OX Whitepaper

Dovecot Anti-Abuse Shield

Version 1.4

Aug 2017

Table of Contents

1. Dovecot Anti-Abuse Shield Overview

Dovecot Anti-Abuse Shield is included along with Dovecot Pro, but works with both Dovecot Pro and OX App Suite as a component to protect against login/authentication abuse.

Anti-Abuse Shield runs on a cluster of servers, and integrates with both OX App Suite and Dovecot to detect abuse, brute force attacks and also to enforce common authentication/authorization policies across the platform.

1.1. Anti-Abuse Shield Features

Features of Dovecot Anti-Abuse Shield include:

- Replicated/clustered architecture – In-Memory DB is replicated between all the servers in a cluster so there is a single view of abuse
- Scriptable Policy Language – Using the Lua language, the functionality of the daemon can be extended to record and protect against a large variety of abusive behavior, as well as implement specific customer policies.
- DNS Lookup Feature – For looking up IPs or domains in blacklists
- GeoIP Lookup Feature – GeoIP lookups can be made, and incorporated into policy decisions.
- Ratelimiting and Tarpitting – Extremely flexible, these can be enabled and enforced based on IP address, login name, geoip location, time windows, etc.
- Flexible In-Memory Statistics Database – A versatile and extensible in-memory database is used to store statistics information about abuse over time periods from a few minutes to many hours.
- Persistent Blacklist – Configurable via a REST API or the Lua policy engine, supports auto-expiry of entries, and uses the Redis DB for persistence.
- Webhooks – Integrate Anti-Abuse Shield with other systems using webhooks to send events
- Integration with Customer Authentication/Authorization Systems – Customers can use the open HTTP REST API to benefit from the

protection of the anti-abuse daemon in their own authentication/authorization systems.

- Admin Console – To retrieve statistics and query server state

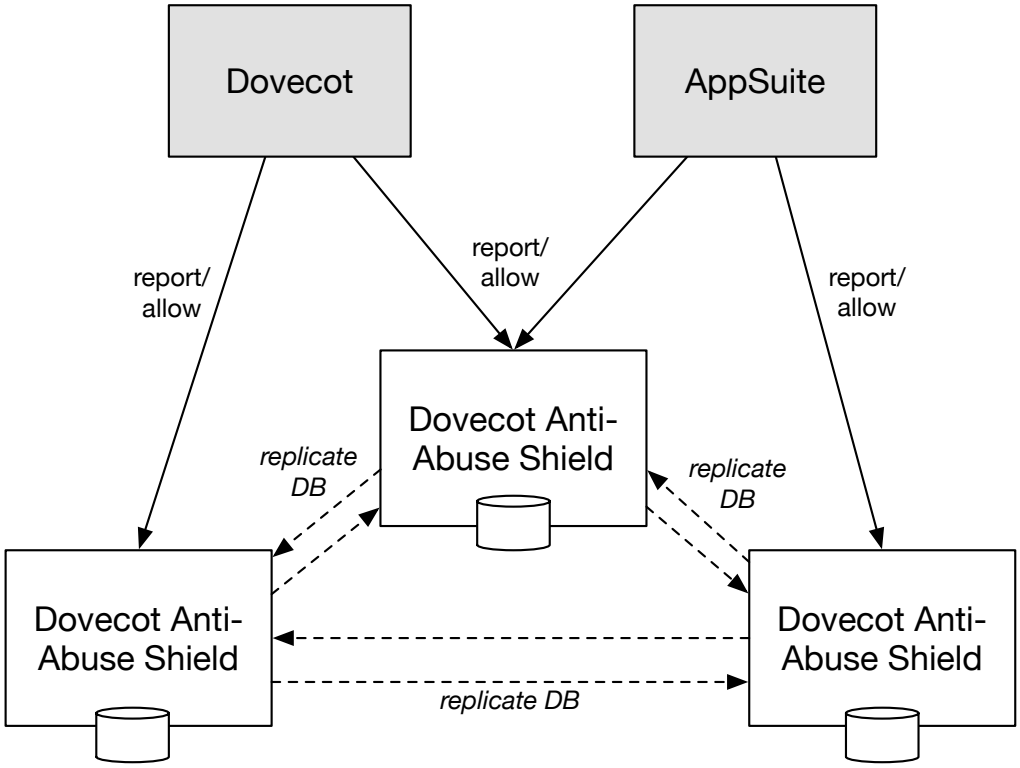
1.2. Customization

The anti-abuse daemon ships with simple policies to protect against brute force attacks. For more sophisticated protection, Open-Xchange can provide customized Lua scripts that handle a wide variety of abuse scenarios as well as implementing customer-specific features and policies. The enhanced scripts and script customization service are Professional Services that can be provided along with the commercial versions of Dovecot and OX App Suite.

2. Clustering

Dovecot Anti-Abuse Shield can be configured to run in a clustered environment, with multiple servers sharing data to achieve a unified view of abusive behavior. Clustering uses the UDP protocol, and all information is encrypted over the wire using symmetric encryption and a shared secret between all servers.

