Configuring Precise for Web

This section includes the following topics:

- About configuring Precise for Web Collectors
- Configuring registry settings
- Adding Web filter parameters
- Grouping consecutive URLs by pattern
- Temporarily stopping the maximum number of table rows test
- Dynamic instrumentation

About configuring Precise for Web Collectors

To configure the Precise for Web Collectors, you can perform one of the following tasks:

1. Add and change parameters to the Precise registry file.
2. Add parameters to the Web server's configuration mechanism.
3. Add and change parameters to the dynamic instrumentation agent via the `instrument.xml` file.

Configuring registry settings

This section describes the registry settings configuration process.

Registry structure and inheritance

Some configurations are relevant per instance (e.g. collect-post-parameters) and some are relevant for the Web Collector in general (e.g. preferred-data-port). Others are relevant for the whole product (e.g. user-agents). Instance configuration resides in the instance registry file:

products\i3fp\registry\instances\www\<*server-name*>\<*instance_name*>\settings.xml

> You can set instance configuration in the `products\i3fp\registry\instances\www\<server-name>\settings.xml` file if you want to set the same setting for all instances on the machine. Consider however that settings in the instance registry will override the settings in the server registry.

Web Collector configuration resides in the registry file:

products\i3fp\registry\instances\www\<*server-name*>\settings.xml

> You can set the configuration for all Web Collectors at once by modifying the file `products\i3fp\registry\products\indepth-web\focal-point.xml` Consider however that settings in the server registry will override the settings in the `focal-point.xml` file.

Product level configuration resides in the registry file: `products\i3fp\registry\products\indepth-web\focal-point.xml` Web Collector behavior is set via the registry entry <web-collector>

Cluster configuration resides in the registry file:

products\i3fp\registry\clusters\www\<cluster-name>\settings.xml

Instance registry

Configuring the instance registry only affects the specific instance. Every element in the instance registry can also be specified in the server registry file, in which case it affects all the instances on that server.

To change instance registry

1. On the relevant Precise node (proxy), manually change the products\i3fp\registry\instances\www\<**server-name**>\<**instance_name**>\settings.xml file.
2. Save the file.
3. Run the `update-main-registry` CLI command on the same Precise node.
    - **UNIX**. /infra/bin/psin_cli.sh -action update-main-registry -i3-user admin -i3-clear-password admin -registry-path /registry/clusters/www/<*server_name*>/<*instance_name*>
    - **Windows**. infra\bin\psin_cli.bat -action update-main-registry -i3-user admin -i3-clear-password admin -registry-path /registry/clusters /www/<*server_name*>/<*instance_name*>

This will force a registry sync from the node's FocalPoint to the main FocalPoint.

Cluster registry

Configuring the cluster registry affects all instances of the specific cluster.

All instances in the cluster need to be restarted via AdminPoint after changing the cluster registry.

To change the cluster registry:

1. On the relevant Precise node (proxy), manually change the products\i3fp\registry\clusters\www\<*cluster-name*>\settings.xml file.
2. Save the file.
3. Run the `update-main-registry` CLI command on the same Precise node.
   - **UNIX**. ./infra/bin/psin_cli.sh -action update-main-registry -i3-user admin -i3-clear-password admin -registry-path /registry/clusters/www/<*cluster_name*>
   - **Windows**. infra\bin\psin_cli.bat -action update-main-registry -i3-user admin -i3-clear-password admin -registry-path /registry/clusters/www/<*cluster_name*>

This will force a registry sync from the node's FocalPoint to the main FocalPoint.

Cluster registry changes override the FocalPoint registry changes.

connection-params (in general section)

The following table describes the registry elements for the connection-params (in the general section).

Table 1 connection-params (in general section)

Registry element	Description
site-name	Specifies the name of the Web application (context root). Only relevant for J2EE Web servers and only when the filter is not installed on the root Web application.
filter-connect-ext-cmd-line	Additional miscellaneous command line parameters for the psi_web_filter_connect utility which is used to send management URLs to the filter.
authorized-ips	Comma separated IPs list that specifies the IPs from which the Precise for Web agent management pages can be browsed. Other IPs that will try to access this status will be blocked. (127.0.0.1 is added by default).

ssl (in connection-params, in general section)

The ssl element in the registry is a "container" element that contains the use-client-certificate, cipher-list, and log-file elements. It also contains another "container" element: client-certificate. This "container" element contains the path, password, and key-path elements.

The following table describes the registry elements that can be used for the ssl instance registry.

Table 2 ssl elements

Registry element	Child element	Description
use-client-certificate		Specifies the use of a client certificate when Precise for Web tries to connect using Secure Socket Layer (SSL). Default value: false.
cipher-list		Specifies additional ciphers that OpenSSL uses on top of the default ciphers for SSL version 2 and 3 and Transport Layer Security (TSL) version 1. For more information, see the following URL: http://www.openssl.org/docs/apps/ciphers.html#cipher_suite_names.
log-file		Specifies the file that OpenSSL logging is written to.
client-certificate		Specifies client certificate details, if the monitored Web server requires it. Container for path, password, and key-path elements.
	path	Specifies the full path of the client certificate.
	password	Specifies the password to be used to decrypt the certificate file. The password itself should be encrypted using the Encrypt command that is part of the Precise CLI Utility. For more information, see the *Precise CLI Utility Reference Guide*.
	key-path	Specifies the path to the private key if it is different than the path to the certificate.

user-authentication (in connection-params, in general section)

The following table describes the registry elements for the user-authentication (in connection-params, in general section).

