# EverQuest Trilogy Client Deconstruction

## Random Name Generation

EverQuest allows players to generate random names for their characters on the character creation screen. The names reflect the chosen race and gender of that character. Below is a breakdown of how the name generation works. The function can be found at **0x004af3d7**. It was reverse engineered from the Trilogy Client using Ghidra and ollydbg.

### Name Fragment Table

To create the random names, the generator assembles a name using a **name fragment table**. The name fragment table is generated at runtime and doesn't contain C style strings directly, but pointers to C style strings. This allows multiple races to reference the same name fragments, useful for common fragments like "er" or "an." Name fragments have a maximum length of four characters. The name fragment table starts at **0x00566838** and contains 50 pointers per race (750 total). The pointers are grouped in sequences of five, in **name clusters**. A name cluster can be thought of as one complete name composed of five name fragments. Below is a visualization of the first three human clusters laid out contiguously in memory.

| **D** | **en** | **e** | **lron** | **a** |
|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 |

| **B** | **al** | **ad** | **orn** | **y** |
|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 |

| **L** | **an** | **a** | **dor** | **e** |
|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 |

The "mixing" of the clusters from which a fragment is selected is what generates a unique name. As there are 10 clusters for each race, a name is formed by randomly selecting a cluster from which to pull a name fragment for each index 0-4. Not all fragment indices are used in each name and the selection logic is detailed below. The fifth index in a cluster is used only for female characters and contains a vowel. There is no guarantee that all clusters have unique string fragment values for certain indices. Clusters 2 and 4, for example, could both have "am" in index 2. This means that the number of unique combinations varies between races. Dwarves, for example, have a high occurrence of repeating values resulting in far fewer unique names that can be generated.

### Selection Logic

Selection for each name fragment is done based on a probability table. The table dictates the likelihood of using a particular index when generating a name. Index 1 and 2 have a 20% chance of being skipped when generating the name and allowing the generation to "skip" certain cluster indices allows names to vary in length. Below are the probabilities that the index in the cluster will be used in the name:

| Fragment Index | Probability |
|---|---|
| 0 | 100 |
| 1 | 80 |
| 2 | 80 |
| 3 | 100 |
| 4 | 0 if male<br>100 if female |

The name is built by determining based on a random number if we will use this fragment index, and then picking getting the associated name fragment from a random cluster associated with that race. The name is then assembled by linking all string fragments together.

Here is an example of the process of generating a name:

Troll Male

1. Calculate the probability table. The results are that fragment indices 1, 2, 3 and 4 are used.
2. Calculate the cluster we will pull values from. The results are clusters 3, 6, 1, and 8.
3. The resulting name is **Rakgak.**

High Elf Female

1. Calculate the probability table. The results are 1, 4 and 5.
2. Calculate the clusters. The results are 2, 4, 5, and 9.
3. The resulting name is **Lwen**.

## Name Validation

Once the name is assembled, it is validated to ensure that what was generated adheres to five rules:

1. The name must be between 4 and 15 characters, inclusive of these values.
2. The name cannot start with a character sequence of three consonants where the second and third are the same value (e.g. “Kgg”). If this is found, a random vowel (including Y) is generated and inserted between the first and second characters. “Kggrol” would for example, become “Koggrol.” Although the length of the name is increased by one, the limits are not rechecked. Even so, I was not able to find any way that a generated name would exceed 15 characters.
3. The name must not contain an instance of four consecutive consonants. If this is detected, the name is regenerated.
4. The name must not contain any bad word strings. These combinations are contained in a bad word table consists of 25 pointers to C strings which are not fixed in size. The table can be found at address **0x005679d0**. This check runs checks to see if the generated name contains any of

these bad words and if so, the name is regenerated. Both strings are converted to lowercase before the check is done.

5. The name must not contain any prohibited per race combinations. These are separate from the bad words and prevent unintended letter combinations from occurring in generated names. The table is found at **0x005673f0** and contains 25 C string pointers for each race. The strings have a fixed length of 4 bytes. Unlike the bad word check, the values are not converted to lowercase so if the check is for the string “Wd”, it is relevant to the beginning of the name only.