



Memory Leak Detector

First Published: May 17, 2004

Last Updated: June 1, 2006

The Memory Leak Detector feature is a tool that can be used to detect memory leaks on a router that is running Cisco IOS software. The Memory Leak Detector feature is capable of finding leaks in all memory pools, packet buffers, and chunks.

History for the Memory Leak Detector Feature

Release	Modification
12.3(8)T1	This feature was introduced.
12.2(25)S	This feature was integrated into Cisco IOS Release 12.2(25)S.
12.2(33)SRA	This feature was integrated into Cisco IOS Release 12.2(33)SRA.

Finding Support Information for Platforms and Cisco IOS and Catalyst OS Software Images

Use Cisco Feature Navigator to find information about platform support and software image support. Cisco Feature Navigator enables you to determine which Cisco IOS and Catalyst OS software images support a specific software release, feature set, or platform. To access Cisco Feature Navigator, go to <http://www.cisco.com/go/cfn>. An account on Cisco.com is not required.

Contents

- [Information About Memory Leak Detector, page 2](#)
- [How to Use Memory Leak Detector, page 3](#)
- [Additional References, page 10](#)
- [Command Reference, page 11](#)



Corporate Headquarters:
Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

© 2004, 2006 Cisco Systems, Inc. All rights reserved.

Information About Memory Leak Detector

Before using the Memory Leak Detector feature, you should understand the following concepts:

- [Memory Leaks, page 2](#)
- [Memory Leak Detection, page 2](#)

Memory Leaks

Memory leaks are static or dynamic allocations of memory that do not serve any useful purpose. Although technology is available for detection of leaks among statically allocated memory, in this document the focus is on memory allocations that are made dynamically.

Memory Leak Detection

From the detection point of view, leaks among the dynamically allocated memory blocks can be classified into the following three types:

- Type 1 leaks have no references. These blocks of memory can not be accessed.
- Type 2 leaks are part of one or more cycles of allocations but none of the blocks in these cycles is accessible from outside of the cycles. Blocks within each cycle have references to other elements in the cycle(s). An example of a Type 2 leak is a circular list that is not needed anymore. Though individual elements are reachable, the circular list is not reachable.
- Type 3 leaks are accessible or reachable but are not needed, for example, elements in data structures that are not needed anymore. A subclass of Type 3 leaks are those where allocations are made but never written to. You can look for these subclass leaks using the **show memory debug reference unused** command.

The Memory Leak Detector feature provides the technology to detect Type 1 and Type 2 memory leaks.

The Memory Leak Detector feature works in the following two modes:

- Normal mode—Where memory leak detector uses memory to speed up its operations.
- Low memory mode—Where memory leak detector runs without attempting to allocate memory.

Low memory mode is considerably slower than the normal mode and can handle only blocks. There is no support for chunks in low memory mode. Low memory mode is useful when there is little or no memory available on the router.

The memory leak detector has a simple interface and can be invoked by the command line interface (CLI) at any time to get a report of memory leaks. For testing purposes, you can perform all tests, then invoke memory leak detector to get a report on leaks. If you are interested only in leaks generated by your test cases alone, memory leak detector has an incremental option, which can be enabled at the start of testing. After testing completes, you can get a report on only the leaks that occurred after the incremental option was enabled.

To reduce false alarms, it is mandatory that memory leak detector be invoked multiple times and that only leaks that consistently appear in all reports be interpreted as leaks. This is especially true for packet buffer leaks.



Note

When submitting defects based on the reports of memory leak detector, please add “memleak-detection” to the attribute field of the defect report.

How to Use Memory Leak Detector

This section contains the following procedures:

- [Displaying Memory Leak Information, page 3](#)
- [Setting the Memory Debug Incremental Starting Time, page 8](#)
- [Displaying Memory Leak Information Incrementally, page 8](#)

Displaying Memory Leak Information

This task describes how to display detected memory leak information.

