JSend NSCA 1.1.1

User & Developer Guide

Contents

Overview 1
Background 1
API Quick Start 2
Pre-requisites 2
Set up API 3
Code to send passive check 3
Complete quick start code 5
Command Line Tool 6
Set up 6
Usage 6
Example 6
Further reading 6

Overview

JSend NSCA is a Java API and command line tool for sending Nagios[1] Passive Checks to the Nagios NSCA add on.

By using the API, you can easily integrate your Java applications into a Nagios monitored environment thereby notifying Nagios of problems and issues during the running of your application.

The command line tool wraps the API and allows you to send passive checks from the command line.

Background

JSend NSCA was developed as the company I'm working for uses Nagios to monitor applications and servers. For existing applications written is Perl and c, there were options available to send passive checks but for Java application, the option available was to shell out and execute the send_nsca command line tool.

Although send_nsca worked in this manner, it's a bit ugly and we preferred having the code within our applications for better performance.

A search on the internet revealed a few options such as the NagiosAppender for log4j but in the end we settled on writing our own client. This client is currently in use thus proving the feasibility of the approach.

On the back of this, I decided to write JSend NSCA from the ground up as an exercise in TDD and thought I would make it available as an open source project so other developers can benefit from the functionality.

[1] For information on Nagios and NSCA see http://www.nagios.org/

API Quick Start

OK, so you want to send passive checks from your code. This quick start will hopefully have you sending checks in no time.

Pre-requisites

You need to have Nagios and NSCA set up. This is beyond the scope of this document so refer to the documentation available on the Nagios web site.

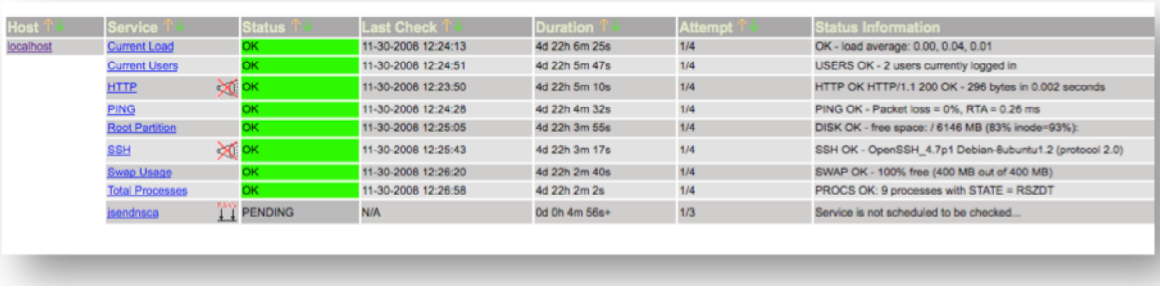
For this quick start, add the following passive check to the localhost.cfg. I will assume Nagios is set up locally, if not make a note of the IP or hostname for use later on.

```
define service{
        use                     template-nsca-service
        host_name               localhost
        service_description     jsendnsca
        check_command           check_dummy
}
```

And restart nagios as shown below

```
raj@ubuntu-server:/usr/local/nagios/etc/objects$ sudo /etc/init.d/nagios restart
Running configuration check...done.
Stopping nagios: done.
Starting nagios: done.
```

