Valimail API Integration Guide for Customers

01.27.2026

Valimail API Integration Guide for Customers

Table of Contents

1. Introduction

Overview

Valimail APIs enable customers to streamline real-time data sharing, manage their accounts, configure email authentication policies (DMARC, SPF, DKIM), manage inbound email security policies (MTA-STS, TLS-RPT), and (where enabled) provision users via SCIM.

Key Benefits for Customers:

- Improved email authentication
- Enhanced security posture
- Actionable insights to mitigate risks.

Target Audience

- This guide is intended for:
  - Data and analytics teams
  - Developers and system administrators

Scope

This document provides:

- A summary of Valimail APIs
- Suggested use cases for each API type
- Integration steps, onboarding actions, and advanced usage

Access and Authentication

Members Gaining Access

- For access to Valimail's APIs, please contact your internal account administrator to obtain the API credentials within the application account settings for the specific APIs you intend to use.

- When making your request, please specify which API you need access to:
    - Configuration API
    - Reporting Data API
    - Account Management API (Partners Only)
- Once access is granted through the application, your Admin will provide the necessary API key details to generate a bearer token.
- If you're unsure who serves as the account Admin, don't hesitate to contact our Product Support team for help with the connectivity process.

Owners Granting Access

- Please ensure you use SSO or MFA when logging into Valimail's platform.
- Access Enforce and select the gear icon located in the top-right corner of the toolbar.
- Select API keys from the left column to create and manage new keys.
- Press the Create API Key button and choose from these options:
    - API Key Name: Name the key based on the user or intended use.
    - Key Access: Select the API(s) for which the user requires access.
- Press the Create button and give the intended user the following credential details:
    - Client ID
    - App ID
    - Token generation Instructions
- After a key is generated and distributed, it will appear in the table as active, allowing for renaming or deletion if necessary.

2. Getting Started

To effectively utilize Valimail's APIs, Customers should consider these suggested workflows tailored to common scenarios:

- Validate API Credentials
    - To validate your credentials, please use the POST/Auth endpoint to confirm authentication.
    - When the authentication is successful, the response will include a bearer token and a timestamp of when the token expires.

- Configure a Sandbox Domain
    - Once you have access, we recommend using the DMARC configuration API on a test domain to familiarize yourself with the available endpoints and set up your preferences without affecting your existing enforcement status.

- Next, you can test the available configuration updates through the API (examples of everyday use cases below) on this test domain to validate that the intended setup functions as expected.
- Once validated, please update the test domain to represent the production domain(s) you intend to use APIs to manage systematically.

- Enhancing DMARC Policy Configuration
  - Start with the DMARC Configuration API to assess the current domain configuration.
  - Update the existing configuration to improve or extend DMARC policies across customer domains/subdomains.
  - Next, a user can review the Reporting Data API to monitor sender authentication and unidentified email activity for opportunities to improve DMARC enforcement.

- Enforcement Monitoring and Reporting
  - Start with the Reporting Data API to access data that helps evaluate the effectiveness of the current DMARC configuration.Employ the Sender's report data to identify high-volume email senders for their customers. By monitoring the passing DMARC rates of these senders, Customers can prioritize outreach to those who can improve email authentication.
  - Next, leverage the Unidentified Senders report data to support customers in identifying and investigating suspicious or unknown email traffic on specific domains/subdomains.
  - Finally, a user can incorporate this data into existing dashboards, workflows, and risk reporting to monitor the status of ongoing authentication and enforcement.

- Policy Configuration (MTA-STS and TLS-RPT)
  - Start with the MTA-STS and TLS-RPT policy endpoints to establish inbound email transport security controls.
  - Use reporting endpoints (where enabled) to review TLS-related telemetry and failure details for troubleshooting.

- Holistic Security Strategy
  - Combine DMARC configuration (domain/sender/DKIM/netblocks) with transport security policies (MTA-STS/TLS-RPT) to improve both authentication and secure transport posture.
  - Enhanced visibility provides opportunities for informed adjustments.

- Custom Reporting Dashboards
  - Start with the Reporting Data API to gather relevant information for custom-built dashboards.
  - Next, integrate this data with key metrics from various APIs or business intelligence tools to visualize real-time email security performance.
  - Aggregating this data across platforms and tooling facilitates strategic planning

and in-depth performance evaluation.

3. API Overview

Valimail offers three core APIs categorized by functionality:

API Name	Purpose
Reporting Credential Authentication & Health Check	Validate credentials and ensure connection status to the Reporting API.
DMARC Configuration	Manage domains, senders, and DMARC policies for email authentication.
Reporting Data	Retrieve email authentication metrics and reports for compliance and insights.

4. Reporting API Credential Authentication and Health Check

Overview

The Reporting Credential Authentication and Health Check API ensures valid credentials and verifies the availability of Valimail's services.

