

SQLWhite Documentation

Amin Astaneh, Qwik.Net Systems Inc.

September 18, 2007

Contents

1	Introduction	2
2	Installation	2
2.1	Requirements	2
2.2	Installation Procedure	2
2.2.1	Create UID/GID	2
2.3	Install the Program	2
3	Configuration	3
3.1	Creating the Database	3
3.2	/etc/sqlwhite/sqlwhite.conf	3
3.3	Configuring Postfix	3
3.4	Activate the Init Script	3
3.5	Optional- Integrating with Xerxes E-Mail Administrator	4
4	Running SQLWhite	4
4.1	Overview of the Whitelist/Blacklist Mechanism	4
4.1.1	Database Schema	5
4.1.2	Wildcarding	5
5	Creating an Effective Policy	6
5.1	Examples	6
5.1.1	Example 1	6
5.1.2	Example 2	6
5.1.3	Example 3	6
5.1.4	Example 4	7
5.2	Using Xerxes to Create Policy	7

1 Introduction

SQLWhite is a Postfix policy server written in Perl, based on Lionel Bouton's SQLGrey. SQLWhite serves as an excellent addition to a wide toolset that a Postfix email administrator can use to combat spam. It allows the administrator to create policy which allows or denies incoming email by source address, destination address, and MTA IP address/domainname. SQLWhite also has support for wildcarding, which makes policy description very easy. Furthermore, SQLWhite's policy can be defined by the Xerxes E-Mail Administrator through a web browser.

2 Installation

Currently SQLWhite is distributed as a tarball. In later releases, SQLWhite will also have .rpm noarch packages as well.

2.1 Requirements

- Perl
- Net::Server (Perl Module)
- IO::Multiplex (Perl Module)
- perl-DBI (Perl Module)
- Postfix 2.1 and above
- MySQL/Postgres/SQLite (MySQL is required for integration with Xerxes)

2.2 Installation Procedure

2.2.1 Create UID/GID

```
groupadd sqlwhite  
adduser -g sqlwhite -d /var/sqlwhite sqlwhite
```

2.3 Install the Program

Download the archive then perform these commands (as root):

```
tar -zxvf sqlwhite.tar.gz  
cd sqlwhite  
make install
```

3 Configuration

3.1 Creating the Database

Since MySQL is the recommended DB, here's the instructions on how to create the database in MySQL:

```
CREATE DATABASE sqlwhite;
GRANT ALL ON sqlwhite.* TO sqlwhite@localhost IDENTIFIED BY "foobaz";
FLUSH PRIVILEGES;
```

3.2 /etc/sqlwhite/sqlwhite.conf

99 percent of the time, you'll only need to configure these settings. Make sure you remove the comments in front of the parameters.

```
## Database settings
# instead of Pg below use "mysql" for MySQL, "SQLite" for SQLite
# any DBD driver is allowed, but only the previous 3 have been tested
db_type = mysql
db_name = sqlwhite
# Note: the following are not used with SQLite
db_host = localhost
db_user = sqlwhite
db_pass = spaces_are_not_supported
```

3.3 Configuring Postfix

In /etc/postfix/main.cf:

```
smtpd_recipient_restrictions =
    ...
    reject_unauth_destination
    check_policy_service inet:127.0.0.1:2502
```

This assumes SQLWhite will listen on the TCP 2502 port (default) and is on the same host.

3.4 Activate the Init Script

Ensure that /etc/init.d/sqlwhite will be activated on runlevels 3, 4, and 5. Each distribution has its own application to manage its init scripts; see your distribution documentation for details.

Start the service:

```
/etc/init.d/sqlwhite start
```

3.5 Optional- Integrating with Xerxes E-Mail Administrator

If you wish to configure SQLWhite to have Xerxes work as it's frontend, you must use MySQL as the database backend. Also, you must connect to the same database that Xerxes is using to administer email. For example: If your Xerxes database configuration looks like this:

```
$DB = array(  
    'TYPE' => 'mysql',  
    'USER' => 'mysqluser',  
    'PASS' => 'password',  
    'PROTO' => 'tcp',          // set to "tcp" for TCP/IP  
    'HOST' => '127.0.0.1',  
    'NAME' => 'mail'  
);
```

Then your SQLWhite database configuration will look like this:

```
db_type = mysql  
db_name = mail  
# Note: the following are not used with SQLite  
db_host = localhost  
db_user = mysqluser  
db_pass = password
```

And finally, in the Xerxes configuration file, this must also be set:

```
$sqlwhite = 1;
```

4 Running SQLWhite

4.1 Overview of the Whitelist/Blacklist Mechanism

Postfix receives an inbound connection from a client (MTA, mail client, etc) This client gives postfix this information when requesting access to deliver mail:

- Source email address
- IP address of the connecting client
- Destination Address

If SQLWhite is installed and running, Postfix will hand off this data to SQLWhite and expect one of three answers:

- "OK"- Unconditionally accept this email (bypassing other spam filters and mechanisms)
- "REJECT"- Reject the mail

- "DUNNO"- Cannot make a decision (send the mail along to other spam filters)

SQLWhite decides on how to handle each email by taking the information given by Postfix and comparing it to a database looking for matches, and replies according to the most specific match. The fact that the most specific match is chosen is important. If there is no match, SQLWhite will tell Postfix "DUNNO", and the mail will go through the rest of the validation mechanism.