Table 3 user-authentication (in connection-params, in general section)

Registry element	Description
user	Specifies the user name for authentication.
password	Specifies the user password for authentication. The password itself should be encrypted using the Encrypt command that is part of the Precise CLI Utility. For more information, see the *Precise CLI Utility Reference Guide*.
domain	Specifies the domain name for authentication.
method	The authentication method. One of the following methods should be used: • basic • digest • ntlm

Server-side

The following table describes the registry elements for the server-side .

Table 4 Registry elements for server-side

Registry element	Description
ext-list	Specifies the extensions used, such as .html, .jsp, and .asp. For example: <ext-list>html,jsp,asp</ext-list> Default values: html, swe, asp, aspx, htm, jsp, php, gbl, cfm, pl, php3, do, IScript_PT_NAV_PAGELET, I_Script_AppHP, IScript_UniHeader_Frame, IScript_PT_NAV_INFRAME, iscript_apphp, iscript_timeoutwarning
ext-list-separator	Specifies the separator of the extension list. Default value: , (a comma)
stat-extension	Specifies the URL of the statistics module. Add this registry element only if you know that the statistics URL differs from the default URL for the specific Web server type. Default values: Apache: server-status?auto iPlanet: .perf WebLogic: weblogicstat WebSphere: websphere_stats
filter-trace	Specifies whether filter logging is used. Default value: false
filter-session-cookie	Specifies the name of the session cookie used by the application. Define this value to improve client-side correlation in the following Web servers: WebLogic, WebSphere, Tomcat, Oracle AS, Sun One, IIS, Apache 2.x. Default value: JSESSIONID
max-i4w-cookie-size	Specifies the maximum size of a Precise for Web cookie (in bytes). Default value: 2048
filter-ignore-files	A comma-separated list of files that the filter has to ignore. If a file contains any of these patterns, it is ignored. Only relevant for J2EE filters.
filter-use-encoding	Specifies the translation of URLs and URL parameters to unicode format. Set this parameter if URLs or parameters may contain non-ASCII characters. Default value: false
filter-encoding-name	Specifies the name of the URL encoding mechanism. Set this parameter if the server handles URLs that use neither UTF-8 nor local encoding, which are the default encoding types.

web-parameter-delimiters	A list of URL parameters delimiters. These delimiters are used to separate the parameters of a URL from the URL itself. For example, the URL: “index.html?param1=abc” has 1 parameter ‘param1’. The parameters delimiter in this URL is ‘?’. Use this element if your application uses non-standard delimiters (like ‘;’). To specify more than one delimiter, concatenate all delimiters into one string, for example: <parameter-delimiters>?;.</parameter-delimiters> sets 3 delimiters: ‘?’, ‘;’ & ‘.’ If more than 1 delimiter is used, the first one that is encountered in the URL is used. Default value: ? (question mark)
collect-post-parameters	If this registry element is set to true, the filter collects parameters that are passed in the request body (as opposed to parameters that are passed with the URL). For enabling the post-parameters collection on an IIS Web server, another action is needed (besides changing this flag) - For more information, see the *Adding post-parameters collection for IIS6* and *Adding post-parameters collection for IIS7* sections in the *Installing Web Tier Collectors* section of the *Precise Installation Guide*. Default value: false (for Siebel this is set to “true” (automatically)) This feature is not supported for Apache 1.3 Only Form post-parameters are supported. This means: Only submitted page forms of which the POST request contains the header `Content-Type: application/x-www-form-urlencoded` are supported. The post-parameters are passed as: `key1-value1&key2=value2&...` Any other type of post-parameters are not monitored. For example - if the POST data is XML. In any case no more than 1500 characters of POST data will be collected.
apache-pid-file-path	Sets the path for apache .pid file (commonly: httpd.pid). Should be set only when the .pid file path is not specified in the configuration file or in any other case, when the statistics agent fails to resolve it.

user-defined-transaction-name

By configuring the elements user-defined-transaction-name, you can change the collection method of the transactions. This is valid for Precise for Web and Precise TPM. The collection method can be represented by one of the following tags:

Table 5 Elements

Elements	Description
cookie-name	Collects the value of the specified cookie.
req-header-name	Collects the value of the specified request header.
parameter-name	Collects the value of the specified GET parameter (not relevant for POST parameters).
dom-element	Collects the value of the specified dom element. Applicable only if the client.-side agent is installed.
display-mode	The way the transaction name is displayed in Precise for Web and Precise TPM.

When configuring a user defined transaction name, you will need to specify the display mode between the <display-mode> tags. Possible values are:

- **prefix**. This will place the value in the before the original transaction name.
- **override**. This will replace the original transaction name with the configure transaction name.

When you configure a user defined transaction name, verify that it is a field with little variance (not many different values). For example, the cookie APP_VIEW is a good candidate to be used. In contrary to the SESSION_ID, which is a bad candidate, as it has a very large range of values. When you have a very large range of values, the Precise for Web schema tables grow very large.