Swagger Documentation: Authentication and Health Check API

Endpoints

- Validate Server Status
  - Endpoint: GET /healthcheck
  - Description: Confirms the server is operational.
  - Response: 200 OK (Service is up).

Authenticate API Credentials

- Endpoint: POST /auth
- Description: Validates Reporting API credentials and generates a bearer token for subsequent requests.
- Request Body:

```
JSON file
   {
client-id: YOUR_CLIENT_ID
     app-id: YOUR_APP_ID
   }
```

- Response:

```
JSON file
    {
token: YOUR_TOKEN
expires_at: 2023-05-10T08:00:00Z
    }
```

- Usage: Add the token to the Authorization header in future requests:

```
JSON file
    {
Authorization: Bearer YOUR_TOKEN
    }
```

- General Use Cases:
    - Ensuring Reliable Service Access
        - Endpoint: GET /healthcheck
        - Use Case: Customers regularly check the health of the API to confirm service availability before executing essential operations.
        - Outcome: Minimizes service disruptions by ensuring the API is operational, leading to smoother integration experiences.
    - Validate Credential Token Authentication
        - Endpoint: POST /auth
        - Use Case: Customers implement automated scripts to validate credentials and generate tokens for scheduled tasks.
        - Outcome: Streamlined authentication processes reduce manual effort and enhance security for ongoing API interactions.

5. DMARC Configuration API

Overview
These endpoints simplify management of domains/subdomains, DMARC policy posture, sender associations, DKIM records, and SPF-related netblocks.

Swagger Documentation: DMARC Configuration

Endpoints

- List Domains/Subdomains

- Endpoint: GET /accounts/{slug}/domains
- Description: Returns a list of domains/subdomains and relevant DMARC configuration data.
- Common Query Filters:
    i. domain[] (filter by domains)
    ii. dmarc-record[] (None, NS, CNAME, TXT)
    iii. enforcement-status[]
    iv. dmarc-policy[] (None, Quarantine, Reject)
    v. sending-status[] (Active, Reporting Only, Blocked)
    vi. reverse (boolean)
    vii. sort-key (e.g., domain_name, dmarc_record_type, enforcement_status, dmarc_policy, sending_status, last_report_date)

- Create Domain/Subdomain

    - Endpoint: POST /accounts/{slug}/domains
    - Description: Creates a domain/subdomain associated with an account.
    - Request Example:

```
JSON file
   {
     "domain": "example.com"
   }
```

- Update Domain/Subdomain Configuration

    - Endpoint: PUT /accounts/{slug}/domains/{domain}
    - Description: Modify DMARC policy or sending status.
    - Request Example:

```
JSON file
   {
        dmarc-policy: Reject
        sending-status: Active
   }
```

- List Senders for a Domain
    - Endpoint: GET /accounts/{slug}/domains/{domain}/senders

    - Description: Returns the list of senders linked to a domain/subdomain.

- Link a Sender to a Domain

- Endpoint: POST /accounts/{slug}/domains/{domain}/senders
- Description: Creates a sender link. (To list available senders: GET /resource/senders)
- Request Example:

JSON file

```
{
  "Slug": "sendgrid"
}
```

- Get a Single Sender Link
  - Endpoint: GET /accounts/{slug}/domains/{domain}/senders/{sender-slug}

- Update Sender Status
  - Endpoint: PUT /accounts/{slug}/domains/{domain}/senders/{sender-slug}
  - Description: Updates sender status to one of: approved, denied, pending_approval.
  - Request Example:

JSON file

```
{
  "Status": "approved"
}
```

- List Available Sender Providers
  - Endpoint: GET /resource/senders
  - Description: Returns the list of senders available to be connected (provider catalog).

- List DKIMs for a Sender
  - Endpoint: GET /accounts/{slug}/domains/{domain}/senders/{sender-slug}/dkims
  - Description: Returns DKIMs for a sender linked to the domain/subdomain.
  - Query: reverse (boolean)

- Create DKIM
  - Endpoint: POST /accounts/{slug}/domains/{domain}/senders/{sender-slug}/dkims
  - Description: Creates a DKIM using either a PublicKey or CnameTarget (not both).
  - Request Example (CNAME target):

JSON file

```
{
  "Selector": "selector1",
  "CnameTarget": "cname-key-here",
  "Comment": "This is a comment"
}
```

- Get DKIM by Selector
  - Endpoint: GET /accounts/{slug}/domains/{domain}/senders/{sender-slug}/dkims/{selector}

- Update DKIM
  - Endpoint: PUT /accounts/{slug}/domains/{domain}/senders/{sender-slug}/dkims/{selector}
  - Description: Updates DKIM comment, sender-owner, domain-owner, and optionally re-links to a new sender slug.
    Updatable Fields (at least one required):
    i. comment
    ii. sender-owner
    iii. domain-owner
    iv. new-sender-slug (optional)
  - Request Example:

