that will be the source of a pending mirror copy.
DestHost	10	The ID of the PVM host that was the destination of the last mirror copy or the host that will be the destination of a pending mirror copy.
CopyState	11	The state of the mirror copy for this PVM mirrored disk. One of: <i>(1) pending</i> <i>(2) inProgress</i> <i>(3) complete</i> <i>(4) failed</i> <i>(5) notPossible</i> <i>(6) unknown</i>
Capacity	12	The capacity of the PVM mirrored disk (examples: 100.000MB, 8.474GB, 1.500TB).

Table 6 mtceverRunPVMMirroredDisks Scalar Entries

Field Name	Child No.	Description
PercentCopied	13	Percent of the PVM mirrored disk that has been copied.
SizeCopied	14	Number of bytes of the PVM mirrored disk that have been copied (e.g., 100.000MB, 8.474GB, 1.500TB).
CopyStartTime	15	The start time of the current or most recent mirror copy.
CopyEndTime	16	The end time of the last mirror copy.
CopyEstEndTime	17	The estimated end time of the current mirror copy.
CopyRate	18	The rate of the current or most recently completed mirror copy, in MB/sec.
CopyType	19	The mirror copy type. One of: <i>(1) full</i> <i>(2) delta</i> <i>(3) none</i> <i>(4) unknown</i>
NumSectorsToCopy	20	Number of sectors to be copied.
NumSectorsCopied	21	Number of sectors already copied.
IsDelta	22	Indicates whether delta copy is enabled on the everRun PVM.
IsDeltaPriv	23	Indicates whether delta copy is enabled on the PVM mirrored disk.
<i>States</i>	100 - 114	See Table 3 on page 24 .

Network Parameters

The **mtceverRunPVMLogNetworks** subnode of the everRun PVM MIBv1 contains objects that describe redirected network adapters configured to the PVM operating system by the user.

These objects are all scalar data types ([Table 7](#)). For brevity, the **mtceverRunPVMLogNetworks** prefix has been removed from each field name.

Table 7 mtceverRunPVMLogNetworks Scalar Entries

Field Name	Child No.	Description
Index	1	An index for the PVM Ethernet adapter name, state, and configuration table.
Name	2	The name of the PVM Ethernet adapter.
MacAddress	3	The MAC address for the PVM Ethernet adapter.
<i>States</i>	100 - 114	See Table 3 on page 24 .

everRun PVM Hosts Parameters

The **mtceverRunPVMHost1** and **mtceverRunPVMHost2** nodes of the everRun MIBv1 contain objects that describe parameters associated with the operation of the PVM on a host. These objects are all scalar data types ([Table 8](#)). For brevity, the **mtceverRunPVHost1** or **mtceverRunPVMHost2** prefix has been removed from each field name.

Table 8 mtceverRunPVMHost Scalar Entries

Field Name	Child No.	Description
Name	1	The name of the PVM host (<i>Host1</i> or <i>Host2</i>).
InitInterval	6	The number of seconds that this host will wait for the other host to become available before automatically starting the PVM on its own.
AutoStart	7	Indicates whether the PVM will start on the host automatically in the absence of the other host.
<i>States</i>	100 - 114	See Table 3 on page 24 .

NICs Parameters

The **mtceverRunPVMHost1NICs** and **mtceverRunPVMHost2NICs** nodes of the **everRun MIBv1** contain objects that describe parameters associated with the operation of NICs on a single host. These objects are all scalar data types ([Table 9](#)). For brevity, the **mtceverRunPVMHost1NICs** or **mtceverRunPVMHost2NICs** prefix has been removed from each field name.

Table 9 mtceverRunPVMHostNIC Scalar Entries

Field Name	Child No.	Description
Index	1	An index for the Ethernet adapter name, state, and configuration table.
Name	2	The name of the Ethernet adapter.
HostId	3	ID of the host where the physical Ethernet adapter resides.
<i>States</i>	100 - 114	See Table 3 on page 24 .

Disks Parameters

The **mtceverRunPVMHost1Disks** and **mtceverRunPVMHost2Disks** nodes of the **everRun MIBv1** contain objects that describe parameters associated with a disk on a single PVM. These objects are all scalar data types ([Table 10](#)). For brevity, the **mtceverRunPVHost1Disks** or **mtceverRunPVMHost2Disks** prefix has been removed from each field name.

