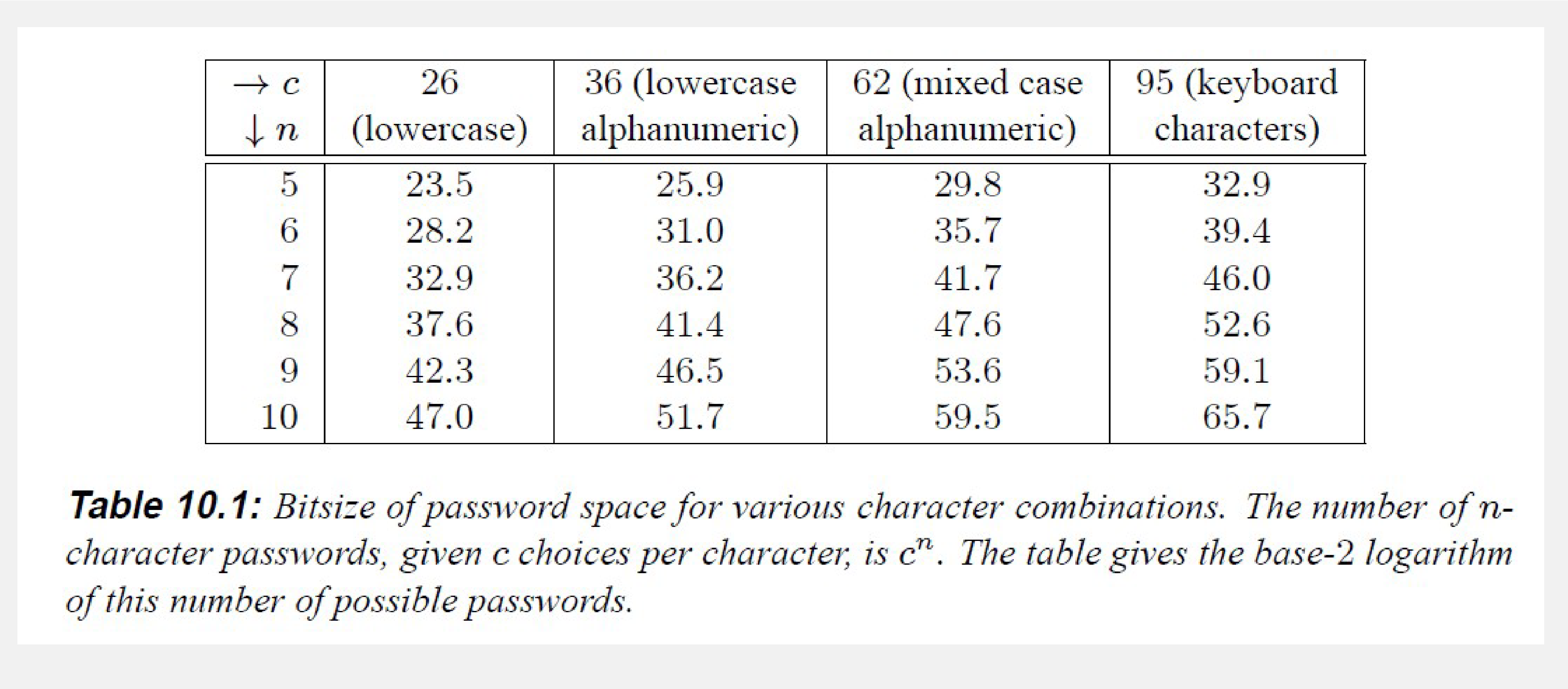
PASSWORD ENTROPY-measured by 2^k



$\rightarrow c$ $\downarrow n$	26 (lowercase)	36 (lowercase alphanumeric)	62 (mixed case alphanumeric)	95 (keyboard characters)
5	23.5	25.9	29.8	32.9
6	28.2	31.0	35.7	39.4
7	32.9	36.2	41.7	46.0
8	37.6	41.4	47.6	52.6
9	42.3	46.5	53.6	59.1
10	47.0	51.7	59.5	65.7

Table 10.1: *Bitsize of password space for various character combinations. The number of n-character passwords, given c choices per character, is c^n. The table gives the base-2 logarithm of this number of possible passwords.*

Source: Menezes et al. Handbook of Applied Cryptography.