JSON file

```
{
  "new-sender-slug": "mailgun",
  "comment": "This is a comment update",
  "sender-owner": "mysenderowner@example.com",
  "domain-owner": "mydomainowner@example.com"
}
```

- List Netblocks
  - Endpoint: GET /accounts/{slug}/domains/{domain}/netblocks
  - Description: Returns netblocks (CIDR ranges) associated with a domain/subdomain.
  - Query: reverse (boolean)

- Create Netblock
  - Endpoint: POST /accounts/{slug}/domains/{domain}/netblocks
  - Description: Creates a new netblock. comment and sender-slug are optional.
    Request Example:

```
JSON file
  {
    "netblock": "203.0.113.0/24",
    "comment": "Outbound mail range",
    "sender-slug": "sendgrid"
  }
```

- Get Netblock
  - Endpoint: GET /accounts/{slug}/domains/{domain}/netblocks/{netblock}

- Update Netblock
  - Endpoint: PUT /accounts/{slug}/domains/{domain}/netblocks/{netblock}
  - Description: Updates comment and/or sender-slug
  - Request Example:

```
JSON file
  {
    "comment": "Updated comment",
    "sender-slug": null
  }
```

- Get MTA-STS Policy for a Domain
  - Endpoint**:** GET /accounts/{slug}/domains/{domain}/mta_sts_policy
  - Description**:** Returns MTA-STS policy configuration, record text, policy text, and SSL certificate validation status (if available).

- Create MTA-STS Policy
  - Endpoint**:** POST /accounts/{slug}/domains/{domain}/mta_sts_policy
  - Request Example**:**

```
JSON file
  {
    "policy-mode": "testing",
    "mx-hosts": ["mx1.example.com", "mx2.example.com"],
    "max-age": 86400
  }
```

- Update MTA-STS Policy
  - Endpoint**:** PUT /accounts/{slug}/domains/{domain}/mta_sts_policy
  - Request Example**:**

```
JSON file
  {
    "policy-mode": "enforce",
    "mx-hosts": ["mx1.example.com", "mx2.example.com", "mx3.example.com"],
    "max-age": 604800
  }
```

- List All MTA-STS Policies for an Account
  - Endpoint: GET /accounts/{slug}/mta_sts_policies

- Get TLS-RPT Policy for a Domain
  - Endpoint: GET /accounts/{slug}/domains/{domain}/smtp_tls_policy

- Create TLS-RPT Policy
  - Endpoint: POST /accounts/{slug}/domains/{domain}/smtp_tls_policy
  - Request Example:

```
JSON file
  {
    "rua-uris": ["mailto:tls-reports@example.com", "mailto:security@example.com"]
  }
```

- Update TLS-RPT Policy
  - Endpoint: PUT /accounts/{slug}/domains/{domain}/smtp_tls_policy
  - Request Example:

```
JSON file
  {
    "rua-uris": ["mailto:tls-reports@example.com"]
  }
```

- List TLS-RPT Policies for an Account
  - Endpoint: GET /accounts/{slug}/smtp_tls_policies

- TLS Report Summary
  - Endpoint: GET /accounts/{slug}/mta-sts/tls

  - Required Query Params:

i. start-date (YYYY-MM-DD)
ii. end-date (YYYY-MM-DD)

- Optional Filters:
  i. domain-names
  ii. policy-mx-host
  iii. policy-mode (none, enforce, testing, unknown)
  iv. limit (default 20)
  v. page (default 1)
  vi. sort (e.g., domain, -policy-mx-host)
- Response Shape:
  i. Count total results
  ii. Next / Previous page URLs
  iii. Results entries with:
  • domain
  • report-id
  • version
  • policy-mode
  • policy-mx-host
  • max-age

- TLS Report Failure Details
  - Endpoint**:** GET /accounts/{slug}/mta-sts/tls/failure-details
  - Required Query Params:
    report-id
    i. domain-name
  - Optional Filters:
    i. policy-mode
    ii. policy-type

Overview: SCIM endpoints allow identity providers to manage users in SCIM format (RFC 7644). SCIM endpoints use, Content-Type: application/scim+json

- List Users (SCIM ListResponse)
  - Endpoint**:** GET /accounts/{slug}/scim/v2/Users
  - Optional Query:
    i. filter with operators: eq, ne, co, sw, ew, gt, lt, ge, le
    ii. Logical operators: and, or, not
  - Filter Example:
    i. userName eq "foo@example.com"
    ii. active eq true and name.givenName co "J"

