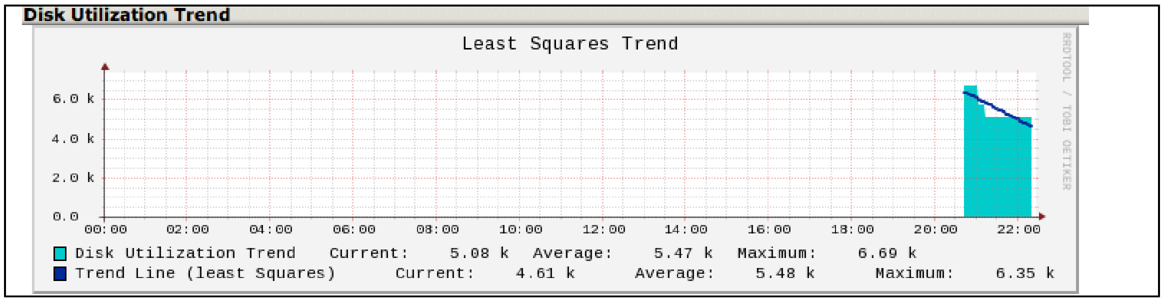
Setting up Trend Graphs

Background

GroundWork includes a plugin called check_rrd_ls.pl for calculating, checking, and displaying trends on RRD-based performance data. This article describes how to use the plugin, and provides a small sample profile. It also describes how to set up thresholds to get alarms when the predicted value is out of a specified threshold range. This can be VERY USEFUL when trying to PREDICT and AVOID outages!

Here is a sample of the graphing output:



```
Disk Utilization Trend    Current:     5.08 k   Average:     5.47 k   Maximum:     6.69 k
Trend Line (least Squares)       Current:     4.61 k       Average:     5.48 k       Maximum:     6.35 k
```

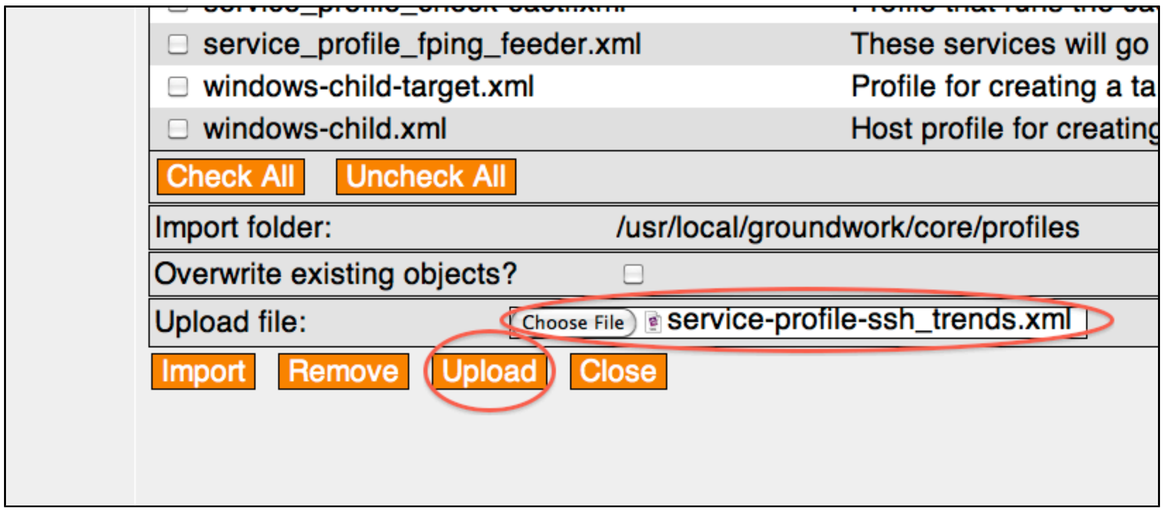
Here is the plugin help text, for refernce:

```
# /usr/local/groundwork/nagios/libexec/check_rrd_ls.pl --help
check_rrd_ls v$Revision: 1.0 $ (nagios-plugins 1.4.15)
The nagios plugins come with ABSOLUTELY NO WARRANTY. You may redistribute
copies of the plugins under the terms of the GNU General Public License.
For more information about these matters, see the file named COPYING.
Copyright (c) 2004 Thomas Stocking

Perl RRD Trend check plugin for Nagios

Usage: check_rrd_ls
        [-r ] (Reference RRD name and path)
        [-R ] (Reference data store name)
        [-w <low:high>] (actual warning threshold)
        [-W ] (Trend Warning - minutes)
        [-c <low:high>] (actual critical threshold - must be outside -w range)
        [-C]  (Trend Critical - minutes. Must be higher than -W)
        [-p] (constrain to positive real values)
        [-i] (Trend interval to consider - minutes)
        [-G] (Graphing RRD toggle)
        [-D] (debug) [-h] (help) [-V] (Version)

-r, --reference_rrd
   The RRD which you want to check for trends (required)
-R, --reference_data_store
   Data store in reference rrd to check (required)
-i, --interval
   Time in minutes to consider for trending (default 60)
-w, --warning
   Actual warning threshold <low:high>
-W, --trend_warning
   Time threshold for predicted warnings (in minutes). Set this to the amount of time within which a
predicted warning will result in a warning state.
-c, --critical
        Actual critical threshold <low:high>
-C, --trend_critical
   Time threshold for predicted critical errors (in minutes). Set this to the amount of time within
which a predicted critical error will result in a critical state.
-p, --positive
   Set if you want to restrict data range considered to positive real values.
-D, --debug
   Turn on debugging.  (Verbose)
-h, --help
   Print help
-V, --Version
   Print version of plugin
```