SUMMARY STEPS

1. **enable**
2. **show memory debug leaks [chunks | largest | lowmem | summary]**

DETAILED STEPS

	Command or Action	Purpose
Step 1	<pre>enable</pre> <p>Example: Router> enable</p>	<p>Enables privileged EXEC mode.</p> <ul style="list-style-type: none"> Enter your password if prompted.
Step 2	<pre>show memory debug leaks</pre> <p>or</p> <pre>show memory debug leaks [chunks]</pre> <p>or</p> <pre>show memory debug leaks [largest]</pre> <p>or</p> <pre>show memory debug leaks [lowmem]</pre> <p>or</p> <pre>show memory debug leaks [summary]</pre> <p>Example: Router# show memory debug leaks or</p> <p>Example: Router# show memory debug leaks chunks or</p> <p>Example: Router# show memory debug leaks largest or</p> <p>Example: Router# show memory debug leaks lowmem or</p> <p>Example: Router# show memory debug leaks summary</p>	<p>Invokes normal mode memory leak detection and displays detected memory leaks. It does not detect memory leaks in chunks.</p> <p>or</p> <p>(Optional) Invokes normal mode memory leak detection and displays detected memory leaks in chunks.</p> <p>or</p> <p>(Optional) Invokes memory leak detection and displays the top ten leaking allocator_pcs and total amount of memory that they have leaked. Additionally, each time this command is invoked it remembers the previous invocation's report and compares it to the current invocation's report.</p> <p>or</p> <p>(Optional) Invokes low memory mode memory leak detection and displays detected memory leaks. The amount of time taken for analysis is considerably greater than that of normal mode. The output for this command is similar to the show memory debug leaks command.</p> <p>or</p> <p>(Optional) Invokes normal mode memory leak detection and displays detected memory leaks based on allocator_pc and then on the size of the block.</p>

Examples

This section provides the following output examples:

- [Sample Output for the show memory debug leaks Command, page 5](#)
- [Sample Output for the show memory debug leaks chunks Command, page 5](#)
- [Sample Output for the show memory debug leaks largest Command, page 6](#)
- [Sample Output for the show memory debug leaks summary Command, page 7](#)

Sample Output for the show memory debug leaks Command

The following example shows output from the **show memory debug leaks** command with no optional keywords specified:

```
Router# show memory debug leaks

Adding blocks for GD...

          PCI memory
Address   Size   Alloc_pc  PID  Name
-----
          I/O memory
Address   Size   Alloc_pc  PID  Name
-----
          Processor memory
Address   Size   Alloc_pc  PID  Name
62DABD28    80 60616750  -2  Init
62DABD78    80 606167A0  -2  Init
62DCF240    88 605B7E70  -2  Init
62DCF298    96 605B7E98  -2  Init
62DCF2F8    88 605B7EB4  -2  Init
62DCF350    96 605B7EDC  -2  Init
63336C28   104 60C67D74  -2  Init
63370D58    96 60C656AC  -2  Init
633710A0   304 60C656AC  -2  Init
63B2BF68    96 60C659D4  -2  Init
63BA3FE0  32832 608D2848 104  Audit Process
63BB4020  32832 608D2FD8 104  Audit Process
```

[Table 1](#) describes the significant fields shown in the display.

Table 1 *show memory debug leaks Field Descriptions*

Field	Description
Address	Hexadecimal address of the leaked block.
Size	Size of the leaked block (in bytes).
Alloc_pc	Address of the system call that allocated the block.
PID	The process identifier of the process that allocated the block.
Name	The name of the process that allocated the block.