- Create User
  - Endpoint**:** POST /accounts/{slug}/scim/v2/Users
  - Response**:** 201 Created
    Request Example:

```
JSON file
  {
    "schemas": [
      "urn:ietf:params:scim:schemas:core:2.0:User",
      "urn:ietf:params:scim:schemas:extension:valimailuserextension:2.0:User"
    ],
    "userName": "foo@somedomain.com",
    "externalId": "string",
    "name": { "givenName": "foo", "familyName": "bar" },
    "urn:ietf:params:scim:schemas:extension:valimailuserextension:2.0:User": {
      "memberType": "owner"
    }
  }
```

- Get User by ID
  - Endpoint**:** GET /accounts/{slug}/scim/v2/Users/{id}

- Patch User
  - Endpoint**:** PATCH /accounts/{slug}/scim/v2/Users/{id}
  - Description**:** Supports SCIM PatchOp with add, replace, remove.
  - Request Example:

```
JSON file
  {
    "schemas": ["urn:ietf:params:scim:api:messages:2.0:PatchOp"],
    "Operations": [
      { "op": "replace", "path": "active", "value": "false" },
      {
        "op": "replace",
        "path":
  "urn:ietf:params:scim:schemas:extension:valimailuserextension:2.0:User.memberType
  ",
        "value": "owner"
      }
    ]
  }
```

General Use Cases

- Automated Domain Management
  - Scenario: Customers use domain endpoints to add/update domain records programmatically.
  - Outcome: Simplifies multi-domain management.
- Centralized Subdomain Policy Management
  - Scenario: A customer centrally manages subdomain DMARC coverage (e.g., enabling/disabling subdomain monitoring or applying consistent policy inheritance) by updating subdomain-related settings within the Configuration API for a parent domain.
  - Outcome: Consistent enforcement and visibility across subdomains with reduced manual configuration drift.
- Subdomain Coverage Audit and Baseline Enforcement
  - Scenario: A customer audits subdomain configurations to identify which subdomains are missing required email authentication controls, then programmatically applies the approved configuration baseline.
  - Outcome: Faster remediation of gaps and improved overall domain posture at scale.
- Proactive Security Improvements
  - Scenario: Identify domains/subdomains with weak policies and strengthen DMARC posture.
  - Outcome: Reduced risk of phishing and spoofing.
- Managing Sender Integrations Programmatically
  - Scenario: Link third-party email services (SendGrid, Mailgun, etc.) to domains/subdomains.
  - Outcome: Streamlined setup and onboarding.
- Visualizing Netblock Coverage
  - Scenario: Pull netblocks to validate SPF coverage across legitimate mail sources.
  - Outcome: Reduced false positives and improved authentication alignment.
- Programmatic MTA-STS Enablement and Updates
  - Scenario: A customer publishes or updates MTA-STS policy settings for one or more domains through the Configuration API as part of a standardized “secure email” rollout.
  - Outcome: Improved transport security by enforcing TLS requirements for inbound mail delivery.
- Staged MTA-STS Policy Version Rollout
  - Scenario: A customer rotates MTA-STS policy versions and validates changes during staged deployments (dev → pilot → production) using API-driven updates rather than manual DNS/web hosting changes.
  - Outcome: Safer rollouts with fewer misconfigurations and clearer change control.
- TLS-RPT Activation and Report Destination Configuration
  - Scenario: A customer enables TLS-RPT reporting for domains and configures the reporting destination(s) using the Configuration API to receive automated feedback on TLS negotiation failures.
  - Outcome: Increased visibility into delivery issues and misconfigurations impacting encrypted transport.
- Fleetwide TLS-RPT Standardization
  - Scenario: A customer periodically reviews TLS-RPT configuration state across many domains, then standardizes reporting URIs and settings to ensure consistent telemetry.
  - Outcome: Reliable, account-wide transport reporting with simplified operational oversight.
- Automated User Lifecycle Provisioning
  - Scenario: A customer uses SCIM-based user and group provisioning to automatically create, update, and deactivate users in Valimail when changes occur in their identity provider (e.g.,

Okta, Azure AD).
    - Outcome: Reduced administrative workload and improved security through timely deprovisioning and least-privilege access.
- Group-Based Access and Role Synchronization
    - Scenario: A customer synchronizes role-based access by mapping identity provider groups to application permissions, ensuring the correct teams have access to the correct accounts and domains.
    - Outcome: More consistent access control with fewer manual errors and faster onboarding/offboarding.

Detailed Business Use Cases

- Streamlined Bulk Domain Management
    - Endpoint: /accounts/{slug}/domains (POST)
    - Use Case: Customers automate adding new domains/subdomains to their DMARC settings as new brands or services are needed.
    - Outcome: Reduces time to compliance for new domains and ensures consistent security policies across all email communications.

- Centralized Subdomain Policy Management
    - Endpoint: /accounts/{slug}/domains/{domain}/subdomains/{subdomain} (PUT)
    - Use Case: Customers centrally manage DMARC-related subdomain settings by programmatically applying consistent policy inheritance (or explicit overrides) across subdomains tied to a parent domain. This supports bulk enablement/disablement of monitoring and ensures approved enforcement behaviors are applied uniformly.
    - Outcome: Consistent enforcement and visibility across subdomains with reduced manual configuration drift.

