# 2. SIEM API Details

## 1. Accounts APIs

This document provides a comprehensive reference for the SIEM (Security Information and Event Management) API. The API allows developers to interact with various components of the SIEM system, including user management, alert handling, log analysis, and reporting.

---

## 2. Authentication

### Obtain JWT Token

- **URL:** `/api/v1/accounts/token/`
- **Method:** `POST`
- **Description:** Obtain a JWT access token by providing valid user credentials.

  **Request:**

  ```
  {
    "email": "user@example.com",
    "password": "password123"
  }
  ```

  **Response:**

  ```
  {
    "refresh": "jwt-refresh-token",
    "access": "jwt-access-token"
  }
  ```

### Refresh JWT Token

- **URL:** `/api/v1/accounts/token/refresh/`
- **Method:** `POST`
- **Description:** Refresh the access token using a valid refresh token.

  **Request:**

  ```
  {
    "refresh": "jwt-refresh-token"
  }
  ```

  **Response:**

```
{
  "access": "new-jwt-access-token"
}
```

---

## 3. API Endpoints

### Accounts API

### User Management

- **List Users**
  - **URL:** `/api/v1/accounts/users/`
  - **Method:** `GET`
  - **Description:** Retrieve a list of all users.
  - **Response:**

```
[
  {
    "user_id": "uuid",
    "email": "user@example.com",
    "role": {
      "role_id": "uuid",
      "name": "Admin",
      "permissions": [
        {"permission_id": "uuid", "permission_name": "Can view users"}
      ]
    },
    "is_active": true,
    "is_staff": false
  },
  ...
]
```

- **Create User**
  - **URL:** `/api/v1/accounts/users/`
  - **Method:** `POST`
  - **Description:** Create a new user.
  - **Request:**

```
{
  "email": "user2@example.com",
  "password": "password123",
  "role_id": "uuid"
}
```

  - **Response:**

```
{
  "user_id": "uuid",
  "email": "user2@example.com",
```

```
    "role": {
      "role_id": "uuid",
      "name": "Analyst"
    },
    "is_active": true,
    "is_staff": false
  }
```

- **Retrieve User**
  - **URL:** `/api/v1/accounts/users/{user_id}/`
  - **Method:** `GET`
  - **Description:** Get details of a specific user by their ID.
  - **Response:**

```
{
  "user_id": "uuid",
  "email": "user@example.com",
  "role": {
    "role_id": "uuid",
    "name": "Admin"
  },
  "is_active": true,
  "is_staff": false
}
```

- **Update User**
  - **URL:** `/api/v1/accounts/users/{user_id}/`
  - **Method:** `PUT`
  - **Description:** Update an existing user's details.
  - **Request:**

```
{
  "email": "updated_user@example.com",
  "role_id": "uuid",
  "is_active": false
}
```

  - **Response:**

```
{
  "user_id": "uuid",
  "email": "updated_user@example.com",
  "role": {
    "role_id": "uuid",
    "name": "Admin"
  },
  "is_active": false,
  "is_staff": false
}
```

- **Delete User**
  - **URL:** `/api/v1/accounts/users/{user_id}/`
  - **Method:** `DELETE`

- **Description:** Delete a user.
- **Response:**

```
{
  "message": "User deleted successfully"
}
```

**Role Management**

- **List Roles**
  - **URL:** `/api/v1/accounts/roles/`
  - **Method:** `GET`
  - **Description:** Retrieve a list of all roles.
  - **Response:**

```
[
  {
    "role_id": "uuid",
    "name": "Admin",
    "permissions": [
      {"permission_id": "uuid", "permission_name": "Can view users"}
    ]
  },
  ...
]
```

- **Create Role**
  - **URL:** `/api/v1/accounts/roles/`
  - **Method:** `POST`
  - **Description:** Create a new role.
  - **Request:**

```
{
  "name": "Editor"
}
```

  - **Response:**

```
{
  "role_id": "uuid",
  "name": "Editor",
  "permissions": []
}
```

- **Retrieve Role**
  - **URL:** `/api/v1/accounts/roles/{role_id}/`
  - **Method:** `GET`
  - **Description:** Retrieve a specific role by ID.
  - **Response:**

```
{
  "role_id": "uuid",
  "name": "Admin",
  "permissions": [
```

```
      {"permission_id": "uuid", "permission_name": "Can view users"}
    ]
  }
```