Installation

The plugin is included, in the folder /usr/local/groundwork/nagios/libexec.

Import the profile xml by using the Profile Importer. You may use it to upload the xml as well as import it, and we suggest you also upload the perfconfig.xml files, as the graphs are a bit tricky to set up the first time. To do this, download the attached xml files to your hard disk, and then navigate to the Configuration-Profiles-Profile Importer. At the bottom of the right-hand panel, select the XML files one by one and Upload them:



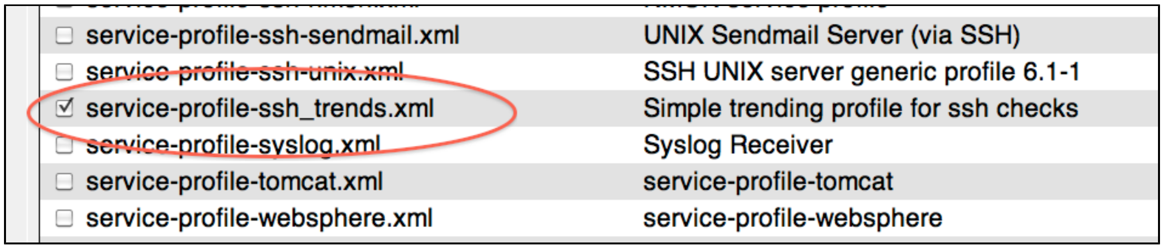
☐ service_profile_fping_feeder.xml	These services will go
☐ windows-child-target.xml	Profile for creating a ta
☐ windows-child.xml	Host profile for creating

