SOAR

Yehia A. Mostafa

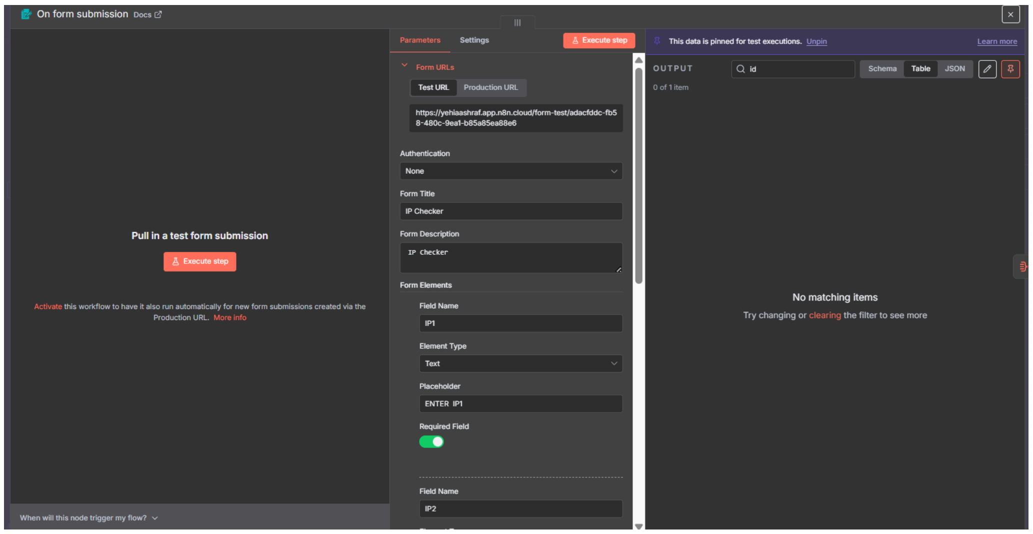
Accounts created: VirusTotal and n8n.

---

1) First Workflow — VirusTotal lookup from a form

1.1 What I built

- Created a new **workflow** in n8n.
- Added two nodes:
  - **On Form Submission** (with an **IP** field on the form).
  - **HTTP Request** (to call VirusTotal's IP intelligence endpoint).



Activate this workflow to have it also run automatically for new form submissions created via the Production URL. More info

Form URLs

https://yehiaashraf.app.n8n.cloud/form-test/adacfddc-fb58-480c-9ea1-b85a85ea88e6

Authentication: None

Form Title: IP Checker

Form Description: IP Checker

Form Elements

Field Name: IP1

Element Type: Text

Placeholder: ENTER IP1

Required Field: ✓

Field Name: IP2

This data is pinned for test executions. Unpin Learn more

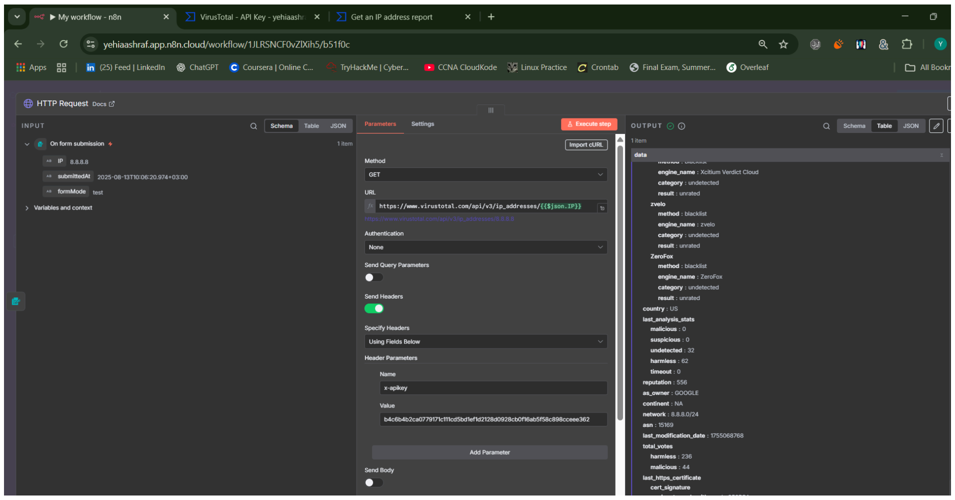
OUTPUT id

0 of 1 item

No matching items

Try changing or clearing the filter to see more

- In **HTTP Request**, added my **VirusTotal API key** (**censored**) and configured a **GET** request using the official documentation URL.



yehiaashraf.app.n8n.cloud/workflow/1JLRSNCF0vZlXih5/b51f0c

HTTP Request

INPUT

```
On form submission                                   1 item
  IP            8.8.8.8
  submittedAt   2025-08-13T10:06:20.974+03:00
  formMode      test
Variables and context
```

Method: GET

URL: `https://www.virustotal.com/api/v3/ip_addresses/{{$json.IP}}`

https://www.virustotal.com/api/v3/ip_addresses/8.8.8.8

Authentication: None

Send Query Parameters: off

Send Headers: on

Specify Headers: Using Fields Below

Header Parameters

Name: x-apikey

Value: b4c6b4b2ca0779171c111cd5bd1ef1d2128d0928cb0f16ab5f58c898cceee362

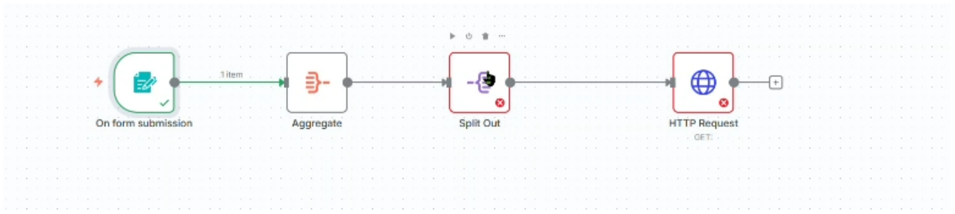
Send Body: off

OUTPUT

```
1 item
data
      engine_name : Xcitium Verdict Cloud
      category : undetected
      result : unrated
    zvelo
      method : blacklist
      engine_name : zvelo
      category : undetected
      result : unrated
    ZeroFox
      method : blacklist
      engine_name : ZeroFox
      category : undetected
      result : unrated
  country : US
  last_analysis_stats
    malicious : 0
    suspicious : 0
    undetected : 32
    harmless : 62
    timeout : 0
  reputation : 556
  as_owner : GOOGLE
  continent : NA
  network : 8.8.8.0/24
  asn : 15169
  last_modification_date : 1755068768
  total_votes
    harmless : 236
    malicious : 44
  last_https_certificate
    cert_signature
```

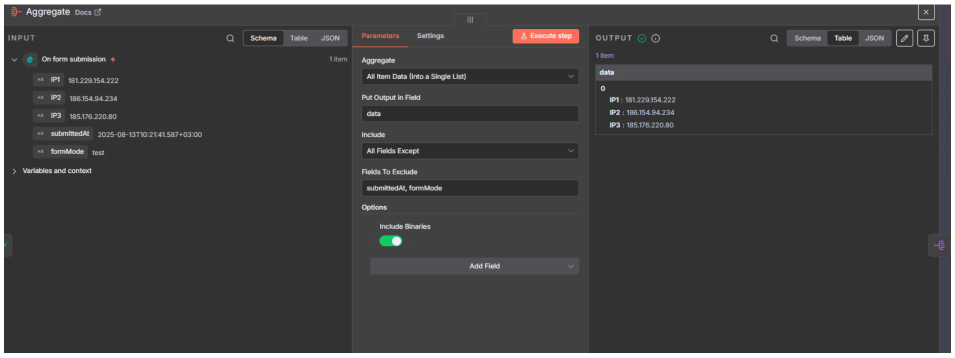
1.2 Multiple IPs (Aggregate + Split Out)

- Built another flow to accept multiple IPs.
- Used **Aggregate** to gather inputs, then **Split Out** to emit one item per IP.



