Vulnerability Report – No Rate Limiting on Password Reset Endpoint

Summary:

The application does not implement proper rate limiting on the password reset functionality, allowing an attacker to send an unlimited number of requests to the endpoint without any restriction. This behaviour can lead to abuse of backend resources, service degradation, and financial implications due to excessive email delivery usage.

Affected Endpoint: https://admin.alwaysdata.com/password/lost/

Technical Description:

The password reset feature lacks a mechanism to detect or limit repeated requests made to the endpoint. During testing, I was able to automate password reset requests for a valid email address without triggering any form of throttling or CAPTCHA challenges.

This kind of misconfiguration introduces multiple risks:

- **Mass Email Bombing**: An attacker can flood the password reset endpoint with thousands of requests, causing the application to send out a massive number of reset emails.
- **Financial Damage**: Since transactional email services (e.g., SendGrid, Mailgun, SES) are typically billed per email, this could inflate operational costs, especially if abused at scale.
- **Service Availability Impact**: This email flooding could result in throttling or delay from the email service provider, potentially affecting legitimate user password reset or notification deliveries.
- **User Experience Degradation**: Users could experience delays or failures when trying to reset their passwords due to backend overload.
- **Brute-force or Enumeration Attack Vector**: While not exploited in this instance, the absence of rate limiting opens up potential for attackers to perform user enumeration or brute-force attacks by cycling through large sets of emails and observing the application's behaviour.

Real-World Impact:

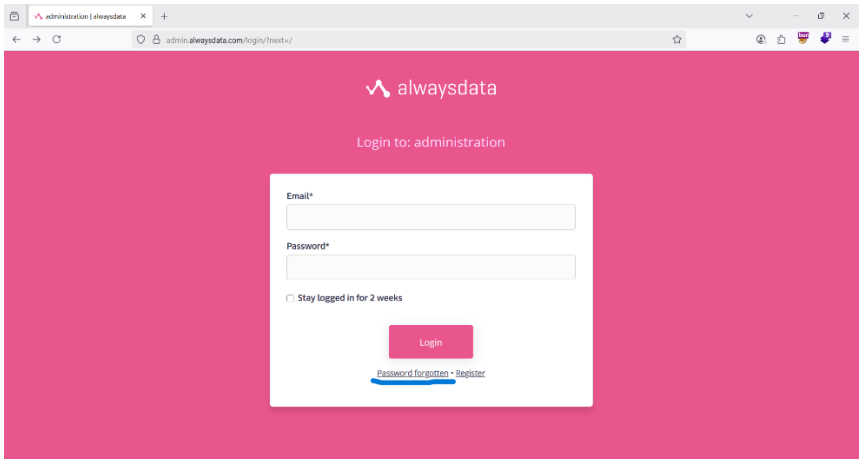
This type of vulnerability has been acknowledged in real-world reports with similar low-severity technical flaws that led to **high business impact**:

“By choking their email server to send such an enormous amount of invite emails, the availability of this functionality was impacted. Legitimate users would experience delay... the company cared enough to see this as an issue that could cost their company.”

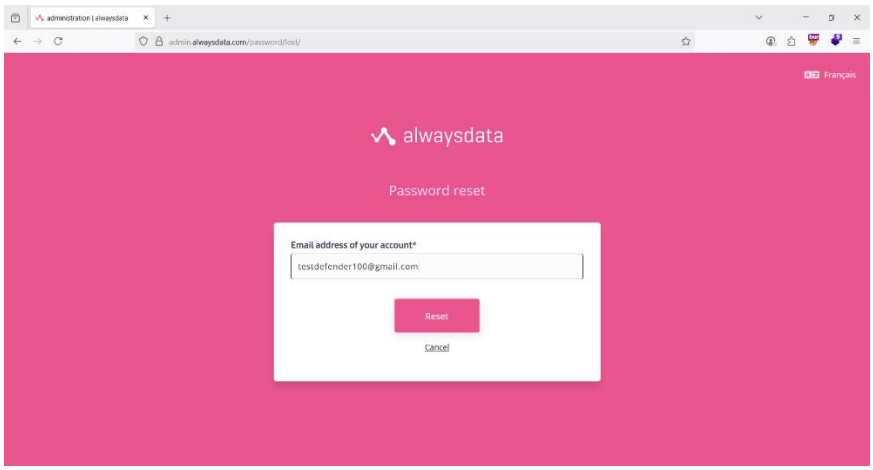
In the current scenario, the **password reset flow** is even more sensitive, as it deals with **account recovery** and **identity assurance**. Abusing this functionality not only affects infrastructure and cost but could also be perceived as a **targeted harassment** against users (by sending repeated reset emails).