- Subdomain Coverage Audit and Baseline Enforcement
    - Endpoint: /accounts/{slug}/domains/{domain}/subdomains (GET)
    - Use Case: Customers retrieve the full inventory of subdomains and their configuration status, identify gaps (e.g., missing monitoring, incomplete policy inheritance, or misaligned enforcement), and then automatically apply a standardized baseline configuration to out-of-compliance subdomains.
    - Outcome: Faster remediation of gaps and improved overall domain posture at scale.

- Identifying Non-Compliant Domains
    - Endpoint: /accounts/{slug}/domains
    - Use Case: Customers use the API to identify customer domains with None or weak enforcement policies and recommend steps to strengthen their DMARC configurations.
    - Outcome: Improved email authentication and reduced risk of phishing and spoofing attacks.

- Configuring DMARC Policies for New Domains
    - Endpoint: /accounts/{slug}/domains/{domain} (PUT)
    - Use Case: Customers provide tools to programmatically update the DMARC policy (e.g., from None to Quarantine or Reject) for newly onboarded domains as they move toward full compliance.
    - Outcome: Seamless adoption of DMARC best practices without disrupting email operations.

- Programmatic MTA-STS Enablement and Updates
  - Endpoint: /accounts/{slug}/domains/{domain}/mta-sts (PUT)
  - Use Case: Customers enable MTA-STS for a domain and update policy parameters (e.g., mode, MX patterns, max-age) using the API as part of a secure transport initiative. This allows security teams to roll out and maintain transport requirements across many domains without manual steps.
  - Outcome: Improved transport security by enforcing TLS requirements for inbound mail delivery.

- Staged MTA-STS Policy Version Rollout
  - Endpoint: /accounts/{slug}/domains/{domain}/mta-sts (PUT)
  - Use Case: Customers roll out MTA-STS changes in stages by updating policy versions through the API (e.g., pilot domains first, then broader rollout). They can validate new settings, revert quickly if issues arise, and promote the validated configuration to production domains.
  - Outcome: Safer rollouts with fewer misconfigurations and clearer change control.

- TLS-RPT Activation and Report Destination Configuration
  - Endpoint: /accounts/{slug}/domains/{domain}/tls-rpt (PUT)
  - Use Case: Customers enable TLS-RPT for a domain and configure one or more report destinations (URIs) through the API, ensuring transport failure telemetry is collected consistently as part of operational monitoring.
  - Outcome: Increased visibility into delivery issues and misconfigurations impacting encrypted transport.

- Fleetwide TLS-RPT Standardization
  - Endpoint: /accounts/{slug}/domains (GET) + /accounts/{slug}/domains/{domain}/tls-rpt (PUT)
  - Use Case: Customers periodically enumerate all domains in an account, evaluate TLS-RPT configuration consistency, and programmatically standardize reporting URIs/settings across the fleet to align with centralized monitoring and incident workflows.
  - Outcome: Reliable, account-wide transport reporting with simplified operational oversight.

- Automated User Lifecycle Provisioning (SCIM)
  - Endpoint: /scim/v2/Users (POST) and /scim/v2/Users/{id} (PATCH/DELETE)
  - Use Case: Customers use SCIM to automatically provision users when hired, update user attributes/entitlements when roles change, and deactivate users upon departure—driven by changes in the identity provider and applied to Valimail without manual admin effort.
  - Outcome: Reduced administrative workload and improved security through timely deprovisioning and least-privilege access.

- Group-Based Access and Role Synchronization (SCIM)
  - Endpoint: /scim/v2/Groups (POST) and /scim/v2/Groups/{id} (PATCH)
  - Use Case: Customers manage access via IdP groups mapped to Valimail roles (e.g., Admin, Analyst, Read-only). SCIM group synchronization ensures the right teams retain appropriate access to the correct accounts/domains as membership changes over time.
  - Outcome: More consistent access control with fewer manual errors and faster onboarding/offboarding.

- Tracking Enforcement Status for Compliance Reporting

  - Endpoint: /accounts/{slug}/domains
  - Use Case: Customers retrieve the enforcement status of all domains in a customer's account and generate compliance reports to share with regulators or executives.
  - Outcome: Improved transparency and alignment with risk compliance or contractual obligations.

- Enabling Smart Alerts for Configuration Errors
  - Endpoint: /accounts/{slug}/domains
  - Use Case: Customers integrate the API into monitoring tools to detect and alert customers when domains are misconfigured or missing DMARC, SPF, or DKIM records across customer domains.
  - Outcome: Faster resolution of configuration errors and identification of policy improvement opportunities.