1.3 Node configurations (Aggregate, Split Out, HTTP Request)

Exact configuration snapshots



Aggregate

INPUT

```
On form submission                                   1 item
  IP1           181.229.154.222
  IP2           186.154.94.234
  IP3           185.176.220.80
  submittedAt   2025-08-13T10:21:41.587+03:00
  formMode      test
Variables and context
```

Aggregate: All Item Data (Into a Single List)

Put Output in Field: data

Include: All Fields Except

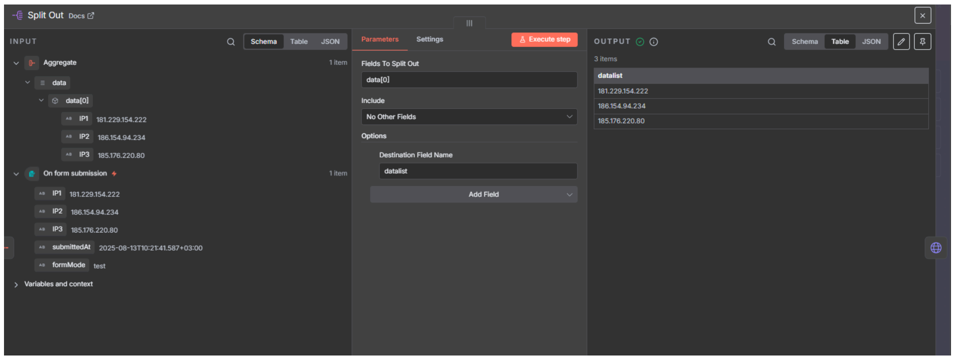
Fields To Exclude: submittedAt, formMode

Options

Include Binaries: on

OUTPUT

```
1 item
data
0
  IP1 : 181.229.154.222
  IP2 : 186.154.94.234
  IP3 : 185.176.220.80
```



Split Out Docs

INPUT

Aggregate — 1 item
- data
  - data[0]
    - IP1 181.229.154.222
    - IP2 186.154.94.234
    - IP3 185.176.220.80

On form submission — 1 item
- IP1 181.229.154.222
- IP2 186.154.94.234
- IP3 185.176.220.80
- submittedAt 2025-08-13T10:21:41.587+03:00
- formMode test

Variables and context

Parameters

Fields To Split Out: data[0]

Include: No Other Fields

Options

Destination Field Name: datalist

OUTPUT

3 items

datalist
181.229.154.222
186.154.94.234
185.176.220.80

Split Out configuration details

Step 1 — Fields To Split Out

- Set **Fields To Split Out** to: `data[0]`

Step 2 — Destination Field Name

- Set **Destination Field Name** to: `datalist`

  This assigns each split item's value to a clean field name.

Step 3 — Verify output

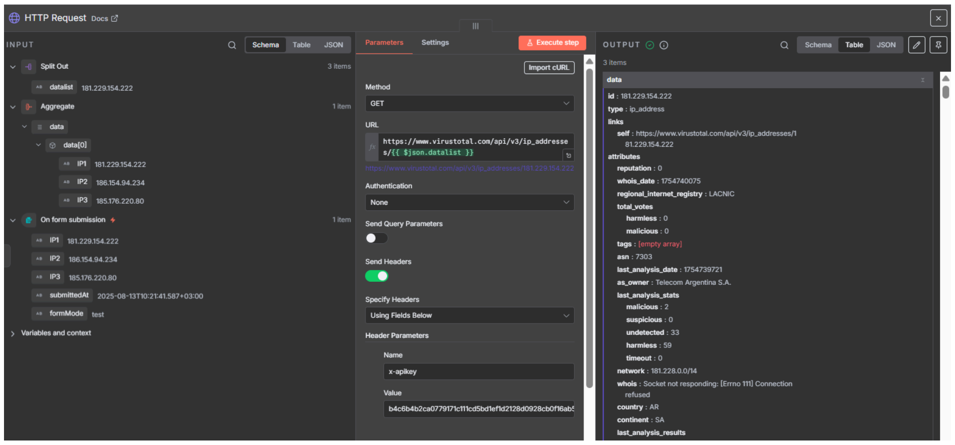
- Confirm output shows **3 items** instead of 1; each item contains one IP's data.

Step 4 — HTTP Request URL

- In **HTTP Request**, set URL to:

  ```
  <https://www.virustotal.com/api/v3/ip_addresses/>{{ $json.datalist }}
  ```

  Uses `$json.datalist` so each request references the individual IP.



HTTP Request Docs

INPUT

Split Out — 3 items
- datalist 181.229.154.222

Aggregate — 1 item
- data
  - data[0]
    - IP1 181.229.154.222
    - IP2 186.154.94.234
    - IP3 185.176.220.80

On form submission — 1 item
- IP1 181.229.154.222
- IP2 186.154.94.234
- IP3 185.176.220.80
- submittedAt 2025-08-13T10:21:41.587+03:00
- formMode test

Variables and context

Parameters

Method: GET

URL:
```
https://www.virustotal.com/api/v3/ip_addresses/{{ $json.datalist }}
```
https://www.virustotal.com/api/v3/ip_addresses/181.229.154.222

Authentication: None

Send Query Parameters: off

Send Headers: on

Specify Headers: Using Fields Below

Header Parameters

Name: x-apikey

Value: b4c6b4b2ca0779171c111cd5bd1ef1d2128d0928cb0f16ab5

OUTPUT

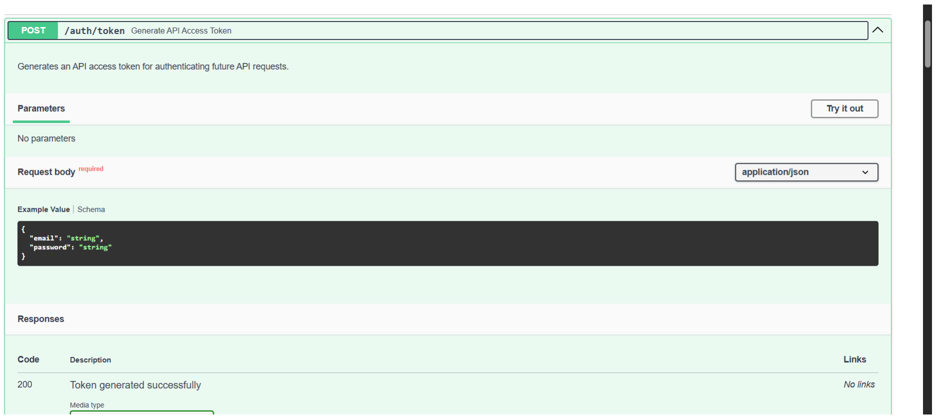
3 items

```
data
id : 181.229.154.222
type : ip_address
links
  self : https://www.virustotal.com/api/v3/ip_addresses/181.229.154.222
attributes
  reputation : 0
  whois_date : 1754740075
  regional_internet_registry : LACNIC
  total_votes
    harmless : 0
    malicious : 0
  tags : [empty array]
  asn : 7303
  last_analysis_date : 1754739721
  as_owner : Telecom Argentina S.A.
  last_analysis_stats
    malicious : 2
    suspicious : 0
    undetected : 33
    harmless : 59
    timeout : 0
  network : 181.228.0.0/14
  whois : Socket not responding: [Errno 111] Connection refused
  country : AR
  continent : SA
  last_analysis_results
```

2) Second Workflow — SIEM-LIKE API at `http://162.216.115.196:8000/docs`

Context

- The exercise used a simulated SIEM API.



POST `/auth/token` Generate API Access Token

Generates an API access token for authenticating future API requests.