Steps to Reproduce:

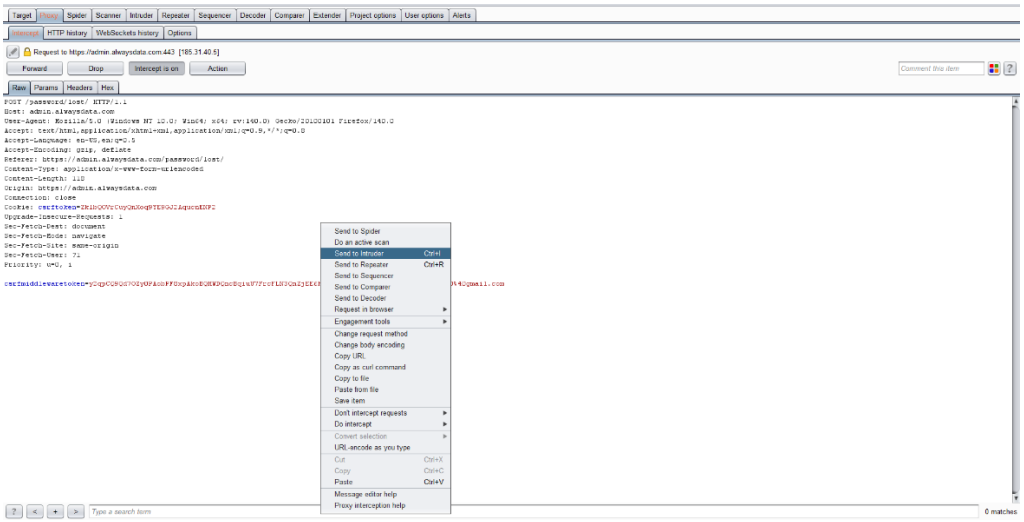
1. Navigate to: https://admin.alwaysdata.com/login/?next=/



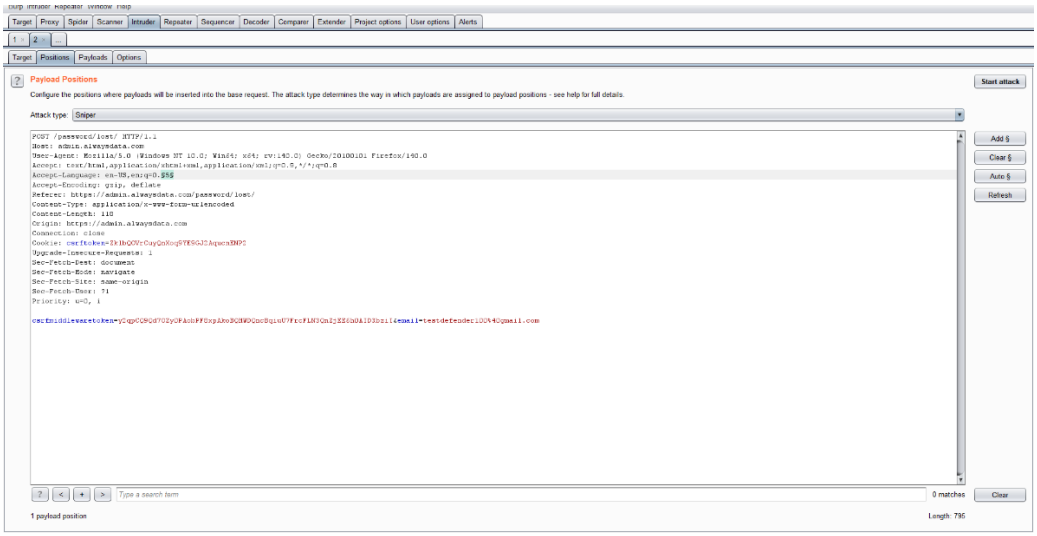
2. Click on the **"Password forgotten"** link.
3. Enter a **registered email address** that you control.



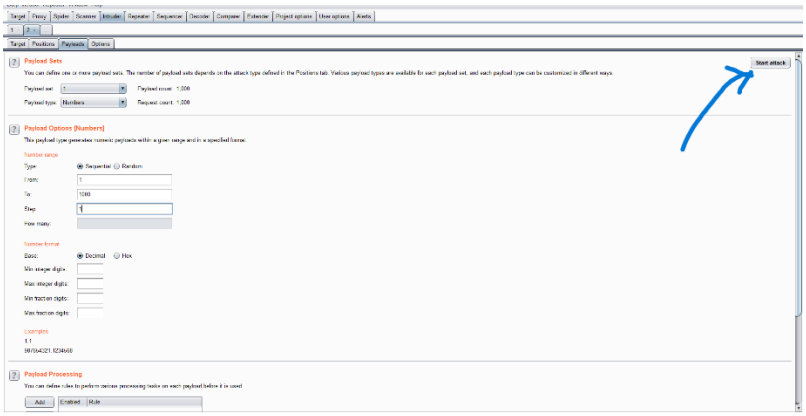
4. Turn on **Burp Suite** with **intercept enabled**.
5. Click the **"Reset"** button — this will capture the password reset request.
6. In Burp, **right-click** the captured request and **send it to Intruder**. Then, turn off intercept.



