Import guide

NConf offers the following data import mechanisms:

- Import existing Nagios configuration files
- Import items from a CSV file

NConf stores its data in a proprietary database, meaning that external data has to be parsed and imported into the NConf data structure in order to make it manageable though NConf.

The import mechanisms are not intended to be used to keep multiple data sources in sync. Importing data is a semi-automatic process. Often, data has to be “tuned” before it can be imported. It can therefore be a tedious process which is intended to be done as a one-time migration.

Nagios import guide

Introduction

This guide is meant to assist users in importing their existing Nagios configurations into NConf. NConf stores its data in a proprietary database, meaning that existing Nagios configuration files have to be parsed and imported into the NConf data structure in order to make them manageable though NConf. NConf does not maintain text files directly. Any changes done through the GUI are first written and stored to NConf's database. You generate and deploy new Nagios config files every time something is updated.

Nagios configurations can be very complex and often contain numerous dependencies between objects. Importing your Nagios configuration can therefore be a tedious process which is intended to be done as a one-time migration.

This guide will give you step by step instructions and will try to make the import process as smooth as possible by trying to prevent most known issues. It does not cover each and every possibility and may therefore require a bit of debugging on the side of the user. We therefore recommend users to keep a copy of their existing Nagios configuration and to do a thorough comparison after the import. Plan for a possible rollback!

Import mechanism

The import mechanism works the following way:

```
  'add_items' script  -->  perl-API  -->  NConf database
```

The 'add_items' script is a perl script which communicates with the database using the NConf perl-API. The script is command-line based. So far, there is no GUI-like web interface for the import process.

As of today, the import script can only import one type of item at the same time (e.g. 'hosts', 'services', 'contacts' etc.). The items must be stored in separate files (e.g. 'hosts.cfg', 'services.cfg', 'contacts.cfg' etc.). Each type of item must be imported separately.

You may break your config down into as many files as you like, but each file may only contain one type of items. The reason for this is that NConf differentiates between several items which Nagios does not (e.g. 'checkcommands' and 'misccommands' are simply 'commands' to Nagios). It is therefore necessary for the user to tell NConf what type of items he is currently importing.

If you have a very complex Nagios configuration with a nested design and you use complex templates or assign multiple items to one object, then you might find it easier to work with the 'objects.cache' file that Nagios generates, instead of the Nagios config files. Please note that the 'objects.cache' file is a 'flattened' version of your config and does not contain templates anymore. As such, your templates will not be imported into NConf. More information on the 'objects.cache' file can be found here [http://nagios.sourceforge.net/docs/3_0/configmain.html#object_cache_file].

The 'add_items' script offers a 'simulation' mode and several loglevels (1-5). **We recommend users to run a simulation before importing any type of item** and to check for any '[WARN]' messages. The simulation mode will not make any changes to the database, but it will detect possible conflicts and/or missing/badly formatted data.

Pre-import tasks

- split your item types into separate files (as described above)
- if you are using parent hosts, nested templates, hostgroups linked to hostgroups or any other inter-object-dependencies, make sure you import your 'parent' items first by putting these into separate files also, or by maintaining the proper order
- remove any sample data that comes with the NConf standard installation and that you do not want to use
- define at least one 'OS' and make sure it's the default OS preselected when adding a host (predefined_value)
- define at least one 'host-preset' and make sure it's the default one preselected when adding a host (predefined_value)
- for simulation purpouses: if there are no items of a certain type yet in NConf, create a 'dummy' entry (e.g. a host called 'AAA') for each type of item you're going to import (this is necessary for proper simulation and can be removed after completing the import)

Importing your configuration

Experience has shown that most conflicts can be avoided by importing your configuration items in the following order:

1. timeperiods
2. misccommands & checkcommands
3. contacts
4. contactgroups (considering that some contactgroups might be linked to others)
5. host-templates
6. parent-hosts, then remaining hosts
7. hostgroups (considering that some hostgroups might be linked to others)

8. host-dependencies
9. service-templates
10. services
11. advanced-services
12. servicegroups (considering that some servicegroups might be linked to others)
13. service-dependencies

If you have services that are assigned to hostgroups or to more than one host, consider importing them as 'advanced-services' instead.

