# Your Question

**Article: 00116**
**Question:**
How to Work with Regular Expression in Net Report?

# Net Report Answer

## Introduction

For the following examples we will be using the Generic Parser in the Net Report Management Console. Through Examples 1 – 9 we will give examples of how to:

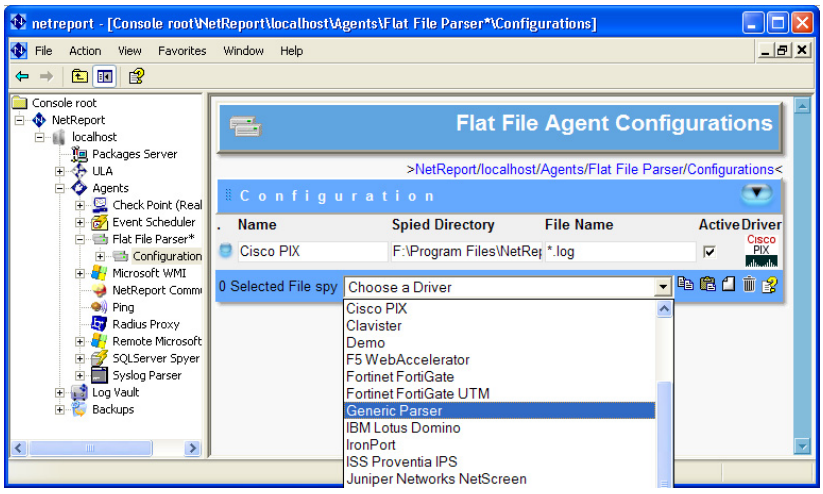
- Write Regular Expressions via the Records Parser.
- Parse Test Values.
- Test the expressions.
- Create Header Records.
- Use the Net Report Includes Parser.

To get started, create a Generic parser and add regular expressions via the Records Parser, please follow the steps below.

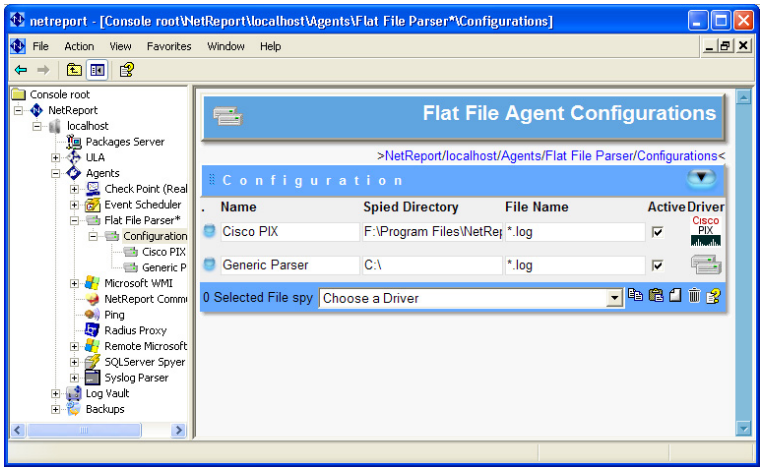
**Creating a Generic Parser and Using the Records Parser to Create and Test Regular Expressions**

### Steps

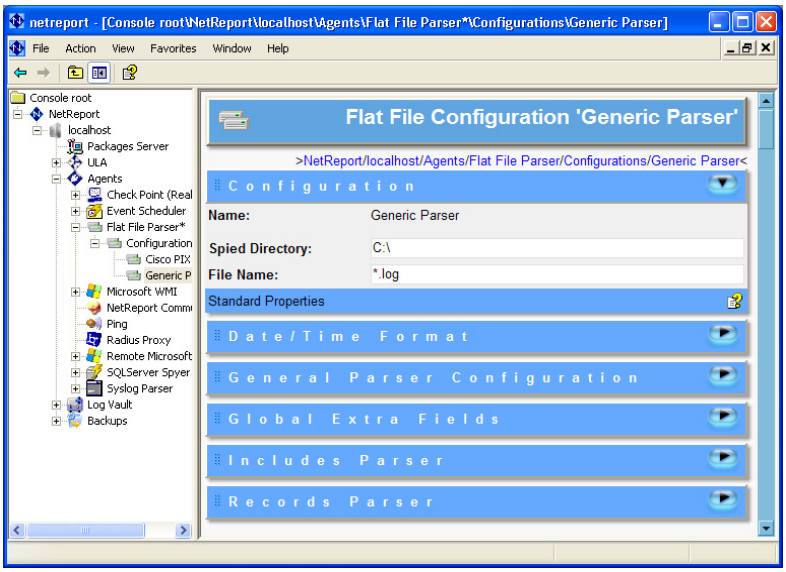
1. Launch the Net Report Management Console, select **Start> All Programs> NetReport> Management Console**. The **Login** dialog box appears.
2. Enter your **Login** and **Password**. The Net Report Management Console appears.
3. Select **NetReport> [localhost]> Agents> Flat File Parser> Configuration** in the left **Console Root** pane.
4. Select **Generic Parser** in the drop-down list.



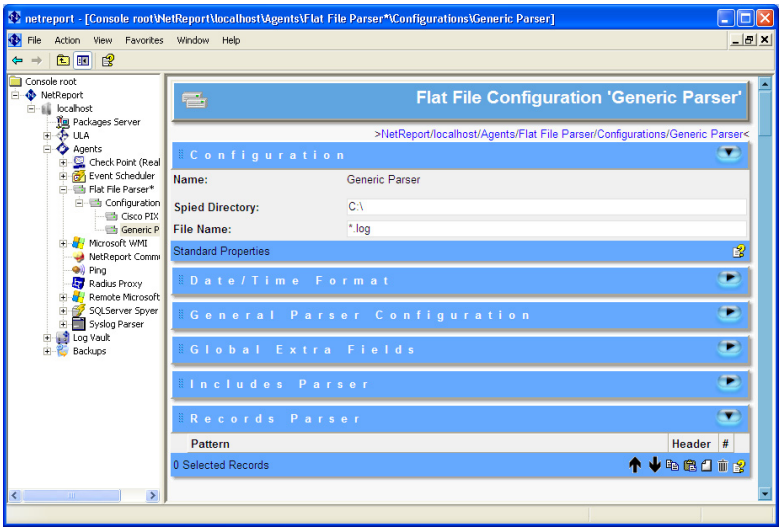
5. Click **New** to add the Generic Parser.