- **Update Role**
  - **URL:** `/api/v1/accounts/roles/{role_id}/`
  - **Method:** `PUT`
  - **Description:** Update an existing role.
  - **Request:**
    ```
    {
      "name": "Moderator"
    }
    ```
  - **Response:**
    ```
    {
      "role_id": "uuid",
      "name": "Moderator",
      "permissions": []
    }
    ```
- **Delete Role**
  - **URL:** `/api/v1/accounts/roles/{role_id}/`
  - **Method:** `DELETE`
  - **Description:** Delete a role.
  - **Response:**
    ```
    {
      "message": "Role deleted successfully"
    }
    ```

**Permission Management**

- **List Permissions**
  - **URL:** `/api/v1/accounts/permissions/`
  - **Method:** `GET`
  - **Description:** Retrieve a list of all permissions.
  - **Response:**
    ```
    [
      {
        "permission_id": "uuid",
        "permission_name": "Can view users"
      },
      ...
    ]
    ```
- **Create Permission**
  - **URL:** `/api/v1/accounts/permissions/`
  - **Method:** `POST`
  - **Description:** Create a new permission.
  - **Request:**
    ```
    {
    ```

```
  "permission_name": "Can edit users"
}
```

- **Response:**

```
{
  "permission_id": "uuid",
  "permission_name": "Can edit users"
}
```

**Role-Permission Management**

- **List Role-Permissions**
  - **URL:** `/api/v1/accounts/role-permissions/`
  - **Method:** `GET`
  - **Description:** Retrieve all role-permission relationships.
  - **Response:**

```
[
  {
    "role": {
      "role_id": "uuid",
      "name": "Admin"
    },
    "permission": {
      "permission_id": "uuid",
      "permission_name": "Can view users"
    }
  },
  ...
]
```

- **Assign Permissions to Role**
  - **URL:** `/api/v1/accounts/role-permissions/`
  - **Method:** `POST`
  - **Description:** Assign a permission to a role.
  - **Request:**

```
{
  "role": "uuid",
  "permission": "uuid"
}
```

  - **Response:**

```
{
  "message": "Permission assigned to role"
}
```

---

## 4. Additional Information

### Error Handling

All endpoints return standardized error messages in the following format:

```
{
  "error": "Error description"
}
```

### Authentication Requirements

Most endpoints require authentication using JWT tokens. Ensure to include the token in the `Authorization` header as follows:

`Authorization: Bearer <your_jwt_token>`

---

## Alerts API Endpoints

The Alerts API allows users to manage and investigate alerts generated by the system. It provides endpoints to create, retrieve, update, and delete alerts, as well as manage investigations.

### Alerts

1. **List Alerts**

   - **Endpoint:** `GET /api/v1/alerts/`
   - **Description:** Retrieves a list of all alerts.
   - **Query Parameters:**
     - `severity`: Filter alerts by severity (INFO, LOW, MEDIUM, HIGH, CRITICAL).
     - `rule__name`: Filter alerts by rule name.
   - **Response:**
     - **200 OK**: A list of alerts.

2. **Create Alert**

   - **Endpoint:** `POST /api/v1/alerts/`
   - **Description:** Creates a new alert.
   - **Request Body:**

     ```
     {
       "rule": <Rule ID>,
       "event": <Event ID>,
       "severity": <Severity Level>,
     ```

```
  "comments": "Optional comments"
}
```

- **Response:**
  - **201 Created**: The created alert object.
  - **400 Bad Request**: Invalid input.

3. **Retrieve Alert**

- **Endpoint:** `GET /api/v1/alerts/{id}/`
- **Description:** Retrieves a specific alert by ID.
- **Response:**
  - **200 OK**: The requested alert object.
  - **404 Not Found**: Alert not found.

4. **Update Alert**

- **Endpoint:** `PATCH /api/v1/alerts/{id}/`
- **Description:** Updates a specific alert by ID.
- **Request Body:**

```
{
  "severity": <New Severity Level>,
  "comments": "Updated comments"
}
```

- **Response:**
  - **200 OK**: The updated alert object.
  - **404 Not Found**: Alert not found.

5. **Delete Alert**

- **Endpoint:** `DELETE /api/v1/alerts/{id}/`
- **Description:** Deletes a specific alert by ID.
- **Response:**
  - **204 No Content**: Successfully deleted.
  - **404 Not Found**: Alert not found.