To add a cookie - example

- If you have a cookie called USER_NAME in your application, you may add it to the user interface (as User Name), by adding the following UserDefined element:

```
<user-defined-transaction-name>
      <cookie-name>USER_NAME</cookie-name>
      <display-mode>prefix</display-mode>
</user-defined-transaction-name>
```

To add a dom element - example

- If you want to add the Title (DOM element) of your Web pages to the user interface (as My Title), you can add the following UserDefined element:

```
<user-defined-transaction-name>
        <dom-element>document.title</dom-element>
        <display-mode>prefix</display-mode>
</user-defined-transaction-name>
```

If the client-side agent is not installed on your Web server (only server-side is installed), you can not collect the Title DOM element.

user-defined-client-ip

By configuring the element user-defined-client-ip, you can change the way that client IP is collected by the server-side agent. You can instruct Precise for Web to collect the client IP from 3 different sources (instead of the HTTP request itself):

- Cookie
- Request header
- URL parameter

> You may want to change the way the client IP is collected when the Web server is behind some proxy, in which case the collected client IP is always the IP of the proxy.

apache-virtual-hosts

To separate Apache Virtual Hosts in different Precise applications, the Precise administrator needs to perform the following steps:

1. Create an Application in Precise for each Virtual Host. (For more information, see the Working with the *Application Installer wizard* section in the *P recise Installation Guide*.)
2. Place the Apache instance in the multiple instance view in each of the defined Precise applications.
3. Go to the instance registry and set the a registry entry for each of the Virtual Hosts using the following format:

```
<virtual-host-settings>
    <vhost-i>
        <vhost-name>name</vhost-name>
        <precise-application-name>application_name</precise-application-name>
    </vhost-i>
    …
        <!-- map all unmapped Virtual Hosts to this env (if not defined the information will be discarded) -->
    <vhost-i>
        <vhost-name>default</vhost-name>
        <precise-application-name>application_name</precise-application-name>
    </vhost-i>
<virtual-host-settings>
```

> In a case of an unmapped Virtual Host, transactions can either be displayed in the “default” application (“default” is set), or discarded if “default” is not set.

4. Restart the Web Collector on the monitored server by performing the following steps.
    a. Go to **AdminPoint>Agents**.
    b. Click **Stop** to stop all Web agents on the monitored server.
    c. Click **Start** to start all Web agents on the monitored server.

After performing these steps, you will see the unique information of each Virtual Host in each of the defined Precise applications.

The limitations of this feature are:

- Once the instance is configured as defined, each of the Virtual Hosts are linked to a specific application in Precise and the information cannot be shared in a multiple instance view.
- J2ee instances which are shared between two applications can cause unexpected results, and thus should be defined in two Virtual Host Precise applications.

Server registry

Configuring the server registry affects the Web collector and the instances on the server.

server-side

Table 6 client-side section

Registry elements	Description
use-tomcat-client-collector-as-proxy	If set to true, the instance will sent client-side data to the client collector. Default: true for Apache 1.3 and iPlanet false for the rest.

Table 7 verify section (inside web-collector)

Registry elements	Description
max-user-agent-length	Maximum allowed user agent (browser type) length. Records exceeding this value will be ignored (not loaded). Units: bytes Default: 256
max-user-defined-length	Maximum allowed user defined 1 & 2 length. Records exceeding this value will be ignored (not loaded). Units: bytes Default: 256

Table 8 server-side section (inside verify, inside web-collector)

Registry elements	Description
max-server-period	Maximum allowed server period (not including network period). Records exceeding this value will be ignored (not loaded). Units: milliseconds Default: 3600000 (1 hour)
max-future-time	Maximum allowed difference between current time and future record time. Records that have a later time the allowed will be ignored (not loaded). Units: milliseconds Default: 86400000 (1 day)
empty-http-status-allowed	If true, records with no HTTP status are not ignored. Default: true
fallback-http-status	If a record does not contain HTTP status and empty-http-status-allowed is set to true, sets the value to use for the HTTP status of that record. Default: 0
min-http-status, max-http-status	If fallback-http-status is not set (or set to 0) HTTP statuses outside this range cause the record to be ignored (not loaded). Default: 0, 599
empty-protocol-allowed	If true, records with no HTTP protocol are not ignored. Default: true
max-protocol-length	If the length of the HTTP protocol of a record exceeds this value, the record will be ignored (not loaded). Units: bytes Default: 5
allowed-protocols	If this is set, only HTTP protocols in this list are allowed, other values will cause the record to be ignored (not loaded). Default: none
empty-http-method-allowed	If true, records with no HTTP method are not ignored. Default: true
fallback-http-method	If a record does not contain HTTP method and empty-http-method-allowed is set to true, sets the value to use for the HTTP method of that record. Default: GET
allowed-http-methods	If this is set, only HTTP methods in this list are allowed, other values will cause the record to be ignored (not loaded). Default: none

Table 9 client-side section (inside verify, inside web-collector)

Registry elements	Description

max-text-period	Maximum allowed text period. Records exceeding this value will be ignored (not loaded). Units: milliseconds Default: 900000 (15 minutes)
max-rendering-period	Maximum allowed rendering period. Records exceeding this value will be ignored (not loaded). Units: milliseconds Default: 900000 (15 minutes)

The entries below are similar for 4 sections (with one exception*):