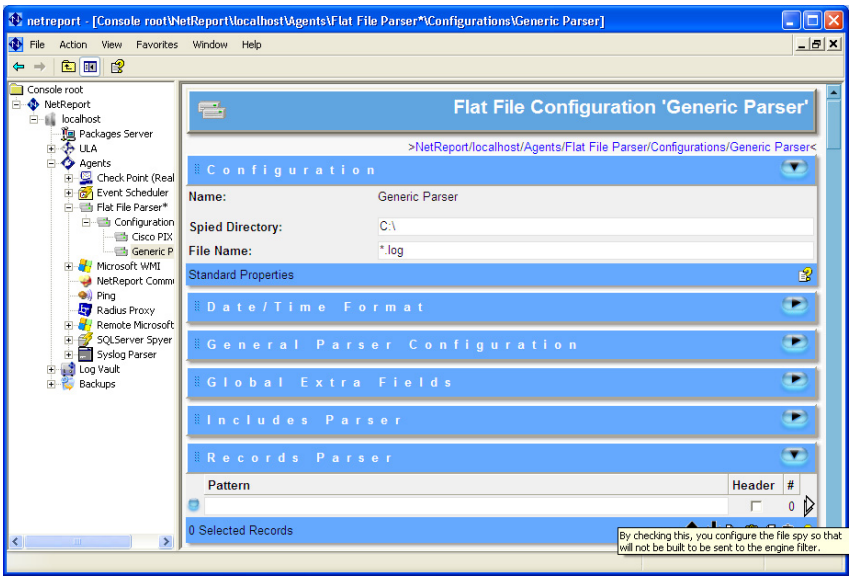

6. Click **Edit this spied configuration** to the right of the Generic Parser row. The **Flat File Configuration 'Generic Parser'** pane appears.



7. Select the **Records Parser** section.




8. Click **New** to add a new pattern (regular expression).




9. Click **Edit Pattern** to write the regular expression, add a test value and test the expression.

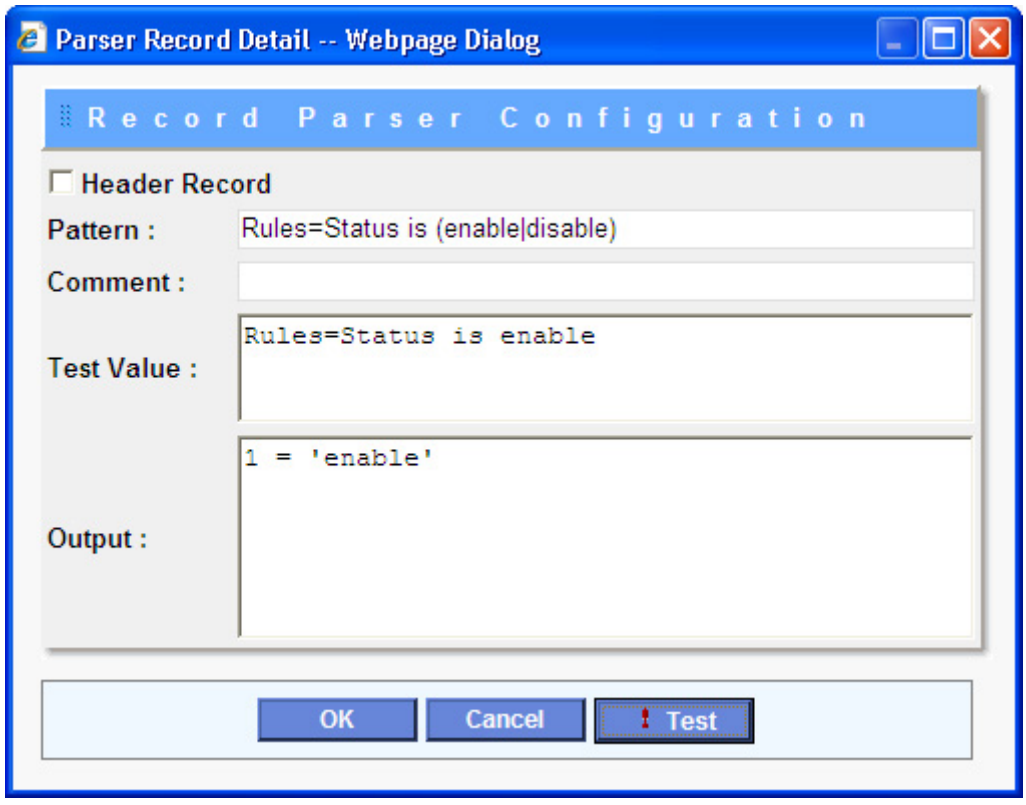
**Example 1: Alternation and Grouping**

If you do not want to capture a substring but need parentheses to group the substring, use the '?:' operator to avoid capturing. For example, to see if a rule's status is Enable or Disable we could use the following regular expression:

Rules=Status is (enable|disable

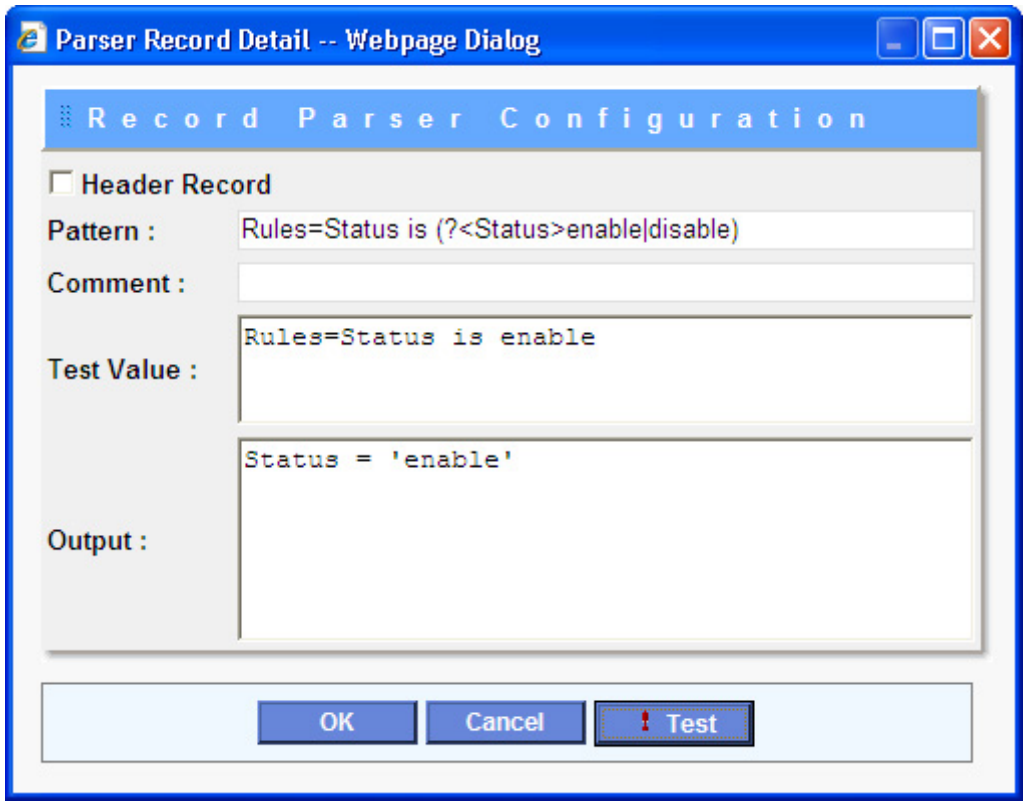
## Steps

1. Click **New** to add a new **Pattern** via the **Records Parser**.
2. Click **Edit Records Parser**.
3. Enter the following pattern in the **Pattern** field:
   Rules=Status is (enable|disable)
4. Enter the following **Test Value**:
   Rules=Status is enable
5. Click **Test** and note the **Output**.




