**Problem**

RecoverController.java generates passwords via org.apache.commons.lang.RandomStringUtils, which uses java.util.Random internally. This PRNG has a 48-bit seed that can easily be bruteforced, leading to trivial privilege escalation attacks.

This PRNG is has a 48-bit seed, that can easily be bruteforced if an attacker is able to get the PRNG's output, for example but resetting their own account multiple times, leading to trivial privileges escalation attacks.

**RecoverController.java**

boolean captchaOk;

if (settingsService.isCaptchaEnabled()) {

String recaptchaResponse = request.getParameter("g-recaptcha-response");

ReCaptcha captcha = new ReCaptcha(settingsService.getRecaptchaSecretKey());

captchaOk = recaptchaResponse != null && captcha.isValid(recaptchaResponse);

} else {

captchaOk = true;

}

if (!captchaOk) {

map.put("error", "recover.error.invalidcaptcha");

} else if (user == null) {

map.put("error", "recover.error.usernotfound");

} else if (user.getEmail() == null) {

map.put("error", "recover.error.noemail");

} else {

**String password = RandomStringUtils.randomAlphanumeric(8);**

if (emailPassword(password, user.getUsername(), user.getEmail())) {

map.put("sentTo", user.getEmail());

user.setLdapAuthenticated(false);

user.setPassword(password);

securityService.updateUser(user);

} else {

map.put("error", "recover.error.sendfailed");

}

}

}

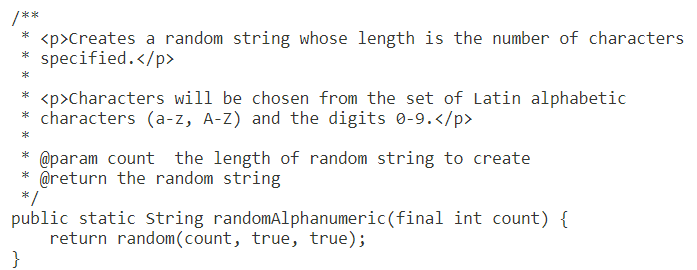
[**https://commons.apache.org/proper/commons-lang/javadocs/api-3.9/org/apache/commons/lang3/RandomStringUtils.html#randomAlphanumeric-int-**](https://commons.apache.org/proper/commons-lang/javadocs/api-3.9/org/apache/commons/lang3/RandomStringUtils.html#randomAlphanumeric-int-)

[**randomAlphanumeric**](https://commons.apache.org/proper/commons-lang/javadocs/api-3.9/org/apache/commons/lang3/RandomStringUtils.html#randomAlphanumeric-int-)(int count)

Creates a random string whose length is the number of characters specified.

Graphical user interface, text, application, email

Description automatically generated



<https://docs.oracle.com/javase/8/docs/api/java/util/Random.html>

Text, application

Description automatically generated with medium confidence

Graphical user interface, text, application, email

Description automatically generated

<https://www.baeldung.com/java-secure-random>

Text

Description automatically generated

A picture containing text, tiled

Description automatically generated

**Fix**

**RecoverController.java**

**import java.security.SecureRandom;**

private static final String SYMBOLS = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890";

private final SecureRandom random = new SecureRandom();

private static final int PASSWORD\_LENGTH = 32;

StringBuilder sb = new StringBuilder(PASSWORD\_LENGTH);

for(int i=0; i<PASSWORD\_LENGTH; i++) {

int index = **random.nextInt**(SYMBOLS.length());

sb.append(SYMBOLS.charAt(index));

}

String password = sb.toString();

[https://docs.oracle.com/javase/8/docs/api/java/security/**SecureRandom**.html](https://docs.oracle.com/javase/8/docs/api/java/security/SecureRandom.html)

protected final int next(int numBits)

Generates an integer containing the user-specified number of pseudo-random bits (right justified, with leading zeros). This method **overrides a java.util.Random method** and serves to provide a source of random bits to all of the methods inherited from that class (for example, nextInt, nextLong, and nextFloat).

**Overrides:**

[next](https://docs.oracle.com/javase/8/docs/api/java/util/Random.html#next-int-) in class [Random](https://docs.oracle.com/javase/8/docs/api/java/util/Random.html)

**Parameters:**

numBits - number of pseudo-random bits to be generated, where 0 <= numBits <= 32.

**Returns:**

an int containing the user-specified number of pseudo-random bits (right justified, with leading zeros).

Text

Description automatically generated with low confidence

Text

Description automatically generated

<https://webpages.charlotte.edu/yonwang/papers/lilesorics.pdf>

[**https://www.geeksforgeeks.org/random-vs-secure-random-numbers-java/**](https://www.geeksforgeeks.org/random-vs-secure-random-numbers-java/)

Text

Description automatically generated