Now **2^35** complexity can be cracked within **a day** on a 3GHz PC (generous est).

1 FPGA Hardware cracker can crack 56 bits within **5 days** (est).

ASIC crackers can be more than 10 times faster than FPGA.

DICTIONARY ATTACK

- Choosing passwords with high entropy prevents brute-force attack.
- However, hashed passwords, especially for human-generated passwords, are still vulnerable to *dictionary attack*.
- This exploits weakness in human-chosen passwords, which tend to derive from words in natural languages.

Users with same password will have same hash value stored in password file.

- Guess some commonly used passwords
- Compute their hash values
- Look for the same hash values in the password file

PRE-COMPUTED HASH TABLE

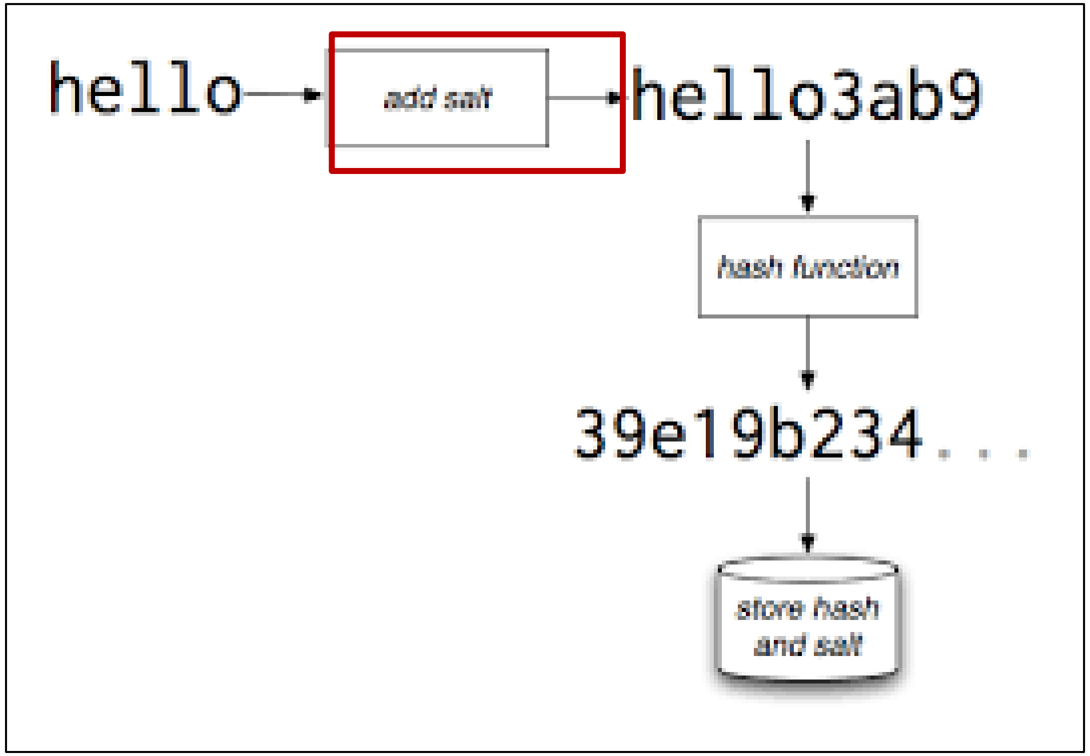
Strategy
A strategy for cracking hashed passwords is to pre-compute a hash table, containing pairs of passwords and their hashes.

- If we have k password candidates and each hash has n bit, then we have a table of size $k \times n$.
- This may not be practical if k is large.

PASSWORD SALTING

Salting

Illustration



- To reduce the effectiveness of offline attacks using pre-computed hashes, a *salt* is added to a password before applying the hash function.
- A salt is just a random string.
- Each password has its own salt.
- The salt value is stored along with the hash of password+salt.
- For a salt of n-bit, the attacker needs to pre-compute 2^n of hashes for *the same password*.

Password Storage Cheat Sheet

Introduction¶

- It is essential to store passwords in a way that prevents them from being obtained by an attacker even if the application or database is compromised.
- After an attacker has acquired stored password hashes, they are always able to brute force hashes offline.
- As a defender, it is only possible to slow down offline attacks by selecting hash algorithms that are as resource intensive as possible.