

# Password Generator 2009 Professional Performance Guide

This document outlines performance test results for the main GUI application (pgapplication.exe) of Password Generator 2009.

## Contents

Password Generator 2009 Professional Performance Guide .....	1
Test Setup.....	1
Results .....	1
Basic Generation Mode.....	1
Advanced Generation Mode .....	2
Mask Generation Mode .....	3
Pronounceable Generation Mode .....	3
Random Number Generators.....	3
Special Tests .....	3
Notes and Explanations.....	3

## Test Setup

The tests were performed on a PC with 3 GB of RAM, Intel Pentium D 940 3.2 GHz processor, and Windows Vista Ultimate Operating System. All tests were executed on **one core** of the processor, thus if both cores were used, the results would have been roughly twice as good.

## Results

The results are given in hh:mm:ss format. For example 00:00:01.4560000 means 1.456 second, or 1456 milliseconds.

## Basic Generation Mode

This section shows performance test results for Basic Generation Mode.

Test 1: Use Character Policies, Allow Duplicates				
Test Description		Count	Total Time	Passwords/Second
<b>Mode:</b> Basic		100	00:00:00.3085740	324
<b>Password Length:</b> 10		1 000	00:00:00.3359160	2 977
<b>Character Policies:</b> Yes		10 000	00:00:00.3700935	27 020
<b>Duplicates Check:</b> No		100 000	00:00:00.5673465	176 259
<b>RNG:</b> XorShift128		1 000 000	00:00:02.6179965	381 971
<b>Priority:</b> Normal		10 000 000	00:00:21.3376295	468 655

Notes: The reason why more passwords are generated per second when generating large amount of passwords versus when generating small amount of passwords comes from the fact that some time is needed for the engine to initialize before it actually begins generating passwords.

Test 2: Don't Use Character Policies, Allow Duplicates			
Test Description	Count	Total Time	Passwords/Second
<b>Mode:</b> Basic	100	00:00:00.2509605	398
<b>Password Length:</b> 10	1 000	00:00:00.3378690	2 960
<b>Character Policies:</b> No	10 000	00:00:00.3534930	28 289
<b>Duplicates Check:</b> No	100 000	00:00:00.4657905	214 689
<b>RNG:</b> XorShift128	1 000 000	00:00:02.4080490	415 274
<b>Priority:</b> Normal	10 000 000	00:00:19.3521095	516 739

Notes: This is probably one of the fastest configurations for a given password length.

Test 3: Don't Use Character Policies, Disallow Duplicates			
Test Description	Count	Total Time	Passwords/Second
<b>Mode:</b> Basic	100	00:00:00.2519370	397
<b>Password Length:</b> 10	1 000	00:00:00.3486105	2 868
<b>Character Policies:</b> No	10 000	00:00:00.4189185	23 871
<b>Duplicates Check:</b> Yes	100 000	00:00:01.4286195	69 998
<b>RNG:</b> XorShift128	1 000 000	00:00:14.7148785	67 958
<b>Priority:</b> Normal	10 000 000	00:02:39.3521095	62 754

Notes: As you can see, checking for duplicates significantly lowers performance if you generate a lot of passwords. However duplicates check improves overall security.

## Advanced Generation Mode

This section shows performance test results for Advanced Generation Mode using (luns) as mask, which basically means that all alphanumeric and special characters are allowed in any order.

Test 1: Use Character Policies, Allow Duplicates, Relative-Random Composition Mode			
Test Description	Count	Total Time	Passwords/Second
<b>Mode:</b> Advanced, Relative-Random	100	00:00:00.2753730	363
<b>Password Length:</b> 10	1 000	00:00:00.3388455	2 951
<b>Character Policies:</b> Yes	10 000	00:00:00.3671640	27 236
<b>Duplicates Check:</b> No	100 000	00:00:00.5810175	172 112
<b>RNG:</b> XorShift128	1 000 000	00:00:03.7341360	267 800
<b>Priority:</b> Normal	10 000 000	00:00:30.6548310	326 213

Notes: In this test, 4 basic character groups are used to generate passwords: lower case characters, upper case characters, numeric characters, and special (punctuation) characters. In Relative-Random mode, which was used in this test, the number of characters appearing in a password from each character group is determined using 1) character group's density property, and 2) some random value. In this test all 4 group's densities were set to 5, meaning that a particular character has equal probability to be chosen from any of the 4 groups, which, in turn makes for precisely 2.5 characters from each character group in password of length 10, as  $10/4 = 2.5$ . But because it is not possible to have 2.5 characters, a random is used to make a choice and make the total number of characters in a password from a particular group to be either 2 or 3.

Test 2: Use Character Policies, Allow Duplicates, Absolute Composition Mode			
Test Description	Count	Total Time	Passwords/Second
<b>Mode:</b> Advanced, Absolute	100	00:00:00.3085740	324
<b>Password Length:</b> 10	1 000	00:00:00.3251745	3 075
<b>Character Policies:</b> Yes	10 000	00:00:00.3945060	25 348
<b>Duplicates Check:</b> No	100 000	00:00:00.7157745	139 709
<b>RNG:</b> XorShift128	1 000 000	00:00:03.8669400	258 602
<b>Priority:</b> Normal	10 000 000	00:00:31.4506201	317 959

Notes: In this test, 4 basic character groups are used to generate passwords: lower case characters, upper case characters, numeric characters, and special (punctuation) characters. In Absolute mode, which was used in this test, the number of characters from each character group that will appear in a password is determined using character group's Min and Max properties. Suppose you want to ensure that your passwords contain at least 2 punctuation characters. Then you would set special character group's Min property to 2, and Max to, for example, 5. In this test all 4 groups had their Min property set to 2, and Max to 5.

### Mask Generation Mode

Coming soon!

### Pronounceable Generation Mode

Coming soon!

### Random Number Generators

Coming Soon!

### Special Tests

Coming soon!

### Notes and Explanations

- If you require top performance, console password generator available in Server Edition is the best choice, as it does not have overhead associated with Graphical User Interface.
- Increasing available memory, when generating large amount of passwords will improve performance!