



Directory Scanner Service

Administrator Manual

Software Version 1.0



EducateIT GmbH

Steinenbergstrasse 7, 4532 Feldbrunnen, Switzerland

Phone: +41 32 355 39 00

E-Mail: info@educateit.ch

Web: <http://www.educateit.ch/>

Directory Scanner Service – Administrator Manual

Copyright: ©2018 by EducateIT GmbH

EducateIT GmbH
Steinenbergstrasse 7
CH-4532 Feldbrunnen
Switzerland

Phone: +41 32 355 39 00
E-Mail: info@educateit.ch
Web: <http://www.educateit.ch/>

Illustrations and Text: TE
Layout and Design: [TE](#)

Last Modification: 2018-01-05 17:30

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by and means, electronic, mechanical, photocopying, recording, or otherwise, without the prior consent of the EducateIT GmbH. The EducateIT GmbH assumes no responsibility for errors or omissions, or for damages that may result from the use of information (especially including example configurations and listings) contained herein.

Table of Contents

Table of Contents	3
Preface	5
Features	5
See Also	5
Configuration	7
network	7
secrets	11
pathMappings	13
allowedPaths	15
cacheFilePath	15
Security Checklist	17

Preface

The *Directory Scanner Service* is a lightweight service to calculate the size of directory trees and check if a directory path exists or not. Permissions to this service can be configured to limit the access to a very narrow set of users and directories.

This service exists to implement a strong privilege separation. Calculating the size of user profile directories requires administrative access, which only this services requires. Clients of this service can run under unprivileged accounts and use this service to calculate directory sizes, without the risk this clients have access to the user data.

Features

- Extensive configuration of permissions.
- Very low *CPU* and network impact when idle.
- Encrypted *SSL* connection and strong authentication.
- Simple *HTTPS API* to use with scripts or custom applications.
- Optional caching to speed up directory size calculations.
- Path mapping and filtering.

See Also

The service is using the *Configuration B* system for the configuration. To understand all aspects of the configuration, it is recommended to read the document “*Configuration B - Administrator Manual*”.

The service also uses the *EducatelT Service Framework* and the *EducatelT Log System*. Both systems provide several features which are explained in their own manuals: “*Service Framework – Administrator Manual*” and “*Log System – Administrator Manual*”.

You should get all these documents along with this manual. If not, please request them from us.

Configuration

The configuration of this application is based on “*Configuration B*” and is supporting all features of this system. The configuration is stored in the file “configuration.xml” which is placed in the program directory beside the executable, but can also be placed in any of the searched directories to be more OS compliant. Read the “*Configuration B*” manual, section “Files and Locations” for details.

[Illustration 1 on page 8](#) shows the structure of the application configuration. The following sections describe each value of the configuration in detail.

network

The network list configures the different protocol and ports of the service. You can define one entry per protocol. If you omit a protocol, it is disabled.

See [Listing 1 on page 7](#) for a common example of this configuration. Both protocols are enabled. The message protocol is only accessible from the local host, but the *HTTP protocol* is accessible from any host. Both protocols use the default port and no customized *SSL profile*.

See [Listing 2 on page 10](#) for an example where only the *HTTP protocol* is enabled. The protocol uses a custom port number 18010 which is accessible from outside of the host. It also uses a custom *SSL profile* with the identifier *SslProfile*.

```
<List name="network">
  <ListEntry>
    <Value name="protocol">Message</Value>
    <Value name="scope">LocalHost</Value>
  </ListEntry>
  <ListEntry>
    <Value name="protocol">HTTP</Value>
    <Value name="scope">Any</Value>
  </ListEntry>
</List>
```

Listing 1 The network configuration with both protocols enabled.

Module	Application
List	allowedPaths
Definition	Value List Definition
Constraints	Has to be a valid regular expression.
Value	cacheFilePath Optional String
List	network Optional
Definition	List Entry Definition
Value	port Optional Integer
Constraints	- Minimum value: 0 - Maximum value: 65535
Default	0
Value	protocol String
Constraints	- Must not be empty. - Has to be one of the following values: Message, HTTP
Value	scope String
Constraints	Has to be one of the following values: None, LocalHost, Any
Default	LocalHost
Value	sslProfile Optional String
Constraints	Maximum length: 64 character(s).
List	pathMappings
Definition	List Entry Definition
Value	match String
Constraints	- Must not be empty. - Has to be a valid regular expression.
Value	replacement String
List	secrets
Definition	List Entry Definition

Illustration 1 The Application module for the service configuration.