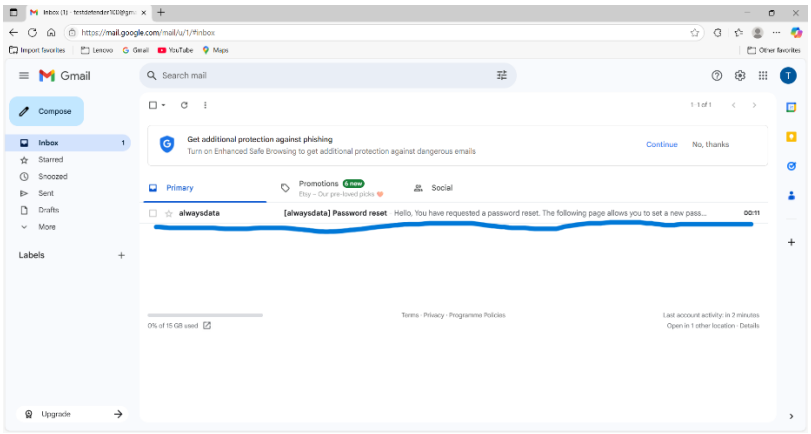
7. Go to the **Intruder** tab → **Positions** section, and leave the default insertion point or choose any one parameter for fuzzing (e.g., the email field).



8. Go to the **Payloads** tab:
    - Set **Payload type** to Numbers.
    - Set **Payload Options** → From: 1, To: <your desired number> (e.g., 1000)



9. Now, **check your inbox** — at this point, you'll see only **one password reset email**, which was triggered when the original intercepted request was submitted manually.



10. Then, click "Start Attack" in Burp
11. Once the attack starts, open the inbox of the email address you used in Step 3.
12. Once the Intruder attack begins, observe your inbox — it will be **flooded with hundreds/thousands of password reset emails**, confirming the vulnerability and demonstrating how an attacker could abuse it to cause email spam or DoS-like effects.

Recommendation:

- Implement **rate limiting** (e.g., 5 requests per IP/user per hour) on all sensitive endpoints such as:
    - Password reset
    - Login
    - OTP/verification codes
- Introduce **CAPTCHA** or **device fingerprinting** for abuse detection.
- Monitor email queue thresholds and alert on anomalies.
- Log and alert on excessive email sends to a single address or IP.

Impact Summary:

Category	**Description**
CWE	CWE-770: Allocation of Resources Without Limits or Throttling
Impact	Financial loss, degraded availability, spam/harassment vector
User Interaction	None required
Attack Vector	Network
Scope	Unchanged
Security Risk	Medium (with potential for High business impact)
Business Risk	High — due to potential cost abuse and service reputation impact

Attachments: Screenshots of full process and PoC

Conclusion:

While this issue may initially seem minor due to the absence of direct account takeover, the cumulative impact on business operations, user trust, and infrastructure cost makes it a highly exploitable and impactful vulnerability. I recommend fixing this swiftly to prevent abuse at scale.

Proof of Concept (PoC):

To demonstrate the impact of this vulnerability, I performed a controlled test using Burp Suite Community Edition, which has a known limitation in sending concurrent or high-speed requests.

- I initiated the password reset process using Burp Intruder with 1000 payloads.
- Due to the slower request rate in the Community Edition, it took approximately 30 minutes to send the first 500 requests.
- As a result, my inbox was flooded with 500 password reset emails within just 30 minutes.

Attached Screenshot:
Shows over 500 password reset emails received within a short window.

Important Note:
If an attacker uses Burp Suite Professional or a tool like ffuf, turbo intruder, or custom Python scripts, the request rate could increase significantly — allowing hundreds or even thousands of reset emails to be sent within seconds. This could easily lead to:

- Exhaustion of email delivery quota
- Delayed or dropped emails for legitimate users
- Reputation damage due to spam-like behaviour
- Financial loss due to increased email costs (especially if using paid services like SendGrid, SES, Mailgun, etc.)