Import folder: /usr/local/groundwork/core/profiles

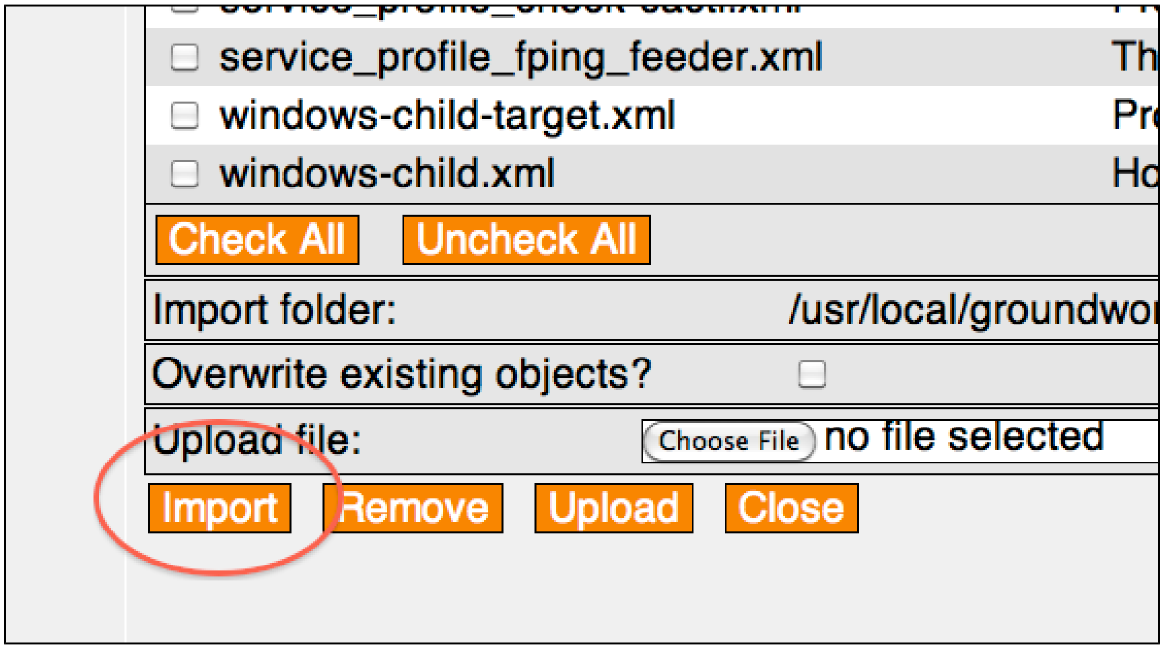
Overwrite existing objects? ☐

Upload file: Choose File service-profile-ssh_trends.xml

Select the service-profile-ssh_trends.xml and import:



☐ service-profile-ssh-sendmail.xml	UNIX Sendmail Server (via SSH)
☐ service-profile-ssh-unix.xml	SSH UNIX server generic profile 6.1-1
☑ service-profile-ssh_trends.xml	Simple trending profile for ssh checks
☐ service-profile-syslog.xml	Syslog Receiver
☐ service-profile-tomcat.xml	service-profile-tomcat
☐ service-profile-websphere.xml	service-profile-websphere



☐ service_profile_fping_feeder.xml	Th
☐ windows-child-target.xml	Pr
☐ windows-child.xml	Ho

Import folder: /usr/local/groundwo

Overwrite existing objects? ☐

Upload file: Choose File no file selected

Using the Plugin

Now you can trend any RRD that exists on the disk. Ususally, this means looking at an RRD that has been set up by an existing service, like "ssh_disk_root", which uses a plugin to check disk space on the root partition in Linux systems. The main service does the check, and creates the RRD, with the datasouce name "root". The supplied profile specifies this as an argument. There are some default thresholds supplied as well, which may or may not make sense for your disk check. Let's break the arguments down, so you can adjust them. Here's the command line:

```
check_rrd_least_squares!ssh_disk_root!root!120!3500!6000!1440!1800
```

- The first argument is ssh_disk_root, which is the service that generates the RRD we are going to check on this host. This is used by the plugin to look for the RRD to check, as:

  ```
  /usr/local/groundwork/rrd/$HOSTNAME$_$ARG1$.rrd
  ```

  note
  Note: This means you can check any rrd on the disk, even one created by another tool. Just adjust the command definition to look at an RRD in another location.

- The second argument is the DS name to trend in the RRD. "root" may not be right in your case, and you should be sure this matches the DS you want. There can be more than one DS in an RRD, but only one can be trended with this plugin.
  You can see what DS names are present in an rrd with the command:

  ```
  rrdtool info <rrdname>
  ```

  Don't forget to load the GroundWork Environment before trying to use this command at the command line on your groundwork server (source /usr/local/groundwork/scripts/setenv.sh)

- The third argument is the interval to consider in minutes. 120 means 2 hours. You might want to set this higher if your data is noisy. For disk space, 120 is probably fine. The larger your interval, the longer the plugin will take, but not by much.