- Domain Prioritization for Security Improvements
  - Endpoint: /accounts/{slug}/domain
  - Use Case: Customers use data on domain sending status (e.g., Active vs. Blocked) and the last report date to help them prioritize their efforts on actively used domains with outdated configurations.
  - Outcome: Focused resource allocation on high-impact domains for security and deliverability.

- Managing Sender Integrations Programmatically
  - Endpoint: /accounts/{slug}/domains/{domain}/senders
  - Use Case: Customers automate the process of linking third-party email services (e.g., SendGrid, Mailgun) with a customer's domain by creating sender connections programmatically.
  - Outcome: Streamlined setup of third-party email services, reducing friction in onboarding and integration.

- Visualizing Netblock Coverage
  - Endpoint: /accounts/{slug}/domains/{domain}/netblocks
  - Use Case: Customers integrate the endpoint to fetch and visualize the netblocks (IP ranges) associated with a domain, enabling customers to ensure their SPF records cover all legitimate email sources.
  - Outcome: Reduced chances of legitimate emails being flagged as spam due to incomplete SPF coverage.

6. Reporting Data API

Overview
The Reporting Data API retrieves insights into email authentication metrics, sender behaviors, and compliance statuses.

Swagger Documentation: Reporting Data

Endpoints
1. Get List of Domains/Subdomains

- Endpoint: GET /accounts/{slug}/reports/senders
- Description: Retrieves a comprehensive list of domains/subdomains associated with a customer's account.

2. Senders Report

   - Endpoint: GET /accounts/{slug}/reports/senders
   - Description: Provides details on DMARC compliance rates for each sender.

3. Unidentified Report
   - Endpoint: GET /accounts/{slug}/reports/unidentified
   - Description: Lists suspicious or unidentified email traffic.

4. Events Log

   - Endpoint: GET /accounts/{slug}/reports/events
   - Description: Provides a detailed log of actions taken on all domains and users, allowing customers to audit and track changes effectively.

5. DKIM Continuous Protection

   - Endpoint: GET /accounts/{slug}/reports/dkim
   - Description: Provides insights into continuously monitoring and protecting DKIM configurations for domains.

6. Senders Continuous Protection

   - Endpoint: GET /accounts/{slug}/reports/senders
   - Description: Returns reports on the continuous protection of email senders associated with the customer's account, ensuring ongoing compliance and performance monitoring.

7. Response Data Fields (Field Name Example value)
   - Get List of Domains/Subdomains
     i. Domain: dmarceverywhere.com
     ii. Dmarc-record: CNAME
     iii. Enforcement-status: Not At Enforcement
     iv. Dmarc-policy: Quarantine
     v. Sending-status: Active
     vi. Account-slug: valimail
     vii. Last-report-date: 2021-01-01

   - Senders Report
     i. Service_name SendGrid
     ii. Service_status Active
     iii. Total_messages_count 1500
     iv. Passing_percentage 95.5

v. Dmarc_passing_count 1400
vi. Dmarc_failing_count 100
vii. Allowed_through_no_enforcement_count 50
viii. Allowed_through_with_enforcement_count 20
ix. Aligned_spf_count 1300
x. Failing_or_unaligned_spf_count 200
xi. Dmarc_overridden_count 30
xii. Quarantined_count 10
xiii. Rejected_count 5

- Unidentified Report
  i. Ptr-name Host1.mail.us
  ii. Source-ip 127.0.0.1
  iii. Geo-country-code US
  iv. Spf-domain Dmarceverywhere.com
  v. Dmarc-policy-org-domain Dmarceverywhere.com
  vi. Passing-percentage 95.5
  vii. Total-messages-count 1500
  viii. Dmarc-passing-count 1400
  ix. Dmarc-failing-count 100
  x. Allowed-through-no-enforcement-count 50
  xi. Allowed-through-with-enforcement-count 20
  xii. Aligned-spf-count 1300
  xiii. Failing-spf-count 200
  xiv. Unaligned-spf-count 100
  xv. Dmarc-overridden-count 30
  xvi. quarantined-count 10
  xvii. Rejected-count 5

- Events Log
  i. Domain: dmarceverywhere.com
  ii. Event-type: Service Deleted
  iii. Event-change: Mailgun by Sinch was deleted from dmarceverywhere.com
  iv. User: user@dmarceverywhere.com
  v. Performed-at: 2021-01-01 15:11

- DKIM Continuous Protection
  i. Sender-name: SendGrid
  ii. Domain-name: financeco.example
  iii. Dkim-selector: google
  iv. Dkim-type: CNAME
  v. Dkim-bits: 1234
  vi. Uploaded-date: 2020-01-01 13:37
  vii. Last-seen-date: 2020-02-01 13:37
  viii. Sender-owner: none

ix. Domain-owner: none

- Senders Continuous Protection
  i. Sender-name: SendGrid
  ii. Domain-name: financeco.example
  iii. Last-seen-date: 2020-02-01 13:37
  iv. Email-volume: 9001
  v. Notes: Added by Matt. 03/03/25. RFC#76354

8. General Use Cases
   - Streamlined Domain Inventory
     i. Scenario: Customers need a quick and easy way to list all domains/subdomains associated with their account to ensure they are correctly configured and monitored.
     ii. Outcome: Customers can efficiently retrieve a comprehensive list of all domains/subdomains, enabling them to verify configurations, identify any missing or misconfigured domains, and maintain an accurate inventory.

