# Account Enumeration Vulnerability: `rimi.ee`

Gregor Eesmaa
gregor.eesmaa@ut.ee
University of Tartu

June 8, 2025

## 1 Introduction

Account enumeration is a security vulnerability enabling attackers to determine if specific user accounts exist on a service. The vulnerability usually lies in the account registration functionality of a service, where an error message is returned, indicating that a user with the specified account identifier is already registered. However, an online service can also leak this information in other, more subtle ways, which are often overlooked by software developers. For example, even without a direct message, small visual differences in responses, or slight variations in how the server behaves (like the exact data returned) for existing versus non-existing accounts, can still reveal if an account is registered.

On 2025-05-31, we reassessed `rimi.ee` and found that despite significant changes, **the service is still vulnerable to account enumeration**. Since the initial findings have been already addressed, this report will now highlight the more subtle residual hints of account existence that were identified.

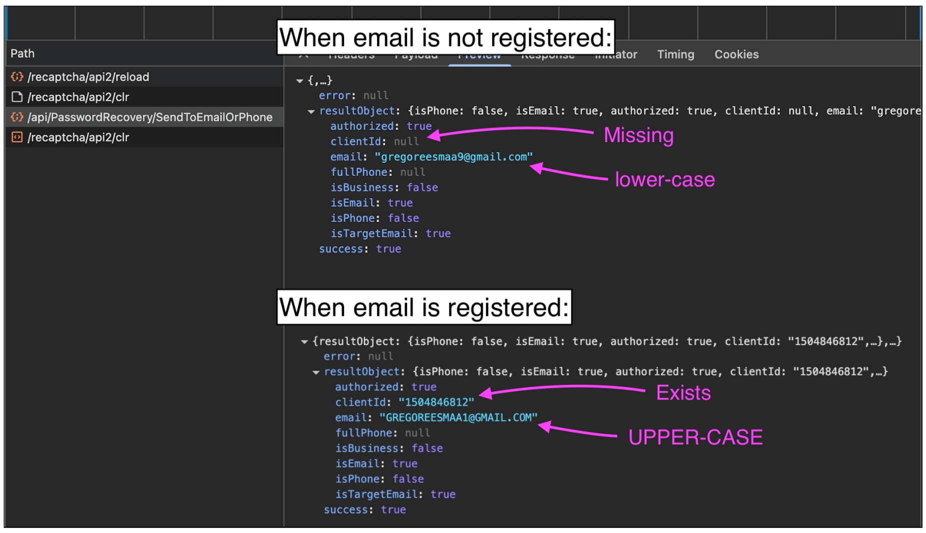
If the account identifier of an online service is personal data (e.g. email address, personal code etc), then the fact, whether it is associated to an account, is also considered personal data. Any disclosure of personal data to third parties without a legal basis constitutes a data breach [1].

We advise you to investigate the potential data breach, and notify the supervisory authority and the affected data subjects, if necessary. After **2025-07-08**, we will reassess the service and notify the Estonian Data Protection Inspectorate in case the vulnerability has not been mitigated. Detailed guidelines for mitigating this type of flaw are available in [2].

## 2 Vulnerabilities Found

We tested the login form, password reset form, account registration form and email change form of `rimi.ee`. No issues appeared on the login form and email change form. However, we identified security issues on the password reset form and account registration form. Additionally, we noticed a likely timing side-channel. The vulnerabilities found are described in more detail in subsections below.

## 2.1 Password Reset Form



When email is not registered:

```
▾ {,…}
    error: null
  ▾ resultObject: {isPhone: false, isEmail: true, authorized: true, clientId: null, email: "gregore
      authorized: true
      clientId: null
      email: "gregoreesmaa9@gmail.com"
      fullPhone: null
      isBusiness: false
      isEmail: true
      isPhone: false
      isTargetEmail: true
    success: true
```

When email is registered:

```
▾ {resultObject: {isPhone: false, isEmail: true, authorized: true, clientId: "1504846812",…},…}
    error: null
  ▾ resultObject: {isPhone: false, isEmail: true, authorized: true, clientId: "1504846812",…}
      authorized: true
      clientId: "1504846812"
      email: "GREGOREESMAA1@GMAIL.COM"
      fullPhone: null
      isBusiness: false
      isEmail: true
      isPhone: false
      isTargetEmail: true
    success: true
```