Table 10 summed section (inside server-side, inside output, inside web-collector) raw section (inside server-side, inside output, inside web-collector), summed section (inside client-side, inside output, inside web-collector), raw section (inside client -side, inside output, inside web-collector)

Registry elements	Description
max-files	Maximum files allowed in the output directory. After exceeding this value, no more files will not be written. Default: 1440
max-recs-in-file	Maximum records allowed in 1 output file. After exceeding this value, a new file will be created. Default: 10000
max-time-to-write-file	Maximum time to have 1 file open. Units: milliseconds * Default for summed: 900000 (15 minutes) *Default for raw: 60000 (1 minute)
max-file-size	Set the maximum for file size. Units: bytes Default: 10485760 (10MB)

Table 11 server-side section (inside tac, inside smartlink, inside web-collector), client-side section (inside tac, inside smartlink, inside web-collector)

Registry elements	Description
max-files	Maximum files allowed in the output directory. After exceeding this value, no more files will not be written. Default: 1440
max-recs-in-file	Maximum records allowed in 1 output file. After exceeding this value, a new file will be created. Default: 10000
max-file-size	Set the maximum for file size. Units: bytes Default: 10485760 (10MB)

Table 12 server-side section (inside input, inside web-collector), client-side section (inside input, inside web-collector)

Registry elements	Description
delete-files	When set to true, input raw files are deleted after being processed. Default: true
max-error-files	Maximum number of error files to move to the errors directory when errors occur on files. Default: 10

Table 13 statistics section (inside ivp, inside web-collector)

Registry elements	Description

enabled	When set to false, validation of the statistics component of the web collector are not performed in the verify action (in AdminPoint). Default: true

Table 14 processing section (inside ivp, inside web-collector)

Registry elements	Description
Enabled	When set to false, validation of the core web collector are not performed in the verify action (in AdminPoint). Default: true
server-amount-limit-for-no-client	When client/server correlation is enabled, the web collector warns - in the verify action (in AdminPoint) – that server records were received but no client records did. This number sets the threshold for this warning. Default: 1000
client-amount-limit-for-no-server	When client/server correlation is enabled, the web collector warns - in the verify action (in AdminPoint) – that client records were received but no server records did. This number sets the threshold for this warning. Default: 500
server-amount-limit-for-no-network	When server/network correlation is enabled, the web collector warns - in the verify action (in AdminPoint) – that server records were received but no network records did. This number sets the threshold for this warning. Default: 1000
network-amount-limit-for-no-server	When server/network correlation is enabled, the web collector warns - in the verify action (in AdminPoint) – that network records were received but no server records did. This number sets the threshold for this warning. Default: 1000

Table 15 web-collector section

Registry elements	Description
preferred-data-port	Will be used by the web collector for communication with server and network agents. If this port is busy, any available port will be used. Default: 20999

Table 16 summary section (inside web-collector)

Registry elements	Description
enabled	When set to false, summary processing is not performed by the web collector. This setting is meaningful only on the monitored server (and not on the FP machine where it is ignored). Setting this to false (along with client-server-correlation) sets the work mode to Basic Mode. Default: true

Table 17 server-side section

Registry elements	Description
Enabled	When set to false, server-side collection is disabled. Default: true
client-server-correlation	When set to false, client/server correlation is disabled and in fact the instance will work in Basic Mode (this does not have meaning on the FP machine). Default: true
server-network-correlation	When set to false, server/network correlation is disabled (this does not have meaning on the FP machine). Default: true
filter-do-collector-settings	When set to false, collector settings activity (handling of URL parameters) is done in the web collector instead of in the web filter. Default: true (processing done on filter)

filter-do-data-patterns	When set to false, data patterns processing is done in the web collector instead of in the web filter. Default: true (processing done on filter)
filter-do-locations	When set to false, data locations processing is done in the web collector, instead of the web filter. Default: true (processing done on filter)

Table 18 client-side section

Registry elements	Description
enabled	When set to false client-side collection is disabled Default: true

Table 19 statistics section

Registry elements	Description
enabled	When set to false statistics collection is disabled Default: true
max-files (in output section)	Maximum files allowed in the output directory. After exceeding this value, no more files will not be written. Default: 2000

Table 20 processing section (inside web-collector)

Registry elements	Description
max-data-processor-threads	Maximum number of threads that will be invoked by the web collector to handle data. Default: 50
write-dp-period-ms	The initial wait time of the serialization timer that serializes the current data to files. Units: milliseconds Default: 120000
write-dp-delay-ms	The cycle time of the serialization timer that serializes the current data to files. Units: milliseconds Default: 120000

Table 21 server-side section (inside processing, inside web-collector)

Registry elements	Description
input-queue-max-size	Set the maximum number of records in the input queue. After the queue reaches this size, we start dropping records. May be used to temporarily ease memory consumption. Default: 15000
aging-period	Max time for server records to wait in memory for correlation (with both client & network records). Units: milliseconds Default: 90000 (1.5 minutes)
aging-period-when-using-client-collector	Max time for server records to wait in memory for correlation (with both client & network records) when this instance uses the client collector. Units: milliseconds Default: 30000 (30 seconds)

Table 22 client-side section (inside processing, inside web-collector)