6. Note the **Output** displays a numbered match: 1='enable' without a name. To name the capture, modify the pattern as follows.

7. Modify the pattern by adding ?<Status> before enable|disable:
   Rules=Status is (?<Status>enable|disable)




8. Note the effect of (?<Status>) in displaying a named capture. Note that the capture is no longer numbered.

**Information**
If we analyse the above expression (pattern) we can note the following key points:

| Character | Comments |
|---|---|
| \| | \| or 'pipe' means 'or', for example enable\|disable matches enable or disable. |
| (enable\|disable) | Matches either enable or disable. For example (none\|enable\|disable) would match none or enable or disable. |
| (?<Status> | Names the capture 'Status'. |

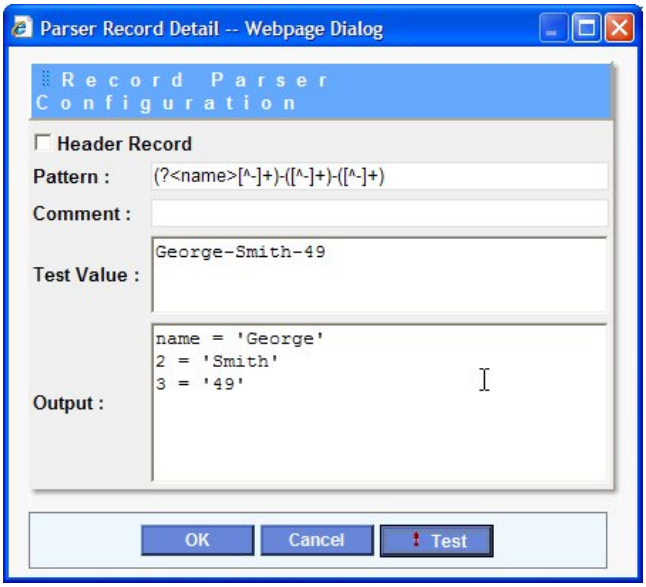
**Example 2: Extracting Fields**

A log line will typically list several fields separated by separators or tabs. Below is a very simple line:

George-Smith-49
To parse this line and extract each field we could use the following expression:
(?<name>[^-]+)-([^-]+)-([^-]+)

## Steps

1. Click **New** to add a new **Pattern** via the **Records Parser**.
2. Click **Edit Records Parser**.
3. Enter the following pattern in the **Pattern** field:
   (?<name>[^-]+)-([^-]+)-([^-]+)
4. Enter the following **Test Value**:
   George-Smith-49
5. Click **Test** and note the **Output**.



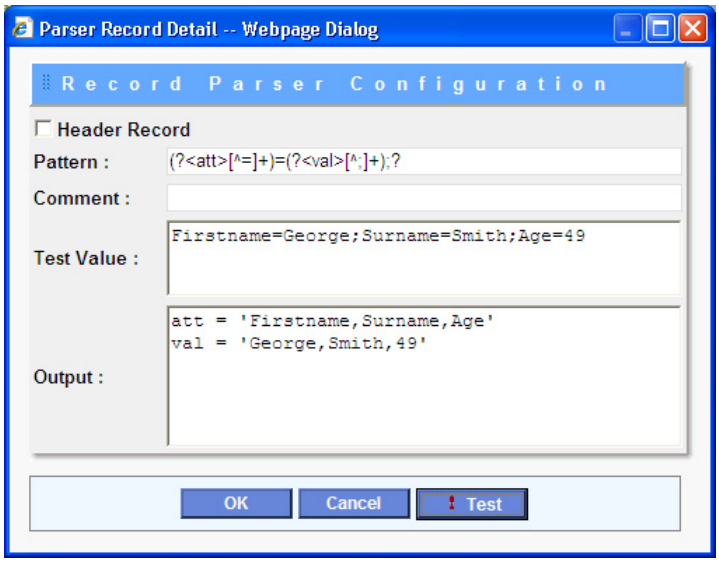

**Information**
If we analyse the above expression we can note the following key points:

| Character | Comments |
|---|---|
| (?<name>[^-]+) | Captures the name, in this case 'George'. Captures all characters except the – (endash) one or more times. |
| ([^-]+) | Captures all characters except the – (endash) one or more times. |

**Example 3: Matching Attributes and Values**
To display all the attributes (for example, Firstname, Surname, Age) in a row, followed by all the values (for example, George, Smith, 49) in the row below, we could use the following expression:
(?<att>[^=]+)=(?<val>[^;]+);?

### Steps

1. Click **New** to add a new **Pattern** via the **Records Parser**.
2. Click **Edit Records Parser**.
3. Enter the following pattern in the **Pattern** field:
   (?<att>[^=]+)=(?<val>[^;]+);?
4. Enter the following **Test Value**:
   Firstname=George;Surname=Smith;Age=49
5. Click **Test** and note the **Output**.




**Information**
If we breakdown the expression above, we can note the following points:

| Character | Comments |
|---|---|
| (?<att> | Captures the attribute name. |
| [^=]+ | Matches all characters except = (equal to) one or more times. |
| (?<val> | Captures the value. |
| [^;]+ | Matches all characters except ; (semi-colon) one or more times. |
| ;? | Matches 0 or once what preceded, in this case the ; semi-colon. |