Table 10 mtceverRunPVMHostDisks Scalar Entries

Field Name	Child No.	Description
Index	1	An index for the disk drive name, state, and configuration table.
Name	2	The name of the disk drive.
HostId	3	ID of the host where the physical disk resides.
GUID	6	Globally unique ID of the disk drive.
IsMirrored	10	Indicates that the physical disk is configured as part of a physical set.
Capacity	13	The capacity of the physical disk (examples: 100.000MB, 8.474GB, 1.500TB).
SectorSize	14	Sector size of this physical disk.
NumSectors	15	Number of sectors on the physical disk.
<i>States</i>	100 - 114	See Table 3 on page 24 .

Link Adapter Parameters

The **mtceverRunPVMHost1LinkAdapters** (and **...Host2LinkAdapters**) nodes of the **everRun MIBv1** contain objects that describe parameters associated a link adapter on a PVM host. These objects are all scalar data types ([Table 11](#)). For brevity, the prefix

mtceverRunPVMHost1LinkAdapters (or **...Host2LinkAdapters**) has been removed from each field name.

Table 11 mtceverRunPVMHostLinkAdapters Scalar Entries

Field Name	Child No.	Description
Index	1	An index for the link adapter name, state, and configuration table.
Name	2	The name of the link adapter.
HostId	3	ID of the host where the physical link adapter resides.
FriendlyName	4	The friendly name of the link adapter.
IPAddress	5	The IP address of the link adapter.
Path1Usage	6	The usage of the link adapter's path 1 (<i>primary</i> or <i>secondary</i>). One of: (1) <i>none</i> (2) <i>primary</i> (3) <i>secondary</i> (4) <i>both</i>
Path2Usage	7	The usage of the link adapter's path 2 (<i>primary</i> or <i>secondary</i>). Same values as the Path1Usage row above.
Path1Speed	8	The speed, in megabytes per second, of the link adapter's path 1.
Path2Speed	9	The speed, in megabytes per second, of the link adapter's path 2.
<i>States</i>	100 - 114	See Table 3 on page 24 .

Compute Instance Parameters

The **mtceverRunPVMHost1ComputeInstance** (and **...Host2ComputeInstance**) nodes of the everRun MIBv1 contain objects that describe parameters associated with

the compute instance on a PVM host. These objects are all scalar data types ([Table 12](#)). For brevity, the **mtceverRunPVMHost1ComputeInstance** (or **...Host2ComputeInstance**) prefix has been removed from each field name.

Table 12 mtceverRunPVMHostComputeInstance Scalar Entries

Field Name	Child No.	Description
Name	1	The name of the compute instance.
AutoSynch	2	Flag indicating whether the compute instance will automatically be synchronized.
AutoBoot	3	Flag indicating whether the compute instance will automatically boot.
<i>States</i>	100 - 114	See Table 3 on page 24 .

XenServer Host Information

The **mtceverRunPVMHost1XenHost** and **mtceverRunPVMHost2XenHost** nodes of the everRun MIBv1 contain objects that describe parameters associated with the XenServer host on which the PVM is configured. These objects are all scalar data types (see [Table 13](#)).

For brevity, the prefix **mtceverRunPVMHost1XenHost** or **mtceverRunPVMHost2XenHost** has been removed from each field name.

Table 13 mtceverRunPVMHostsXenHost Scalar Entries

Field Name	Child No.	Description
Name	1	The name of the XenServer host that is the location for this PVM host.
IpAddr	2	The IP address of the XenServer host that is the location for this PVM host.
PoolMasterName	3	The name of the XenServer host that is the pool master for this PVM host.

Table 13 mtceverRunPVMHostsXenHost Scalar Entries

Field Name	Child No.	Description
PoolMasterIPAddr	4	The IP address of the XenServer host that is the pool master for this PVM host
PoolMasterStates	100 - 114	See Table 3 on page 24 .

everRun IP Isolation Address Information

The **mtceverRunPVMHost1IpIsolation** and **mtceverRunPVMHost2IpIsolation** nodes of the everRun MIBv1 contain objects that describe parameters associated with the everRun IP Isolation Address set via the eAC or the everRun CLI. These objects are all scalar data types (see [Table 14](#)).

For brevity, the prefix **mtceverRunPVMHost1IpIsolation** or **mtceverRunPVMHost2IpIsolation** has been removed from the field name.