6. **Assign Alert**

- **Endpoint:** `POST /api/v1/alerts/{id}/assign/`
- **Description:** Assigns an alert to an investigator.
- **Request Body:**

```
{
  "assigned_to": <User ID>
}
```

- **Response:**
  - **201 Created**: Alert assigned successfully.
  - **400 Bad Request**: No analyst ID provided.
  - **404 Not Found**: Analyst not found.

7. **Latest Alerts**

- **Endpoint:** `GET /api/v1/alerts/latest_alerts/`
- **Description:** Retrieves the latest 4 alerts.
- **Response:**
  - **200 OK**: A list of the latest alerts.

### Investigate Alerts

1. **List Investigate Alerts**

- **Endpoint:** `GET /api/v1/investigate/`
- **Description:** Retrieves a list of investigation alerts.
- **Query Parameters:**
  - `assigned_to__email`: Filter investigations by assigned analyst's email.
  - `alert__severity`: Filter investigations by alert severity.
  - `status`: Filter investigations by status (OPEN, IN PROGRESS, CLOSED).
- **Response:**
  - **200 OK**: A list of investigation alerts.

2. **Create Investigate Alert**

- **Endpoint:** `POST /api/v1/investigate/`
- **Description:** Creates a new investigation alert.
- **Request Body:**

```
{
  "alert": <Alert ID>,
  "assigned_to": <User ID>,
  "notes": "Optional notes"
}
```

- **Response:**
  - **201 Created**: The created investigation alert object.
  - **400 Bad Request**: Invalid input.

3. **Retrieve Investigate Alert**

- **Endpoint:** `GET /api/v1/investigate/{id}/`
- **Description:** Retrieves a specific investigation alert by ID.
- **Response:**
  - **200 OK**: The requested investigation alert object.
  - **404 Not Found**: Investigation alert not found.

4. **Update Investigation Alert**

- **Endpoint:** `PATCH /api/v1/investigate/{id}/update_investigation/`
- **Description:** Updates the investigation alert.
- **Request Body:**

```
{
  "status": <New Status>,
  "notes": "Updated notes"
}
```

- **Response:**
  - **200 OK**: The updated investigation alert object.
  - **403 Forbidden**: User does not have permission to update.
  - **404 Not Found**: Investigation alert not found.

5. **Investigation Status Count**

- **Endpoint:** `GET /api/v1/investigate/investigation_status_count/`
- **Description:** Retrieves counts of investigations by status (CLOSED, OPEN, IN PROGRESS).
- **Response:**
  - **200 OK**: A count of investigations per status.

6. **Open Investigations**

- **Endpoint:** `GET /api/v1/investigate/open_investigations/`
- **Description:** Fetches open investigations for a specific user or all users (for ADMIN).
- **Response:**
  - **200 OK**: A list of open investigations.
  - **403 Forbidden**: Unauthorized access.

## Models

### Alert

- `id`: Integer (primary key)
- `rule`: ForeignKey to `Rule`

- `event`: ForeignKey to `BronzeEventData`
- `severity`: String (choices: INFO, LOW, MEDIUM, HIGH, CRITICAL)
- `comments`: TextField (optional)

**InvestigateAlert**

- `id`: Integer (primary key)
- `alert`: OneToOneField to `Alert`
- `assigned_to`: ForeignKey to `User`
- `status`: String (choices: OPEN, IN PROGRESS, CLOSED)
- `notes`: TextField (optional)

## Status Codes

- `200 OK`: Successful requests.
- `201 Created`: Resource successfully created.
- `204 No Content`: Resource successfully deleted.
- `400 Bad Request`: Invalid input or missing required fields.
- `403 Forbidden`: Access denied.
- `404 Not Found`: Resource not found.

**Core Endpoints**

1. **List All Rules**

- **Endpoint**: `/api/v1/rules/`
- **Method**: `GET`
- **Description**: Fetch a list of all rules defined in the system.
- **Response**:
  - **200 OK**: Returns a list of rules.
  - **Example**:

```
[
  {
    "id": 1,
    "name": "Login Failure Alert",
    "description": "Trigger on 5 failed login attempts",
    "conditions": "Failed logins > 5",
    "severity": "HIGH"
  }
]
```

2. **Retrieve a Single Rule**

- **Endpoint**: `/api/v1/rules/{id}/`
- **Method**: `GET`
- **Description**: Fetch details of a specific rule by ID.
- **Response**:

- **200 OK**: Returns details of the rule.
- **404 Not Found**: Rule not found.
- **Example**:

```
{
  "id": 1,
  "name": "Login Failure Alert",
  "description": "Trigger on 5 failed login attempts",
  "conditions": "Failed logins > 5",
  "severity": "HIGH"
}
```

3. **Create a New Rule**

- **Endpoint**: `/api/v1/rules/`
- **Method**: `POST`
- **Description**: Create a new rule by providing the necessary data.
- **Request Body**:
  - **Example**:

```
{
  "name": "File Deletion Alert",
  "description": "Trigger on mass file deletions",
  "conditions": "Files deleted > 10",
  "severity": "CRITICAL"
}
```

- **Response**:
  - **201 Created**: Rule successfully created.
  - **400 Bad Request**: Validation error.
  - **Example**:

```
{
  "id": 2,
  "name": "File Deletion Alert",
  "description": "Trigger on mass file deletions",
  "conditions": "Files deleted > 10",
  "severity": "CRITICAL"
}
```

4. **Update an Existing Rule**

- **Endpoint**: `/api/v1/rules/{id}/`
- **Method**: `PUT`
- **Description**: Update the details of a specific rule.
- **Request Body**:
  - **Example**:

```
{
  "name": "Updated Rule Name",
  "description": "Updated rule description",
```

```
    "conditions": "Updated conditions",
    "severity": "MEDIUM"
}
```

- **Response**:
  - **200 OK**: Rule successfully updated.
  - **404 Not Found**: Rule not found.
  - **400 Bad Request**: Validation error.

**5. Delete a Rule**

- **Endpoint**: `/api/v1/rules/{id}/`
- **Method**: `DELETE`
- **Description**: Delete a specific rule by ID.
- **Response**:
  - **204 No Content**: Rule successfully deleted.
  - **404 Not Found**: Rule not found.

## Models

### Rule Model

Represents the structure of a rule.

- **id**: Primary key, auto-incremented integer.
- **name**: A descriptive name for the rule.
- **description**: A short description explaining the rule.
- **conditions**: The conditions that trigger the rule, stored as text.
- **severity**: The severity level of the rule (choices: `INFO`, `LOW`, `MEDIUM`, `HIGH`, `CRITICAL`).

Example:

```
{
  "id": 1,
  "name": "Login Failure Alert",
  "description": "Trigger on 5 failed login attempts",
  "conditions": "Failed logins > 5",
  "severity": "HIGH"
}
```

## Logs API Endpoint

## Base URL

`/api/v1/logs/`

## Bronze Event Data

### List Bronze Events

- **Endpoint:** `GET /api/v1/logs/bronze-events/`
- **Description:** Retrieve a list of bronze event data.
- **Query Parameters:**
  - `query`: Search term for hostname, AccountName, or message (optional).
  - `start_time`: Filter results to include events starting from this timestamp (optional).
  - `end_time`: Filter results to include events up to this timestamp (optional).
  - `event_type`: Filter results to include events of this type (optional).
- **Response:**
  - **200 OK**: Returns a list of bronze event data.
  - **Example Response:**

```
[
  {
    "id": 1,
    "iso_timestamp": "2023-01-01T00:00:00Z",
    "hostname": "example-host",
    "EventType": "Error",
    "EventID": "1001",
    "AccountName": "admin",
    "message": "An error occurred"
  }
]
```

### Export Bronze Events as PDF

- **Endpoint:** `GET /api/v1/logs/bronze-events/export_pdf/`
- **Description:** Export the list of bronze events as a PDF report.
- **Response:**
  - **200 OK**: Returns a PDF document.

### Count Bronze Events

- **Endpoint:** `GET /api/v1/logs/bronze-events/count/`
- **Description:** Retrieve the total count of bronze events.
- **Response:**
  - **200 OK**: Returns the count of bronze events.
  - **Example Response:**

```
{
  "count": 100
}
```

## Router Data

### List Router Data

- **Endpoint:** `GET /api/v1/logs/router-data/`
- **Description:** Retrieve a list of router data logs.
- **Query Parameters:**
  - `query`: Search term for hostname or message (optional).
  - `start_time`: Filter results to include logs starting from this timestamp (optional).
  - `end_time`: Filter results to include logs up to this timestamp (optional).
  - `process`: Filter results to include logs from this process (optional).
- **Response:**
  - **200 OK**: Returns a list of router data logs.
  - **Example Response:**

```
[
  {
    "id": 1,
    "date_time": "2023-01-01T00:00:00Z",
    "hostname": "router-1",
    "process": "router_process",
    "message": "Router log message"
  }
]
```

### Export Router Data as PDF

- **Endpoint:** `GET /api/v1/logs/router-data/export_pdf/`
- **Description:** Export the list of router data logs as a PDF report.
- **Response:**
  - **200 OK**: Returns a PDF document.