- The 4th and 5th arguments are the normal warning and critical thresholds - these should probably match what you set on the source service, but you may want them tighter, to increase the probability of getting trend alarms. If desired, you can have these specified as low:high pairs. If they are set as bare numbers, as they are here, they will be considered high thresholds, and critical should be above warning.

  Note
  These numbers are in relation to the metric as stored in the DS. That may or may not be the same as the numbers used in the original service as thresholds, since performance data may be collected and graphed in different units of measure than is used for thresholds.

- The 6th and 7th arguments are the the Trend Warning and Trend Critical thresholds. These thresholds are TIME-based. This means that you will get a warning if the *predicted value* at the Trend warning threshold time in the future is above the Warning (argument 4) level. If it's above the warning and not above the critical threshold, the result will be a warning. If it's above the critical threshold, it will also be critical, since this will violate the later critical threshold.
  In this example, if the predicted value is over 3500 and below 6000 at 1400 minutes (1 day) from now, the result will be a warning. If it is over 6000, it will be critical if it is predicted to happen before 1800 minutes (30 hours).

Note that the plugin will take two more arguments:
-p This makes the metric considered restricted to positive numbers. This helps to have the thresholds make sense in this case, since otherwise the trending needs to accept negative thresholds.
-G This makes the plugin generate a temporary rrd, with the same name as the reference rrd with _graph.rrd appended to the file name. This ancillary RRD is used in graphing.

Graphing the trend

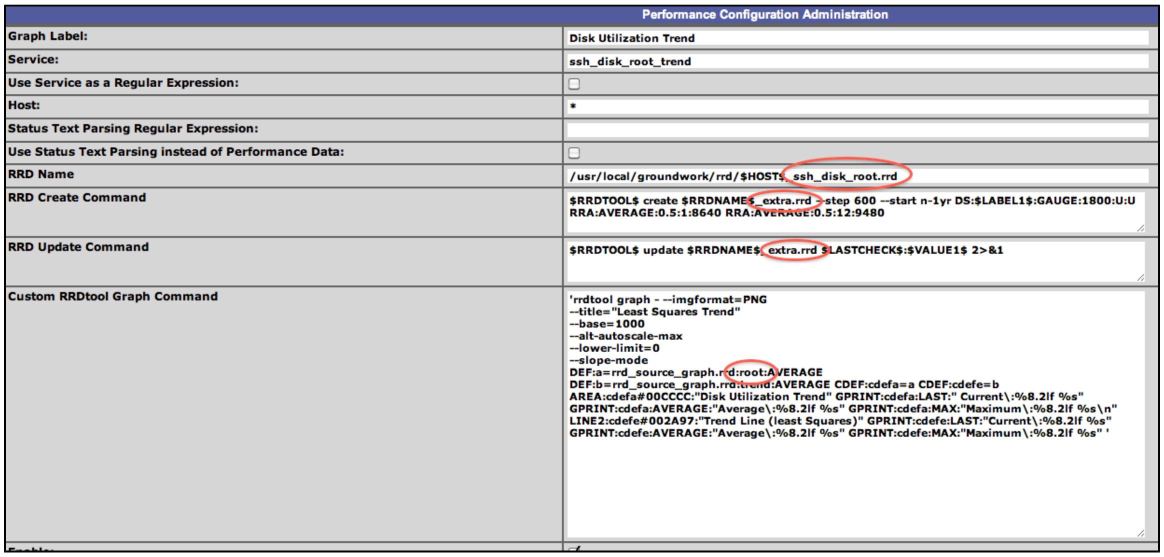
The RRD that the plugin produces with the -G option can be used for graphing, but the graph command needs to specify this special RRD. If you are not familiar with the custom RRD graph commands, you might want to review the bookshelf section called "Home > USING APPLICATIONS > Configuration > Configuration Scenarios > Creating Performance Graphs". This explains the various available macros and substitutions available in this section. This example will work as imported, but you will probably want to extend it. Here are exact instructions:

- We will use the "rrd_source" macro, which is replaced by the RRD pat specified in the RRD Name parameter. That means that we don't need to specify the host name, and can apply this same performance graphing configuration to the same service on multiple hosts.

