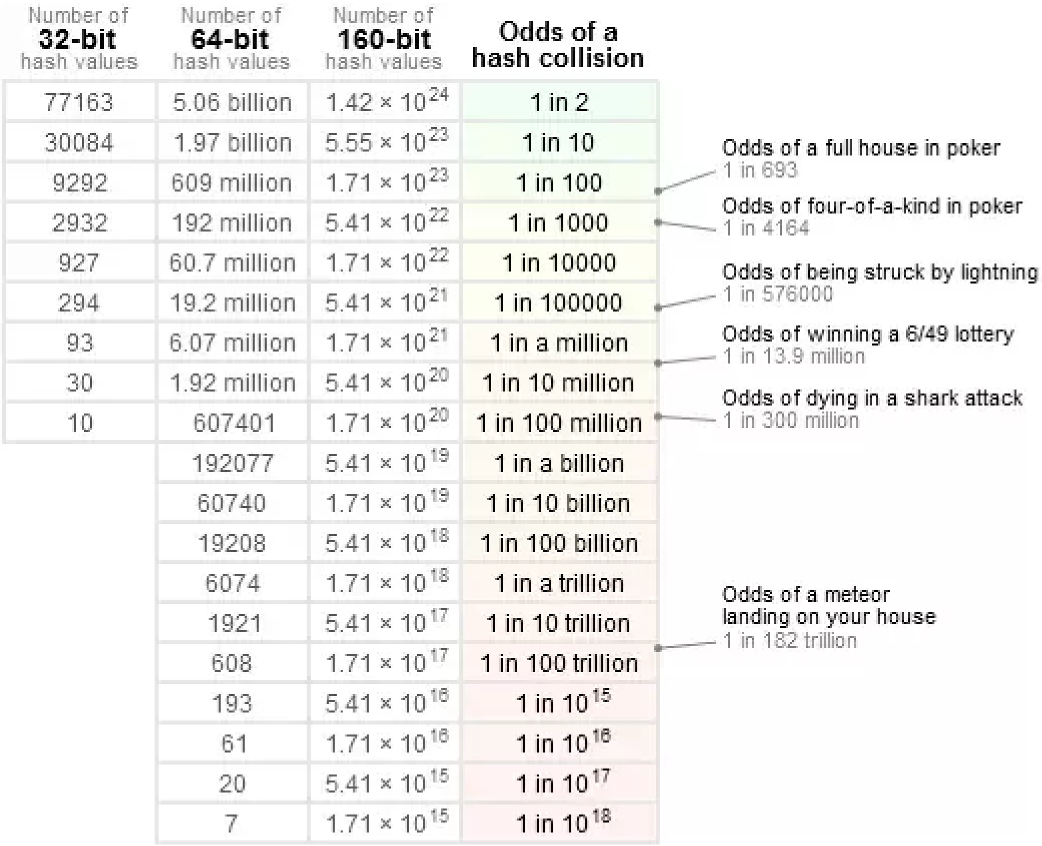
SHA1 is definitely more accurate than CRC32, just see the table below:



Number of **32-bit** hash values	Number of **64-bit** hash values	Number of **160-bit** hash values	**Odds of a hash collision**
77163	5.06 billion	1.42×10^{24}	1 in 2
30084	1.97 billion	5.55×10^{23}	1 in 10
9292	609 million	1.71×10^{23}	1 in 100
2932	192 million	5.41×10^{22}	1 in 1000
927	60.7 million	1.71×10^{22}	1 in 10000
294	19.2 million	5.41×10^{21}	1 in 100000
93	6.07 million	1.71×10^{21}	1 in a million
30	1.92 million	5.41×10^{20}	1 in 10 million
10	607401	1.71×10^{20}	1 in 100 million
	192077	5.41×10^{19}	1 in a billion
	60740	1.71×10^{19}	1 in 10 billion
	19208	5.41×10^{18}	1 in 100 billion
	6074	1.71×10^{18}	1 in a trillion
	1921	5.41×10^{17}	1 in 10 trillion
	608	1.71×10^{17}	1 in 100 trillion
	193	5.41×10^{16}	1 in 10^{15}
	61	1.71×10^{16}	1 in 10^{16}
	20	5.41×10^{15}	1 in 10^{17}
	7	1.71×10^{15}	1 in 10^{18}

Odds of a full house in poker
1 in 693

Odds of four-of-a-kind in poker
1 in 4164

Odds of being struck by lightning
1 in 576000

Odds of winning a 6/49 lottery
1 in 13.9 million

Odds of dying in a shark attack
1 in 300 million

Odds of a meteor
landing on your house
1 in 182 trillion

(Image taken from a great article: Hash Collision Probabilities).

CRC32 is 32-bit, SHA1 is 160-bit. So if you check 100 files, there will be approx. 1 in a million chance that for two of them CRC32 hashes will collide and report files as duplicates when they are not. On the other hand, with SHA1 the chance will be 9 orders of magnitude less: approx. 1 in 10^15.

However SHA1 is also more complex to compute, whereas CRC32 is rather simple. It might make sense to employ both: CRC32 first, and then SHA1 to make sure that files with same CRC32 hashes are indeed duplicates of each other.

But if you are writing an utility that will find and delete duplicate files, think if you can afford even such miniscule probability of a false positive. Probably you'll have to do a bit-by-bit comparison of the files as the final check.

Update: also consider that for large files, more efficient method is to not use hashes of the whole file, but read and hash files by blocks.This way, you'll be able to quit checking as soon as you hit first different block.