Figure 1: The vulnerability in the password reset form

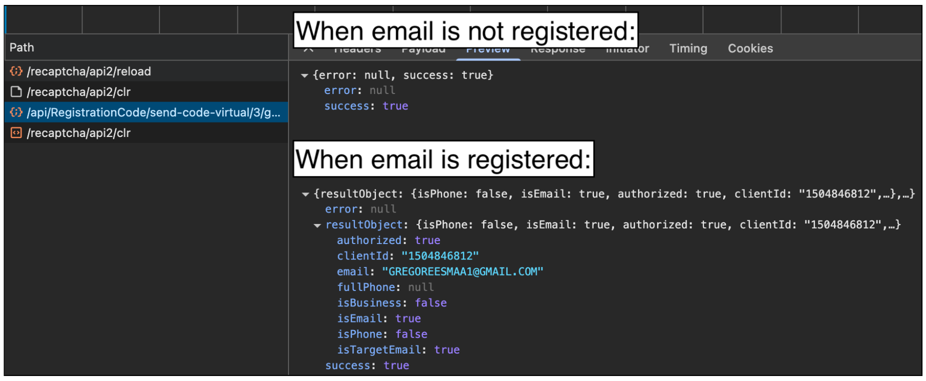
The password reset form is susceptible to account enumeration attacks. This is because when a password reset is requested for an email address that is not registered with the service, there are subtle differences in the response received from the server, compared to when the email is registered (see Figure 1).

It is also crucial to eliminate any side-channels that an attacker could exploit to differentiate between account existence and non-existence. For example, the response should not be faster for an existing account than for an email with which an account does not exist.

**To mitigate the flaw**, the response must be uniform for both registered and unregistered email addresses. This uniformity must apply to the message displayed to the user as well as the underlying HTTP response details (like status codes, headers, and body content).

For example, the indistinguishable user-facing message could be: "A password reset link has been sent if an account with this email exists". [2]

## 2.2 Account Registration Form



```
/recaptcha/api2/reload
/recaptcha/api2/clr
/api/RegistrationCode/send-code-virtual/3/g...
/recaptcha/api2/clr
```

When email is not registered:

```
▾ {error: null, success: true}
    error: null
    success: true
```

When email is registered:

```
▾ {resultObject: {isPhone: false, isEmail: true, authorized: true, clientId: "1504846812",…},…}
    error: null
  ▾ resultObject: {isPhone: false, isEmail: true, authorized: true, clientId: "1504846812",…}
      authorized: true
      clientId: "1504846812"
      email: "GREGOREESMAA1@GMAIL.COM"
      fullPhone: null
      isBusiness: false
      isEmail: true
      isPhone: false
      isTargetEmail: true
    success: true
```

Figure 2: The vulnerability in the account registration form

The account registration form is also susceptible to account enumeration attacks. This is because when the provided email address is already taken, there are subtle differences in the response received from the server, compared to when the email is not taken (see Figure 2).

It is also crucial to eliminate any side-channels that an attacker could exploit to differentiate between account existence and non-existence. For example, the response should not be faster for an existing account than for an email with which an account does not exist.

**To mitigate the flaw**, the response must be uniform for both registered and unregistered email addresses. This uniformity must apply to the message displayed to the user as well as the underlying HTTP response details (like status codes, headers, and body content).

For example, the indistinguishable user-facing message could be: “We have sent further instructions to the provided email address”. Send an email in both cases, but differentiate the content based on account existence. For example, for new registration, provide means for account activation, and for existing accounts, provide means for account recovery. [2]

## 2.3 Timing Side-Channel Vulnerability

There is a side-channel in the registration and password reset forms, that allows for account enumeration. By measuring server response times, an attacker could reliably determine if an email address is registered.

`POST /api/RegistrationCode/send-code-virtual/3/{emailAddress}` endpoint responds significantly faster for registered emails than for unregistered ones. As shown in Table 1, requests for non-registered users take nearly ten times longer than requests for registered users. This discrepancy suggests that when an email is not found, the application performs additional, time-consuming operations, such as querying secondary systems or iterating over all database rows.

| **Attempt** | **Registered (ms)** | **Unregistered (ms)** |
|---|---|---|
| 1 | 534 | 4993 |
| 2 | 667 | 4618 |
| 3 | 497 | 4801 |
| *Average* | *566* | *4804* |

Table 1: Response times comparison for the registration code endpoint.

**To mitigate the flaw**, the application's response time must be made constant, regardless of whether the email is registered. The potential solutions are:

1. **Asynchronous Processing:** Immediately return a generic success message while processing the request (e.g., user lookup, sending an email) as a background job. This makes the perceived response time fast and consistent for all requests.
2. **Uniform Execution Path:** Ensure the server performs the same amount of work for all requests. This would involve removing any "early exit" conditions, making all responses consistently take the longer time ($\sim$ 5 seconds). For example, the process should query all potential data sources for every request, even if the email is found in the first one. If the problem is with a single database query, adding an index to the email column could also help.

**About** This vulnerability report is part of an ongoing study on user enumeration vulnerabilities in Estonian online services. The study is conducted by the University of Tartu master's student Gregor Eesmaa (supervised by Arnis Paršovs - arnis.parsovs@ut.ee). The findings of this study will be published in a master's thesis scheduled for defence in August 2025.

## References

[1] European Union. *General Data Protection Regulation (GDPR): Regulation (EU) 2016/679*. Official Journal of the European Union, L 119/1. 2016. URL: `https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32016R0679`.

[2] OWASP. *Authentication Cheat Sheet - Authentication and Error Messages*. Accessed: 2025-01-26. URL: `https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html#authentication-and-error-messages`.