Registry elements	Description

input-queue-max-size	Set the maximum number of records in the input queue. After the queue reaches this size, we start dropping records. May be used to temporarily ease memory consumption. Default: 10000
aging-period	Max time for client records to wait in memory for correlation (with server records). Units: milliseconds Default: 90000 (1.5 minutes)
aging-period-when-using-client-collector	Max time for client records to wait in memory for correlation (with server records) when this instance uses the client collector. Units: milliseconds Default: 180000 (3 minutes)
max-text-period-before-fix	Max time in milliseconds for client records first byte time. If the first byte time exceeds the maximum, it will be fixed by taking the server + network time. Default: 900000 milliseconds

Table 23 network section (inside processing, inside web-collector)

Registry elements	Description
input-queue-max-size	Set the maximum number of records in the input queue. After the queue reaches this size, we start dropping records. May be used to temporarily ease memory consumption. Default: 15000
aging-period	Max time for network records to wait in memory for correlation (with server records). Units: milliseconds Default: 90000 (1.5 minutes)

Table 24 server-side section (inside filtering, inside web-collector)

Registry elements	Description
enabled	When set to false, server-side filtering is disabled Default: true
reset-identifiers	A list of identifiers that will be reset in the filtering-summary process (when needed). The identifiers will be reset, one by one, in the order they appear here until the top N limit is reached. Valid values: • url • domain • userAgent • userDefined1 • userDefined2 • appUserName • clientIP • location • city • state • country • siebelView • urlParams • protocol • httpMethod • sessionID Default: httpMethod, userAgent, protocol

Table 25 client-side section (inside filtering, inside web-collector)

Registry elements	Description

enabled	When set to false, client-side filtering is disabled Default: true
reset-identifiers	A list of identifiers that will be reset in the filtering-summary process (when needed). The identifiers will be reset, one by one, in the order they appear here until the top N limit is reached. Valid values: • url • domain • protocol • userAgent • userDefined1 • userDefined2 • appUserName • clientIP • location • city • state • country • siebelView • title • host • connectionType • privateIP • login Default: userAgent, protocol

FocalPoint registry

The following section describes the FocalPoint registry element.

user-agents

The user-agent element consists of a set of definitions of user agents. A user agent is a device that may access a Web server. A typical user agent is a browser like Microsoft Internet Explorer or Firefox. Other types of user agents are automatic HTTP traffic generators, like Precise Insight Inquire. The Precise for Web Collector agent, automatically collects the user agent of every request that it monitors. To extract a comprehensive user agent name and version from the user agent header of the request, the agent uses the definitions in the user-agents element. By default, the user-agent element contains a definition for most of the industry's known Web servers, like Microsoft Internet Explorer, Firefox, Netscape, Mozilla as well as the integrative Precise Insight Inquire traffic generator. However, you can add any number of user agents to the list, or change the definition of an existing user agent to adjust it to customers private needs or conventions. For example, the definition for the Netscape browser is:

```
<user-agent id="Netscape" display-string="Netscape version">
    <version-search-string>Netscape</version-search-string>
    <search-string value="Netscape"/>
</user-agent>
```

The default user agents that the product supports out-of-the-box are:

- Precise Insight Inquire
- Microsoft Internet Explorer
- Firefox
- Opera
- Netscape
- Mozilla
- Konqueror
- iCab
- Safari

The following table describes the registry elements for the user-agent in the FocalPoint registry.

Table 26 Registry elements

Registry elements	Description
user-agent	A user agent definition that Precise for Web uses to find and describe user agents.
user-agent.id	A string identifier for the user agent. This identifier is not the display name.
user-agent.display-string	The display string that is shown in the user interface for this user agent. You can use the *version* token which is replaced with the detected user agent version.

Adding Web filter parameters

If you use an Apache, iPlanet, Sun ONE, WebSphere, Tomcat, Oracle Application Server, SAP, J2EE, or BEA WebLogic server, you can specify some Web filter settings that the Web server loads on startup. These parameters are the same for all Web servers. To add them, follow the instructions for the specific Web server. For more information, see Addable parameters.

> Modification of IIS Web filter parameters is not yet supported.

To add parameters on an Apache server

1. Open the httpd.conf file.
2. After the LoadModule pssfilter_module line, add the parameter.

For example:

LoadModule pssfilter_module... PssFilterTrace 1

To add parameters on an iPlanet server

1. Open the magnus.conf files.
2. At the end of the Init fn=pss-init line, add the parameter.

For example:

Init fn=pss-init PssFilterTrace="1"

To add parameters on any J2EE server by editing the web.xml file

1. Open the web.xml file.
2. Add the parameters to the <filter> element in the following format:

```
<init-param>
<param-name>xxx</param-name>
<param-value>yyy</param-value>
</init-param>
```

For example:

```
<filter>
<filter-name>pssFilter</filter-name>
    <description></description>
    <filter-class>com.precise.ifweb.Collection.Filters.pssFilter
    .WLFilter</filter-class>
    <init-param>
        <param-name>PssFilterTrace</param-name>
       <param-value>1</param-value>
    </init-param>
</filter>
```

Addable parameters

The following table lists all parameters that you can add to the Web filters. This mechanism should only be used for the instance name and `<i3_root>`. Any other parameter should only be added after consultation with the Precise Enterprise Support Team.