   - Proactive Threat Analysis
     i. Scenario: Customers use /reports/unidentified to detect and investigate suspicious email traffic.
     ii. Outcome: Early threat detection and mitigation.

   - Benchmarking Security Metrics
     i. Scenario: Customers use aggregated sender data to provide customers with industry benchmarks.
     ii. Outcome: Helps customers improve their email security posture.

9. Detailed Business Use Cases

Senders and Unidentified Senders Reports

- Identifying Known Senders for Classification
  i. Endpoint: /accounts/{slug}/reports/senders
  ii. Use Case: A Customer integrates the sender's report data to identify high-volume email senders for their customers. By monitoring these senders' passing DMARC rates, customers can prioritize resources to improve email authentication compliance.
  iii. Outcome: Enhanced email deliverability and customer trust due to secure sender identification.

- Domain-Specific Threat Analysis
  i. Endpoint: /accounts/{slug}/reports/unidentified
  ii. Use Case: A Customer uses unidentified report data to help customers identify and investigate suspicious or unidentified email traffic on specific domains/subdomains.
  iii. Outcome: Early detection and mitigation of phishing or spoofing attempts improve the customer's security posture.

- Optimizing Email Campaigns with Sender Insights
    i. Endpoint: /accounts/{slug}/reports/senders
    ii. Use Case: Customers analyze sender reports to identify email marketing campaigns or transactional emails with lower DMARC compliance. They advise their customers to update configurations or switch to trusted Senders.
    iii. Outcome: Improved email engagement rates and higher ROI on marketing campaigns.

- Monitoring Email Authentication Metrics for Compliance
    i. Endpoint: /accounts/{slug}/reports/senders
    ii. Use Case: Customers use the data to ensure their customers' third-party email providers comply with DMARC and SPF policies.
    iii. Outcome: Customers can hold providers accountable for upholding email security standards.

- Proactively Blocking Untrusted Senders
    i. Endpoint: /accounts/{slug}/reports/senders
    ii. Use Case: By identifying senders with consistently low DMARC compliance or high failure rates, Customers can recommend policies to block untrusted senders for their customers.
    iii. Outcome: Reduced exposure to spam and phishing attempts, increasing trust in the email ecosystem.

- Regional Compliance & Reporting
    i. Endpoint: /accounts/{slug}/reports/unidentified
    ii. Use Case: Customers use geo-country-code data to track email traffic from specific geographic locations and ensure compliance with regional regulations such as GDPR or CCPA.
    iii. Outcome: Improved adherence to regulatory requirements and avoidance of compliance penalties.

- Reporting for Executive Dashboards
    i. Endpoint: /accounts/{slug}/reports/senders and /accounts/{slug}/reports/unidentified
    ii. Use Case: Customers aggregate sender and unidentified report data into executive dashboards, giving customers insights into their email traffic authentication performance and potential risks.
    iii. Outcome: Improved decision-making at the executive level (CISO) backed by real-time data.

- Data-Driven Prioritization of Security Investments
    i. Endpoint: /accounts/{slug}/reports/unidentified
    ii. Use Case: Customers prioritize investments in email security by identifying high-risk areas such as domains/subdomains with significant unidentified traffic or low compliance rates.
    iii. Outcome: Efficient allocation of resources to address critical security gaps.

- Custom Alerts for Non-Compliant Traffic
    i. Endpoint: /accounts/{slug}/reports/senders
    ii. Use Case: Customers integrate with this API to build custom alerts when DMARC passing rates fall below a threshold, enabling proactive responses to potential email authentication failures.
    iii. Outcome: Minimized downtime and better reputation management through early action.

- Benchmarking Across Industry Peers
    i. Endpoint: /accounts/{slug}/reports/senders
    ii. Use Case: Customers use aggregated data from multiple customers to offer anonymized industry benchmarks, helping customers compare their email security posture with peers.
    iii. Outcome: Competitive advantage and improved email security practices aligned with industry leaders.

Events Log Report

- Enhanced Security Audit Trail
    i. Endpoint: GET /accounts/{slug}/reports/events
    ii. Scenario: Customers must maintain a detailed record of all changes to their domain and user configurations for compliance and security purposes.
    iii. Outcome: A customer can retrieve a comprehensive log of events, including who made the changes, when they were made, and what was changed, providing a clear audit trail for security audits and regulatory compliance.