Sample Output for the show memory debug leaks chunks Command

The following example shows output from the **show memory debug leaks chunks** command:

```
Router# show memory debug leaks chunks

Adding blocks for GD...

          PCI memory
Address   Size   Alloc_pc  PID  Name
-----
Chunk Elements:
Address   Size   Parent   Name
-----
          I/O memory
Address   Size   Alloc_pc  PID  Name
-----
Chunk Elements:
```

```

Address  Size  Parent  Name

Processor memory
Address  Size  Alloc_pc  PID  Name
62DABD28      80 60616750 -2  Init
62DABD78      80 606167A0 -2  Init
62DCF240      88 605B7E70 -2  Init
62DCF298      96 605B7E98 -2  Init
62DCF2F8      88 605B7EB4 -2  Init
62DCF350      96 605B7EDC -2  Init
63336C28     104 60C67D74 -2  Init
63370D58      96 60C656AC -2  Init
633710A0     304 60C656AC -2  Init
63B2BF68      96 60C659D4 -2  Init
63BA3FE0    32832 608D2848 104  Audit Process
63BB4020    32832 608D2FD8 104  Audit Process

```

```

Chunk Elements:
Address  Size  Parent  Name
62D80DA8     16 62D7BFD0 (Managed Chunk )
62D80DB8     16 62D7BFD0 (Managed Chunk )
62D80DC8     16 62D7BFD0 (Managed Chunk )
62D80DD8     16 62D7BFD0 (Managed Chunk )
62D80DE8     16 62D7BFD0 (Managed Chunk )
62E8FD60    216 62E8F888 (IPC Message He)

```

Table 2 describes the significant fields shown in the display.

Table 2 *show memory debug leaks chunks Field Descriptions*

Field	Description
Address	Hexadecimal address of the leaked block.
Size	Size of the leaked block (in bytes).
Alloc_pc	Address of the system call that allocated the block.
PID	The process identifier of the process that allocated the block.
Name	The name of the process that allocated the block.
Size	(Chunk Elements) Size of the leaked element (bytes).
Parent	(Chunk Elements) Parent chunk of the leaked chunk.
Name	(Chunk Elements) The name of the leaked chunk.

Sample Output for the show memory debug leaks largest Command

The following example shows output from the **show memory debug leaks largest** command:

```

Router# show memory debug leaks largest

Adding blocks for GD...

          PCI memory
Alloc_pc  total leak size

          I/O memory
Alloc_pc  total leak size

          Processor memory
Alloc_pc  total leak size
608D2848  32776      inconclusive

```

```

608D2FD8      32776      inconclusive
60C656AC      288        inconclusive
60C67D74      48         inconclusive
605B7E98      40         inconclusive
605B7EDC      40         inconclusive
60C659D4      40         inconclusive
605B7E70      32         inconclusive
605B7EB4      32         inconclusive
60616750      24         inconclusive

```

The following example shows output from the second invocation of the **show memory debug leaks largest** command:

```
Router# show memory debug leaks largest
```

```
Adding blocks for GD...
```

```

          PCI memory
Alloc_pc  total leak size

          I/O memory
Alloc_pc  total leak size

          Processor memory
Alloc_pc  total leak size
608D2848  32776
608D2FD8  32776
60C656AC  288
60C67D74  48
605B7E98  40
605B7EDC  40
60C659D4  40
605B7E70  32
605B7EB4  32
60616750  24

```

Sample Output for the show memory debug leaks summary Command

The following example shows output from the **show memory debug leaks summary** command:

```
Router# show memory debug leaks summary
```

```
Adding blocks for GD...
```

```

          PCI memory

Alloc PC      Size      Blocks      Bytes      What

          I/O memory

Alloc PC      Size      Blocks      Bytes      What

          Processor memory

Alloc PC      Size      Blocks      Bytes      What

0x605B7E70    0000000032 0000000001 0000000032  Init
0x605B7E98    0000000040 0000000001 0000000040  Init
0x605B7EB4    0000000032 0000000001 0000000032  Init
0x605B7EDC    0000000040 0000000001 0000000040  Init
0x60616750    0000000024 0000000001 0000000024  Init
0x606167A0    0000000024 0000000001 0000000024  Init

```

```

0x608D2848 0000032776 0000000001 0000032776 Audit Process
0x608D2FD8 0000032776 0000000001 0000032776 Audit Process
0x60C656AC 0000000040 0000000001 0000000040 Init
0x60C656AC 0000000248 0000000001 0000000248 Init
0x60C659D4 0000000040 0000000001 0000000040 Init
0x60C67D74 0000000048 0000000001 0000000048 Init

```

Table 3 describes the significant fields shown in the display.