Table 14 mtceverRunPVMHostIpIsolation Scalar Entries

Field Name	Child No.	Description
Addr	1	The IP address of the XenServer host that is the location for this PVM host.
AddrStates	100 - 114	See Table 3 on page 24 .

everRun PVM Availability Link Parameters

The **mtceverRunPVMALink** node of the everRun MIBv1 contains objects that describe parameters associated with the operation of the availability link. These objects are all scalar data types ([Table 15](#)). For brevity, the **mtceverRunPVMALink** prefix has been removed from each field name.

Table 15 mtceverRunPVMALink Scalar Entries

Field Name	Child No.	Description
Name	1	The name of the availability link.
<i>States</i>	100 - 114	See Table 3 on page 24 .

A-Link Path Parameters

The **mtceverRunPVMALinkPath** node of the everRun MIBv1 contains objects that describe parameters associated with the operation of the availability link. These objects are all scalar data types ([Table 16](#)). For brevity, the **mtceverRunPVMALinkPath** prefix has been removed from each field name.

Table 16 mtceverRunPVMALinkPath Scalar Entries

Field Name	Child No.	Description
Index	1	An index for the table containing the name and state of each availability link path.
ID	2	The ID number for the A-link path (<i>Path 1</i> or <i>Path 2</i>).
<i>States</i>	100- 114	See Table 3 on page 24 .

everRun PVM Quorum Service Parameters

The **mtceverRunPVMQuorumService** node of the everRun MIBv1 contains objects that describe parameters associated with the operation of the Quorum Service. These

objects are all scalar data types (Table 17). For brevity, the table omits the prefix **mtceverRunPVMQuorumService** from each field name.

Table 17 mtceverRunPVMQuorumService Scalar Entries

Field Name	Child No.	Description
IsEnabled	1	A non-zero value indicates that the quorum service is enabled. A zero value indicates that the quorum service is disabled.
BootIsBlocked	2	A non-zero value indicates that the PVM boot is blocked by the quorum manager.
BootBlockedReason	3	The reason why the quorum manager blocked the PVM boot. One of the following: <i>(1) none</i> <i>(2) lockoutDenied</i> <i>(3) blockedByQuorumService</i> <i>(4) noQuorumService</i>
JoinBlocked	4	A non-zero value indicates that the host join is blocked by the quorum manager.
JoinBlockedReason	5	The reason why the quorum manager blocked the host join. One of the following: <i>(1) none</i> <i>(2) NoQuorumService</i>
CurrentQuorumServerName	100	The computer name of the currently elected quorum server.
CurrentQuorumServerAddr	101	The IP address of the currently elected quorum server.
PreferredQuorumServerName	200	The computer name of the preferred quorum server.
PreferredQuorumServerAddr	201	The IP address of the preferred quorum server.

Table 17 mtceverRunPVMQuorumService Scalar Entries

Field Name	Child No.	Description
AlternateQuorumServerName	300	The computer name of the alternate quorum server.
AlternateQuorumServerAddr	301	The IP address of the alternate quorum server.

Preferred and Alternate Quorum Service Parameters

[Table 18](#) indicates the reachability state of the preferred and alternate quorum servers on each host. The only valid states are *good*, which means the quorum service is reachable, and *offline*, which means the quorum service is unreachable.

The objects shown in [Table 18](#) are all scalar data types. For brevity, the **mtceverRunPVM** prefix has been removed from each field name.

Table 18 mtceverRunPVMHostQuorumServer Scalar Entries

Field Name	Child No.	Description
Host1PreferredQuorumServerStates	400.100-114	See Table 3 on page 24 .
Host1AlternateQuorumServerStates	500.100-114	See Table 3 .
Host2PreferredQuorumServerStates	600.100-114	See Table 3 .
Host2AlternateQuorumServerStates	700.100-114	See Table 3 .

For More Information on everRun Traps

The following chapter contains information about the format of everRun traps. everRun states and their meanings are documented in [Table 3 on page 24](#).

This chapter describes the everRun SNMP extension agent traps (event notifications) that may be generated by an implementing agent.

About the everRun SNMP Extension Agent

The everRun SNMP extension agent, running on the everRun PVM, includes two major pieces of functionality. At the highest level, it:

- Provides read-only access to everRun server management information.
- Monitors the state of the everRun server and generates SNMP traps when state transitions occur in the everRun PVM.