Command syntax

The 'add_items' script is used like this:

```
Usage:
bin/add_items_from_nagios.pl -c class -f /path/to/file [-x (1-5)] [-s]

Help:

  required

  -c  Specify the class of items that you wish to import. Must correspond to an NConf class
      (e.g. "host", "service, "hostgroup", "checkcommand", "contact", "timeperiod"...)

  -f  The path to the file which is to be imported. CAUTION: Make sure you have
      only items of one class in the same file (e.g. "hosts.cfg", "services.cfg"...)
      Also make sure you import host- or service-templates separately ("host" or
      "service" items containing a "name" attribute)

  optional

  -x  Set a custom loglevel (1 = lowest, 5 = most verbose)

  -s  Simulate only. Do not make any actual modifications to the database.
```

Script output will look somewhat like this:

```
$> cd /path/to/nconf/
$> bin/add_items_from_nagios.pl -c service -f /path/to/services.cfg -s

[ Initializing NConf perl-API (library version 0.3, written by A. Gargiulo) ]
[ Copyright (c) 2006 - 2012 Sunrise Communications AG, Zurich, Switzerland  ]

[INFO]  Started executing bin/add_items_from_nagios.pl
[INFO]  Running in simulation mode. No modifications will be made to the database!
[INFO]  Adding service 'testserver-01;;Memory'
[INFO]  Successfully added service 'testserver-01;;Memory'
[INFO]  Finished running bin/add_items_from_nagios.pl
```

To import your items in the order described above, one would have to run the script with the following options (remove the '-s' parameter to do the final import):

```
cd /path/to/nconf/
bin/add_items_from_nagios.pl -c timeperiod -f /path/to/timeperiods.cfg -s
bin/add_items_from_nagios.pl -c misccommand -f /path/to/misccommands.cfg -s
bin/add_items_from_nagios.pl -c checkcommand -f /path/to/checkcommands.cfg -s
bin/add_items_from_nagios.pl -c contact -f /path/to/contacts.cfg -s
bin/add_items_from_nagios.pl -c contactgroup -f /path/to/contactgroups.cfg -s
bin/add_items_from_nagios.pl -c host-template -f /path/to/host_templates.cfg -s
bin/add_items_from_nagios.pl -c host -f /path/to/parent-hosts.cfg -s
bin/add_items_from_nagios.pl -c host -f /path/to/hosts.cfg -s
bin/add_items_from_nagios.pl -c hostgroup -f /path/to/hostgroups.cfg -s
bin/add_items_from_nagios.pl -c host-dependency -f /path/to/host_dependencies.cfg -s
bin/add_items_from_nagios.pl -c service-template -f /path/to/service_templates.cfg -s
bin/add_items_from_nagios.pl -c service -f /path/to/services.cfg -s
bin/add_items_from_nagios.pl -c advanced-service -f /path/to/advanced-services.cfg -s
bin/add_items_from_nagios.pl -c servicegroup -f /path/to/servicegroups.cfg -s
bin/add_items_from_nagios.pl -c service-dependency -f /path/to/service_dependencies.cfg -s
```

Post-import tasks

- remove any 'dummy' (AAA) items you might have added earlier
- edit all checkcommands: set default check parameters, amount of parameters and command description
- add any additional OS you might need and assign them to your hosts using 'multi-modify'
- add any additional host-presets and assign them to your hosts using 'multi-modify'
- make sure all host-presets have a host-alive check assigned to them
- make sure all hosts have a host-preset assigned to them
- add at least one 'collector' server and assign it to your hosts using 'multi-modify' (set 'monitored by' attr)

- make sure all additional options for your collector server(s) are set correctly (templates for active/passive checking assigned etc.)
- if you imported your collector servers as 'hosts', make sure you set the 'host is collector' flag for these hosts appropriately
- add a 'monitor' server, if required
- if required, assign host- & service-templates to your timeperiods, nagios-collectors and nagios-monitors

The import mechanism does not have any functionality to auto-create templates. It can import host- & service-template definitions, but it does not have any logic to determine values used in common and to "magically" create template definitions.