Value	authorizedNetworks	Optional	String
Value	authorizedPath	Optional	String
Constraints	Has to be a valid regular expression.		
Default	.*		
Value	authorizedProtocols	Optional	String
Constraints	Has to be one of the following values: Any, Message, HTTP		
Default	Any		
Value	authorizedServices	String List	
Constraints	Each element of in this list has to be one of the following values: DirectorySize, DirectoryExists		
Value	key	String	
Constraints	Must not be empty.		
Value	label	String	
Constraints	Must not be empty.		
Value	secret	Password	
Constraints	Must not be empty.		

```

<List name="network">
  <ListEntry>
    <Value name="protocol">HTTP</Value>
    <Value name="scope">Any</Value>
    <Value name="port">18010</Value>
    <Value name="sslProfile">SslProfile</Value>
  </ListEntry>
</List>

```

Listing 2 A network configuration where only the *HTTP* protocol is enabled.

protocol

The `protocol` value configures the protocol identifier for the entry. There are only two value protocol identifiers which are case insensitive:

- Message
- HTTP

The identifier `Message` configures the message interface which is the compressed and fast protocol used by most of the EducatelT applications.

The identifier `HTTP` configures the *HTTP/REST* based protocol which can be used with own scripts and services.

port

The optional `port` value configures the port where the service should listen for the specified protocol. If you omit this value, the default port for the protocol is used.

Protocol	Default Port
Message	17661
HTTP	17662

scope

The `scope` value configures the scope for the specified protocol. It configures if the port for the protocol is just available on the local host, or if the port is accessible from outside of the host.

Scope	Description
Any	Port is accessible from outside of the host.
LocalHost	Port is only accessible from local host.
None	The protocol is disabled.

sslProfile

The optional `sslProfile` value selects a custom *SSL profile* for the protocol. Using a custom profile you can define a own SSL certificate and certificate authority for the service and also control if the client certificate shall be verified.

The profile itself is configured in a own SSL module block in the configuration. See the document “*Configuring Custom SSL Certificates*” for details.

secrets

The `secrets` list contains a list of key and secret pairs for authentication with the service. Each entry also controls the allowed services, protocols and paths which can be used with the given key.

Best practice is to create a own key and secret pair for each individual client instance in the network. Label this entry with a unique identifier which names this client and access in a logical way.

See [Listing 3 on page 11](#) for a common configuration of the secrets list. It defines two entries. The first entry is used for the user directory extension of *Raptor*, also this entry is limited to the message protocol.

The second entry is used for a custom scrip which is checking the size of some caches. This entry is limited to the *HTTP protocol* and can only access the `DirectorySize` service. It also uses a regular expression to limit the access to the directory “cache” on the server “example05” or “example20”.

```
<ListEntry>
  <Value name="label">RaptorUserDirectory</Value>
  <Value name="key">pfxYhg...VgnV2(~I3u]M</Value>
  <Value name="secret">*1F+pw...NZa7v8dA==</Value>
  <Value name="authorizedProtocols">Message</Value>
  <Value name="authorizedServices">
    DirectorySize
    DirectoryExists
  </Value>
</ListEntry>
<ListEntry>
  <Value name="label">CacheCheckScript</Value>
  <Value name="key">bTK3tr... (Q7o3AH</Value>
  <Value name="secret">*oMGuy...qp0Vs7A==</Value>
  <Value name="authorizedProtocols">HTTP</Value>
  <Value name="authorizedServices">
    DirectorySize
  </Value>
  <Value name="authorizedPath">(?!i)^//example(?:05|20)/cache/</Value>
</ListEntry>
```

Listing 3 A common configuration for the secrets list.

This example demonstrates the basic principle to restrict the access to the minimum required by each client. It also assumes that there is a proper `allowedPaths` configuration is in place to limit each access to some given locations.

label

The `label` value configures an unique label for the entry to identify the access to the service. This label is used in the log files, therefore you should choose one which clearly identifies the client which is accessing the service.

key

The `key` value configures the key you created with the *EducateIT Secret Generator* application.

secret

The `secret` value configures the secret you created with the *EducateIT Secret Generator* application.

authorizedNetworks

The optional `authorizedNetworks` value can contain a white-space separated list of networks from where this secret can be used. The *IP address* of each incoming connection is checked, if it is part of one of the specified networks.

We recommend limiting the access to a given network if you know the source *IP range* for any connections.

authorizedServices

The required value `authorizedServices` controls which services can be accessed by the used secret entry. There are two supported values:

- `DirectorySize`
- `DirectoryExists`

The `DirectorySize` service allows to start the calculation of the size of a directory tree.

The `DirectoryExists` service checks a number of paths, if they exist.

authorizedProtocols

The optional `authorizedProtocols` value limits the protocols supported by the given secret entry. The following identifier are valid for this value:

- `Any`
- `Message`
- `HTTP`

authorizedPath

The optional `authorizedPath` value configures a regular expression which has to match each path submitted by the client using this secret entry. If you omit this value, all paths from the client will be accepted.

This regular expression checks the path after normalization, but before any path mapping is done. Normalization removes any invalid characters or directory names from the path and converts back slashes into regular slashes.

The regular expressions is a *Perl* compatible (*PCRE*) one, supporting almost the complete range of features. To make the whole regular expression case insensitive, add for example the prefix `(?i)`.

pathMappings

The `pathMappings` list can contain mappings between paths sent to the service and the actual local path which is used for the size calculation or existing checks. While the list has to be configured, it can be left empty if there are no path mappings.