Table 27 Addable Web filter parameters

Parameter	Description
PssInstance Name	Specifies the name of the filter's instance.
PssInstanceID	The instance ID as specified in the infrastructure database.
PssI3Root	Specifies the path to the Precise installation directory.
PssReadConfFile	Specifies whether to read the configuration files on filter initialization. If this parameter is set to FALSE, parameters are taken from the Web server only. Default: TRUE
PssRestartFilesPath	Specifies an alternate location for the pss_restart file. By default this file is located under the directories of the Web server itself (the exact location is web server type dependent). Set this to a different path if the file cannot be created at the default location.

PssInitLog	Specifies whether to write initialization log messages to standard output. In some cases the standard output is written to the Web server log. Applies only to J2EE-based application servers (Sun ONE, WebSphere, Tomcat, Oracle Application Server, SAP J2EE, or BEA WebLogic)

The following table gives a comparison of the parameter and its equivalent registry element.

Table 28 Parameter and Registry element

Parameter	Registry element
PssFilterTrace	filter-trace
PssIgnoreFiles	filter-ignore-files
PssSessionCookieName	filter-session-cookie
PssMaxI4WCookieSize	max-i4w-cookie-size
PssCorrelationMode	web-disable-correlation The opposite of this registry element works as its equivalent parameter.
PssExtList	ext-list
PssExtListSeparator	ext-list-separator
pcs_shm_dir	general-pcs-shm-dir
PssURLEncodingName	filter-encoding-name
PssUseURLEncoding	filter-use-encoding

Grouping consecutive URLs by pattern

The Activity tab of Precise for Web displays information on the URLs and pages visited. In some cases, the database tables that contain this information can easily swell and harm the performance of the user interface, for example when URLs are generated automatically.

To reduce the accumulation frequency of URLs, Precise for Web lets you group consecutive URLs by pattern by editing the data-patterns registry element. Each of the data-pattern elements (urls, pages, titles, domains, and user-defined-transaction-name) can be specified with methods. The available methods are prefix, suffix, prefix-by-key, and regex.

You can add as many methods and patterns to any of the sections (<urls>, <pages>, <titles>, <domains>, <user-defined-transaction-name>).

The tags <methodN> and <patternN> should be numbered so that 'N' is replaced with a consecutive number.

In URLs and Pages, pattern definitions only work on the path part. For example, in the URL `/index.jsp?id=&eventid=`, the path part is `/index.jsp` and the parameters part is `id=&eventid=`. To handle collection definitions (aka grouping) on the parameters part, refer to the Collector Settings section. In the above example, any pattern that is defined on the URL, ignores the parameters part (`id=&eventid=`).

The following table describes the available methods.

Table 29 methods

Method	Description
prefix	Take only the defined prefix. For example: <pattern1>abraham</pattern1> - For URL 'abraham123.html' you will see 'abraham%'.
suffix	Take only the defined suffix. For example: <pattern1>html</pattern1> - For URL 'abraham123.html' you will see '%html'.

prefix-by-key	Take text up to the defined key (not including that key). For example: <pattern1>;jsessionid</pattern1> - For URL 'abraham123.html;jsessionid' you will see 'abraham123.html%' (if showReplaceChar="false", you will get 'abraham123.html').
regex	Use the given regular expression to change the given URL. For example: <pattern1 replace="$1">service(.*)</pattern1> - For URL 'service123' you will get '123' (for more on regular expressions, consult your support representative).

The following code is an example of the element's content for an URL element when all the grouping methods are activated:

```
<data-patterns>
    <urls>
        <execute>true</execute>
        <methods>
            <method1>
                <name>prefix</name>
                <patterns>
                    <pattern1>/redirectlogon.html</pattern1>
                </patterns>
            </method1>
            <method2>
                <name>suffix</name>
                <patterns>
                    <pattern2>ads.com</pattern2>
                </patterns>
            </method2>
            <method3>
                <name>prefix-by-key</name>
                <patterns>
                    <pattern3>:jsessionid=</pattern3>
                </patterns>
            </method3>
            <method4>
                <name>regex</name>
                <patterns>
                    <pattern4 replace="$1">\d+: (.*)</pattern4>
                <patterns/>
            </method4>
        </methods>
    </urls>
</data-patterns>
```

When URL pattern prefix grouping is activated, a percentage sign (%) is added at the end of a URL. For example, the URL /redirectlogon.html/121231245 is saved in the database as /redirectlogon.html%.

To group consecutive URLs by pattern

1. Open the relevant registry file (instance registry, server registry, or focal-point registry) and find the data-patterns element (If it does not exist, you can add it as appears in the server registry)
2. Edit the data-patterns element in the following way:
    - Set the required <execute> elements to **true**.
    - Choose the method (prefix, suffix, prefix-by-key, or regex) in the required element (urls, pages, titles, domains).
    - Define the URL patterns in the <pattern> elements. You can include as many <pattern> elements as required.
    - If you choose the regex method, use the replacement string. For example:
      <pattern replace="$1">\d+: (.*)</pattern>
3. Save and close the file.
4. Restart the Precise for Web server-side and dynamic instrumentation agents monitoring the relevant environment.

Temporarily stopping the maximum number of table rows test

If the database tables that contain the URLs and pages that are visited exceed the maximum number of rows, the installation verification procedure (IVP) issues an error message. To prevent this message from reappearing until the accumulation frequency of URLs has been lowered, you can temporarily stop the 'maximum number of table rows' test.