### Count Router Data Logs

- **Endpoint:** `GET /api/v1/logs/router-data/router_log_count/`
- **Description:** Retrieve the total count of router data logs.
- **Response:**
  - **200 OK**: Returns the count of router logs.
  - **Example Response:**

```
{
  "router_log_count": 50
}
```

## Log Percentage

### Get Log Percentages

- **Endpoint:** `GET /api/v1/logs/log-percentage/log_percentages/`
- **Description:** Retrieve the percentage of logs from bronze event data and router data.
- **Response:**
  - **200 OK**: Returns the percentages.
  - **Example Response:**

```
{
  "windows_os_percentage": 66.67,
  "network_percentage": 33.33
}
```

## Log Aggregation

### Logs Per Hour

- **Endpoint:** `GET /api/v1/logs/logs-aggregation/logs_per_hour/`
- **Description:** Retrieve the count of logs per hour for both bronze events and router data.
- **Response:**
  - **200 OK**: Returns the hourly log count.
  - **Example Response:**

```
[
  {
    "name": "12:00:00",
    "Computer": 10,
    "Networking": 5
  }
]
```

## Events Today

### Count Events Today

- **Endpoint:** `GET /api/v1/logs/events-today/events_today/`
- **Description:** Count the number of events that occurred today.
- **Response:**
  - **200 OK**: Returns the count of today's events.
  - **Example Response:**

```
{
  "events_today": 20
}
```

## Hostname Count

### Count Unique Hostnames

- **Endpoint:** `GET /api/v1/logs/hostname-count/`
- **Description:** Count the number of unique hostnames across bronze event data and router data.
- **Response:**
  - **200 OK**: Returns the total number of unique hostnames.
  - **Example Response:**

```
{
  "total_devices": 15
}
```

## Reports API Endpoints

### List Incident Reports

**GET** `/api/v1/reports/incident-reports/`

**Description**: Retrieve a list of all incident reports.

**Query Parameters**: - `search` (optional): Search incident reports by title, description, rule name, or user email. - `type` (optional): Filter by report type (e.g., `security_incident`, `network_traffic`). - `status` (optional): Filter by report status (e.g., `draft`, `open`). - `user__email` (optional): Filter by user email. - `last_update` (optional): Filter by the last update date.

**Response**: - **Status**: `200 OK` - **Body**: List of incident reports.

---

### Create Incident Report

**POST** `/api/v1/reports/incident-reports/`

**Description**: Create a new incident report.

**Request Body**:

```
{
  "title": "Incident Title",
  "type": "security_incident",
  "status": "open",
  "description": "Detailed description of the incident.",
  "rule_ids": [1, 2, 3]  // List of related rule IDs
}
```

**Response**: - **Status**: `200 OK` - **Body**: The created incident report.

**Retrieve Incident Report**

**GET** `/api/v1/reports/incident-reports/{id}/`

**Description**: Retrieve a specific incident report by ID.

**Response**: - **Status**: `200 OK` - **Body**: The incident report details.

**Update Incident Report**

**PUT** `/api/v1/reports/incident-reports/{id}/`

**Description**: Update an existing incident report by ID.

**Response**: - **Status**: `200 OK` - **Body**: The updated incident report.

**Delete Incident Report**

**DELETE** `/api/v1/reports/incident-reports/{id}/`

**Description**: Delete an incident report by ID.

**Response**: - **Status**: `204 No Content` - **Body**: Success message.

**Generate PDF Report**

**GET** `/api/v1/reports/incident-reports/{id}/generate_pdf/`

**Description**: Generate a PDF report for a specific incident report by ID.

**Response**: - **Status**: `200 OK` - **Body**: The PDF content of the incident report.

**Error Responses**

- **404 Not Found**: If the specified incident report ID does not exist.
- **400 Bad Request**: If the request body is invalid.
- **403 Forbidden**: If the user does not have permission to perform the action.