Parameters

No parameters

Request body required — application/json

Example Value | Schema

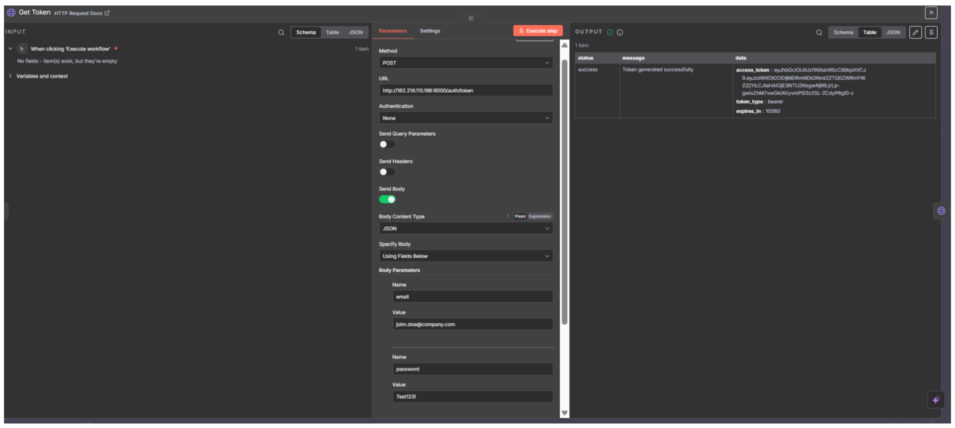
```
{
  "email": "string",
  "password": "string"
}
```

Responses

Code	Description	Links
200	Token generated successfully	*No links*

Media type

- We **authenticate** with **email + password** to obtain a **token**. Example creds used to generate a token:
  - **Email:** `john.doe@company.com`
  - **Password:** `Test123!`



Get Token — HTTP Request Docs

INPUT — When clicking 'Execute workflow' — 1 item
No fields - item(s) exist, but they're empty
Variables and context

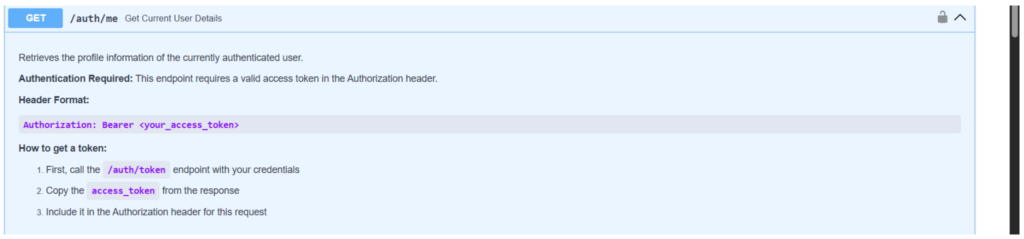
Parameters
- Method: POST
- URL: http://162.216.115.196:8000/auth/token
- Authentication: None
- Send Query Parameters: off
- Send Headers: off
- Send Body: on
- Body Content Type: JSON
- Specify Body: Using Fields Below
- Body Parameters:
  - Name: email — Value: john.doe@company.com
  - Name: password — Value: Test123!

OUTPUT — 1 item

status	message	data
success	Token generated successfully	access_token : eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiI2ODljMDRmNDk5NmI2ZTQ0ZWRmYWZiZjYiLCJleHAiOjE3NTU2NzgwNjR9.jrLp-gwIxZhM7vwGkiAVyvmP5i3z2SL-ZCdyP6gI0-c token_type : bearer expires_in : 10080

- The token can be used to get user info (this was only to demonstrate authorization; **not strictly required** for the rest of the task).

2.1 Authorization elaboration



GET `/auth/me` Get Current User Details

Retrieves the profile information of the currently authenticated user.

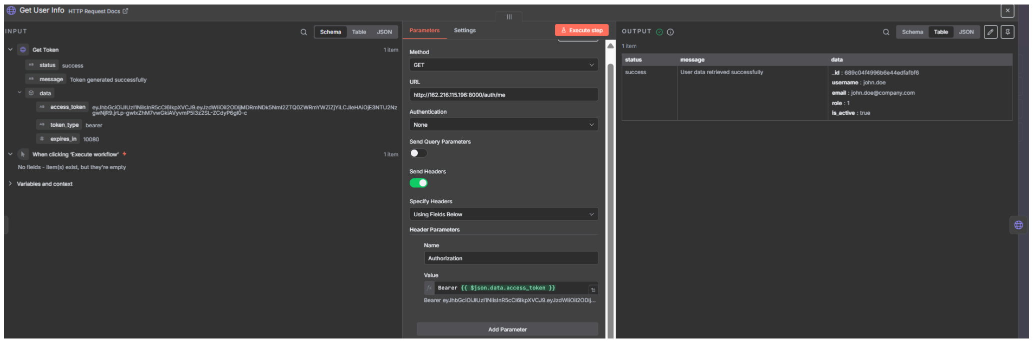
Authentication Required: This endpoint requires a valid access token in the Authorization header.

Header Format:

```
Authorization: Bearer <your_access_token>
```

How to get a token:

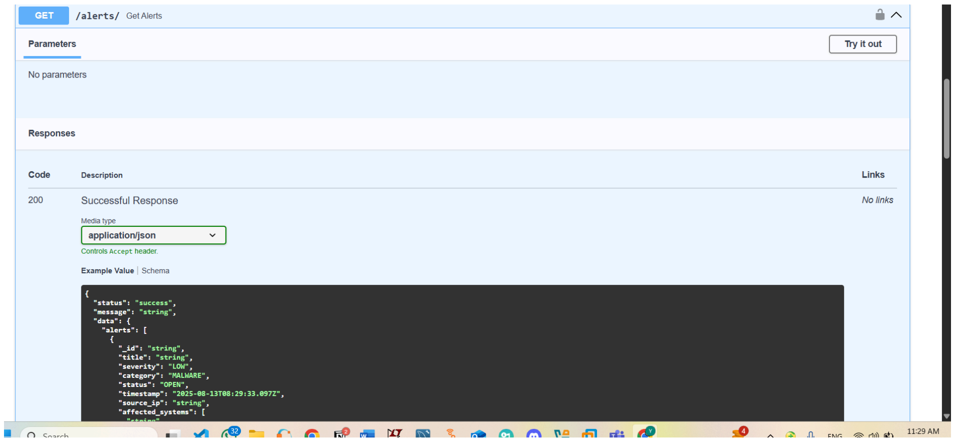
1. First, call the `/auth/token` endpoint with your credentials
2. Copy the `access_token` from the response
3. Include it in the Authorization header for this request



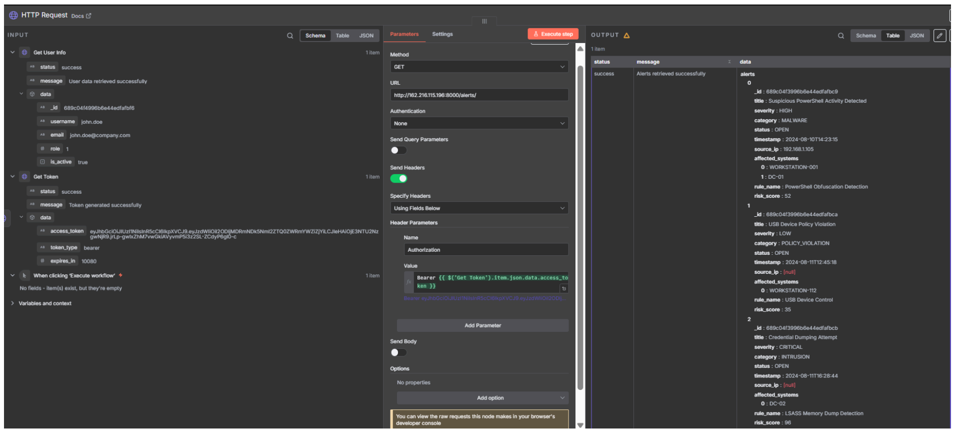
```
Get User Info  HTTP Request Docs

INPUT                                   Schema  Table  JSON
Get Token                                            1 item
  status  success
  message  Token generated successfully
  data
    access_token  eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiI2ODljMDRmNDk5NmI2ZTQ0ZWRmYWZiZjYiLCJleHAiOjE3NTU2Nz
                  gwNjR9.jrLp-gwlxZhM7vwGkiAVyvmP5i3z2SL-ZCdyP6gl0-c
    token_type  bearer
    expires_in  10080
When clicking 'Execute workflow'                     1 item
  No fields - item(s) exist, but they're empty
Variables and context

Parameters  Settings                          Execute step
Method: GET
URL: http://162.216.115.196:8000/auth/me
Authentication: None
Send Query Parameters: off
Send Headers: on
Specify Headers: Using Fields Below
Header Parameters
  Name: Authorization
  Value: Bearer {{ $json.data.access_token }}
         Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiI2ODlj...
Add Parameter

OUTPUT                                  Schema  Table  JSON
1 item
status   | message                          | data
success  | User data retrieved successfully | _id : 689c04f4996b6e44edfafbf6
         |                                  | username : john.doe
         |                                  | email : john.doe@company.com
         |                                  | role : 1
         |                                  | is_active : true
```