When a PVM undergoes a state change, the everRun agent sends a trap, along with associated state changes and other relevant variables — including the name of the PVM where the event occurred.

About everRun Traps (Event Notifications)

The everRun MIB specifies the event notifications that may be generated by an implementing agent. Because every system transition involves multiple state changes, traps are not generated for every state change, but only for events that are deemed major transitions.

Table 19 lists specific state transitions for each entity that causes an event notification. For your convenience, traps are organized by category (PVM traps, mirrored disk traps, and so forth). Table 20 on page 45 lists variables that are common to all manageable components.

NOTE: For brevity, the prefix **mtceverRunPVMTrap** has been omitted from the trap name on the left side of [Table 19](#).

Table 19 everRun SNMP Agent Traps

Trap Name	Trap Description	Additional Variables
PVM Traps		
Good	everRun PVM State Becomes Good	
Informational	everRun PVM State Informational	
Warning	everRun PVM State Warning	
Unknown	everRun PVM State Unknown	
Error	everRun PVM State Error	
Mirrored Disk Traps		
MirroredDiskGood	Mirrored Disk State Becomes Good	<i>Name</i> <i>LogScsiIds</i> <i>IsBootDevice</i>
MirroredDiskInformational	Mirrored Disk State Informational	Same as above
MirroredDiskWarning	Mirrored Disk State Warning	Same as above
MirroredDiskUnknown	Mirrored Disk State Unknown	Same as above
MirroredDiskError	Mirrored Disk State Error	Same as above

Table 19 everRun SNMP Agent Traps

Trap Name	Trap Description	Additional Variables
Logical Network Traps		
LogicalNetworkGood	Logical Network State Becomes Good	<i>Name</i> <i>MacAddress</i>
LogicalNetworkInformational	Logical Network State Informational	Same as above
LogicalNetworkWarning	Logical Network State Warning	Same as above
LogicalNetworkUnknown	Logical Network State Unknown	Same as above
LogicalNetworkError	Logical Network State Error	Same as above
Compute Instance Traps		
ComputeInstance1StateGood ComputeInstance2StateGood	Compute Instance 1 or Compute Instance 2 State Becomes Good	<i>Name</i>
ComputeInstance1StateInformational ComputeInstance2StateInformational	Compute Instance 1 or Compute Instance 2 State Informational	Same as above
ComputeInstance1StateWarning ComputeInstance2StateWarning	Compute Instance 1 or Compute Instance 2 State Warning	Same as above
ComputeInstance1StateUnknown ComputeInstance2StateUnknown	Compute Instance 1 or Compute Instance 2 State Unknown	Same as above
ComputeInstance1StateError ComputeInstance2StateError	Compute Instance 1 or Compute Instance 2 State Error	Same as above

Table 19 everRun SNMP Agent Traps

Trap Name	Trap Description	Additional Variables
Host Traps		
Host1Good Host2Good	everRun PVM Host 1 or Host 2 State Becomes Good	<i>Name</i>
Host1Informational Host2Informational	everRun PVM Host 1 or Host 2 State Informational	Same as above
Host1Warning Host2Warning	everRun PVM Host 1 or Host 2 State Warning	Same as above
Host1Unknown Host2Unknown	everRun PVM Host 1 or Host 2 State Unknown	Same as above
Host1Error Host2Error	everRun PVM Host 1 or Host 2 State Error	Same as above
Host Disk Traps		
Host1DiskGood Host2DiskGood	PVM Host 1 or Host 2 Physical Disk State Becomes Good	<i>Name</i> <i>HostID</i> <i>GUID</i> <i>PhysScsiIds</i>
Host1DiskInformational Host2DiskInformational	PVM Host 1 or Host 2 Physical Disk State Informational	Same as above
Host1DiskWarning Host2DiskWarning	PVM Host 1 or Host 2 Physical Disk State Warning	Same as above
Host1DiskUnknown Host2DiskUnknown	PVM Host 1 or Host 2 Physical Disk State Unknown	Same as above
Host1DiskError Host2DiskError	PVM Host 1 or Host 2 Physical Disk State Error	Same as above