Each entry consists of a regular expression match which has to match the input path. This regular expression is tested after normalization of the path. If an entry matches, the matched part is replaced with the string in the replacement value of the entry.

All entries in the list are tested in the configured order. The processing does not stop after a match. If a path matches and modifies the path, the next entries will process the modified path, not the original one.

See [Listing 4 on page 13](#) for an example configuration for the `pathMappings` list. This example configures two mappings.

The first mapping checks if the normalized path starts with `//example05/users/`, in this case this part of the path is replaced with the local path `c:/files/users`. The idea begin

```
<List name="pathMappings">
  <ListEntry>
    <Value name="match">(?i)^//example05/users/</Value>
    <Value name="replacement">c:/files/users/</Value>
  </ListEntry>
  <ListEntry>
    <Value name="match">(?i)^//example02/caches/</Value>
    <Value name="replacement">//example05/movedCaches/</Value>
  </ListEntry>
</List>
```

Listing 4 An example configuration for the `pathMappings` list.

this mapping is to use a faster access method if the directory scanner service is actually running on server “example05”. Local file access is always faster than network file access.

The second entry demonstrates a simple redirection. There is a script which is checking the cache sizes in regular intervals, but the caches were moved to another server. Each access to “//example02/caches/” is automatically mapped to “//example05/movedCaches/”. Now the scripts accessing the old paths can be migrated one by one.

match

The match value configures the regular expression to match with the normalized input path. If this regular expression matches, the matched part will be replaced with the string in the replacement value.

The regular expressions is a *Perl* compatible (*PCRE*) one, supporting almost the complete range of features. To make the whole regular expression case insensitive, add for example the prefix (*?i*).

If the regular expression matches multiple times in the same path, all matching parts are replaced.

To create a regular expression which matches the whole path, make sure you use the begin (^) and end (\$) character in the expression.

replacement

The replacement value configures the string which is used as replacement for each matching part of the regular expression in the match value.

You can configure an empty string to remove the matching part from the path.

```
<List name="allowedPaths">
</List>
```

Listing 6 This empty allowedPaths list accepts any path from the client.

```
<List name="allowedPaths">
  <Value>(?!)^//example(?:01|02|10|23)/users/</Value>
  <Value>(?!)^//server[a-c]/profiles/</Value>
</List>
```

Listing 5 An example for the allowedPaths list.

You can also use the \1, \2, \3, ... placeholder to use text from capture groups of the regular expression.

allowedPaths

The `allowedPaths` list configures a number of regular expressions for paths which are allowed to use with the service.

If this list is left empty, all paths are allowed.

As soon there is at least one entry in this list, the input paths are checked. If *one* regular expression from this list matches the input path, the path is allowed for the service.

The regular expressions is a *Perl* compatible (*PCRE*) one, supporting almost the complete range of features. To make the whole regular expression case insensitive, add for example the prefix `(?i)`.

To create a regular expression which matches the whole path, make sure you use the begin (^) and end (\$) character in the expression.

See [Listing 6 on page 14](#) for an example of an empty `allowedPaths` list. In this case, the service will accept any path from the client. This is bad practice and should not be used for productive environments.

See [Listing 5 on page 14](#) for a common example of an `allowedPaths` list. It defines a number of regular expressions which have to match the path from the client. If one of this two regular expressions matches the normalized path from the client, it is accepted by the service.

cacheFilePath

The optional `cacheFilePath` value configures the absolute path to the cache database file which is used to speed up directory size calculations.

If you omit this value, the caching is disabled.

The database contains all scanned directories, therefore can contain sensitive information which has to be protected. Make sure you set the permissions of the created file in a way only administrators and the directory scanner service have read or write access to the file.

Security Checklist

Before you deploy a directory scanner service in a productive environment, please make sure you check the following points.

- ☐ The service is running under the correct account. This account requires only read access to all the directories for size calculation and checks.
- ☐ The configuration file can **only be modified** by the administrator and **only be read** by the service and the administrator.
- ☐ The network list configures **only protocols which are actually used**. See “network” on page 7.
- ☐ If a protocol is **only used on the local host**, the scope of this protocol is set to LocalHost. See “network” on page 7.
- ☐ Each service and script has **its own entry** in the secrets list with individual key and secret pairs. See “secrets” on page 11.
- ☐ Each entry in the secrets list has a **correct label** which names the client which is using this service. See “secrets” on page 11.
- ☐ Each entry in the secrets list **limits the access to a network** using the authorized-Networks value. See “secrets” on page 11.
- ☐ Scripts with a single purpose have a secret entry which **limit the accessible path** using the authorizedPaths value. See “secrets” on page 11.
- ☐ The allowedPaths **is not empty** and contains a list of regular expressions with all allowed paths. See “allowedPaths” on page 15.
- ☐ Make sure you removed all commented out sections from the production configuration file and any configuration values related to the test environment.



EducateIT GmbH
Steinenbergstrasse 7, 4532 Feldbrunnen, Switzerland
Phone: +41 32 355 39 00
E-Mail: info@educateit.ch
Web: <http://www.educateit.ch/>