Caution
During import, NConf will compare each attribute in its database with the data in your import files. If it detects an attribute, which is missing in the import file, but which exists within NConf as a non-mandatory attribute, then it will try to use the default value defined within NConf. In most situations this behavior is OK, but it may sometimes also lead to unwanted settings in your config.
To trace these kind of events, the import process must be run at **loglevel 4**. For better visibility, run the import and pipe output through the following grep: *grep -E "INFO|WARN|Attribute .* missing"*

Debugging

During import you will most likely encounter several warning or error messages. Here is a list of the most common messages and what causes them:

```
[WARN]  Mandatory attribute '...' missing for '...' Using default value: '...'
```

This message means that NConf expects a mandatory attribute, which is not in the config file you are importing. This is not a critical error, since missing attributes can be set after the import. Some attributes are very likely to be missing in your files, because NConf might expect them a little differently (e.g. NConf might store 'max_check_attempts' in a timeperiod, whereas Nagios expects this to be specified within a host). There are also attributes which are NConf-specific, and therefore not found in any Nagios files. NConf will always try to set the default value for these attributes. Keep track of these warnings, but don't panic if you get them.

```
[WARN]  Mandatory attribute '...' missing for '...'.
[ERROR] Failed to add '...'. Aborting
```

This message means that NConf expects a mandatory Nagios-specific attribute, which is not in the config file you are importing. These attributes are necessary for proper operation and cannot be omitted (e.g. the 'host_name'). This error is considered a critical error and the import process will terminate in order to maintain proper data consistency.

```
[WARN]  Could not find attribute '...' belonging to class '...'. Skipping import of this attribute.
```

This message means that the config file you are trying to import contains an attribute which is unknown to NConf, or that is specified elsewhere (e.g. the attribute 'active_checks_enabled' will not be added directly to a host, because it is either defined globally for all hosts monitored by a specific 'collector', or it is defined within a template).

```
[DEBUG] Attribute '...' missing for ... Using default value: '...'.
(run with '-x 4' and do a "|grep 'Attribute .* missing'" to get this)
```

This message means that NConf expects an attribute, which is not in the config file you are importing. The attribute is not defined as mandatory, so NConf will try to use the default value and simply ignore this, because it is not a critical error. It is usually safe to ignore these messages, that's why they are only visible at loglevel 4.

in simulation mode:

```
[WARN]  Failed to resolve attr ID for '...' using getAttrId(). Aborting checkItemsLinked().
[WARN]  Could not link ... with '...' (attr '...'). No such item.
```

This message means that the simulation mode failed to test if it could link two items. This is usually the case when there are no items of the type that you are adding in NConf yet. Since no modifications to the database are allowed in simulation mode, the import function will not be capable to test everything, because it requires a test element of the same type. To avoid these messages simply add a 'dummy' entry (e.g. a host called 'aaa'). You may delete the test entry after completing the import.

not in simulation mode:

```
[WARN]  Could not link host with host '...' (attr 'parents'). No such item.
[WARN]  Could not link service with service '...' (attr 'dependent_service_description'). No such item.
```

This message means that some items could not be linked. This might be the case if a dependency could not be fulfilled because the parent item hasn't been imported yet. This happens frequently with parent hosts. NConf will not import the parent hosts, if they do not exist at the time of the import.

```
[WARN]  Could not link service with checkcommand 'service_is_stale' (attr 'check_command'). No such item.
```

This message means that you are probably trying to import the services.cfg file written for your monitor server. Since no active checking is done there, the services are all linked to the dummy checkcommand 'service_is_stale'. To prevent this error, import the services from one of your collector servers instead.

Known errors

The following errors are very likely to happen when you import certain items. As long as you proceed as indicated in this guide, and perform all post-import tasks, you may safely ignore these errors.

checkcommands:

```
[WARN]  Mandatory attribute 'command_param_count' missing for checkcommand '...'. Using default value: '1'.
[DEBUG] Attribute 'default_params' missing for current checkcommand. Using default value: '!'.
[DEBUG] Attribute 'command_syntax' missing for current checkcommand. Using default value: 'ARG1=...,ARG2=...,ARG3=...'.
```

contacts:

```
[DEBUG] Attribute 'nc_permission' missing for current contact. Using default value: 'user'.
[DEBUG] Attribute 'nagios_access' missing for current contact. Using default value: 'enabled'.
```