Table 3 *show memory debug leaks summary Field Descriptions*

Field	Description
Alloc PC	Address of the system call that allocated the block.
Size	Size of the leaked block.
Blocks	Number of blocks leaked.
Bytes	Total amount of memory leaked.
What	Name of the process that owns the block.

Setting the Memory Debug Incremental Starting Time

This task describes how to set the starting time for incremental analysis of memory leaks. For incremental analysis, you can define a starting point by using the **set memory debug incremental starting-time** command. When the starting time is set, only memory allocated after the starting time will be considered for reporting as leaks.

SUMMARY STEPS

1. **enable**
2. **set memory debug incremental starting-time**

DETAILED STEPS

	Command or Action	Purpose
Step 1	enable Example: Router> enable	Enables privileged EXEC mode. <ul style="list-style-type: none">• Enter your password if prompted.
Step 2	set memory debug incremental starting-time Example: Router# set memory debug incremental starting-time	Sets the starting time for incremental analysis to the time when the command is issued.

Displaying Memory Leak Information Incrementally

This task describes how to display memory leak information after a starting time has been established.

SUMMARY STEPS

1. **enable**
2. **set memory debug incremental starting-time**
3. **show memory debug incremental {allocations | leaks [lowmem] | status}**

DETAILED STEPS

	Command or Action	Purpose
Step 1	<pre>enable</pre> <p>Example: Router> enable</p>	<p>Enables privileged EXEC mode.</p> <ul style="list-style-type: none"> • Enter your password if prompted.
Step 2	<pre>set memory debug incremental starting-time</pre> <p>Example: Router# set memory debug incremental starting-time</p>	<p>Sets the starting time for incremental analysis to the time when the command is issued.</p>
Step 3	<pre>show memory debug incremental allocations</pre> <p>or</p> <pre>show memory debug incremental leaks</pre> <p>or</p> <pre>show memory debug incremental leaks lowmem</pre> <p>or</p> <pre>show memory debug incremental status</pre> <p>Example: Router# show memory debug incremental allocations</p> <p>or</p> <p>Example: Router# show memory debug incremental leaks</p> <p>or</p> <p>Example: Router# show memory debug incremental leaks lowmem</p> <p>or</p> <p>Example: Router# show memory debug incremental status</p>	<p>Displays all the memory blocks that were allocated after the issue of a set memory debug incremental starting-time command. The displayed memory blocks are just memory allocations, they are not necessarily leaks.</p> <p>or</p> <p>Displays output similar to the show memory debug leaks command, except that it displays only memory that was leaked after the issue of a set memory debug incremental starting-time command.</p> <p>or</p> <p>Forces memory leak detection to work in low memory mode. The output for this command is similar to the show memory debug leaks command, except that it displays only memory that was leaked after the issue of a set memory debug incremental starting-time command.</p> <ul style="list-style-type: none"> • In low memory mode, the analysis time is considerably greater than it is in normal mode. • You can use this command when you already know that normal mode memory leak detection will fail (perhaps by an unsuccessful previous attempt to invoke normal mode memory leak detection). <p>or</p> <p>Displays whether a starting point for incremental analysis has been set and the elapsed time since then.</p>

Examples

This section provides the following output examples:

- [Sample Output for the show memory debug incremental allocations Command, page 10](#)
- [Sample Output for the show memory debug incremental status Command, page 10](#)

Sample Output for the show memory debug incremental allocations Command

The following example shows output from the **show memory debug incremental** command when entered with the **allocations** keyword:

```
Router# show memory debug incremental allocations
```

```
Address      Size  Alloc_pc  PID  Name
62DA4E98    176  608CDC7C  44   CDP Protocol
62DA4F48     88  608CCCC8  44   CDP Protocol
62DA4FA0     88  606224A0  3    Exec
62DA4FF8     96  606224A0  3    Exec
635BF040     96  606224A0  3    Exec
63905E50    200  606A4DA4  69   Process Events
```

Sample Output for the show memory debug incremental status Command

The following example shows output from the **show memory debug incremental** command entered with the **status** keyword:

```
Router# show memory debug incremental status
```

```
Incremental debugging is enabled
Time elapsed since start of incremental debugging: 00:00:10
```

Additional References

The following sections provide references related to Memory Leak Detector.