- Rapid Configuration Error Detection
    i. Endpoint: GET /accounts/{slug}/reports/events
    ii. Scenario: Customers suspect a recent misconfiguration is causing email delivery issues and need to identify the source of the problem quickly.
    iii. Outcome: By filtering the events log by event type (e.g., "domain_deleted," "dmarc_policy_set_to_none") and timeframe, a customer can quickly pinpoint the configuration change that caused the issue, enabling faster troubleshooting and resolution.

- User Activity Monitoring
    i. Endpoint: GET /accounts/{slug}/reports/events
    ii. Scenario: Customers want to monitor the actions of specific users within their Valimail account to ensure they adhere to security policies and best

practices.

iii. Outcome: Customers can filter the events log by user ID to track individual users' actions, providing insights into their activities and helping to identify potential security risks or policy violations.

Continuous Protection Reports

- Proactive DKIM Key Rotation Management
  i. Endpoint: GET /accounts/{slug}/reports/continuous-protection/dkim
  ii. Scenario: Customers want to ensure their DKIM keys are regularly rotated to maintain optimal security and prevent key compromise.
  iii. Outcome: By monitoring the "uploaded-date" and "last-seen-date" of their DKIM keys, the customer can identify keys nearing their rotation deadline and proactively schedule key rotations to maintain a strong security posture.

- Timely Identification of DKIM Configuration Issues
  i. Endpoint: GET /accounts/{slug}/reports/continuous-protection/dkim
  ii. Scenario: A customer needs to quickly identify any issues with their DKIM configurations that could impact email deliverability.
  iii. Outcome: By regularly reviewing the DKIM Continuous Protection report, the customer can identify any misconfigured or non-compliant DKIM keys and immediately resolve the issues, preventing email deliverability problems and maintaining a positive sender reputation.

- Sender Authentication Validation
  i. Endpoint: GET /accounts/{slug}/reports/continuous-protection/dkim
  ii. Scenario: A customer wants to validate that email senders properly authenticate their messages using DKIM to prevent spoofing and phishing attacks.
  iii. Outcome: By filtering the DKIM Continuous Protection report by sender name, the customer can verify that their senders are using valid DKIM keys and that their messages are properly authenticated, reducing the risk of email-based attacks.

- Sender Performance Monitoring and Optimization
  i. Endpoint: GET /accounts/{slug}/reports/continuous-protection/senders
  ii. Scenario: A customer wants to monitor the performance of their email senders to identify areas for improvement and optimize their email campaigns.
  iii. Outcome: By tracking the "email-volume" and "last-seen-date" of their senders, customers can identify senders with low engagement or outdated configurations and improve their performance, such as updating their email

authentication settings or adjusting their sending practices.

- Proactive Identification of Inactive Senders
    i. Endpoint: GET /accounts/{slug}/reports/continuous-protection/senders
    ii. Scenario: A customer wants to identify senders no longer actively sending emails to streamline their email infrastructure and reduce unnecessary overhead.
    iii. Outcome: By filtering the Senders Continuous Protection report by "last-seen-date," the customer can identify inactive senders and remove them from their Valimail configuration, simplifying their email infrastructure and reducing potential security risks.

- Enhanced Sender Security Management
    i. Endpoint: GET /accounts/{slug}/reports/continuous-protection/senders
    ii. Scenario: A customer wants to ensure that all their email senders adhere to security best practices and are not being used maliciously.
    iii. Outcome: By regularly reviewing the Senders Continuous Protection report and investigating any unusual activity, the customer can proactively identify and address potential security threats, such as compromised sender accounts or unauthorized email sending activity.Error Handling and Standard Responses

7. Error Handling and Standard Responses

The Configuration API surface includes (at minimum) the following response codes:

Code	Message	Description
200	Ok – Success of Request	The request was completed successfully.
201	Created	Resource successfully created (commonly used by SCIM create).
400	Bad Request	Invalid input, missing parameters, or invalid payload/filter syntax.
401	Unauthorized	Authentication failed or missing credentials.
403	Forbidden	Authorization failed or quota/permission constraints (e.g., sender quota exceeded).

404	Not Found	The resource does not exist (domain/sender/dkim/user/report not found).
409	Conflict	Duplicate resource or conflicting state (e.g., existing selector or user already exists).
422	Unprocessable Entity	Validation/normalization errors (e.g., invalid domain, invalid policy mode).
429	Too Many Requests	Rate limit exceeded.
500	Internal Server Error	Unexpected server error.

8. Helpful Links

- Valimail API Help Center Articles
- Swagger Documentation
  - DMARC Configuration
  - Reporting Data

Thank You

Contact Name
Support@Valimail.com