Table 19 everRun SNMP Agent Traps

Trap Name	Trap Description	Additional Variables
Host NIC Traps		
Host1NICGood Host2NICGood	PVM Host 1 or Host 2 Physical Ethernet Adapter State Becomes Good	<i>Name</i> <i>HostID</i>
Host1NICInformational Host2NICInformational	PVM Host 1 or Host 2 Physical Ethernet Adapter State Informational	Same as above
Host1NICWarning Host2NICWarning	PVM Host 1 or Host 2 Physical Ethernet Adapter State Warning	Same as above
Host1NICUnknown Host2NICUnknown	PVM Host 1 or Host 2 Physical Ethernet Adapter State Unknown	Same as above
Host1NICError Host2NICError	PVM Host 1 or Host 2 Physical Ethernet Adapter State Error	Same as above
Link Adapter Traps		
LinkAdapter1Good LinkAdapter2Good	Link Adapter 1 or 2 State Becomes Good	<i>Name</i> <i>Host ID</i>
LinkAdapter1Informational LinkAdapter2Informational	Link Adapter 1 or 2 State Informational	Same as above
LinkAdapter1Warning LinkAdapter2Warning	Link Adapter 1 or 2 State Warning	Same as above
LinkAdapter1Unknown LinkAdapter2Unknown	Link Adapter 1 or 2 State Unknown	Same as above
LinkAdapter1Error LinkAdapter2Error	Link Adapter 1 or 2 State Error	Same as above

Table 19 everRun SNMP Agent Traps

Trap Name	Trap Description	Additional Variables
Availability Link (A-Link) Traps		
ALinkGood	Availability Link State Becomes Good	<i>Name</i>
ALinkInformational	Availability Link State Informational	Same as above
ALinkWarning	Availability Link State Warning	Same as above
ALinkUnknown	Availability Link State Unknown	Same as above
ALinkError	Availability Link State Error	Same as above
Quorum Server Traps		
Host2PreferredQuorumServerGood	Preferred Quorum Server is Reachable on PVM Host 2	<i>Name</i> <i>Addr</i>
Host2PreferredQuorumServerWarning	Preferred Quorum Server is Unreachable on PVM Host 2	Same as above
Host2AlternateQuorumServerGood	Alternate Quorum Server is Reachable on PVM Host 2	Same as above
Host2AlternateQuorumServerWarning	Alternate Quorum Server is Unreachable on PVM Host 2	Same as above
Pool Master Traps		
Host1PoolMasterGood Host2PoolMasterGood	XenServer Pool Master Server is Reachable on PVM Host 1 or 2	<i>Name</i>

Table 19 everRun SNMP Agent Traps

Trap Name	Trap Description	Additional Variables
Host1PoolMasterWarning Host2PoolMasterWarning	XenServer Pool Master Server is Unreachable on PVM Host 1 or 2	Same as above
everRun IP Isolation Traps		
Host1IpIsolationAddressGood Host2IpIsolationAddressGood	The everRun IP Isolation Address is Reachable on PVM Host 1 or 2	<i>IpIsolationAddr</i>
Host1IpIsolationAddressWarning Host2IpIsolationAddressWarning	The everRun IP Isolation address is Unreachable on PVM Host 1 or 2	Same as above

The behavior described in the table is the default behavior. There are registry parameters that can be changed to modify the trap generation by an agent. Contact Marathon customer support if you require more information.

Variables Common to All Traps

NOTE: For brevity, the component-specific prefix has been removed from the variable names in [Table 20](#). For example, the complete variable name for the first entry in the table is **mtceverRunPVMAgentSystemName**.

Table 20 Variables Common to All Traps

Variable Name	Description and Values
AgentSystemName	The host name of the system on which the everRun SNMP extension agent is running.
AgentLocation	PVM operating system

Table 20 Variables Common to All Traps

Variable Name	Description and Values
State	The state of the component: (1) <i>failed</i> (2) <i>offline</i> (3) <i>disabled</i> (4) <i>degraded</i> (5) <i>transitioning</i> (6) <i>good</i> (7) <i>unknown</i> (8) <i>idle</i>
StateString	The state of the component in string form.
SubState	The substate of the component.
SubStateString	The substate of the component in string form.
Severity	The severity of the current state: (1) <i>error</i> (2) <i>unknown</i> (3) <i>warning</i> (4) <i>info</i> (5) <i>good</i>
PreviousState	The previous state of the component. Same values as listed in the State row above.
PreviousStateString	The previous state of the component in string form.
PreviousSubState	The previous substate of the component. Same values as listed in the SubState row above.
PreviousSubStateString	The previous substate of the component in string form.