Related Documents

Related Topic	Document Title
Additional commands: complete command syntax, command mode, defaults, usage guidelines, and examples	The <i>Cisco IOS Configuration Fundamentals and Network Management Command Reference</i> , Release 12.3 T

Standards

Standards	Title
None	—

MIBs

MIBs	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
None	—

Technical Assistance

Description	Link
The Cisco Technical Support & Documentation website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Command Reference

This section documents modified commands only.

- [set memory debug incremental starting-time](#)
- [show memory debug incremental](#)
- [show memory debug leaks](#)

set memory debug incremental starting-time

To set the current time as the starting time for incremental analysis, use the **set memory debug incremental starting-time** command in privileged EXEC mode.

set memory debug incremental starting-time [none]

Syntax Description	none (Optional) Resets the defined start time for incremental analysis.
---------------------------	--

Defaults	No default behavior or values.
-----------------	--------------------------------

Command Modes	Privileged EXEC
----------------------	-----------------

Command History	Release	Modification
	12.3(8)T1	This command was introduced.
	12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
	12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines	For incremental analysis, a starting point can be defined by using the set memory debug incremental starting-time command. When a starting time is set, only memory allocated after that starting time will be considered for reporting as leaks.
-------------------------	--

Examples	The following example shows the command used to set the starting time for incremental analysis to the time when the command was issued:
-----------------	---

```
Router# set memory debug incremental starting-time
```

Related Commands	Command	Description
	show memory debug incremental allocation	Displays all memory blocks that were allocated after the issue of the set memory debug incremental starting-time command.
	show memory debug incremental leaks	Displays only memory that was leaked after the issue of the set memory debug incremental starting-time command.
	show memory debug incremental leaks lowmem	Forces incremental memory leak detection to work in low memory mode. Displays only memory that was leaked after the issue of the set memory debug incremental starting-time command.
	show memory debug incremental status	Displays if the starting point of incremental analysis has been defined and the time elapsed since then.
	show memory debug leaks	Displays detected memory leaks.

show memory debug incremental

To display information about memory leaks after a starting time has been established, use the **show memory debug incremental** command in privileged EXEC mode.

```
show memory debug incremental {allocations / leaks [lowmem] / status}
```

Syntax Description		
	allocations	Displays all memory blocks that were allocated after issuing the set memory debug incremental starting-time command.
	leaks	Displays only memory that was leaked after issuing the set memory debug incremental starting-time command.
	lowmem	(Optional) Forces the memory leak detector to work in low memory mode, making no memory allocations.
	status	Displays all memory blocks that were allocated after issuing the set memory debug incremental starting-time command.

Command Modes	
	Privileged EXEC

Command History	Release	Modification
	12.3(7)T	This command was introduced.
	12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

The **show memory debug incremental allocations** command displays all the memory blocks that were allocated after the issue of a **set memory debug incremental starting-time** command. The displayed memory blocks are just memory allocations, they are not necessarily leaks.

The **show memory debug incremental leaks** command provides output similar to the **show memory debug leaks** command, except that it displays only memory that was leaked after the issue of a **set memory debug incremental starting-time** command.

The **show memory debug incremental leaks lowmem** command forces memory leak detection to work in low memory mode. The amount of time taken for analysis is considerably greater than that of normal mode. The output for this command is similar to the **show memory debug leaks** command, except that it displays only memory that was leaked after the issue of a **set memory debug incremental starting-time** command. You can use this command when you already know that normal mode memory leak detection will fail (perhaps by an unsuccessful previous attempt to invoke normal mode memory leak detection).