2.2 Getting alerts (`/alerts`) and headers

- To fetch alerts, call the `/alerts` index and **authorize using headers** as per documentation.
- Observation: I **still needed authorization**, even when that part of the docs didn't explicitly mention it (**documentations aren't always 100% precise**). I followed the documented header approach to access it.



```
GET  /alerts/  Get Alerts

Parameters                                              Try it out
No parameters

Responses
Code  Description                                       Links
200   Successful Response                               No links
      Media type
      application/json
      Controls Accept header.
      Example Value | Schema
{
  "status": "success",
  "message": "string",
  "data": {
    "alerts": [
      {
        "_id": "string",
        "title": "string",
        "severity": "LOW",
        "category": "MALWARE",
        "status": "OPEN",
        "timestamp": "2025-08-13T08:29:33.097Z",
        "source_ip": "string",
        "affected_systems": [
```



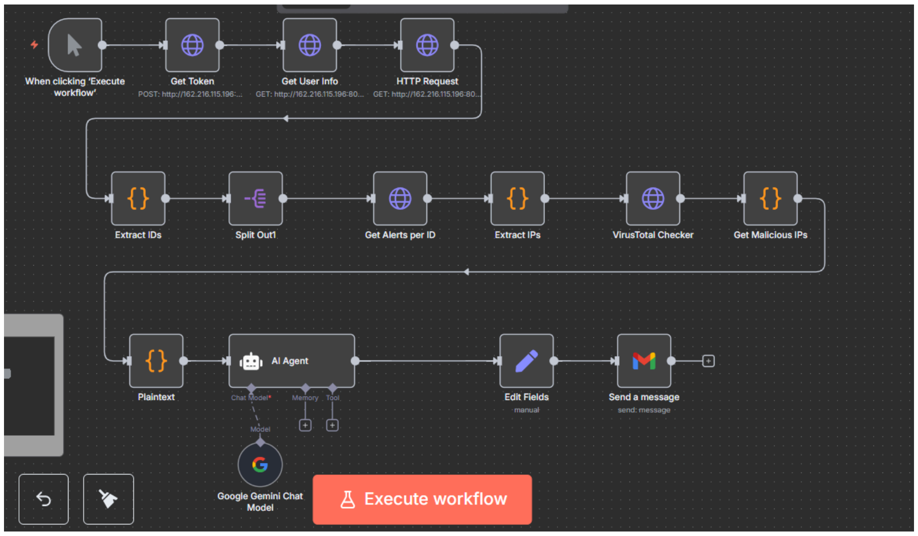
```
HTTP Request  Docs

INPUT                                   Schema  Table  JSON
Get User Info                                        1 item
  status  success
  message  User data retrieved successfully
  data
    _id  689c04f4996b6e44edfafbf6
    username  john.doe
    email  john.doe@company.com
    role  1
    is_active  true
Get Token                                            1 item
  status  success
  message  Token generated successfully
  data
    access_token  eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiI2ODljMDRmNDk5NmI2ZTQ0ZWRmYWZiZjYiLCJleHAiOjE3NTU2Nz
                  gwNjR9.jrLp-gwlxZhM7vwGkiAVyvmP5i3z2SL-ZCdyP6gl0-c
    token_type  bearer
    expires_in  10080
When clicking 'Execute workflow'                     1 item
  No fields - item(s) exist, but they're empty
Variables and context

Parameters  Settings                          Execute step
Method: GET
URL: http://162.216.115.196:8000/alerts/
Authentication: None
Send Query Parameters: off
Send Headers: on
Specify Headers: Using Fields Below
Header Parameters
  Name: Authorization
  Value: Bearer {{ $('Get Token').item.json.data.access_token }}
         Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiI2ODlj...
Add Parameter
Send Body: off
Options
  No properties
Add option
You can view the raw requests this node makes in your browser's developer console

OUTPUT                                  Schema  Table  JSON
1 item
status   | message                      | data
success  | Alerts retrieved successfully| alerts
         |                              |   0
         |                              |     _id : 689c04f3996b6e44edfafbc9
         |                              |     title : Suspicious PowerShell Activity Detected
         |                              |     severity : HIGH
         |                              |     category : MALWARE
         |                              |     status : OPEN
         |                              |     timestamp : 2024-08-10T14:23:15
         |                              |     source_ip : 192.168.1.105
         |                              |     affected_systems
         |                              |       0 : WORKSTATION-001
         |                              |       1 : DC-01
         |                              |     rule_name : PowerShell Obfuscation Detection
         |                              |     risk_score : 52
         |                              |   1
         |                              |     _id : 689c04f3996b6e44edfafbca
         |                              |     title : USB Device Policy Violation
         |                              |     severity : LOW
         |                              |     category : POLICY_VIOLATION
         |                              |     status : OPEN
         |                              |     timestamp : 2024-08-11T12:45:18
         |                              |     source_ip : [null]
         |                              |     affected_systems
         |                              |       0 : WORKSTATION-112
         |                              |     rule_name : USB Device Control
         |                              |     risk_score : 35
         |                              |   2
         |                              |     _id : 689c04f3996b6e44edfafbcb
         |                              |     title : Credential Dumping Attempt
         |                              |     severity : CRITICAL
         |                              |     category : INTRUSION
         |                              |     status : OPEN
         |                              |     timestamp : 2024-08-11T16:28:44
         |                              |     source_ip : [null]
         |                              |     affected_systems
         |                              |       0 : DC-02
         |                              |     rule_name : LSASS Memory Dump Detection
         |                              |     risk_score : 96
```

2.3 Task requirements

1. Get **all alerts**.
2. Get **each alert individually** (use the **Get Alert by ID** API; do **not** just split the list response).
3. **Extract** the IPs from the alerts.
4. **Remove duplicates**.

5. **Check IP reputation** on VirusTotal.
6. **Filter malicious** IPs.
7. **Send** malicious IPs to **Gemini** to generate a **report**.

