

Client Email Protocol

- [WebMarketers Client Email Protocol](#)
- [Guide: Setting Up Amazon SES Usage Spike Alarms \(250% Increase\)](#)

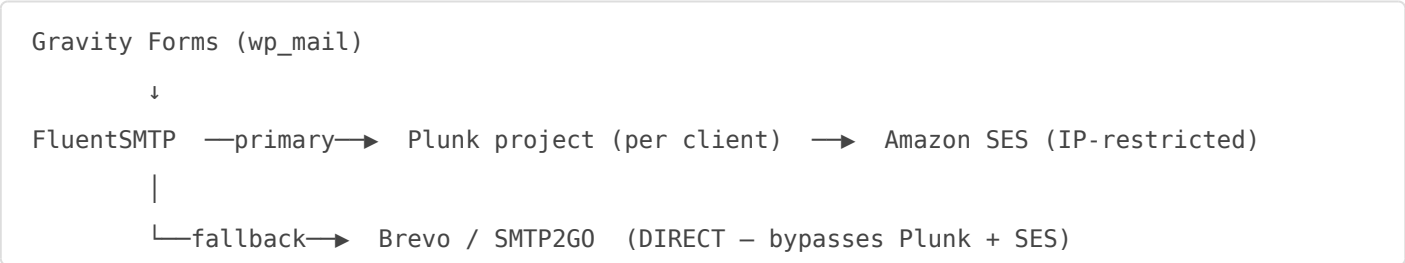
WebMarketers Client Email Protocol

Purpose: Isolate client sites from Amazon SES so one compromised site can't tank the whole account's reputation — and so we can swap providers without touching every site.

The Problem

Client mailboxes/sites get compromised → spam goes out through SES → SES flags the account for bad reputation → we're forced to reconfigure email for *all* 150+ sites at once. This protocol contains the blast radius and gives us fast rotation + fallback.

The Stack



Layer	Role
FluentSMTP	WordPress plugin. Intercepts all <code>wp_mail()</code> , handles routing, fallback, and per-site logging. Free. Replaces PostSMTP.
Plunk	Central relay over SES. One project per client (multiple domains per project). Gives per-client isolation, credential rotation, and a kill switch. Exposes standard SMTP (465/587).
Amazon SES	Actual delivery. Credentials IP-restricted to our servers via IAM <code>aws:SourceIp</code> .
Brevo / SMTP2GO	Fallback provider. Free tier, sits idle until needed. Connected directly , not through Plunk.

Key Decisions

- **Fallback goes direct, not through Plunk.** If Plunk or SES is the failure point, the site still sends. Routing fallback through Plunk would take out both paths at once.
- **Form responses stay native.** Gravity Forms notifications are `wp_mail()`, so they already flow through Plunk over SMTP automatically. No separate API integration needed.
- **Transactional ≠ marketing.** Form receipts (admin notification + visitor auto-response) are transactional. Only promote a submitter into a Plunk contact/automation on sites that actually run nurture campaigns — as a separate, consented marketing path.

Detection & Alerting

Two distinct signals — we need **both**:

- **Failure notification** (site-detected) → catches *outages* (SES down/banned). Triggers the fallback. ☐ already planned.
- **Volume/anomaly alerting** → catches the real reputation-killer: a hacked site sending spam *successfully* (returns 250 OK, looks normal). Watch Plunk per-project send spikes and/or SES CloudWatch alarms on **send rate + complaint rate**. ⚠ must add.

💡 Reminder: IP whitelisting stops *leaked-credential* abuse from outside. It does **not** stop a hacked site sending through its own legit relay — that's what the anomaly alerting is for. Prevention still lives at the source (patching, WAF, form CAPTCHA/rate-limiting).

Per-Client Setup Checklist

1. Create a Plunk **project** for the client; add their domain(s) with DKIM/SPF.
2. Point the project's upstream at the shared **IP-restricted SES** credentials.
3. On each client site, install **FluentSMTP**:
 - **Primary connection** → the client's Plunk SMTP credentials.
 - **Fallback connection** → Brevo or SMTP2GO (direct).
 - Enable **email logging**.
4. Confirm Gravity Forms notifications send and appear in the FluentSMTP log.
5. Add the site/project to **anomaly alerting** (send-rate + complaint-rate).

Blast-Radius (optional, recommended)

Split clients across **separate SES sub-accounts by risk tier** and/or **configuration sets per client**. One compromised site then degrades a segment, not all 150.

Guide: Setting Up Amazon SES Usage Spike Alarms (250% Increase)

This guide provides step-by-step instructions for configuring an Amazon CloudWatch alarm to notify you when your Amazon SES send usage increases by more than 250% in 1 day.

You can choose to set this up **manually via the AWS Console** or **programmatically via AWS CloudShell (CLI)**.

Part 1: Manual Setup via AWS Console

Step 1: Create an SNS Topic (Notification Channel)

1. Navigate to the **Amazon SNS Console**.
2. Click **Topics** on the left menu, then **Create topic**.
3. Choose **Standard** as the type.
4. Name the topic (e.g., `SES-Usage-Alerts`) and click **Create topic**.