The **show memory debug incremental status** command displays whether a starting point for incremental analysis has been set and the elapsed time since then.



Note

All memory leak detection commands invoke normal mode memory leak detection, except when the low memory option is specifically invoked by use of the **lowmem** keyword. In normal mode, if memory leak detection determines that there is insufficient memory to proceed in normal mode, it will display an appropriate message and switch to low memory mode.

Examples**show memory debug incremental allocations Command Example**

The following example shows output from the **show memory debug incremental** command when entered with the **allocations** keyword:

```
Router# show memory debug incremental allocations
```

```
Address      Size  Alloc_pc  PID  Name
62DA4E98    176  608CDC7C  44   CDP Protocol
62DA4F48     88  608CCCC8  44   CDP Protocol
62DA4FA0     88  606224A0  3     Exec
62DA4FF8     96  606224A0  3     Exec
635BF040     96  606224A0  3     Exec
63905E50    200  606A4DA4  69   Process Events
```

show memory debug incremental status Command Example

The following example shows output from the **show memory debug incremental** command entered with the **status** keyword:

```
Router# show memory debug incremental status
```

```
Incremental debugging is enabled
Time elapsed since start of incremental debugging: 00:00:10
```

Related Commands

Command	Description
set memory debug incremental starting-time	Sets the current time as the starting time for incremental analysis.
show memory debug leaks	Displays detected memory leaks.

show memory debug leaks

To display detected memory leaks, use the **show memory debug leaks** command in privileged EXEC mode.

show memory debug leaks [**chunks** | **largest** | **lowmem** | **summary**]

Syntax Description		
chunks	(Optional)	Displays the memory leaks in chunks.
largest	(Optional)	Displays the top ten leaking allocator_pcs based on size, and the total amount of memory they have leaked.
lowmem	(Optional)	Forces the memory leak detector to work in low memory mode, making no memory allocations.
summary	(Optional)	Reports summarized memory leaks based on allocator_pc and size of the memory block.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.3(8)T1	This command was introduced.
	12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
	12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines If no optional keywords are specified, the **show memory debug leaks** command invokes normal mode memory leak detection and does not look for memory leaks in chunks.

The **show memory debug leaks chunks** command invokes normal mode memory leak detection and looks for leaks in chunks as well.

The **show memory debug leaks largest** command displays the top ten leaking allocator_pcs and the total amount of memory that they have leaked. Additionally, each time this command is invoked it remembers the previous invocation's report and compares it to the current invocation's report. If there are new entries in the current report they are tagged as "inconclusive." If the same entry appears in the previous invocation's report and the current invocation's report, the inconclusive tag is not added. It would be beneficial to run memory leak detection more than once and to consider only the consistently reported leaks.

The **show memory debug leaks lowmem** command forces memory leak detection to work in low memory mode. The amount of time taken for analysis is considerably greater than that of normal mode. The output for this command is similar to the **show memory debug leaks** command. You can use this command when you already know that normal mode memory leak detection will fail (perhaps by an unsuccessful previous attempt to invoke normal mode memory leak detection).

The **show memory debug leaks summary** command reports memory leaks based on allocator_pc and then on the size of the block.

**Note**

All memory leak detection commands invoke normal mode memory leak detection, except when the low memory option is specifically invoked by use of the **lowmem** keyword. In normal mode, if memory leak detection determines that there is insufficient memory to proceed in normal mode, it will display an appropriate message and switch to low memory mode.