Clustering is achieved by configuring each Anti-Abuse Shield server to replicate the in-memory DB to a configured set of servers. Typically, those other servers would be configured to also replicate their DB to the same set of servers, thus creating a “mesh”. In this way, Dovecot Anti-Abuse Shield can be configured to create a highly available cluster, where all servers in the cluster have all the information about abuse, and can thus be accessed in the same way by clients. This is shown in the following figure:



The previously supported scenario of creating a set of “report-only” servers, which receive reports from clients and “spread” those reports to a set of “allow-only” servers, is no longer supported.

3. Anti-Abuse Policy

3.1. Overview

Dovecot Anti-Abuse Shield provides a policy engine for detecting and blocking abuse using the Lua scripting engine, which can be configured to provide almost any policy a customer, desires.

There are three main commands that the Anti-Abuse Shield accepts, each of which can be configured to implement the required policy in Lua:

- Report – The Report command allows a client (Dovecot, OX App Suite or a customer-specific client) to report a login event success or failure, along with associated information such as the IP address, login username, and any arbitrary additional information the client wishes to provide, in key-value pairs.

- Allow – The Allow command allows a client (Dovecot, OX App Suite or a customer-specific client) to query whether a new login attempt should be allowed. Again information pertinent to the login attempt is provided.

- Reset – The Reset command allows a client to reset all the statistics associated with a particular username and/or IP address.

Additionally there is a “canonicalization” function that is run as part of each command, which is used to canonicalize login names to a standard form, for example “foo” and “foo@foo.com” would both be canonicalized to “foo@foo.com”.

3.2. Lua Scripting Engine

The Lua language is integrated with the core code of Dovecot Anti-Abuse Shield, which itself is written in C++ for speed and efficiency. The Lua interpreter is full-featured, allowing full access to the Lua language, and multiple threads and Lua stats can be configured to optimize performance.

Lua is used for the configuration of Dovecot Anti-Abuse Shield as well as implementing the commands described above.

Lua can also be used to extend the standard HTTP API described in Section 5, by creating custom HTTP endpoints which call installation-specific Lua functions.

3.3. Lua Policy Functions

3.3.1. Time Window Statistics Database

The Time Window Statistics Database is an in-memory database that stores information in a rolling time-windows of configurable number and length, providing the ability to store statistics from a few minutes to a whole day, depending on memory constraints. Statistics in the oldest time window are expired and a new time window is created at regular intervals, depending on the configuration of the database. Multiple databases can be created, tracking statistics over multiple time periods. All database statistics can be replicated to other members of the cluster – replication can be enabled on a per-database basis.

Statistics can be stored using an arbitrary “key”, which can be an IP address, login name, or any other string. The database can be configured with IP v4 and IPv6 prefixes, which causes the aggregation of IP-level statistics according to the prefix. Each database can be configured with a set of named sub-keys, which are used to stored different statistics for the main key.

Sub-keys can be of the following types:

- Integers – Used to count the number of events in a stream of data
- HyperLogLog – Use to estimate cardinality (or count-distinct) of events in a stream of data
- CountMin – Serves as a frequency table of events in a stream of data

3.3.2. GeoIP Support

GeoIP support is provided, enabling lookups from Lua of IP address to Country Code, City/Region information, as well a ISP name (the capabilities depend on the GeoIP DBs available, some of which may only be available with a subscription).

This provides the ability to write policies that depend on the source country, city or ISP of the login attempt, as well as using the country code, city name or ISP name as a key for statistics, or as a value for statistics functions. For example, the CountMin type can be used to track the frequency of logins from individual countries for a given username.

New in the 1.4 release:

- Support for City and ISP GeoIP Databases
- Support for reloading GeoIP Databases without restarting

3.3.3. DNS Lookups

DNS Lookup support is provided, enabling DNS Lookups from the Lua interpreter. Support for lookup of A records, PTR records, as well as RBL formatted records is provided.

DNS support allows integration with third-party or customer-created IP blacklists, as well as interrogation of DNS PTR information from the supplied IP address.

3.3.4. Persistent Replicated Blacklists

Persistent, replicated blacklists are a feature which enable Anti-Abuse Shield to store permanent or time-expiring entries for IP addresses and logins which are blacklisted. Blacklist entries are optionally persisted in a Redis DB, and optionally replicated to all cluster members. Blacklist entries can be viewed and manipulated from Lua or via the HTTP REST API.

3.3.5. Webhooks

Webhooks can be configured to send events to configured webhook endpoints over HTTP(S). The events that can be sent via a webhook are:

- Report
- Allow
- Add Blacklist Entry
- Delete Blacklist Entry
- Expire Blacklist Entry

Custom webhooks can also be created, which are triggered from Lua, thus providing complete flexibility in terms of how and when data is exported from the platform.