Solution overview



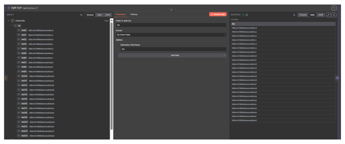
2.3.1 Extract all alert IDs from `/alerts`

- Upstream nodes up to `/alerts` already discussed.
- Used this **JavaScript** in a **Code** node to extract all `_id` values and emit them in a single item for later splitting:

```
// Extract all _id values from the alerts array
const alerts = $input.first().json.data.alerts;
const ids = alerts.map(alert => alert._id);

return [{ json: { ids: ids } }];
```

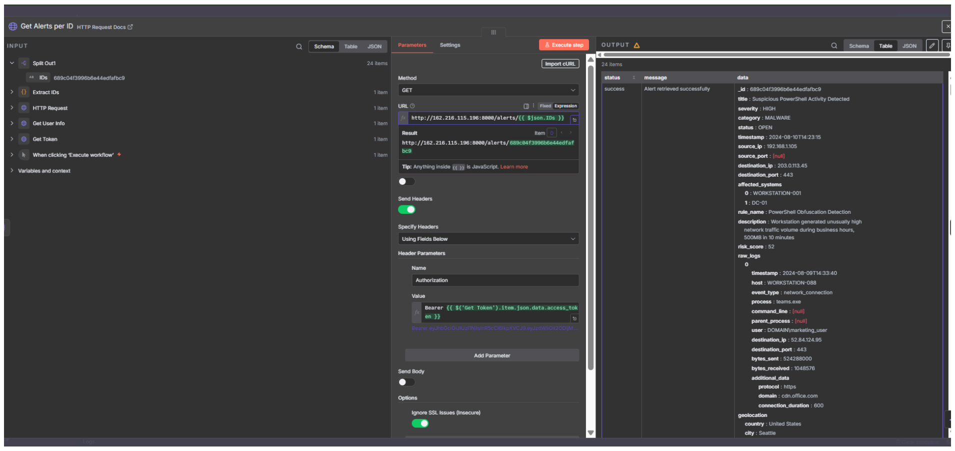
2.3.2 Split IDs into individual items

- **Split Out** produced **24** items (24 IDs).



2.3.3 Get alerts per ID

- Node **Get Alerts per ID** fetched all **24 alerts** individually.



```
Split Out1                                   24 items
  IDs 689c04f3996b6e44edfafbc9
Extract IDs                                  1 item
HTTP Request                                 1 item
Get User Info                                1 item
Get Token                                    1 item
When clicking 'Execute workflow'             1 item
Variables and context
```

Method: GET

URL (Fixed / Expression):

```
http://162.216.115.196:8000/alerts/{{ $json.IDs }}
```

Result (Item 0):

```
http://162.216.115.196:8000/alerts/689c04f3996b6e44edfafbc9
```

Tip: Anything inside {{ }} is JavaScript. Learn more

Send Headers: on

Specify Headers: Using Fields Below

Header Parameters

Name: Authorization

Value:

```
Bearer {{ $('Get Token').item.json.data.access_token }}
```

Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiI2ODljM…

Add Parameter

Send Body: off

Options

Ignore SSL Issues (Insecure): on

24 items

```
status   message                       data
success  Alert retrieved successfully  _id : 689c04f3996b6e44edfafbc9
                                       title : Suspicious PowerShell Activity Detected
                                       severity : HIGH
                                       category : MALWARE
                                       status : OPEN
                                       timestamp : 2024-08-10T14:23:15
                                       source_ip : 192.168.1.105
                                       source_port : [null]
                                       destination_ip : 203.0.113.45
                                       destination_port : 443
                                       affected_systems
                                         0 : WORKSTATION-001
                                         1 : DC-01
                                       rule_name : PowerShell Obfuscation Detection
                                       description : Workstation generated unusually high
                                         network traffic volume during business hours,
                                         500MB in 10 minutes
                                       risk_score : 52
                                       raw_logs
                                         0
                                           timestamp : 2024-08-09T14:33:40
                                           host : WORKSTATION-088
                                           event_type : network_connection
                                           process : teams.exe
                                           command_line : [null]
                                           parent_process : [null]
                                           user : DOMAIN\marketing_user
                                           destination_ip : 52.84.124.95
                                           destination_port : 443
                                           bytes_sent : 524288000
                                           bytes_received : 1048576
                                           additional_data
                                             protocol : https
                                             domain : cdn.office.com
                                             connection_duration : 600
                                       geolocation
                                         country : United States
                                         city : Seattle
```

2.3.4 Extract public, non-redundant, non-null, non-documentation/test IPv4s

- Used this **JavaScript** to:
  - Traverse all items.
  - Only capture from `source_ip` and `destination_ip` fields.
  - Validate IPv4 format and range.
  - Exclude private ranges (10.0.0.0/8, 172.16.0.0/12, & 192.168.0.0/16), loopback, link-local, multicast/reserved, and **RFC5737 documentation/test** blocks.
  - Remove duplicates.

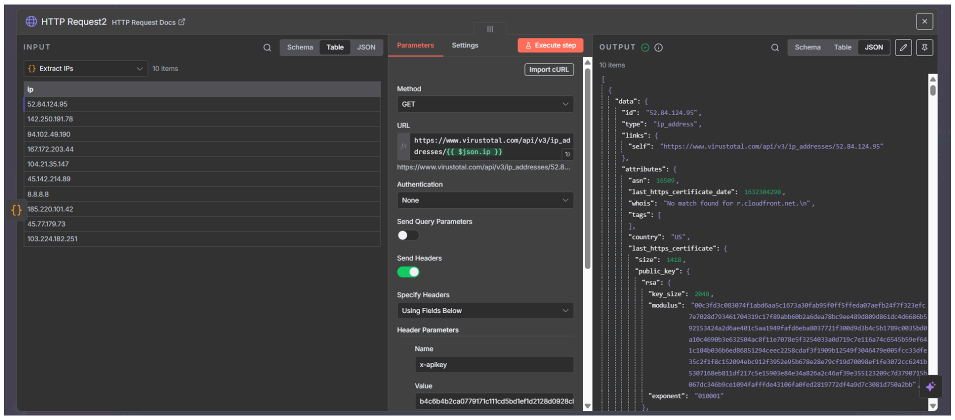
```
function isValidPublicIp(ip) {
  if (!ip || typeof ip !== 'string') return false;

  // quick IPv4 shape check
  if (!/^(?:\d{1,3}\.){3}\d{1,3}$/.test(ip)) return false;

  const parts = ip.split('.').map(Number);
  if (parts.length !== 4 || parts.some(n => n < 0 || n > 255)) return false;

  const [a, b, c] = parts;

  // RFC1918 private ranges
  if (a === 10) return false;                         // 10.0.0.0/8
  if (a === 172 && b >= 16 && b <= 31) return false;   // 172.16.0.0/12
  if (a === 192 && b === 168) return false;           // 192.168.0.0/16

  // Loopback, link-local, multicast/reserved
  if (a === 127) return false;                        // 127.0.0.0/8
  if (a === 169 && b === 254) return false;           // 169.254.0.0/16
  if (a >= 224) return false;                         // 224.0.0.0/4 and up

  // RFC5737 documentation/test CIDRs
  if (a === 192 && b === 0 && c === 2) return false;   // 192.0.2.0/24
  if (a === 198 && b === 51 && c === 100) return false;// 198.51.100.0/24
  if (a === 203 && b === 0 && c === 113) return false; // 203.0.113.0/24

  return true;
}

function collectPublicIpsFromObject(obj, set) {
  if (!obj || typeof obj !== 'object') return;

  if (Array.isArray(obj)) {
    for (const v of obj) collectPublicIpsFromObject(v, set);
    return;
  }

  for (const [key, value] of Object.entries(obj)) {
    // Only capture from source_ip / destination_ip fields
    if ((key === 'source_ip' || key === 'destination_ip') && typeof value === 'string') {
      if (isValidPublicIp(value)) set.add(value);
    }

    if (value && typeof value === 'object') collectPublicIpsFromObject(value, set);
  }
}

// n8n: read all input items, walk each, collect unique public IPs
const inputItems = $input.all();
const ips = new Set();

for (const item of inputItems) {
  const payload = (item.json && item.json.data) ? item.json.data : item.json;
  collectPublicIpsFromObject(payload, ips);
}

// Return in n8n items format
return Array.from(ips).map(ip => ({ json: { ip } }));
```

2.4.5 Check IP reputation on VirusTotal

- **HTTP Request** node with GET:

```
<https://www.virustotal.com/api/v3/ip_addresses/>{{ $json.ip }}
```

- Include **API Key** authentication **on every request**.



Extract IPs — 10 items

ip
52.84.124.95
142.250.191.78
94.102.49.190
167.172.203.44
104.21.35.147
45.142.214.89
8.8.8.8
185.220.101.42
45.77.179.73
103.224.182.251

Method: GET

URL: https://www.virustotal.com/api/v3/ip_addresses/{{ $json.ip }}

https://www.virustotal.com/api/v3/ip_addresses/52.8...