Table 20 Variables Common to All Traps

Variable Name	Description and Values
PreviousSeverity	The severity of the previous state. Same values as listed in the Severity row above.
LastStateChangeReason	The reason for the last everRun PVM state transition.
LastStateChangeReason String	The reason for the last everRun PVM state transition in string form.
StateChangeTime	Date and time of the last component state transition.

Windows Event Log Entries

5



The everRun SNMP extension agent makes entries in the Application section of the Windows event log to indicate internal situations that may be of interest to the everRun system administrator. This chapter contains a summary of the entries currently generated.

Summary of Windows Event Log Messages

The following entries may be written to the Windows event log:

Message:	The everRun SNMP extension agent has started successfully.
Severity:	Informational
Description:	The everRun SNMP extension agent DLL has been loaded and initialized by the Windows SNMP service.
Action:	No action is required. This event is used primarily to provide an entry in the event log with a time stamp to indicate that the extension agent has started.
Message:	The everRun SNMP extension agent is now enabled.
Severity:	Informational
Description:	The everRun extension agent is enabled and becomes an extension of the Window SNMP service.
Action:	No action is required. This event is issued when the everRun SNMP extension agent is enabled through the Marathon menu command.

Message:	The everRun SNMP extension agent is now disabled.
Severity:	Informational
Description:	This everRun SNMP extension agent is disabled and is removed from the Window SNMP service.
Action:	No action is required. This event is issued when the everRun SNMP extension agent is disabled through the Marathon menu command.
Message:	The everRun configuration data have been read successfully.
Severity:	Informational
Description:	This event indicates that the agent has re-established access to the everRun server configuration data.
Related Messages:	<i>The everRun configuration data could not be read.</i>
Action:	No action is necessary. The problem previously reported has been rectified.
Message:	The everRun configuration data could not be read.
Severity:	Error
Description:	The extension agent background thread runs periodically to obtain the latest everRun configuration information. This information is used to populate the MIB database and to check for events that require an SNMP trap to be generated. The event indicates that an error occurred while attempting to access the configuration information via the everRun system management API. This message is repeated periodically until the agent restores access to the configuration data. Most MIB data is not available and no SNMP traps are generated until the agent restores access to the configuration data.
Related Messages:	<i>The everRun configuration data has been successfully read.</i>

Action: This error indicates an internal problem with the everRun system management components. The event should be reported to customer support. Record and include the values of the additional data contained in the event log when reporting the error. If the agent re-establishes access to the configuration data, an event will be generated to report the restored access. If the problem persists, the agent should be restarted by stopping the Windows SNMP service and restarting it, using the Computer Management utility accessible via the Administrative tools in the Control Panel.

Message: **The everRun SNMP extension agent has successfully terminated.**

Severity: Informational

Description: The everRun extension agent has shut down successfully.

Action: No action is required. This event is issued when SNMP service is shut down, or when the everRun SNMP extension agent is shut down as a result of a Marathon menu command.

Message: **The everRun SNMP extension agent failed to load the everRun system management API DLL.**

Severity: Error

Description: The everRun SNMP extension agent failed to load the everRun system management API, **Mtcsmapi.dll** — a DLL that is dynamically linked when the Windows SNMP service loads the everRun extension agent. The everRun agent terminates when this error occurs.

Action: The extension agent first tries to resolve the version of the Marathon software running on the CI, using the information in the SOFTWARE\Marathon registry. After successful resolution, it tries to load the appropriate system management API DLL from the Marathon folder. A failure here indicates that either the SOFTWARE\Marathon registry does not contain the correct version information, or the Marathon folder does not contain the correct system management DLL.

Make sure the SOFTWARE\Marathon registry has correct data, and that the Marathon software folder contains the correct system management API DLL. The agent can be restarted by stopping the Window SNMP service and restarting it, using the Computer Management utility accessible via the Administrative tools in the Control Panel. If problem persists, contact Marathon customer support to report the error.

Message: **The everRun SNMP extension agent failed to initialize.**

Severity: Error

Description: The everRun SNMP extension agent failed to initialize successfully. The everRun agent terminates when this error occurs.