- To do this, we just adjust the name of the RRD to be the name of the host, combined with the names of the original base service for the RRD we are trending.
  That is:
  RRD Name is "/usr/local/groundwork/rrd/$HOST$_ssh_disk_root.rrd"
  In our example, we removed the "$SERVICE$" macro from the default RRD name, and put in the original service that is the source of the trend data: (ssh_disk_root).

- Then the rrd_source macro in the Customer Graph Command is changed to be the full path to the *special* RRD created by the -G option on the plugin: (rrd_source_graph.rrd)
- In our example we have "root" specified in the custom graph command as the first DS to graph, along with the "trend" DS. These are generated with by the plugin. If you decide to use a different DS name, say, for a different service, match this with the argument you give the plugin for the DS to get trending data for - it will create the graphing RRD using that name.
- Finally, we need to change the name of the RRD we are creating and updating to avoid self-refernceing problems. The RRD that is created for the trend service itself is not used for anything, but is crated normally and updated with the plugin output, which is just the last value harvested from the source RRD. Having this happen without errors allows the processing to go through smoothly, and avoids errors in the log file that might otherwise make you think there is a problem. To do this, add a string like "extra.rrd" to the path in the rrd create and update commands.

This is what the perfdata configuration looks like in our example:



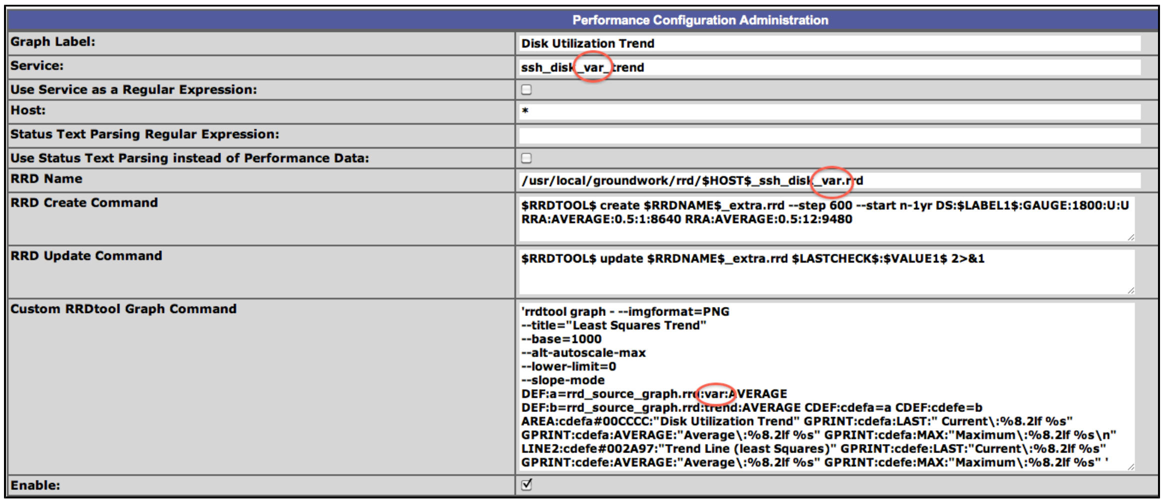
Performance Configuration Administration	
Graph Label:	Disk Utilization Trend
Service:	ssh_disk_root_trend
Use Service as a Regular Expression:	☐
Host:	*
Status Text Parsing Regular Expression:	
Use Status Text Parsing instead of Performance Data:	☐
RRD Name	/usr/local/groundwork/rrd/$HOST$_ssh_disk_root.rrd
RRD Create Command	$RRDTOOL$ create $RRDNAME$_extra.rrd --step 600 --start n-1yr DS:$LABEL1$:GAUGE:1800:U:U RRA:AVERAGE:0.5:1:8640 RRA:AVERAGE:0.5:12:9480
RRD Update Command	$RRDTOOL$ update $RRDNAME$_extra.rrd $LASTCHECK$:$VALUE1$ 2>&1
Custom RRDtool Graph Command	see below

```
'rrdtool graph - --imgformat=PNG
--title="Least Squares Trend"
--base=1000
--alt-autoscale-max
--lower-limit=0
--slope-mode
DEF:a=rrd_source_graph.rrd:root:AVERAGE
DEF:b=rrd_source_graph.rrd:trend:AVERAGE CDEF:cdefa=a CDEF:cdefe=b
AREA:cdefa#00CCCC:"Disk Utilization Trend" GPRINT:cdefa:LAST:" Current\:%8.2lf %s"
GPRINT:cdefa:AVERAGE:"Average\:%8.2lf %s" GPRINT:cdefa:MAX:"Maximum\:%8.2lf %s\n"
LINE2:cdefe#002A97:"Trend Line (least Squares)" GPRINT:cdefe:LAST:"Current\:%8.2lf %s"
GPRINT:cdefe:AVERAGE:"Average\:%8.2lf %s" GPRINT:cdefe:MAX:"Maximum\:%8.2lf %s" '
```