Examples**show memory debug leaks Command Example**

The following example shows output from the **show memory debug leaks** command:

```
Router# show memory debug leaks

Adding blocks for GD...

          PCI memory
Address   Size   Alloc_pc  PID  Name

          I/O memory
Address   Size   Alloc_pc  PID  Name

          Processor memory
Address   Size   Alloc_pc  PID  Name
62DABD28    80 60616750  -2  Init
62DABD78    80 606167A0  -2  Init
62DCF240    88 605B7E70  -2  Init
62DCF298    96 605B7E98  -2  Init
62DCF2F8    88 605B7EB4  -2  Init
62DCF350    96 605B7EDC  -2  Init
63336C28   104 60C67D74  -2  Init
63370D58    96 60C656AC  -2  Init
633710A0   304 60C656AC  -2  Init
63B2BF68    96 60C659D4  -2  Init
63BA3FE0  32832 608D2848  104  Audit Process
63BB4020  32832 608D2FD8  104  Audit Process
```

[Table 4](#) describes the significant fields shown in the display.

Table 4 *show memory debug leaks Field Descriptions*

Field	Description
Address	Hexadecimal address of the leaked block.
Size	Size of the leaked block (in bytes).
Alloc_pc	Address of the system call that allocated the block.
PID	The process identifier of the process that allocated the block.
Name	The name of the process that allocated the block.

show memory debug leaks chunks Command Example

The following example shows output from the **show memory debug leaks chunks** command:

```
Router# show memory debug leaks chunks

Adding blocks for GD...

          PCI memory
Address   Size   Alloc_pc  PID  Name
```

```

Chunk Elements:
Address  Size  Parent  Name

          I/O memory
Address  Size  Alloc_pc  PID  Name

Chunk Elements:
Address  Size  Parent  Name

          Processor memory
Address  Size  Alloc_pc  PID  Name
62DABD28      80 60616750  -2  Init
62DABD78      80 606167A0  -2  Init
62DCF240      88 605B7E70  -2  Init
62DCF298      96 605B7E98  -2  Init
62DCF2F8      88 605B7EB4  -2  Init
62DCF350      96 605B7EDC  -2  Init
63336C28     104 60C67D74  -2  Init
63370D58      96 60C656AC  -2  Init
633710A0     304 60C656AC  -2  Init
63B2BF68      96 60C659D4  -2  Init
63BA3FE0    32832 608D2848  104  Audit Process
63BB4020    32832 608D2FD8  104  Audit Process

Chunk Elements:
Address  Size  Parent  Name
62D80DA8     16 62D7BFD0 (Managed Chunk )
62D80DB8     16 62D7BFD0 (Managed Chunk )
62D80DC8     16 62D7BFD0 (Managed Chunk )
62D80DD8     16 62D7BFD0 (Managed Chunk )
62D80DE8     16 62D7BFD0 (Managed Chunk )
62E8FD60    216 62E8F888 (IPC Message He)

```

Table 5 describes the significant fields shown in the display.

Table 5 *show memory debug leaks chunks Field Descriptions*

Field	Description
Address	Hexadecimal address of the leaked block.
Size	Size of the leaked block (in bytes).
Alloc_pc	Address of the system call that allocated the block.
PID	The process identifier of the process that allocated the block.
Name	The name of the process that allocated the block.
Size	(Chunk Elements) Size of the leaked element (bytes).
Parent	(Chunk Elements) Parent chunk of the leaked chunk.
Name	(Chunk Elements) The name of the leaked chunk.

show memory debug leaks largest Command Example

The following example shows output from the **show memory debug leaks largest** command:

```

Router# show memory debug leaks largest

Adding blocks for GD...

          PCI memory
Alloc_pc  total leak size

```

```

                I/O memory
Alloc_pc      total leak size

                Processor memory
Alloc_pc      total leak size
608D2848     32776      inconclusive
608D2FD8     32776      inconclusive
60C656AC     288        inconclusive
60C67D74     48         inconclusive
605B7E98     40         inconclusive
605B7EDC     40         inconclusive
60C659D4     40         inconclusive
605B7E70     32         inconclusive
605B7EB4     32         inconclusive
60616750     24         inconclusive

```

The following example shows output from the second invocation of the **show memory debug leaks largest** command:

```
Router# show memory debug leaks largest
```

```
Adding blocks for GD...
```

```

                PCI memory
Alloc_pc      total leak size

                I/O memory
Alloc_pc      total leak size

                Processor memory
Alloc_pc      total leak size
608D2848     32776
608D2FD8     32776
60C656AC     288
60C67D74     48
605B7E98     40
605B7EDC     40
60C659D4     40
605B7E70     32
605B7EB4     32
60616750     24