Authentication: None

Send Query Parameters: off

Send Headers: on

Specify Headers: Using Fields Below

Header Parameters

Name: x-apikey

Value: b4c6b4b2ca0779171c111cd5bd1ef1d2128d0928cl

10 items

```
[
  {
    "data": {
      "id": "52.84.124.95",
      "type": "ip_address",
      "links": {
        "self": "https://www.virustotal.com/api/v3/ip_addresses/52.84.124.95"
      },
      "attributes": {
        "asn": 16509,
        "last_https_certificate_date": 1632304298,
        "whois": "No match found for r.cloudfront.net.\n",
        "tags": [
        ],
        "country": "US",
        "last_https_certificate": {
          "size": 1418,
          "public_key": {
            "rsa": {
              "key_size": 2048,
              "modulus": "00c3fd3c083074f1abd6aa5c1673a30fab95f0ff5ffeda07aefb24f7f323efc
                         7e7028d793461704319c17f89abb60b2a6dea78bc9ee489d809d861dc4d6686b5
                         92153424a2d6ae401c5aa1949fafd6eba8037721f300d9d3b4c5b1789c0035bd0
                         a10c4690b3e632504ac8f11e7078e5f3254033a0d719c7e116a74c6545b59ef64
                         1c104b036b6ed86851294ceec2258cdaf3f1909b12549f3046479e005fcc33dfe
                         35c2f1f8c152094ebc912f3952e95b678e28e79cf19d70098ef1fe3072cc6241b
                         5307168eb811df217c5e15903e84e34a826a2c46af39e355123209c7d3790715b
                         067dc346b9ce1094fafffde43106fa0fed2819772df4a9d7c3081d750a2bb",
              "exponent": "010001"
            },
```

2.4.6 Keep only malicious IPs

- **Code** node "Get Malicious IP" filters to IPs with `last_analysis_stats.malicious > 0` and returns one item per malicious IP with a summary string:

```
// Get all n8n input items (VirusTotal API results)
const inputData = $input.all();

// Store results
const maliciousIps = [];

for (const item of inputData) {
  const data = item.json.data;

  if (data && data.attributes && data.attributes.last_analysis_stats) {
    const stats = data.attributes.last_analysis_stats;

    // Only keep IPs where malicious count > 0
    if (stats.malicious && stats.malicious > 0) {
      maliciousIps.push({
        json: {
          ip: data.id,
          category: `malicious: ${stats.malicious}, harmless: ${stats.harmless || 0}, undetected: ${stats.undetected ||
0}`
        }
      });
    }
  }
}

// Return one item per malicious IP
return maliciousIps;
```

2.4.7 Convert JSON items to plain text (for AI input)

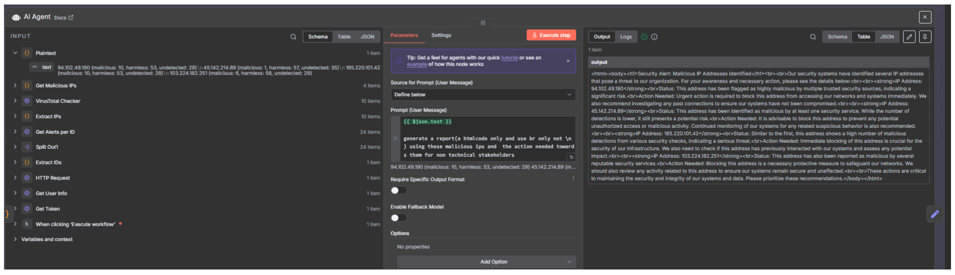
- **Code** node "Plaintext" converts items to newline-separated text like `IP (category)` to make it readable for **Gemini**

```
// Get all input items from previous node
const items = $input.all();

// Convert to plain text paragraph
const paragraph = items
  .map(item => {
    const ip = item.json.ip || '';
    const category = item.json.category || '';
    return `${ip} (${category})`;
  })
  .join('\n'); // Or use ' ' for a single-line paragraph

// Return as single string in a JSON object
return [
  {
    json: {
      text: paragraph
```

```
        }
    }
];
```

2.4.8 AI Agent (Google Gemini) to generate a report

- **AI Agent Node** configured to **Google Gemini Chat Model**.



text: 94.102.49.190 (malicious: 10, harmless: 53, undetected: 29)\n 45.142.214.89 (malicious: 1, harmless: 57, undetected: 35)\n 185.220.101.42 (malicious: 10, harmless: 53, undetected: 28)\n 103.224.182.251 (malicious: 6, harmless: 58, undetected: 29)

Source for Prompt (User Message): Define below

Prompt (User Message):

```
{{ $json.text }}

generate a report(a htmlcode only and use br only not \n
) using these malicious ips and  the action needed toward
s them for non technical stakeholders
```

94.102.49.190 (malicious: 10, harmless: 53, undetected: 29) 45.142.214.89 (m...

Require Specific Output Format

Enable Fallback Model

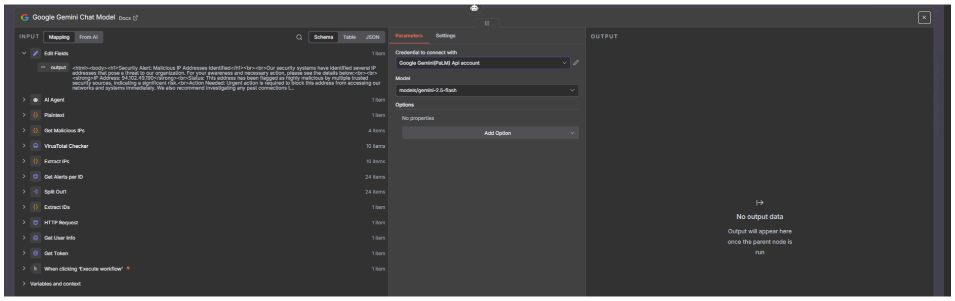
Options

No properties

Add Option

output

```
<html><body><h1>Security Alert: Malicious IP Addresses Identified</h1><br><br>Our security systems have identified several IP addresses that pose a threat to our organization. For your awareness and necessary action, please see the details below:<br><br><strong>IP Address: 94.102.49.190</strong><br>Status: This address has been flagged as highly malicious by multiple trusted security sources, indicating a significant risk.<br>Action Needed: Urgent action is required to block this address from accessing our networks and systems immediately. We also recommend investigating any past connections to ensure our systems have not been compromised.<br><br><strong>IP Address: 45.142.214.89</strong><br>Status: This address has been identified as malicious by at least one security service. While the number of detections is lower, it still presents a potential risk.<br>Action Needed: It is advisable to block this address to prevent any potential unauthorized access or malicious activity. Continued monitoring of our systems for any related suspicious behavior is also recommended.<br><br><strong>IP Address: 185.220.101.42</strong><br>Status: Similar to the first, this address shows a high number of malicious detections from various security checks, indicating a serious threat.<br>Action Needed: Immediate blocking of this address is crucial for the security of our infrastructure. We also need to check if this address has previously interacted with our systems and assess any potential impact.<br><br><strong>IP Address: 103.224.182.251</strong><br>Status: This address has also been reported as malicious by several reputable security services.<br>Action Needed: Blocking this address is a necessary protective measure to safeguard our networks. We should also review any activity related to this address to ensure our systems remain secure and unaffected.<br><br>These actions are critical to maintaining the security and integrity of our systems and data. Please prioritize these recommendations.</body></html>
```



output: <html><body><h1>Security Alert: Malicious IP Addresses Identified</h1>

Our security systems have identified several IP addresses that pose a threat to our organization. For your awareness and necessary action, please see the details below:

<strong>IP Address: 94.102.49.190</strong>
Status: This address has been flagged as highly malicious by multiple trusted security sources, indicating a significant risk.
Action Needed: Urgent action is required to block this address from accessing our networks and systems immediately. We also recommend investigating any past connections t...