hosts:

```
[WARN]  Could not find attribute 'notifications_enabled' belonging to class 'host'. Skipping import of this attribute.
[WARN]  Could not find attribute 'active_checks_enabled' belonging to class 'host'. Skipping import of this attribute.
[WARN]  Could not find attribute 'passive_checks_enabled' belonging to class 'host'. Skipping import of this attribute.
[WARN]  Could not find attribute 'max_check_attempts' belonging to class 'host'. Skipping import of this attribute.
[WARN]  Could not find attribute 'notification_options' belonging to class 'host'. Skipping import of this attribute.
[WARN]  Could not find attribute 'check_command' belonging to class 'host'. Skipping import of this attribute.
[WARN]  Could not find attribute 'notification_interval' belonging to class 'host'. Skipping import of this attribute.
[WARN]  Mandatory attribute 'os' missing for host '...'. Using default value: 'Linux'.
[WARN]  Mandatory attribute 'host-preset' missing for host '...'. Using default value: 'linux-server'.
[WARN]  Mandatory attribute 'host_is_collector' missing for host '...'. Using default value: 'no'.
```

services:

```
[WARN]  Could not find attribute 'retry_check_interval' belonging to class 'service'. Skipping import of this attribute.
[WARN]  Could not find attribute 'check_freshness' belonging to class 'service'. Skipping import of this attribute.
[WARN]  Could not find attribute 'notifications_enabled' belonging to class 'service'. Skipping import of this attribute.
[WARN]  Could not find attribute 'active_checks_enabled' belonging to class 'service'. Skipping import of this attribute.
[WARN]  Could not find attribute 'freshness_threshold' belonging to class 'service'. Skipping import of this attribute.
[WARN]  Could not find attribute 'passive_checks_enabled' belonging to class 'service'. Skipping import of this attribute.
[WARN]  Could not find attribute 'max_check_attempts' belonging to class 'service'. Skipping import of this attribute.
[WARN]  Could not find attribute 'notification_options' belonging to class 'service'. Skipping import of this attribute.
[WARN]  Could not find attribute 'normal_check_interval' belonging to class 'service'. Skipping import of this attribute.
[WARN]  Could not find attribute 'notification_interval' belonging to class 'service'. Skipping import of this attribute.
[DEBUG] Attribute 'check_params' missing for current service. Using default value: '!'.
```

CSV import guide

This page needs to be reviewed. Content might not be up to date anymore.

Introduction

This guide explains how to import objects from a CSV file into NConf. CSV stands for “comma-separated values”, a file containing tabular data, which is separated by commas or semicolons. Data is typically first edited as a spreadsheet in Excel and then exported to CSV format.

NConf comes with two different CSV import scripts in the 'nconf/bin/' directory:

- **add_items_from_csv.pl** lets you freely define the contents of your CSV file and imports any type of items
- **add_items_from_special_csv.pl** dictates a specific CSV file structure and only imports hosts & services

Both scripts will process your CSV input file and create new items in NConf accordingly. Please note that you can only import one type of item at once, e.g. you cannot import hosts and contacts at the same time. That would require two different CSV files (the second script does let you import hosts & services at once). The order of the attributes in your input file is irrelevant.

If you wish to import a field that has to be selected from a list (drop-down) in NConf, then you must make sure the value you're importing actually exists for that field, otherwise the import mechanism will not be able to map your data.

Example: If you wish to set the “notification_period” of an item to “24×7”, you must make sure a timeperiod with that name actually exists in NConf.

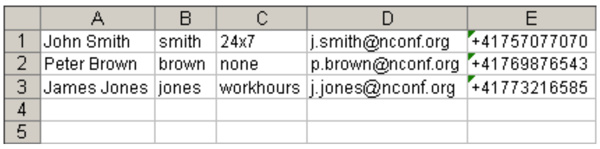
Importing from a user-defined CSV file

Say for example you were trying to import contacts from a CSV file. The **add_items_from_csv.pl** script lets you define the structure of your CSV file as you like. The only restriction is that your CSV file must contain all attributes that are mandatory for that object type in NConf.

The mandatory attributes for 'contact' items in NConf are:

- contact_name
- alias
- email

If you're working with a spreadsheet tool, your sheet would look somewhat like this:



	A	B	C	D	E
1	John Smith	smith	24x7	j.smith@nconf.org	+41757077070
2	Peter Brown	brown	none	p.brown@nconf.org	+41769876543
3	James Jones	jones	workhours	j.jones@nconf.org	+41773216585
4					
5					

After exporting your spreadsheet to CSV format, or if you're working with a text editor, the data would look somewhat like this:

```
John Smith;smith;24x7;j.smith@nconf.org;+41757077070
Peter Brown;brown;none;p.brown@nconf.org;+41769876543
James Jones;jones;workhours;j.jones@nconf.org;+41773216585
```

As you can see, all three mandatory attributes and more are available, assuming “alias” is the real name and “contact_name” is the user id.