Now check localhost on Nagios to see the jsendnsca service as shown below



Host	Service	Status	Last Check	Duration	Attempt	Status Information
localhost	Current Load	OK	11-30-2008 12:24:13	4d 22h 6m 25s	1/4	OK - load average: 0.00, 0.04, 0.01
	Current Users	OK	11-30-2008 12:24:51	4d 22h 5m 47s	1/4	USERS OK - 2 users currently logged in
	HTTP	OK	11-30-2008 12:23:50	4d 22h 5m 10s	1/4	HTTP OK HTTP/1.1 200 OK - 296 bytes in 0.002 seconds
	PING	OK	11-30-2008 12:24:28	4d 22h 4m 32s	1/4	PING OK - Packet loss = 0%, RTA = 0.26 ms
	Root Partition	OK	11-30-2008 12:25:05	4d 22h 3m 55s	1/4	DISK OK - free space: / 6146 MB (83% inode=93%):
	SSH	OK	11-30-2008 12:25:43	4d 22h 3m 17s	1/4	SSH OK - OpenSSH_4.7p1 Debian-8ubuntu1.2 (protocol 2.0)
	Swap Usage	OK	11-30-2008 12:26:20	4d 22h 2m 40s	1/4	SWAP OK - 100% free (400 MB out of 400 MB)
	Total Processes	OK	11-30-2008 12:26:58	4d 22h 2m 2s	1/4	PROCS OK: 9 processes with STATE = RSZDT
	jsendnsca	PENDING	N/A	0d 0h 4m 56s+	1/3	Service is not scheduled to be checked...

OK, so now we have our service set up, we can move onto the code.

Set up API

Download jsendnsca-core.zip or the tar.gz and unpack somewhere. The contents are listed below.

```
jsendnsca-core-1.1.1
|-- javadocs
|-- jsendnsca-core-1.1.1.jar
```

Copy the jsendnsca-core-1.1.1.jar to your classpath so it can be referenced by your code.

Code to send passive check

To show sending of a passive check, we will build up a class to send the check

Set up a basic class as below

```
package com.jsendnsca.quickstart;

public class QuickStart {

	public static void main(String[] args) {
		sendPassiveCheck();
	}

	public static void sendPassiveCheck() {

	}
}
```

Now lets set our settings for Nagios up in the sendPassiveCheck method. The settings below are default so we could have actually just constructed the object, but for illustration purposes, I've set them explicitly. If nagios is running on another host, set the nagios host accordingly with the correct password.

```
final NagiosSettings nagiosSettings = new NagiosSettings();
nagiosSettings.setNagiosHost("localhost");
nagiosSettings.setPassword("hasturrocks");
nagiosSettings.setEncryptionMethod(NagiosSettings.XOR_ENCRYPTION);
nagiosSettings.setTimeout(10000);
nagiosSettings.setPort(5667);
```

Now our settings have been set, we can create the passive check itself.

```
final MessagePayload messagePayload = new MessagePayload();
messagePayload.setHostname("localhost");
messagePayload.setLevel(MessagePayload.LEVEL_OK);
messagePayload.setServiceName("jsendnsca");
messagePayload.setMessage("It works!");
```

The message we have have constructed is being sent from localhost, is OK, for service jsendnsca and the message is “It works!”

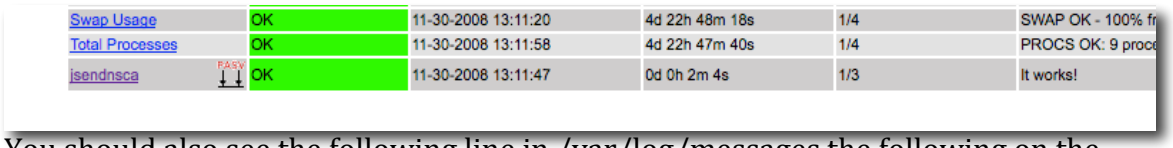
We are explicitly setting the hostname to localhost in this example. Alternatively, you can call the method below. If useCanonical is true, the fully qualified domain name of the host where you are sending from will be determined. If false, it will set the hostname to the short hostname of the host.

```
messagePayload.setHostname(boolean useCanonical)
```

To send the passive check we have created, we construct a NagiosPassiveCheckSender using our settings and send the MessagePayload.

```
final NagiosPassiveCheckSender nagiosPassiveCheckSender =
                        new NagiosPassiveCheckSender(nagiosSettings);
try {
    nagiosPassiveCheckSender.send(messagePayload);
} catch (NagiosException e) {
    e.printStackTrace();
} catch (IOException e) {
    e.printStackTrace();
}
```