4.1.1 Database Schema

An example database looks like this:

sender_name	sender_domain	src	mta_fqdn	dest_address	policy
*	foo.com	66.162.173.130	*	foo@foo.com	1
*	bar.com	66.162.173.130	*	*@foo.com	1
*	baz.com	*	*	*@foo.com	2
*	*	*	*.bunchaspammers.com	*@foo.com	2

The database consists of five fields per entry:

- sender_name- the 'user' part of the sender's email address ie: user@domain.tld
- sender_domain- the 'domain' part of the sender's email address ie: user@domain.tld
- src- the source IP address of the connecting client
- mta_fqdn- The fully-qualified domain name of the connecting client
- dest_address- the destination email address
- policy- the rule determining whether to accept(1) or deny(2) the email

4.1.2 Wildcarding

All fields except the policy field may accept the wildcard symbol "*". This symbol allows the mail administrator to set whitelist/blacklist rules to mask off entire domains of incoming email addresses, destination email addresses, and MTAs. SQLWhite will automatically prioritize by specificity of the rule. For example, an admin might want to blacklist an entire domain from delivering, except for one address. The rule pertaining to the exception would be chosen in priority to the general rule when an incoming email is addressed to the exception user.

NOTE: Wildcard functionality for the dest_address is restricted to this format: '*@domain.com'. Also, mta_fqdn may take on different forms. For MTA mail.foo.bar.com, valid entries can be '*.foo.bar.com', '*.bar.com', or '*'. This is so entire subdomains can be masked off.

5 Creating an Effective Policy

Creating an useful SQLWhite policy begins with asking yourself these questions:

- Are there email addresses that I should always accept/reject?
- Are there email domains that I should always accept/reject?
- Is there a particular MTA IP/domainname that I should always accept/reject?
- Are there domain-wide or user-wide restrictions from that set above?

5.1 Examples

5.1.1 Example 1

A domain administrator for foo.com wants no emails delivered except from bar.com.

sender_name	sender_domain	src	mta_fqdn	dest_address	policy
*	*	*	*	*@foo.com	2
*	bar.com	*	*	*@foo.com	1

The top rule masks all incoming emails as rejected email for foo.com. The second rule is the exception to the first rule; where all bar.com incoming email is allowed.

5.1.2 Example 2

A domain administrator for bar.com wants to deny delivery from baz.com, except for a few trusted addresses.

sender_name	sender_domain	src	mta_fqdn	dest_address	policy
*	baz.com	*	*	*@bar.com	2
trusted1	baz.com	*	*	*@bar.com	1
trusted2	baz.com	*	*	*@bar.com	1
trusted3	baz.com	*	*	*@bar.com	1

The top rule restricts all mail from baz.com. The other three rules allows delivery from those specific addresses.

5.1.3 Example 3

A domain administrator for quux.com wants to allow all email from a particular trusted MTA of the ip address 32.23.32.2.

sender_name	sender_domain	src	mta_fqdn	dest_address	policy
*	*	32.23.32.2	*	*@quux.com	1

This rule allows all sending usernames from all domains from IP 32.23.32.2 to send to any user of quux.com.

5.1.4 Example 4

A domain administrator for quuux.com wants to deny all email from a pool of MTAs that have been confirmed in sending spam.

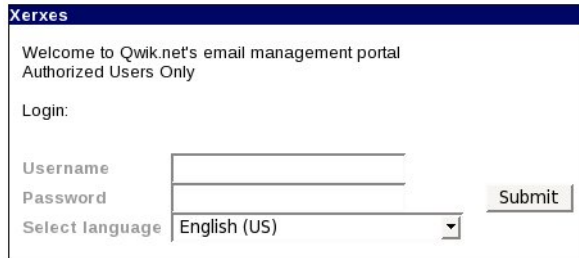
sender_name	sender_domain	src	mta_fqdn	dest_address	policy
*	*	*	*.spammers.com	*@quuux.com	2

This disallows all delivery for any email server underneath the subdomain spammers.com, ie: mta-a.spammers.com, mta-b.spammers.com, etc.

5.2 Using Xerxes to Create Policy

If Xerxes E-Mail Administrator is properly configured and installed, it can be used to define SQLWhite policy. In Xerxes, policy is defined separately for each recipient domain. For example, suppose that we administer a domain called testqwik.net, and wish to create a policy object for it.

We first successfully authenticate at the login page:



Next, we select the 'Whitelist' tab associated with testqwik.net (in red):

Edit Domain	Delete Domain	accounts	Aliases	Whitelist	testqwik.net	20	20000
-----------------------------	-------------------------------	--------------------------	-------------------------	---------------------------	--------------	----	-------

Then, we select the 'Add new whitelist rule' button:

[Add new whitelist rule](#)

Finally, we come to the dialog where you can create a policy object.



Each field in the dialog corresponds with the fields in each policy object in

the database. Just add the desired information, then click 'Add Rule'. You also have the capability to edit and delete entries.