Credential to connect with: Google Gemini(PaLM) Api account

Model: models/gemini-2.5-flash

Options

No properties

Add Option

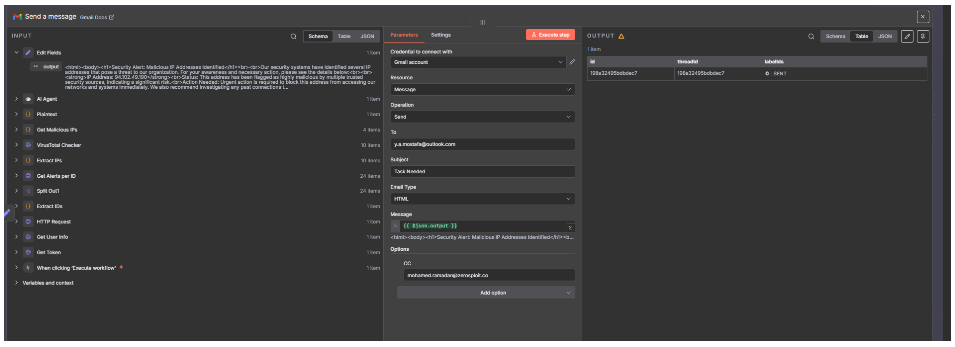
- Requires a Google **Developer API Key** from https://aistudio.google.com/apikey?_gl=1*jujqno*_up*MQ..&gclid=CjwKCAjw7_DEBhAeEiwAWKiCC-wdEY7LYDlp3uTTdjR9ORCe3PvAlfQHMKz31eOBgwwpKkzPQJiwtxoCWsoQAvD_BwE&gclsrc=aw.ds&gbraid=0AAAAACn
- Prompt: `{{ $json.text }} generate a report(a htmlcode only and use br only not \n ) using these malicious ips and  the action needed towards them for non technical stakeholders`

2.4.9 Clean the AI output into valid HTML

- Add an **Edit Fields** node that transforms the output by removing code extras (AI Error) and the leading ` ``` ` & `html\n` : `{{ $json.output.replaceAll("```", "").replace("html\n","") }}`

2.4.10 Email the report via Gmail

- Use a **GMAIL** node to send the generated HTML report.



output: <html><body><h1>Security Alert: Malicious IP Addresses Identified</h1>

Our security systems have identified several IP addresses that pose a threat to our organization. For your awareness and necessary action, please see the details below:

<strong>IP Address: 94.102.49.190</strong>
Status: This address has been flagged as highly malicious by multiple trusted security sources, indicating a significant risk.
Action Needed: Urgent action is required to block this address from accessing our networks and systems immediately. We also recommend investigating any past connections t...

Credential to connect with: Gmail account

Resource: Message

Operation: Send

To: y.a.mostafa@outlook.com

Subject: Task Needed

Email Type: HTML

Message: `{{ $json.output }}`

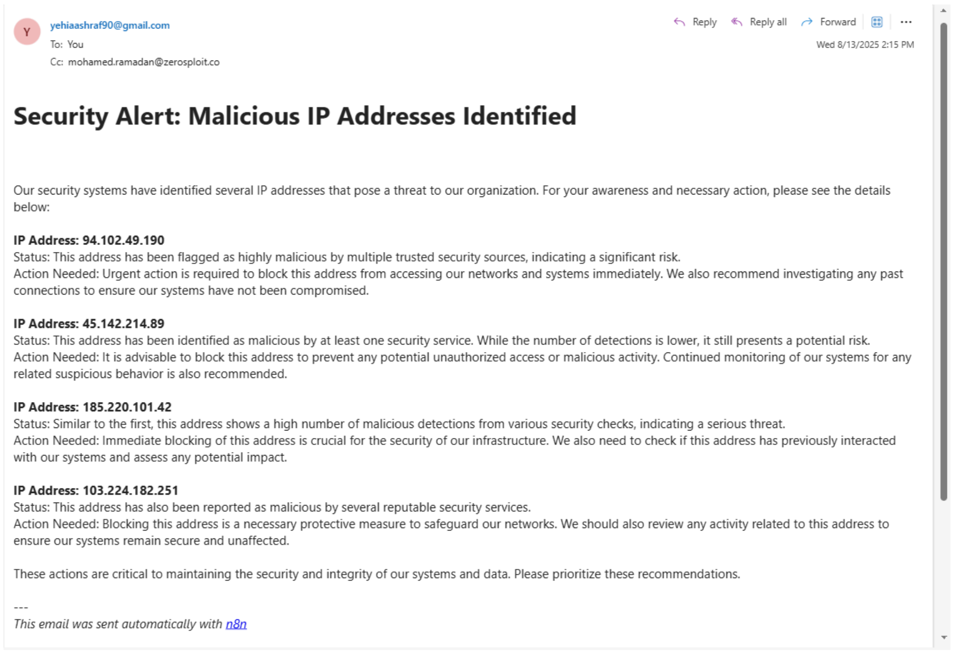
<html><body><h1>Security Alert: Malicious IP Addresses Identified</h1><b...

Options

CC: mohamed.ramadan@zerosploit.co

Add option

id	threadId	labelIds
198a32495bdbdec7	198a32495bdbdec7	0 : SENT


yehiaashraf90@gmail.com
To: You
Cc: mohamed.ramadan@zerosploit.co

Wed 8/13/2025 2:15 PM

Security Alert: Malicious IP Addresses Identified

Our security systems have identified several IP addresses that pose a threat to our organization. For your awareness and necessary action, please see the details below:

IP Address: 94.102.49.190
Status: This address has been flagged as highly malicious by multiple trusted security sources, indicating a significant risk.
Action Needed: Urgent action is required to block this address from accessing our networks and systems immediately. We also recommend investigating any past connections to ensure our systems have not been compromised.

IP Address: 45.142.214.89
Status: This address has been identified as malicious by at least one security service. While the number of detections is lower, it still presents a potential risk.
Action Needed: It is advisable to block this address to prevent any potential unauthorized access or malicious activity. Continued monitoring of our systems for any related suspicious behavior is also recommended.

IP Address: 185.220.101.42
Status: Similar to the first, this address shows a high number of malicious detections from various security checks, indicating a serious threat.
Action Needed: Immediate blocking of this address is crucial for the security of our infrastructure. We also need to check if this address has previously interacted with our systems and assess any potential impact.

IP Address: 103.224.182.251
Status: This address has also been reported as malicious by several reputable security services.
Action Needed: Blocking this address is a necessary protective measure to safeguard our networks. We should also review any activity related to this address to ensure our systems remain secure and unaffected.

These actions are critical to maintaining the security and integrity of our systems and data. Please prioritize these recommendations.