4. Standardized Policy

Previous versions of anti-abuse shield shipped with a simple policy, suitable for very basic abuse protection. The latest version (1.4) ships with a full-features and extensible example policy that is extremely straightforward to use and configure, and requires no programming or scripting knowledge.

Features of the standardized policy include:

- Out of the box support for RBLs
- Sixteen behavioral policies that can be enabled/disabled, tuned and customized without needing scripting knowledge.
- Support for IP and User-level Whitelists
- Support for country-level Blacklists
- Support for custom policies to enhance the existing policy

5. HTTP REST API

The HTTP REST API is the only supported interface for providing report information and issuing queries, and provides support for the following commands listed below:

- Report – Report a login success/failure
- Allow – Query whether a login should be allowed
- Reset – Reset the stats for a login/IP
- Ping – Check whether the server is running
- Add Blacklist Entry
- Delete Blacklist Entry
- Get Blacklist – Retrieve all entries in the blacklist
- Get DB Stats – Retrieve stats about a particular IP/user
- Call Custom Endpoint – Call a custom Lua function

The HTTP REST API supports HTTP 1.0 and 1.1, including persistent connections for higher performance, as well as chunked encoding in requests and responses. Authentication is provided by HTTP Basic Authentication.

6. Admin Console

The Dovecot Anti-Abuse Shield product also includes an admin console, which can be used for introspection, debugging and retrieving statistical information. The admin console connects to a particular Anti-Abuse Shield instance on a custom port, and provides the following functions:

- Dynamic ACL Configuration
- Statistics Functions on the number of successful/failed logins, uptime and query latency
- Create Cryptographic Keys for Console Authentication

7. Integration with OX App Suite and Dovecot

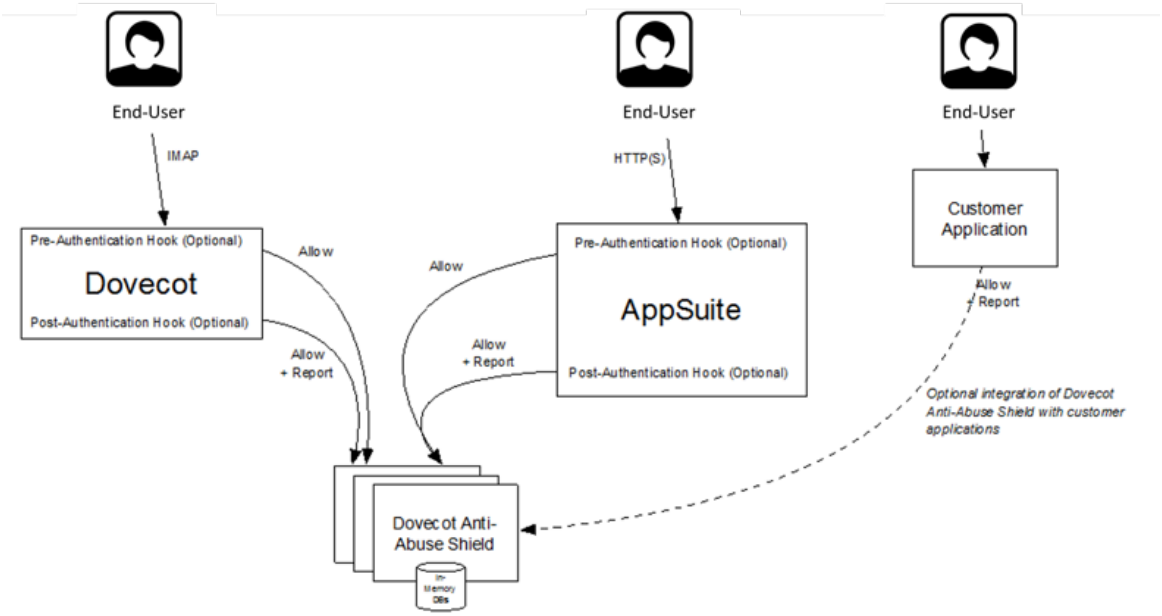
OX App Suite and Dovecot are both integrated with the Anti-Abuse Shield in a similar manner:

- Pre-Authentication Hook – The pre-authentication hook allows Dovecot or OX App Suite to query Anti-Abuse Shield if a particular login should be

allowed before authentication takes place. This allows for efficient rejection of abusive connections.

- Post-Authentication Hook – The post-authentication hook allows Dovecot or OX App Suite to send login reports to Anti-Abuse Shield, as well as perform post-authentication queries. Post-Authentication queries can contain information about the user such as data from the OX App Suite DB or LDAP Directory, and can be used to control user login based on policy.

The integration between Dovecot, OX App Suite and Anti-Abuse Shield is shown in the following figure:



8. Integration with other Authentication Systems

The well-documented (using Swagger/OpenAPI specification) API for Anti-Abuse Shield means that it is straightforward to integrate with systems other than Dovecot and AppSuite. Examples would include:

- Single Sign On (SSO)/ Identity Management Systems
- Web Portals
- Operating System logins
- Etc.