```

show memory debug leaks summary Command Example

The following example shows output from the **show memory debug leaks summary** command:

```
Router# show memory debug leaks summary
```

```
Adding blocks for GD...
```

```

                PCI memory

Alloc PC      Size      Blocks      Bytes      What

                I/O memory

Alloc PC      Size      Blocks      Bytes      What

                Processor memory

Alloc PC      Size      Blocks      Bytes      What

```

```

0x605B7E70 0000000032 0000000001 0000000032 Init
0x605B7E98 0000000040 0000000001 0000000040 Init
0x605B7EB4 0000000032 0000000001 0000000032 Init
0x605B7EDC 0000000040 0000000001 0000000040 Init
0x60616750 0000000024 0000000001 0000000024 Init
0x606167A0 0000000024 0000000001 0000000024 Init
0x608D2848 0000032776 0000000001 0000032776 Audit Process
0x608D2FD8 0000032776 0000000001 0000032776 Audit Process
0x60C656AC 0000000040 0000000001 0000000040 Init
0x60C656AC 0000000248 0000000001 0000000248 Init
0x60C659D4 0000000040 0000000001 0000000040 Init
0x60C67D74 0000000048 0000000001 0000000048 Init

```

Table 6 describes the significant fields shown in the display.

Table 6 *show memory debug leaks summary Field Descriptions*

Field	Description
Alloc_pc	Address of the system call that allocated the block.
Size	Size of the leaked block.
Blocks	Number of blocks leaked.
Bytes	Total amount of memory leaked.
What	Name of the process that owns the block.

Related Commands

Command	Description
set memory debug incremental starting-time	Sets the current time as the starting time for incremental analysis.
show memory debug incremental allocation	Displays all memory blocks that were allocated after the issue of the set memory debug incremental starting-time command.
show memory debug incremental leaks	Displays only memory that was leaked after the issue of the set memory debug incremental starting-time command.
show memory debug incremental leaks lowmem	Forces incremental memory leak detection to work in low memory mode. Displays only memory that was leaked after the issue of the set memory debug incremental starting-time command.
show memory debug incremental status	Displays if the starting point of incremental analysis has been defined and the time elapsed since then.

CCVP, the Cisco logo, and Welcome to the Human Network are trademarks of Cisco Systems, Inc.; Changing the Way We Work, Live, Play, and Learn is a service mark of Cisco Systems, Inc.; and Access Registrar, Aironet, Catalyst, CCDA, CCDP, CCIE, CCIP, CCNA, CCNP, CCSP, Cisco, the Cisco Certified Internetwork Expert logo, Cisco IOS, Cisco Press, Cisco Systems, Cisco Systems Capital, the Cisco Systems logo, Cisco Unified Enterprise/Solver, EtherChannel, EtherFast, EtherSwitch, Fast Step, Follow Me Browsing, FormShare, GigaDrive, HomeLink, Internet Quotient, iPod, iPhone, IP/TV, iQ Expertise, the iQ logo, iQ Net Readiness Scorecard, iQuick Study, LightStream, Linksys, MeetingPlace, MGX, Network Networking Academy, Network Registrar, PIX, ProConnect, ScriptShare, SMARTnet, StackWise, The Fastest Way to Increase Your Internet Quotient, and TransPath are registered trademarks of Cisco Systems, Inc. and/or its affiliates in the United States and certain other countries.

All other trademarks mentioned in this document or Website are the property of their respective owners. The use of the word partner does not imply partnership relationship between Cisco and any other company. (0711R)

Any Internet Protocol (IP) addresses used in this document are not intended to be actual addresses. Any examples, command display output, and figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses in illustrative content is unintentional and coincidental.

© 2004, 2006 Cisco Systems, Inc. All rights reserved.