Once you have this working, you can copy the configuration and create more trending services, adjusting the "ssh_disk_root" and "root" strings to match the new services you make. For example, say you want to trend disk usage on the "var" partition. You will need a source service like "ssh_disk_var", which creates an RRD with the DS named "var".
Then you can create a service "ssh_disk_var_trend", with a command line like:

```
check_rrd_least_squares!ssh_disk_root!var!120!3500!6000!1440!1800
```

Then copy the ssh_disk_root_trend performance config, and make it look like this:



Performance Configuration Administration	
Graph Label:	Disk Utilization Trend
Service:	ssh_disk_var_trend
Use Service as a Regular Expression:	☐
Host:	*
Status Text Parsing Regular Expression:	
Use Status Text Parsing instead of Performance Data:	☐
RRD Name	/usr/local/groundwork/rrd/$HOST$_ssh_disk_var.rrd
RRD Create Command	$RRDTOOL$ create $RRDNAME$_extra.rrd --step 600 --start n-1yr DS:$LABEL1$:GAUGE:1800:U:U RRA:AVERAGE:0.5:1:8640 RRA:AVERAGE:0.5:12:9480
RRD Update Command	$RRDTOOL$ update $RRDNAME$_extra.rrd $LASTCHECK$:$VALUE1$ 2>&1
Custom RRDtool Graph Command	see below
Enable:	✓

```
'rrdtool graph - --imgformat=PNG
--title="Least Squares Trend"
--base=1000
--alt-autoscale-max
--lower-limit=0
--slope-mode
DEF:a=rrd_source_graph.rrd:var:AVERAGE
DEF:b=rrd_source_graph.rrd:trend:AVERAGE CDEF:cdefa=a CDEF:cdefe=b
AREA:cdefa#00CCCC:"Disk Utilization Trend" GPRINT:cdefa:LAST:" Current\:%8.2lf %s"
GPRINT:cdefa:AVERAGE:"Average\:%8.2lf %s" GPRINT:cdefa:MAX:"Maximum\:%8.2lf %s\n"
LINE2:cdefe#002A97:"Trend Line (least Squares)" GPRINT:cdefe:LAST:"Current\:%8.2lf %s"
GPRINT:cdefe:AVERAGE:"Average\:%8.2lf %s" GPRINT:cdefe:MAX:"Maximum\:%8.2lf %s" '
```

That should get you trending on the usage of the var partition. The same approach can be used for *any* data you want to make trends for.

Files

Here are the 3 files, referenced above. The original service "ssh_disk_root" and the trend service "ssh_disk_root_trend" are included, as is their respective performance data definitions. This pair of service should be a good template for you to work from in setting up other services. Happy trending!

Name	Size	Creator	Creation Date	Comment
service-profile-ssh_disk_trends.xml	4 kB	Thomas Stocking	Apr 20, 2011 22:12	
perfconfig_ssh_disk.xml	2 kB	Thomas Stocking	Apr 20, 2011 22:12	
perfconfig_ssh_disk_root_trend.xml	1 kB	Thomas Stocking	Apr 20, 2011 22:12	