If you run this class now, it will send a passive check. After checking the service on Nagios, you should see the entry change as shown below



You should also see the following line in /var/log/messages the following on the Nagios host

```
Nov 30 13:11:48 ubuntu-server nagios: EXTERNAL COMMAND:
PROCESS_SERVICE_CHECK_RESULT;localhost;jsendnsca;0;It works!
```

Complete quick start code

```
package com.jsendnsca.quickstart;

import java.io.IOException;

import com.googlecode.jsendnsca.core.MessagePayload;
import com.googlecode.jsendnsca.core.NagiosException;
import com.googlecode.jsendnsca.core.NagiosPassiveCheckSender;
import com.googlecode.jsendnsca.core.NagiosSettings;

public class QuickStart {

	public static void main(String[] args) {
		sendPassiveCheck();
	}

	public static void sendPassiveCheck() {
		final NagiosSettings nagiosSettings = new NagiosSettings();
		nagiosSettings.setNagiosHost("localhost");
		nagiosSettings.setPassword("hasturrocks");
		nagiosSettings.setEncryptionMethod(NagiosSettings.XOR_ENCRYPTION);
		nagiosSettings.setTimeout(20000);
		nagiosSettings.setPort(5667);

		final MessagePayload messagePayload = new MessagePayload();
		messagePayload.setHostname("localhost");
		messagePayload.setLevel(MessagePayload.LEVEL_OK);
		messagePayload.setServiceName("jsendnsca");
		messagePayload.setMessage("It works!");

		final NagiosPassiveCheckSender nagiosPassiveCheckSender =
			new NagiosPassiveCheckSender(nagiosSettings);
		try {
			nagiosPassiveCheckSender.send(messagePayload);
		} catch (NagiosException e) {
			e.printStackTrace();
		} catch (IOException e) {
			e.printStackTrace();
		}
	}
}
```

Command Line Tool

In addition to the core API, a command line tool is available for sending passive checks from the command line.

Set up

Download the jsendnsca-cli-1.1.1.zip or tar.gz and unpack. The contents are shown below.

```
jsendnsca-cli-1.1.1
|-- bin
|   |-- jsend_nsca
|   |-- jsend_nsca.bat
|   `-- jsendnsca-cli-1.1.1.jar
`-- lib
    |-- commons-cli-1.1.jar
    |-- commons-lang-2.4.jar
    `-- jsendnsca-core-1.1.1.jar
```

For unix users, add execute permissions to the jsend_nsca shell script and add the executable to your path if you like. Windows users can use the .bat script with the same functionality.

Usage

If you run the script without any arguments or with –-help, the following usage information is displayed.

```
Raj:bin rajpatel$ jsend_nsca
usage: jsend-nsca [OPTIONS] OK|WARNING|CRITICAL servicename message
[OPTIONS]
     --alertinghost <alerting host>          the host sending the passive check, defaults to using
                                             the hostname of the machine
     --nagioshost <nagios host>              the host where nagios is running, defaults to localhost
     --password <nsca password>              the password configured in NSCA, defaults to
                                             hasturrocks
     --port <port>                           the port on which NSCA is running, defaults to 5667
     --timeout <send timeout>                the timeout to use when sending the passive check in
                                             ms, defaults to 10000
```

Example

The example below shows a passive check being sent for our jsendnsca service.

```
Raj:bin rajpatel$ jsend_nsca --nagioshost localhost --password hasturrocks WARNING
jsendnsca ooops
Sent Passive Check Successfully
Raj:bin rajpatel$
```

On checking the service, you should see the following

jsendnsca	PASV	WARNING	11-30-2008 14:30:43	0d 0h 3m 4s	3/3	ooops

Further reading

For more information on the API, see the javadoc provided in the jsendnsca-core package.