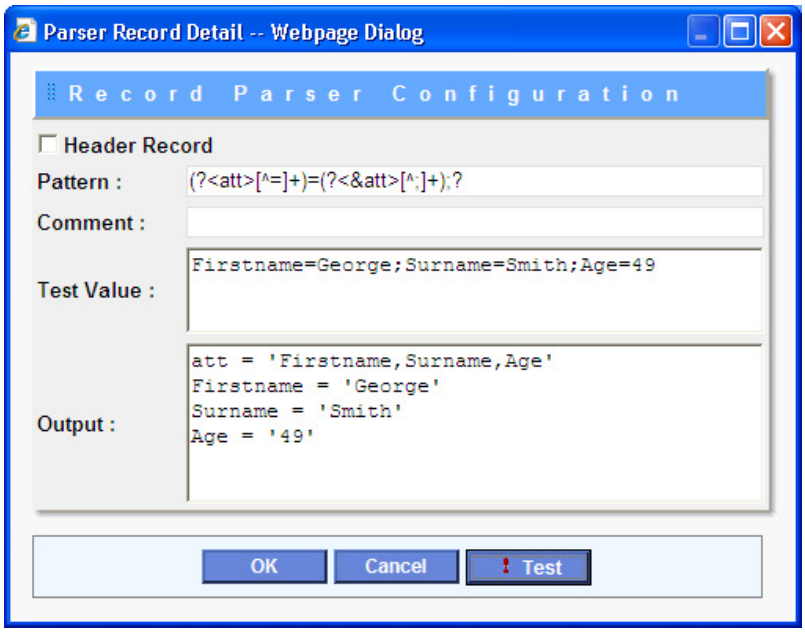
**Example 4: Displaying Each Attribute with its Value**
To display the attributes in a row (for example, Firstname, Surname, Age), followed by each attribute separately with its specific value (for example, Surname = Smith), we could use the following expression:
(?<att>[^=]+)=(?<&att>[^;]+);?

### Steps

1. Edit the previous Pattern, replace <val> by <&att>:
   (?<att>[^=]+)=(?<&att>[^;]+);?

2. Keep the previous **Test Value**:
   Firstname=George;Surname=Smith;Age=49

3. Click **Test** and note the **Output**.



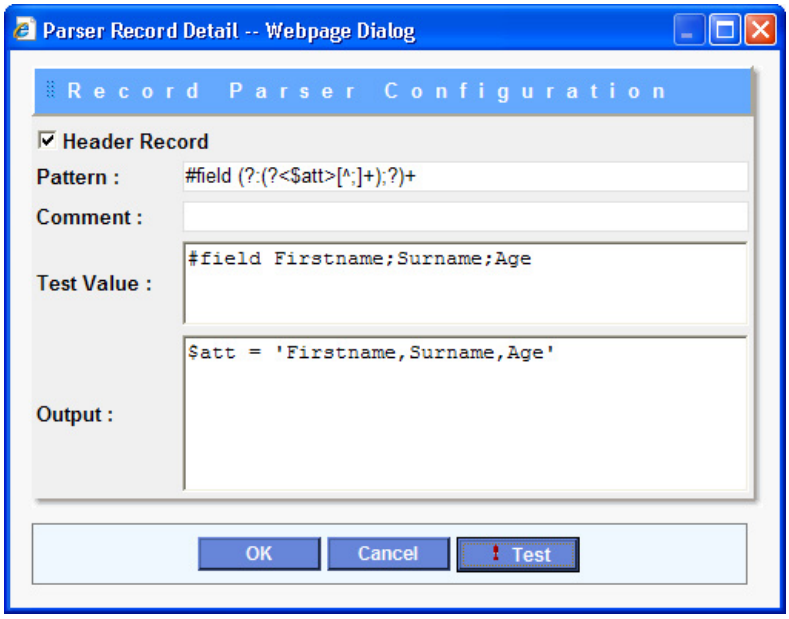

**Information**
If we breakdown the expression above, we can note the following points:

| Character | Comments |
|---|---|
| (?<att> | Captures the attribute name. |
| [^=]+ | Matches all characters except the = (equal to) one or more times. |
| (?<&att> | Analyses the Attribute's value with reference to the Attribute Name captured. |
| [^;]+ | Matches all characters except the ; (semi-colon) one or more times. |
| ;? | Matches 0 or once what preceded, in this case the ; semi-colon. |

**Example 5: Defining a Header Record**
It is possible to create a Header Record, that is, a pattern which is used to parse the header in a log file. In this example, we are going to use the following pattern:
#field (?:(?<$att>[^;]+);?)+

### Steps

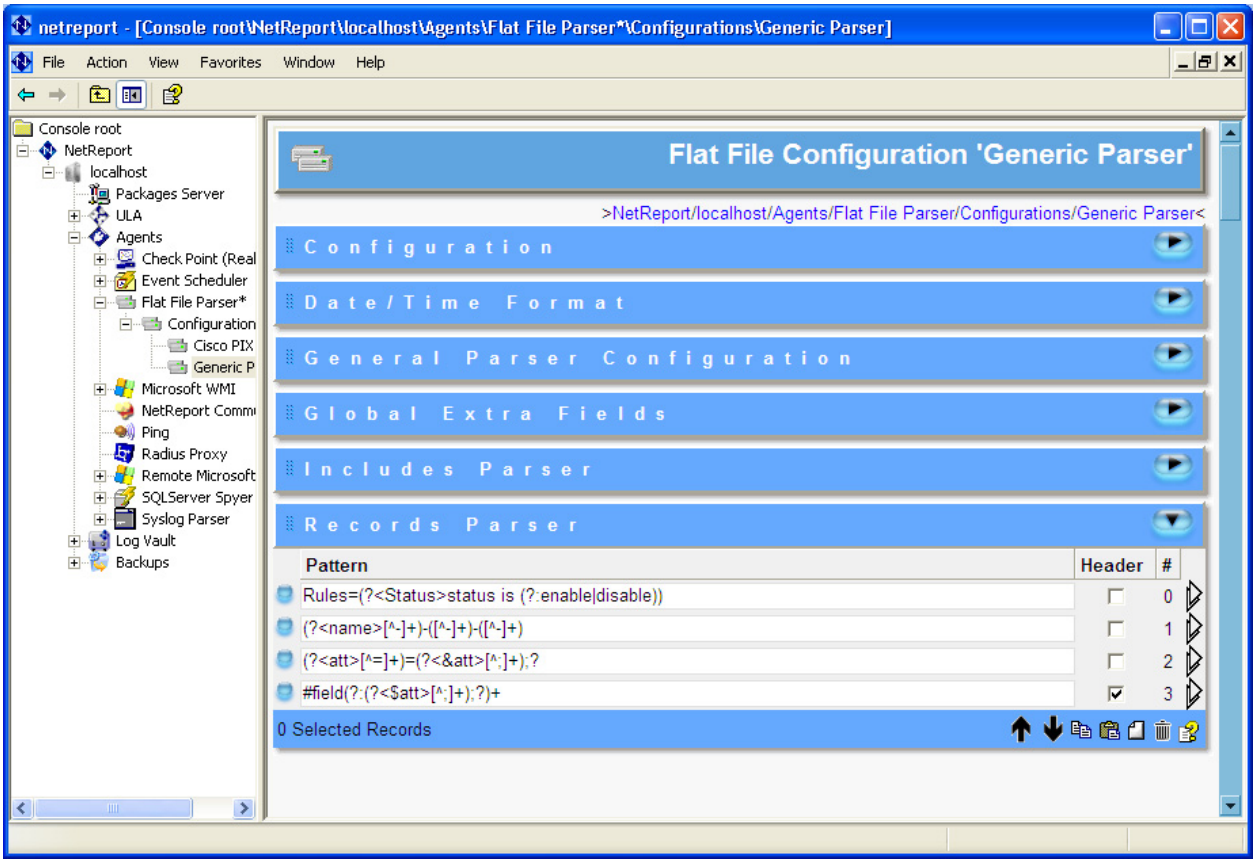
1. Click **New** to add a new **Pattern** via the **Records Parser**.
2. Click **Edit Records Parser**.
3. Select the **Header Record** check box.
4. Enter the following pattern in the **Pattern** field:
   #field (?:(?<$att>[^;]+);?)+
5. Enter the following **Test Value**:
   #field Firstname;Surname;Age
6. Click **Test** and note the **Output**.