This email was sent automatically with n8n

success

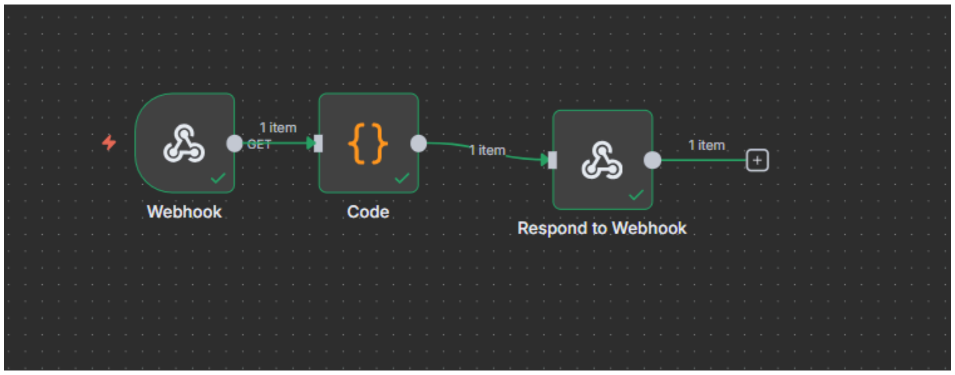
---

3) Third Workflow — Simulate an API via Webhook in n8n

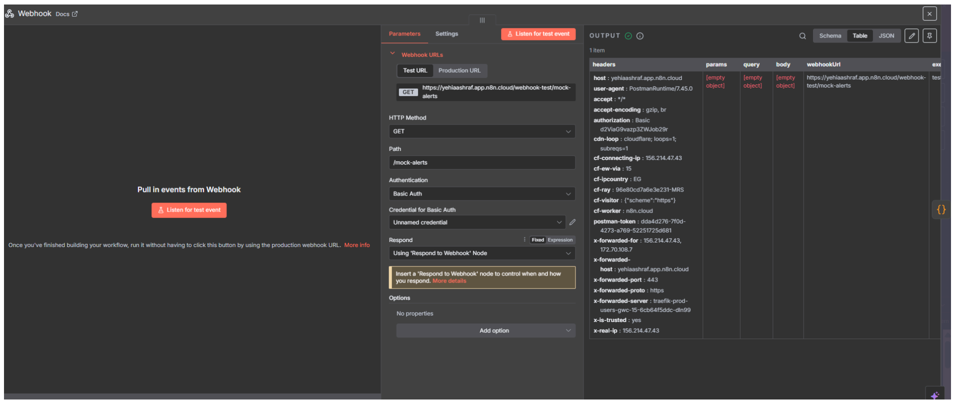
Requirements

1. API uses authentication (**Basic user/password** or **token header**).
2. API replies with **mock data** of all alerts in the system.

3.1 Workflow structure



- **Webhook** node: exposes an endpoint (configure security as needed: **Basic Auth** user/pass or header-based token).



Once you've finished building your workflow, run it without having to click this button by using the production webhook URL. More info

GET https://yehiaashraf.app.n8n.cloud/webhook-test/mock-alerts

HTTP Method: GET

Path: /mock-alerts

Authentication: Basic Auth

Credential for Basic Auth: Unnamed credential

Respond (Fixed | Expression): Using 'Respond to Webhook' Node

Insert a 'Respond to Webhook' node to control when and how you respond. More details

Options: No properties

Add option

1 item

headers	params	query	body	webhookUrl	exe
host : yehiaashraf.app.n8n.cloud user-agent : PostmanRuntime/7.45.0 accept : */* accept-encoding : gzip, br authorization : Basic d2ViaG9vazp3ZWJob29r cdn-loop : cloudflare; loops=1; subreqs=1 cf-connecting-ip : 156.214.47.43 cf-ew-via : 15 cf-ipcountry : EG cf-ray : 96e80cd7a6e3e231-MRS cf-visitor : {"scheme":"https"} cf-worker : n8n.cloud postman-token : dda4d276-7f0d-4273-a769-52251725d681 x-forwarded-for : 156.214.47.43, 172.70.108.7 x-forwarded-host : yehiaashraf.app.n8n.cloud x-forwarded-port : 443 x-forwarded-proto : https x-forwarded-server : traefik-prod-users-gwc-15-6cb64f5ddc-dln99 x-is-trusted : yes x-real-ip : 156.214.47.43	[empty object]	[empty object]	[empty object]	https://yehiaashraf.app.n8n.cloud/webhook-test/mock-alerts	tes

- **Code** node: acts as the handler that returns mock **alerts** data (as if it were a real GET to a backend). Exact code used:

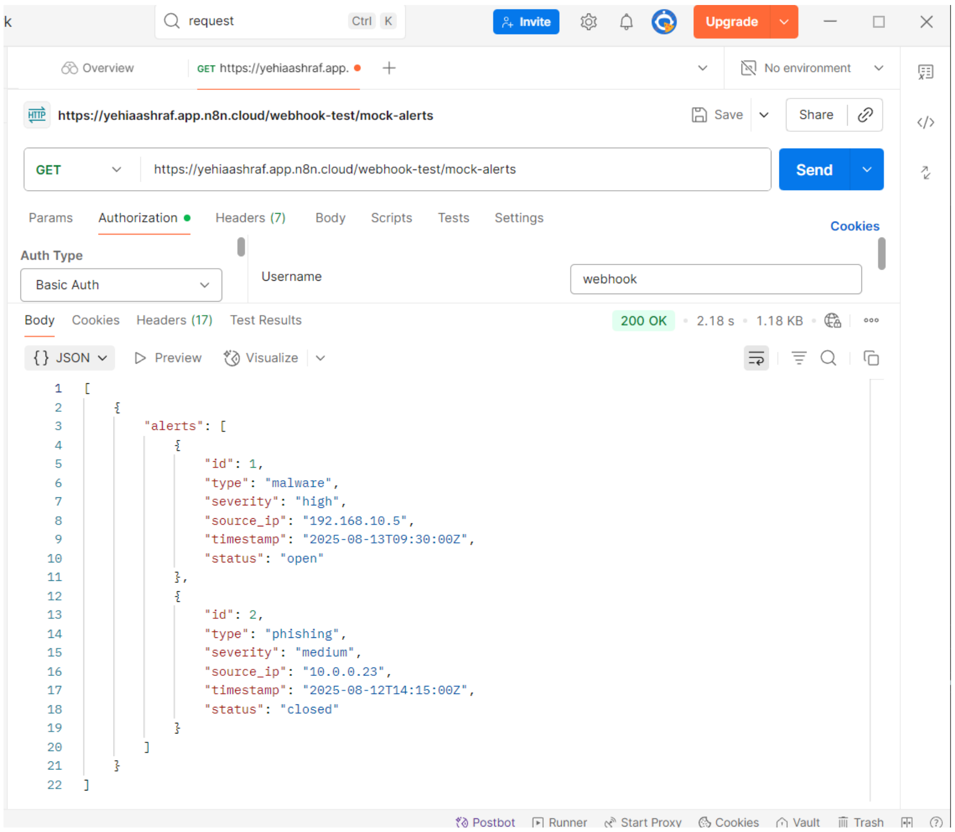
```
return [
  {
    json: {
      alerts: [
        {
          id: 1,
          type: "malware",
          severity: "high",
          source_ip: "192.168.10.5",
          timestamp: "2025-08-13T09:30:00Z",
          status: "open"
        },
        {
          id: 2,
          type: "phishing",
          severity: "medium",
          source_ip: "10.0.0.23",
          timestamp: "2025-08-12T14:15:00Z",
          status: "closed"
        }
      ]
    }
  }
];
```

- **Respond to Webhook** node: ensures the **Webhook** does **not** respond until this node executes, so the response contains the output from **Code**.
  - **Important:** configure the **Webhook** node **not to respond immediately**, but to **respond using the "Respond to Webhook" node**.

3.2 Testing with Postman

- Open the workflow URL in Postman:
  - `Ctrl + T` → **New Request** → paste the **webhook URL**.
  - In **Authorization**, set **Basic Auth** with:
    - **Username:** `webhook`

        - **Password:** `webhook`
    - Send **GET**.



https://yehiaashraf.app.n8n.cloud/webhook-test/mock-alerts

GET https://yehiaashraf.app.n8n.cloud/webhook-test/mock-alerts

Auth Type: Basic Auth

Username: webhook

```
[
    {
        "alerts": [
            {
                "id": 1,
                "type": "malware",
                "severity": "high",
                "source_ip": "192.168.10.5",
                "timestamp": "2025-08-13T09:30:00Z",
                "status": "open"
            },
            {
                "id": 2,
                "type": "phishing",
                "severity": "medium",
                "source_ip": "10.0.0.23",
                "timestamp": "2025-08-12T14:15:00Z",
                "status": "closed"
            }
        ]
    }
]
```

success

Key Highlights

- VirusTotal calls: always include your **API key** with **each HTTP request**.
- The SIEM docs example showed that you may **still need authorization** on endpoints even when the page doesn't explicitly call it out.