To temporarily stop the 'maximum number of table rows' test

1. Open the *<i3_root>*\products\www\ivp\resources\functions_ivp.xml file in an editor:
2. In the file, set the <max-rows-for-alert> element to **0**.
3. Save and close the file.

Dynamic instrumentation

When you install a Web server agent, you are asked whether you would like to dynamically instrument Web pages. Also when you select an existing instance, you can edit it, so that the dynamic instrumentation can be applied. In both cases this results in an additional entry in the AdminPoint window. The new entry shows identical instance information for all but one column: the Agent column. In this column the agent is shown as Web Instrument agent. The Instrument.xml file handles the dynamic instrumentation configuration.

About the instrumentation configuration file (Instrument.xml)

If the instance is part of a cluster, the Instrument.xml file is located in the <*FP server*>\products\i3fp\registry\clusters\www\<*name of cluster*> directory.

If the instance is not part of a cluster, the `Instrument.xml` file is located in the <FP server>\products\i3fp\registry\instances\www\<monitored instance name>\<instance name>\<instance name> directory. If you make changes in the `Instrument.xml` file, then you will also have to make the changes in the proxy (or proxies). The recommended way to do so it you use the CLI command as describe before. See Instance registry.

Following is an example of the file structure that is explained in the section following the example:

```
<dynamic-inst>
        <enable>true</enable>
        <handle-js>true</handle-js>
        <prolog-string> <![CDATA[<!DOCTYPE html PUBLIC "-//W3C//DTD
        XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/ xhtml1-transitional.dtd">]]>
        </prolog-string>
        <no-recompress>true</no-recompress>
        <accumulate-on-parsing>true</accumulate-on-parsing>
        <max-parsing-bytes>2048</max-parsing-bytes>
        <js-mime-type>text/javascript</js-mime-type>
        <instance-id>1234</instance-id>
        <strict-servlet-api>true</strict-servlet-api>
        <private-log>false</private-log>
        <i3-log>false</i3-log>
    <webgarden>
        <enable>false</enable>
        <sampling-interval>2000</sampling-interval>
    </webgarden>
    <user-agents>
            <user-agent name="debug">
                <i3-log>false</i3-log>
                <private-log>false</private-log>
                <accumulation>default</accumulation>
                <parsing>false</parsing>
                <second-script>false</second-script>
                <enable>true</enable>
            </user-agent>
            <user-agent name="MSIE">
                <i3-log>false</i3-log>
                <private-log>false</private-log>
                <accumulation>default</accumulation>
                <parsing>false</parsing>
                <second-script>false</second-script>
                <enable>true</enable>
            </user-agent>
            <user-agent name="Default">
                <i3-log>false</i3-log>
                <private-log>false</private-log>
                <accumulation>default</accumulation>
                <parsing>false</parsing>
                <second-script>false</second-script>
                <enable>true</enable>
            </user-agent>
    </user-agents>
    <include-ip>
        <range from="123.23.1.1" to="123.23.1.255" />
        <range from="125.2.1.1" to="125.2.1.255" />
    </include-ip>
    <url-exclude>
        <suffix>.gif</suffix>
        <suffix>.jpeg</suffix>
    </url-exclude>
    <url-include>
        <prefix>/products</prefix>
        <prefix>/petstore/accounts</prefix>
        <suffix>createData.dll</suffix>
    </url-include>
    <include-char-encoding>
        <char-enc>shift_jis</char-enc>
        <char-enc>jis</char-enc>
    </include-char-encoding>
    <req-header-exclude>
        <header name="content-type" value="application/x-www-form- urlencoded" />
    </req-header-exclude>
</dynamic-inst>
```

About the Instrument.xml tags

The following table contains the tags that can be encountered in the Instrument.xml file. "not applicable" means that the tag has no influence on that specifically chosen instance type.

Table 30 Instrument.xml tags

Tag	Description

dynamic-inst	The first tag of the dynamic instrumentation configuration file. J2EE: mandatory ISAPI: mandatory
enable	Enables or disables the instrumentation filter globally. J2EE: optional ISAPI: optional Values: true, false Default value: true
handle-js*	Enables or disables the JavaScript handling by the filter. J2EE: optional ISAPI: optional Values: true, false Default value: true
strict-servlet-api (J2EE)	Forces compatibility with the J2EE servlets specifications. J2EE: optional ISAPI: not applicable Values: true, false Default value: true
js-mime-type	Uses js-mime-type protocol. J2EE: optional ISAPI: optional Values: if possible to use dynamically obtained, otherwise use application/x-javascript Default value: if possible use dynamically obtained, otherwise use application/x-javascript
instance-id*	The filter init process fills this tag: • If the tag exists and is empty, it is filled with the right instance-id. • If the tag exists with a value, the value is not changed. • If the tag does not exist, it is not created. • If the filter is able to read the value of the tag, it inserts it to a cookie on the response. The tag should only have an empty value on load balanced environments with no server side installed. Default value: empty
prolog-string*	An extra string that is injected during instrumentation (in addition to the regular callout to JavaScript). Usually it is used to add the <!Doctype element (because according to some browsers this tag should come at the beginning of the Web page). J2EE: optional ISAPI: optional
no-recompress	For compressed content. If set to true, insert the instrumentation script without recompression. Apache: applicable for version 2.0 and higher Values: true, false Default value: true