**Information**
If we breakdown the expression above, we can note the following points:

| Character | Comments |
|---|---|
| #field | Matches #field |
| ?: | Regroups what follows between parantheses. |
| $att | Stores the attribute name for it to be used as reference later (please see Example 6). |
| [^;]+ | Matches all characters except the ; (semi-colon) one or more times. |

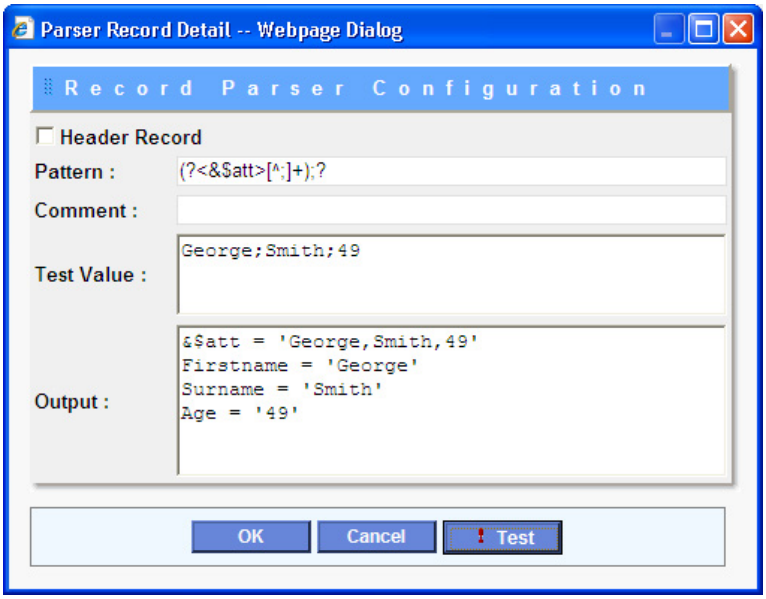
**Status:** if we return to the **Flat File Configuration 'Generic Parser'** screen we can note the following patterns have been added and that **Pattern** number 3 (#3) is selected as a **Header** Record.



**Example 6: Parsing with reference to a Header Record**
We are now going to parse a new record with reference to the Header Record we created in Example 5, using the following pattern.
(?<&$att>[^;]+);?

### Steps

1. Click **New** add a new **Pattern** via the **Records Parser**.
2. Click **Edit Records Parser**.
3. Enter the following pattern in the **Pattern** field:
   (?<&$att>[^;]+);?
4. Enter the following **Test Value**:
   George;Smith;49
5. Click **Test** and note the **Output**.




**Information**
If we breakdown the expression above, we can note the following points:

| Character | Comments |
|---|---|
| <&$att> | Refers to the $att which we defined in the Header Record in Example 5. |
| [^;]+ | Matches all characters except the ; (semi-colon) one or more times. |
| ;? | Matches 0 or once what preceded, in this case the ; semi-colon. |

**Example 7: Creating Includes Parser Patterns**

To make regular expressions as readable and user friendly as possible, Net Report enables you to create predefinitions. For each one of them, a name and a pattern must be defined. For this example we are going to use the following test pattern:
Src=192.168.0.1;dst=15.12.12.01;user=George;Service=80

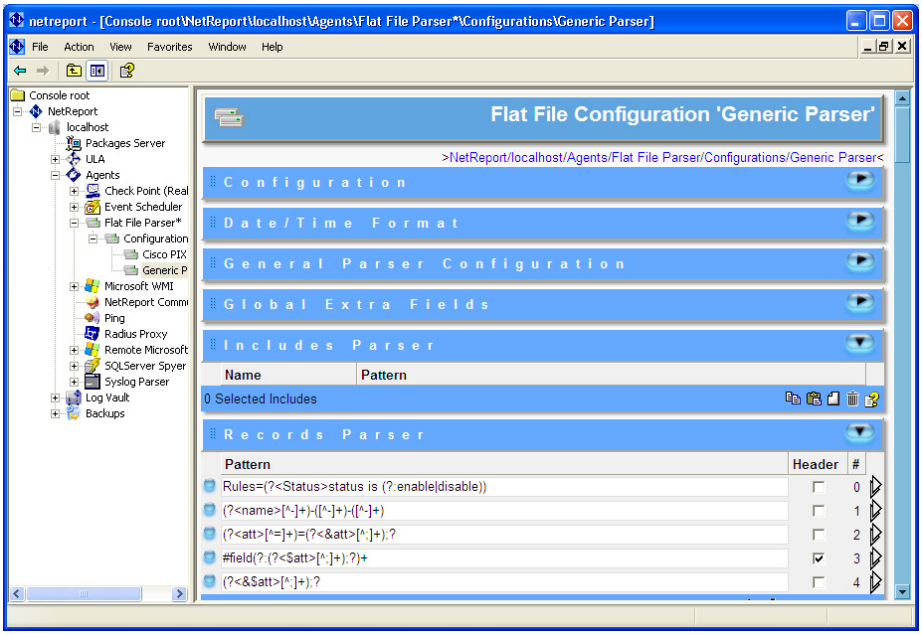
We therefore need to create Includes Parser patterns for numbers, letters and IP Addresses. For this example we are going to create the following patterns:

| Name | Pattern |
|---|---|
| num | [0-9]+ |
| alpha | \w+ |
| ip | [0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3} |

To add these patterns via the Includes Parser, please follow the steps below:

