

Introduction

Shinken supports a feature that does **freshness** checking on the results of host and checks. The purpose of freshness checking is to ensure that host and service checks are being provided passively by external applications on a regular basis.

Freshness checking is useful when you want to ensure that passive checks are being received as frequently as you want. This can be very useful in distributed and failover monitoring environments.

How Does Freshness Checking Work?

Shinken periodically checks the freshness of the results for all hosts services that have freshness checking enabled.

- A freshness threshold is calculated for each host or check.
- For each host/service, the age of its last check result is compared with the freshness threshold.
- If the age of the last check result is greater than the freshness threshold, the check result is considered "**stale**".
- If the check results is found to be stale, Shinken will force an active check of the host or check by executing the command specified by in the host or check definition.

An active check is executed even if active checks are disabled on a program-wide or host- or check-specific basis.

For example, if you have a freshness threshold of 60 for one of your checks, Shinken will consider that check to be stale if its last check result is older than 60 seconds.

Enabling Freshness Checking

Here's what you need to do to enable freshness checking.

- Enable freshness checking on a program-wide basis with the `check_service_freshness` and `check_host_freshness` directives.
- Use `service_freshness_check_interval` and `host_freshness_check_interval` options to tell Shinken how often it should check the freshness of service and host results.
- Enable freshness checking on a host- and check-specific basis by setting the "`check_freshness`" option in your host and service definitions to a value of 1.
- Configure freshness thresholds by setting the **freshness threshold** option in your host and check definitions.
- Configure the **check command** option in your host or check definitions to reflect a valid command that should be used to actively check the host or service when it is detected as stale.
- The **check period** option in your host and check definitions is used when Shinken determines when a host or check can be checked for freshness, so make sure it is set to a valid timeperiod.

If you do not specify a host- or check-specific **freshness threshold** value (or you set it to zero), Shinken will automatically calculate a threshold automatically, based on a how often you monitor that particular host or service. I would recommended that you explicitly specify a freshness threshold, rather than let Shinken pick one for you.

Example

An example of a check that might require freshness checking might be one that reports the status of your nightly backup jobs. Perhaps you have a external script that submit the results of the backup job to Shinken once the backup is completed. In this case, all of the checks/results for the service are provided by an external application using passive checks. In order to ensure that the status of the backup job gets reported every day, you may want to enable freshness checking for the check. If the external script doesn't submit the results of the backup job, you can have Shinken fake a critical result by doing something like this.

Here's what the definition for the check looks like:

Property	Value	Note
Description	Backup Job	
Active checks enabled	False	Active checks are NOT enabled
Passive checks enabled	True	Passive checks are enabled (this is how results are reported)

Check freshness	True	
Freshness threshold	93600	26 hour threshold, since backups may not always finish at the same time
Check command	no-backup-report	This command is run only if the service results are "stale"

Notice that active checks are disabled. This is because the results for the check are only made by an external application using passive checks. Freshness checking is enabled and the freshness threshold has been set to 26 hours. This is a bit longer than 24 hours because backup jobs sometimes run late from day to day (depending on how much data there is to backup, how much network traffic is present, etc.). The **no-backup-report** command is executed only if the results of the service are determined to be stale. The definition of the **no-backup-report** command might look like this...

Property	Value
Name	no-backup-report
Command Line	/var/lib/shinken/libexec/check_dummy 2 "CRITICAL: Results of backup job were not reported!"

If Shinken detects that the service results are stale, it will run the **no-backup-report** command as an active check. This causes the **check_dummy** plugin to be executed, which returns a critical state to Shinken. The check will then go into a Critical state (if it isn't already there) and someone will probably get notified of the problem.