- **How to check:** Once created, you should see the Topic details page with a unique Topic ARN displayed at the top.

Step 2: Subscribe Your Email to the Topic

1. On the topic details page you just created, click the **Create subscription** button.
2. Under **Protocol**, select **Email**.
3. Under **Endpoint**, enter the email address where you want to receive alerts.

4. Click **Create subscription**.

- **How to check:**

- The subscription status will initially show as `Pending confirmation`.
- Check your email inbox for a message from "AWS Notifications". Click the **Confirm subscription** link in the email.
- Return to the SNS console, refresh the page, and the status should now say `Confirmed`.

Step 3: Create the CloudWatch Alarm with Metric Math

1. Navigate to the **Amazon CloudWatch Console**.

2. Go to **Alarms > All alarms > Create alarm**.

3. Click **Select metric**.

4. Search for `SES Send`, go to **SES > Account Metrics**, check the box next to the **Send** metric, and click the **Graphed metrics** tab.

5. Change **Statistic** to **Sum** and **Period** to **1 hour**.

6. Click **Add math > Start with empty expression** to add a new row (`e1`). Set the expression to `DIFF(m1)`.

- *Tip: Uncheck the box under the "Select" column for `m1` and `e1` so they don't trigger the alarm.*

7. Click **Add math > Start with empty expression** again to add row (`e2`). Set the expression to `IF(m1 - e1 > 0, (e1 / (m1 - e1)) * 100, 0)`. Keep this row checked.

8. Click the **Select metric** button at the bottom.

9. Under **Conditions**, set:

- **Threshold type:** Static
- **Whenever e2 is:** Greater than
- **than:** 250

10. Expand **Additional configuration** and set **Treat missing data as:** to `good (not breaching)`. Click **Next**.

11. Under **Notification**, choose **In alarm**, select **Select an existing SNS topic**, and choose `SES-Usage-Alerts`. Click **Next**.

12. Name the alarm (e.g., `SES-Spike-250Percent-Increase`) and click **Create alarm**.

- **How to check:**

- Go to **Alarms > All alarms**.
- Locate your new alarm. The state should initially say `Insufficient data` and then transition to `OK` within a few minutes if data is flowing correctly.

Part 2: Automated Setup via AWS CloudShell (CLI)

If you prefer using the terminal, you can do this entirely inside your browser using **AWS CloudShell**.

Step 1: Open AWS CloudShell

1. In the AWS Management Console, click the **CloudShell icon** () located in the top navigation bar next to the search bar.
2. Wait a few seconds for the terminal to initialize.

- **How to check:** You should see a command prompt ending with `[cloudshell-user@ip-... ~]$`.

Step 2: Create SNS Topic and Subscribe

Paste the following commands into CloudShell (replace `your-email@example.com` with your actual email):

```
# Create the SNS Topic
TOPIC_ARN=$(aws sns create-topic --name SES-Usage-Alerts --query "TopicArn" --output text)
echo "Topic ARN: $TOPIC_ARN"

# Subscribe your email
aws sns subscribe \
  --topic-arn "$TOPIC_ARN" \
  --protocol email \
  --notification-endpoint "your-email@example.com"
```

- **How to check:**
 - Run `aws sns list-subscriptions-by-topic --topic-arn "$TOPIC_ARN"`. The output should list your email with a `SubscriptionArn` of `PendingConfirmation`.
 - Check your email inbox, click the confirmation link, and run the exact same command again. The output should now show an active ARN instead of `PendingConfirmation`.

Step 3: Create the CloudWatch Alarm

Paste the following command into CloudShell to create the metric math alarm:

```
aws cloudwatch put-metric-alarm \  
  --alarm-name "SES-Spike-250Percent-Increase" \  
  --alarm-description "Triggers when daily SES send volume grows by more than 250%" \  
  --alarm-actions "$TOPIC_ARN" \  
  --evaluation-periods 1 \  
  --comparison-operator GreaterThanThreshold \  
  --threshold 250 \  
  --treat-missing-data notBreaching \  
  --metrics '[  
    {  
      "Id": "m1",  
      "MetricStat": {  
        "Metric": {  
          "Namespace": "AWS/SES",  
          "MetricName": "Send"  
        },  
        "Period": 86400,  
        "Stat": "Sum"  
      },  
      "ReturnData": false  
    },  
    {  
      "Id": "e1",  
      "Expression": "DIFF(m1)",  
      "ReturnData": false  
    },  
    {  
      "Id": "e2",  
      "Expression": "IF(m1 - e1 > 0, (e1 / (m1 - e1)) * 100, 0)",  
      "Label": "Percentage Increase",  
      "ReturnData": true  
    }  
  ]'
```

- **How to check:**

- Run `aws cloudwatch describe-alarms --alarm-names "SES-Spike-250Percent-Increase"`.

- The output will return a JSON object confirming the alarm's exact configuration. Look for the `"StateValue"` line—it should be `"OK"` (or `"INSUFFICIENT_DATA"` until new data arrives).