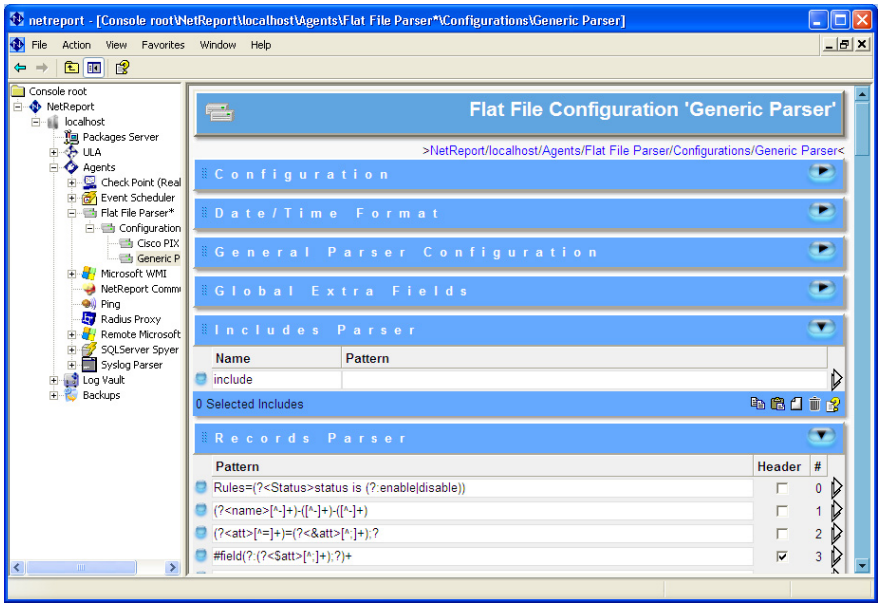
## Steps

1. Select the **Includes Parser** section.

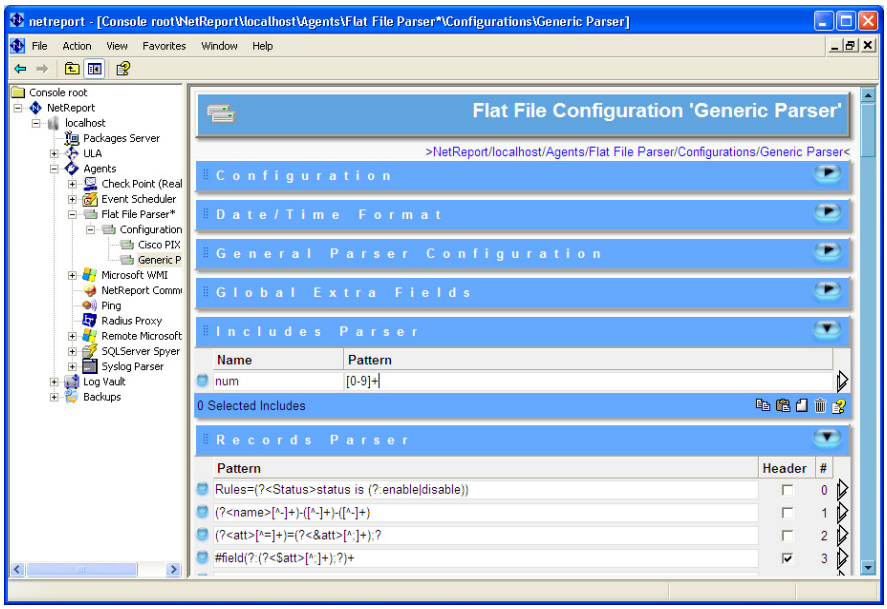


2. Click **New** to add a new pattern (regular expression). **Note:** for each Include you must define the following:

   **Name:** the name of the Include.
   **Pattern:** the Include's pattern.

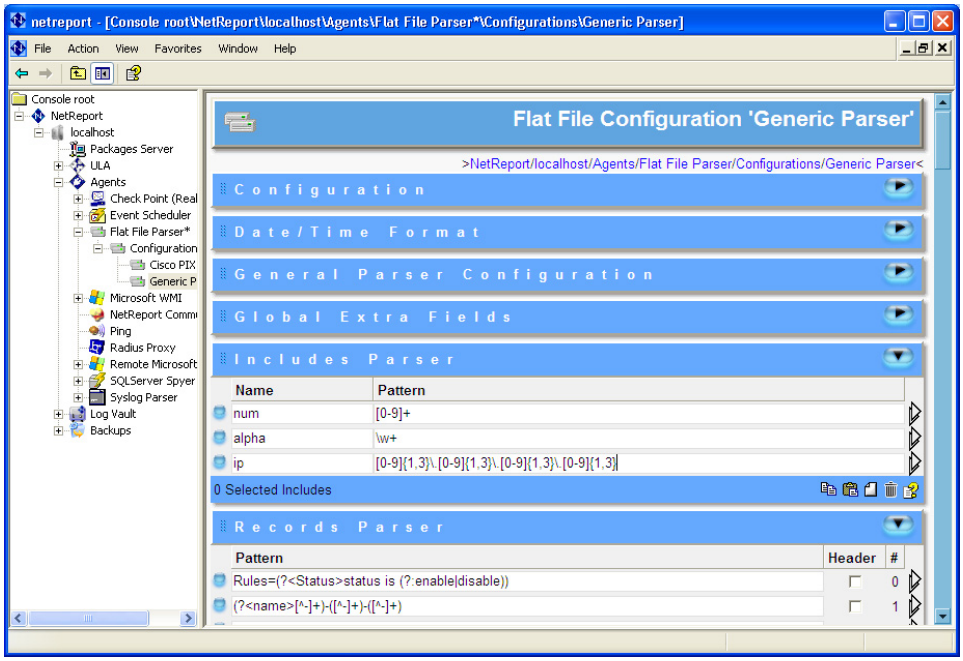



3. Replace the 'include' **Name** by num and enter the **Pattern**: [0-9]+:

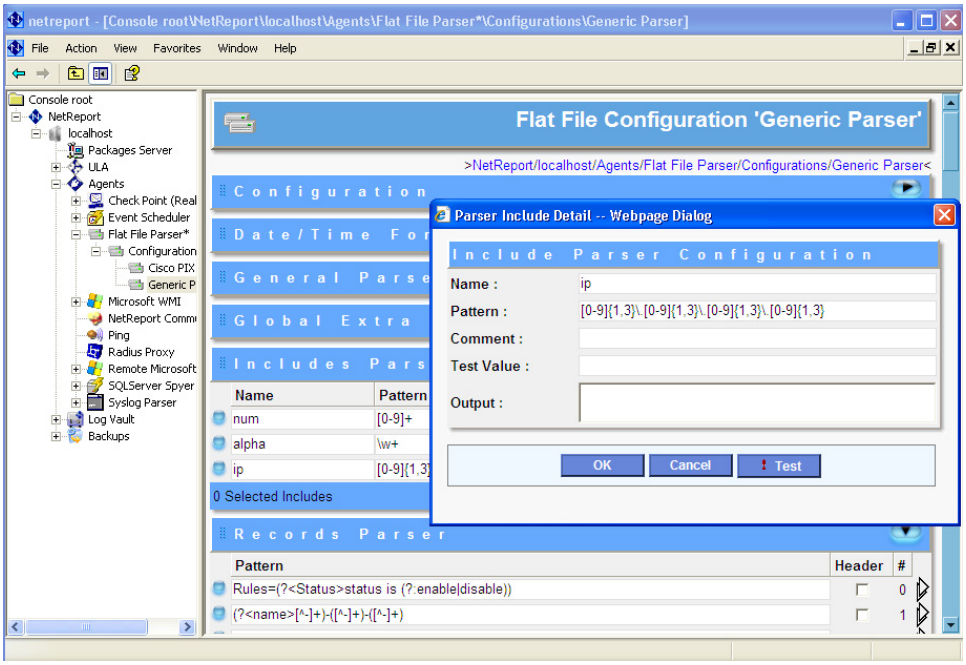


4. Create the following two Includes predefinitions

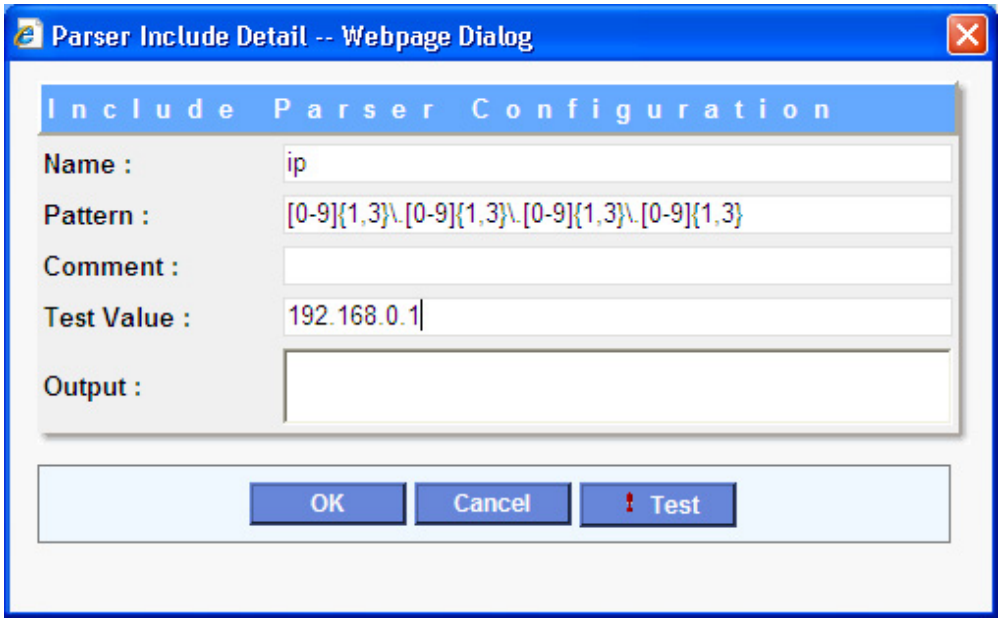
| Name | Pattern |
|---|---|
| alpha | \w+ |
| ip | [0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3} |



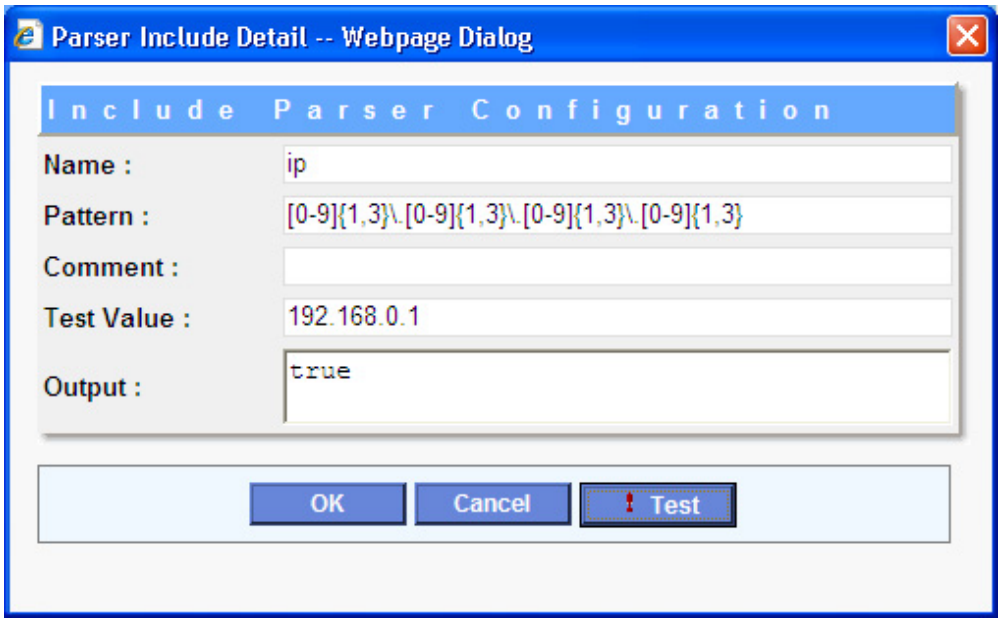

5. Click **Show Include Detail** to the right of ip to edit the ip Include predefinition. The **Include Parser Configuration** dialog box appears.