Next, we need to define the syntax of the CSV file, so that the import script knows how to map the data. This is done by editing the **add_items_from_csv.pl** script (make a backup copy first).

Open the script with a text editor. On one of the first lines you will find a set of commands looking like this:

```
push(@csv_syntax, ...
```

This is where we tell the import script how the structure of the CSV file looks. The idea is to tell the script which column (from left to right) corresponds to what attribute in NConf. Uncomment or remove any “push” commands that you do not need.

For the example above, the necessary syntax would look like this:

```
push(@csv_syntax, 'alias');
push(@csv_syntax, 'contact_name');
push(@csv_syntax, 'host_notification_period');
push(@csv_syntax, 'email');
push(@csv_syntax, 'pager');
```

Make sure you use the exact attribute names as they would appear in a Nagios configuration file. The order of the “push” commands must correspond exactly to the order of the CSV file (from left to right).

Once you've defined the syntax, you can execute the import script. This is how the script is used:

```
Usage:
./add_items_from_csv.pl -c class -n naming-attr -f /path/to/file [-x (1-5)] [-s]

Help:

  required
  -c  Specify the class of items that you wish to import. Must correspond to an NConf class
      (e.g. "host", "service", "hostgroup", "checkcommand", "contact", "timeperiod"...)
  -n  Specify the naming attribute for the class to be imported. Must correspond with the
```

```
      naming attr in NConf (e.g. host: "host_name", service: "service_description"...)
  -f  The path to the file which is to be imported. CAUTION: Make sure you have
      only items of one class in the same file (e.g. "hosts", "services"...)
      Also make sure you import host- or service-templates separately ("host" or
      "service" items containing a "name" attribute)

  optional
  -x  Set a custom loglevel (1 = lowest, 5 = most verbose)
  -s  Simulate only. Do not make any actual modifications to the database.
```

To import our contacts, we would execute the script like this:

```
$> cd /path/to/nconf/bin/
$> ./add_items_from_csv.pl -c contact -n contact_name -f /path/to/import.csv
```

This should give us the following output:

```
[ Initializing NConf perl-API (library version 0.2, written by A. Gargiulo) ]
[ Copyright (c) 2006-2009 Sunrise Communications AG, Zurich, Switzerland    ]

[INFO]  Started executing ./add_items_from_csv.pl
[INFO]  Adding contact 'smith'
[INFO]  Successfully added contact 'smith'
[INFO]  Adding contact 'brown'
[INFO]  Successfully added contact 'brown'
[INFO]  Adding contact 'jones'
[INFO]  Successfully added contact 'jones'
[INFO]  Finished running ./add_items_from_csv.pl
```

Finally, check your new contact items in NConf to make sure the import was successful and the attributes were mapped correctly.

Chances are you might encounter one or more errors during import. Here are a few known errors:

```
[WARN]  Could not find attribute '...' belonging to class '...'. Skipping import of this attribute.
```

This message indicates that your CSV file might contain an attribute that is unknown to NConf. These attributes will be skipped during import.

```
[WARN]  Mandatory attribute '...' missing for ...
[ERROR] Failed to add ... Aborting
```

This message indicates that the import script could not find one or more mandatory attributes in your CSV file. This will cause the import process to abort.

Importing hosts & services only

If you wish to import hosts & services only, you may use the script **add_items_from_special_csv.pl**. This script requires your CSV file to have a specific structure. Each host and service must have a fixed amount of attributes. The amount of services you can add per host is not limited.

The CSV file must be made up of two parts: The first 10 columns are reserved for host specific attributes. The 5 subsequent columns are service specific and can be repeated as many times as necessary. Each service must always consist of 5 columns.

The following columns are required (in this order from left to right):

host attributes:

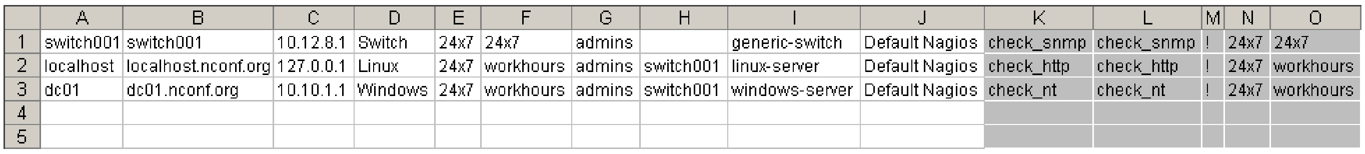
1. host_name
2. alias
3. address
4. os
5. check_period
6. notification_period
7. contact_groups
8. parents
9. host-preset
10. monitored_by

service attributes:

1. service_description
2. check_command
3. check_params
4. check_period
5. notification_period

(this block can be repeated as many times as needed)

If you're working with a spreadsheet tool, the CSV file would look somewhat like this. Attributes can be left empty, as long as they are not considered mandatory by NConf:



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	switch001	switch001	10.12.8.1	Switch	24x7	24x7	admins		generic-switch	Default Nagios	check_snmp	check_snmp	!	24x7	24x7
2	localhost	localhost.nconf.org	127.0.0.1	Linux	24x7	workhours	admins	switch001	linux-server	Default Nagios	check_http	check_http	!	24x7	workhours
3	dc01	dc01.nconf.org	10.10.1.1	Windows	24x7	workhours	admins	switch001	windows-server	Default Nagios	check_nt	check_nt	!	24x7	workhours
4															
5															

Notice the “gray” part. These are the 5 service specific columns. These 5 columns can be repeated, allowing you to add as many services for each host as you like. The amount of services can differ from host to host (leave unneeded columns empty).

The import script will process the CSV file sequentially, so make sure parent hosts are listed before the hosts that are dependent on them.

Once you are done editing your spreadsheet, export it to CSV format. This is not necessary if you were already editing the file with a text editor. Your data should look somewhat like this:

```
switch001;switch001;10.12.8.1;Switch;24x7;24x7;admins;;generic-switch;Default Nagios;check_snmp;check_snmp;!;24x7;24x7
localhost;localhost.nconf.org;127.0.0.1;Linux;24x7;workhours;admins;switch001;linux-server;Default Nagios;check_http;check_http;!;24x7;workhours
dc01;dc01.nconf.org;10.10.1.1;Windows;24x7;workhours;admins;switch001;windows-server;Default Nagios;check_nt;check_nt;!;24x7;workhours
```

You are now ready to run the import script. This is how the script is used:

```
Usage:
./add_items_from_special_csv.pl -f /path/to/file [-x (1-5)] [-s]

Help:

  required
  -f  The path to the CSV file which is to be imported.

  optional
  -x  Set a custom loglevel (1 = lowest, 5 = most verbose)
  -s  Simulate only. Do not make any actual modifications to the database.
```

Executing the script with the above data should give the following output:

```
$> ./add_items_from_special_csv.pl -f /path/to/special_import.csv

[ Initializing NConf perl-API (library version 0.2, written by A. Gargiulo) ]
[ Copyright (c) 2006-2009 Sunrise Communications AG, Zurich, Switzerland    ]

[INFO]  Started executing ./add_items_from_special_csv.pl
[INFO]  Adding host 'switch001'
[WARN]  Mandatory attribute 'host_is_collector' missing for host 'switch001'. Using default value: 'no'.
[INFO]  Successfully added host 'switch001'
[INFO]  Adding service 'switch001;;check_snmp'
[INFO]  Successfully added service 'switch001;;check_snmp'
[INFO]  Adding host 'localhost'
[WARN]  Mandatory attribute 'host_is_collector' missing for host 'localhost'. Using default value: 'no'.
[INFO]  Successfully added host 'localhost'
[INFO]  Adding service 'localhost;;check_http'
[INFO]  Successfully added service 'localhost;;check_http'
[INFO]  Adding host 'dc01'
[WARN]  Mandatory attribute 'host_is_collector' missing for host 'dc01'. Using default value: 'no'.
[WARN]  Could not link host with os 'Windows' (attr 'os'). No such item.
[INFO]  Successfully added host 'dc01'
[INFO]  Adding service 'dc01;;check_nt'
[INFO]  Successfully added service 'dc01;;check_nt'
[INFO]  Finished running ./add_items_from_special_csv.pl
```

As you can see, the import script will inform you about any missing attributes and any values it couldn't map. If you omit any mandatory NConf attributes, the import process will terminate with an ERROR. It's therefore recommended to run the import in simulation mode (-s) prior to doing the actual import.

```
[WARN]  Mandatory attribute '...' missing for ...
[ERROR] Failed to add host ... Aborting
```

Finally, check your new host and service items in NConf to make sure the import was successful and the attributes were mapped correctly.