accumulate-on-parsing	If parsing is on, accumulate response buffers till the <html> or <head> tag is found (see also the note after this table). J2EE: optional ISAPI: optional Apache: applicable for version 2.0 and higher Values: true, false Default value: true
max-parsing-bytes	Looks for the <html> tag or the <head> tag only in the first n bytes specified. J2EE: optional ISAPI: optional Apache: applicable for version 2.0 and higher Values: number of bytes, -1 for full parsing Default value: 2048
private-log	Controls the writing of messages that need to be logged before the user-agent is obtained, to the private log. J2EE: optional ISAPI: optional Values: true, false Default value: false
i3-log	Enables the i3 log globally. J2EE: not applicable ISAPI: not applicable Values: true, false Default value: false
webgarden*	Introduces the Webgarden management section.
enable	Enables or disables “Webgarden” support. Values: true, false Default value: true for IIS and SAP WAS Web servers, false on any other Web server
sampling-interval	Setting the sampling interval (in milliseconds) in which each process in the “Webgarden” updates its status. Value: any number higher than 500 Default value: 2000
user-agents	Introduces the user-agents section. J2EE: mandatory ISAPI: mandatory
user-agent	Describes the configuration per user-agent. J2EE: mandatory, only for user-agent with name "Default" ISAPI: mandatory, only for user-agent with name "Default"
accumulation	Controls whether to perform accumulation of chunks/parts of the Web content and then preforming instrumentation all together. The default is not to use it and perform instrumentation on-the-fly (and perform accumulation only if necessary). There is an option to force accumulation (true) or to disable instrumentation when accumulation must be done (false). J2EE: mandatory ISAPI: mandatory Values: default, true, false Default value: default

parsing	Controls whether to perform light-parsing to inject the first script after <html> or <head> (what is recognized first). "false" means to inject the script at the beginning of the page, without parsing. J2EE: mandatory ISAPI: mandatory Values: true, false Default value: true
enable	Enables or disables the instrumentation filter per user-agent. J2EE: mandatory ISAPI: mandatory Values: true, false Default value: false
second-script	Controls whether to inject a second-script at the end of the page. This option is reserved for future use. J2EE: mandatory ISAPI: mandatory Values: true, false Default value: false
i3-log	Enables or disables the i3 log per user-agent. J2EE: mandatory ISAPI: mandatory Values: true, false Default value: false
private-log	Enables or disables the private log. J2EE: mandatory ISAPI: mandatory Values: true, false Default value: false
user-agent name	Relates to the browser user-agent that includes this value. The order in the list is important. J2EE: mandatory ISAPI: mandatory Values: Default (mandatory) or free text (such as MSIE or FireFox) that is part of the user-agent string. Default should always be the last user in the list and the values are case-sensitive. Default value: For more information, see the following URL: http://www.zytrax.com/tech/web/browser_ids.htm
include-ip	Introduces the section in which you define the IP addresses that should be included in the dynamic instrumentation process. By default all IPs are included. J2EE: optional ISAPI: optional

range from	Beginning of the IP address range. J2EE: optional ISAPI: optional Value: IP address
range to	End of the IP address range. J2EE: optional ISAPI: optional Value: IP address
url-exclude	Introduces the section in which you define the URLs that should be excluded. By default all URLs are included. J2EE: optional ISAPI: optional
url-include*	Introduces the section in which you define the URLs that should be included. By default all URLs are included. J2EE: optional ISAPI: optional
Prefix	Defines the URL prefix.
suffix*	Defines the URL suffix.
include-char-encoding	Introduces the section for additional character encoding that is supported in the filter (in addition to the built-in character support). J2EE: optional ISAPI: optional
char-enc	Defines additional character encoding that is supported in the filter (in addition to the built-in character support). Use if your pages are written in none Unicode character encoding. J2EE: optional ISAPI: optional
req-header-exclude	Introduces the section that defines the HTTP request headers for which HTTP requests is excluded. J2EE: optional ISAPI: optional
header name	Defines the HTTP header name.
header value	Defines the specific HTTP header value or "*".

The tags that are marked with an asterisk (*) appear in the Frequently Asked Questions document.

The terms “accumulation”, “parsing (without accumulation)”, and “accumulate on parsing” can be described as follows:

- **Accumulation**. Keep all response buffers till you have all the document and if parsing is on, only then parse the document and instrument if the <html> or <head> tag is found.
- **Parsing (without accumulation)**. On the fly check each response buffer and find the <html> or <head> tag. Once found - instrument.
- **Accumulate on parsing**. Keep response buffers only until the on the fly parsing finds the <html> or <head> tag or till the max-parsing-bytes is reached, and then release the buffers with the right content-length.

About include and exclude tags

The following list describes the order in which the include and exclude tags are handled in the instrument.xml file:

- The optional tag "enable" can disable the whole instrumentation process for all HTTP responses.
- The "include-ip" section is checked first. If there is no "include-ip" section, all IPs can be instrumented.
- The "url-include" section is checked before the "url-exclude" section.

If one "url-include item is specified, this HTTP request must be instrumented. If no "url-include" section is specified, all URLs are included.

If one or more "url-exclude" item(s) is (are) specified, the HTTP request(s) must be skipped.

- If the filter recognizes a "content-type" that does not contain "text/html", the instrumentation process is skipped for that particular HTTP response.