6. Enter an IP Address as a **Test Value**, for example: 192.168.0.1:



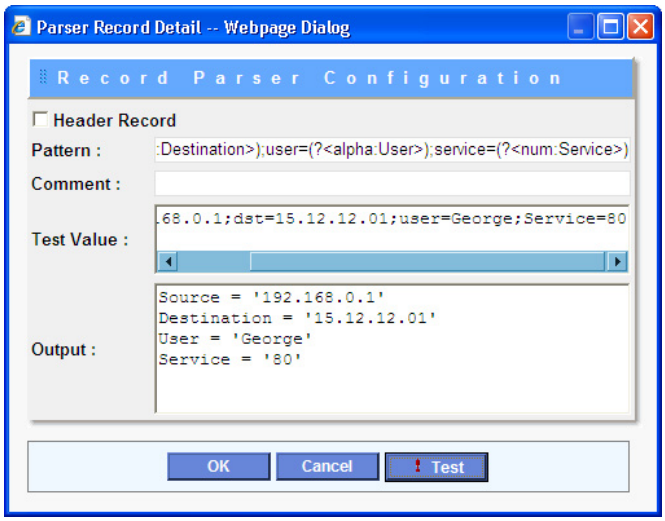

7. Click **Test** and note the **Output**.



**Example 8: Parsing a Record with the Includes Parser**
To parse a record using the Includes Parser definitions we defined in Example 7, please follow the steps below.

### Steps

1. Select Records Parser Patterns # 2 and #4.
2. Click **Delete** to delete these patterns.
3. Click **New** to add a new **Pattern** via the **Records Parser**.
4. Click **Show Includes Detail**.
5. Enter the following pattern in the **Pattern** field:
   src=(?<ip:Source>);dst=(?<ip:Destination>);user=(?<alpha:User>);service=(?<num:Service>)
6. Enter the following **Test Value**:
   Src=192.168.0.1;dst=15.12.12.01;user=George;Service=80
7. Click **Test** and note the **Output**.