Action: The extension agent generates this event log entry when it fails to initialize internal data structures successfully during startup, as the agent experiences an unexpected failure from a Win32 API call.

The event should be reported to Marathon customer support. Record and include the values of the additional data contained in the event log when reporting the error. The agent can be restarted by stopping the Windows SNMP service and restarting it, using the Computer Management utility accessible via the Administrative tools in the Control Panel.

Message: **The everRun SNMP extension agent background thread failed unexpectedly.**

Severity: Error

Description: The background thread that polls the everRun server, to retrieve MIB data and to watch for events that require SNMP traps, has failed unexpectedly. The failures are related to unexpected errors from Win32 API calls. The agent terminates when this error occurs.

Action: The extension agent generates this error during normal operation when it encounters an unexpected failure from a Win32 API call. The agent is terminated. Contact Marathon customer support to report the error. Record and include the values of the additional data contained in the event log when reporting the error. The agent can be restarted by stopping the Windows SNMP service and restarting it, using the Computer Management utility accessible via the Administrative tools in the Control Panel.

Message:	The everRun SNMP extension agent could not allocate memory resources.
-----------------	--

Severity:	Error
------------------	-------

Description:	This event indicates that the agent could not allocate the memory resources needed to produce the MIB data. This event may be generated periodically until the necessary memory resources become available. Some or all MIB data may not be available until the memory resources become available and some traps may not be generated.
---------------------	--

Action:	If the problem persists, the agent should be restarted by stopping the Windows SNMP service and restarting it, using the Computer Management utility accessible via the Administrative tools in the Control Panel.
----------------	--

Message:	The everRun SNMP extension agent could not allocate background thread resources.
-----------------	---

Severity:	Error
------------------	-------

Description:	This event indicates that the agent could not allocate the thread resource during execution. This event may be generated periodically until the necessary thread resources become available. Some or all queries may return error conditions.
---------------------	---

Action:	If the problem persists, the agent should be restarted by stopping the Windows SNMP service and restarting it, using the Computer Management utility accessible via the Administrative tools in the Control Panel.
----------------	--

SNMP Glossary



enumerated integers

A list of labeled integer values. Many of the data values passed by the agent to a remote management station are enumerated integers. The agent sends an integer value which is then interpreted into a text string by the management application. This provides a mechanism to separate the value being sent from the text string equivalent, allowing for language- or site-specific translations of the values.

everRun SNMP extension agent

A user-mode extension DLL (dynamic load library) loaded by the Windows SNMP Service when the service starts. The agent, in turn, loads the System Management API (SMAPI) DLL and uses it to access the everRun management information. The agent utilizes the API exported by the Microsoft SNMP service to provide access to this information to SNMP management applications running on remote systems.

MIB

Management Information Base. In a global sense, this refers to the virtually infinite set of all data that exists to be managed in any network environment. In the context of this document, it refers to the everRun data that is available to be managed.

OID

Object Identifier. Each node in the MIB tree, and each data object in a MIB, has an associated object identifier of the format *a.b.c.x.y.z*, where each letter represents a number. The length of the OIDs (the count of separate numbers) is always less than 128.

protected VM (PVM)

A virtual machine (VM) to which everRun availability software has been applied.

PVM operating system

In this document, this term is used to refer to the instance of the Windows operating system that is executing in the redundant context of an everRun PVM.

RFC

Request for Comments. RFCs are used by the Internet Engineering Task Force as a vehicle for publishing and tracking standards and proposed standards. The SNMP protocol itself is defined in a series of RFCs.

SNMP

Simple Network Management Protocol. A protocol designed to allow for a common mechanism to manage diverse devices in a network environment. It provides basic functions to read (GET) and write (SET) management data and to generate event notifications (TRAPS) when changes in

state or data occur. The first version of the SNMP protocols (SNMPv1), which is supported by the everRun extension agent, is widely used. SNMPv2c (version 2, without certain security features) also enjoys wide use. SNMPv3 is now a standard; however it is not yet widely used.

trap

A system event notification based on the SNMP protocol, following a change in the object state variables in the MIB data.

Windows SNMP service

A Windows service, based on the Simple Network Management Protocol (SNMP), that is used by everRun to enable everRun SNMP extension agents that monitor the activity (via event notifications) in network devices and report to the remote console workstation.