**Information**
If we breakdown the expression above, we can note the following points:

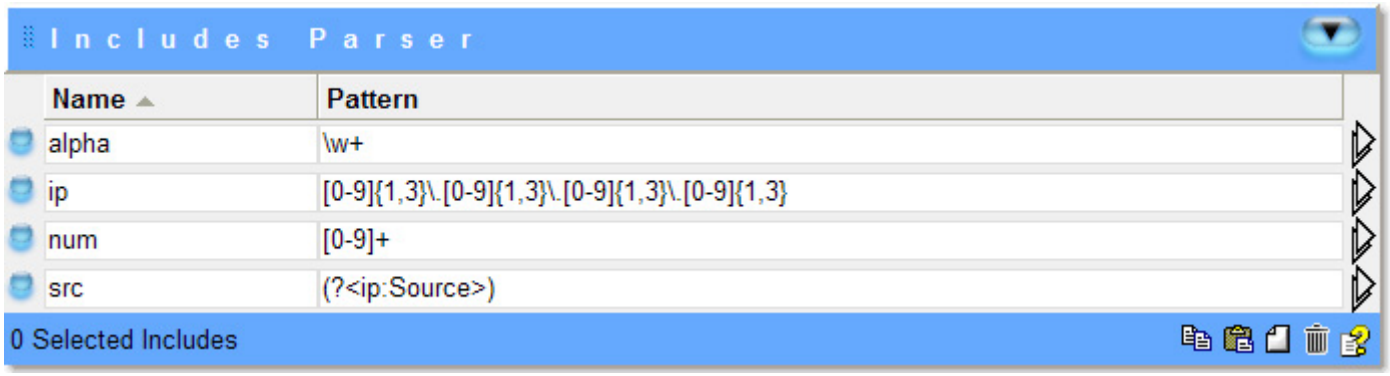
| Character | Comments |
|---|---|
| (?<ip:Source>) | Captures the value for the Includes Parser 'ip' which we defined in Example 7. |
| (?<ip:Destination>) | Captures the value for the Includes Parser 'ip' which we defined in Example 7. |
| (?<alpha:User>) | Captures the value for the Includes Parser 'alpha' which we defined in Example 7. |
| (?<num:Service>) | Captures the value for the Includes Parser 'num' which we defined in Example 7. |

**Example 9: Includes Parser Named Captures**
To parse a record using an Includes Parser named capture, please follow the steps below.

### Steps

1. Create the following Includes predefinitions:

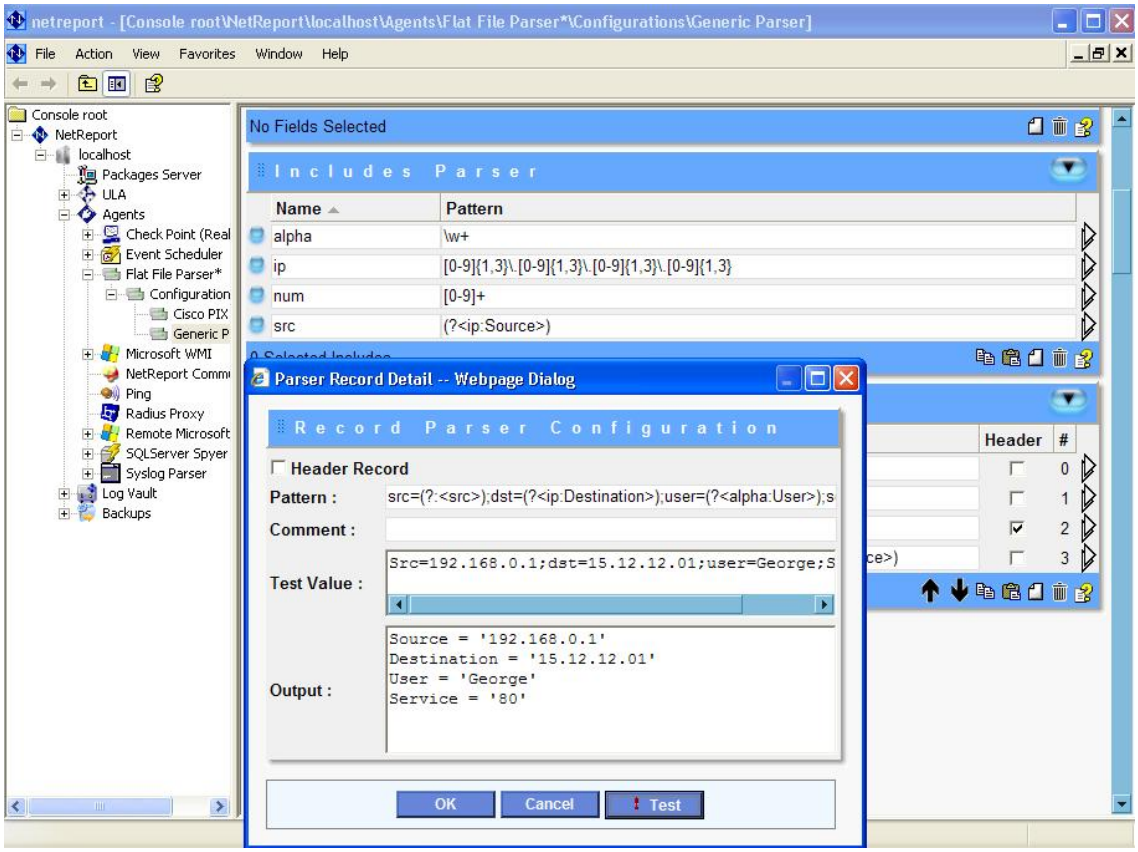
| Name | Pattern |
|---|---|
| src | (?<ip:Source>) |




2. Click **Show Includes Detail** to the right of the **Records Parser** pattern we created in Example 8: src=(?<ip:Source>);dst=(?<ip:Destination>);user=(?<alpha:User>);service=(?<num:Service>)

3. Modify the beginning of the above pattern to refer to the src Includes predefinition we just defined as follows:
   Replace
   src=*(?<ip:Source>)*;dst=(?<ip:Destination>);user=(?<alpha:User>);service=(?<num:Service>)
   by
   src=*(?:<src>)*; dst=(?<ip:Destination>);user=(?<alpha:User>);service=(?<num:Service>)

4. Click **Test** and note the **Output**.




**Information**
If we breakdown the expression above, we can note the following points:

| Character | Comments |
|---|---|
| (?:<src>) | Captures the value for the Includes Parser ‘src’ which we defined in Example 8. |