What is an *Online* Password Guessing Attack?

**Offline**
Attackers steal the database of hashed passwords and guess passwords to hash

**Online**
Attackers connect to website and attempt to login with guessed passwords (often most-common first)
Best Practices for Stopping Online Attacks

• Block accounts after too many failed login attempts
  • Effective if your system sees targeted attacks on a tiny fraction of users
  • Disastrous if attacker tries guessing popular passwords against all accounts

• Block IP addresses that make too many failed attempts
  • Effective if attacker only has one IP address, but becomes less effective as the number of IP addresses available to the attacker increases
  • Potentially disastrous if legitimate user has app that repeatedly tries to login with the users’ previous password
IP-Based Blocking Requirements

• Minimize the opportunities for an IP to login by guessing a password
• Minimize the chance that we block a legitimate user’s login
• Minimize all other information revealed to attackers
Evidence Collected When an IP Attempts to Login

- Is the password provided the correct password?
- Is this an incorrect password that we’ve seen for this user before?
- Did client provide a cookie proving a previous successful login for the account it is trying to login to?

Already used in practice

- Is the password provided one that is being guessed frequently?
- Is the password provided a typo’s distance away from being correct?

Appear to require access to plaintext or unsalted passwords

By finding ways to collect this evidence safely, we can improve decision making
Identifying Typos: a Catch 22

**Immediate Approach:**
When incorrect password provided, compare to correct password.
• Problem: If storage of correct password is compromised, attackers get the plaintext passwords.

**Posthoc Approach:**
Store incorrect passwords, and compare when user next logs in with the correct password.
• Problem: If storage of incorrect passwords is compromised, attackers get typos that make it easy to guess some accounts’ passwords correctly.
Typos: How We Secure Posthoc Checking

• Each account has public/secret key pair \((p, s)\)
• For each account, we store \(p\), Encrypt\(_{\text{password}}\)(\(s\))
  \((p\) in plaintext and \(s\) encrypted with the user’s correct password\)
• On login with incorrect password, log
  Encrypt\(_p\)(\text{password\_provided})
• On login with correct password, log
  \(s = \text{Decrypt}_{\text{correct\_password}}(s)\)
  \(\text{previous\_password\_provided} = \text{Decrypt}_s(\text{password\_provided})\)
  \(\text{IsTypo} = \text{Distance}(\text{correct\_password},\)
  \(\text{previous\_password\_provided}) < \text{threshold}\)

Even if incorrect passwords leak information about the correct password, one cannot decrypt the incorrect passwords without the correct one.
Identifying popular guesses

- Count incorrect guesses using a probabilistic data structure (binomial ladder frequency filter [Schechter2018])
  - Like a bloom filter, but that keeps counts instead of set membership

The probability we will track an unsalted password that is not being guessed frequently can be made arbitrarily small by tuning the filter.

Strategies

• Penalize frequently-guessed passwords
• Defend weakest accounts more
• Penalize typos less than other failures
• Ignore repeat account/password pairs
• Treat invalid accounts differently
• Offset failures with successes
Simulator for Evaluations

• Flexible configuration:
  • We allow variations of parameters

• Simulating legitimate users:
  • We simulate different behaviors: login frequency, IP/cookies, typo, wrong password, wrong account, outdated password

• Simulating attackers:
  • Descending-popularity attack
  • Weighted attack,
  • Detection-avoidance attack
Evaluations - Effectiveness of the baseline threshold-based IP blocking
Evaluations - Effectiveness of StopGuessing

- All strategies are effective for all three attacks
- Most effective: ignore repeat account/password pairs, adjust the blocking threshold based on the account password’s popularity
Evaluations-Combine with Forbidding Users from Having Common Passwords

• With a ban on the top 10,000 passwords:
  • Less than 20 compromised accounts while keeping false positives down to double digits for 5 million accounts
Practical Impact

- A customized version is used to protect Microsoft Azure
- Open source implementations for different scales of website/app developers [https://github.com/Microsoft/StopGuessing](https://github.com/Microsoft/StopGuessing)

- ASP.NET developers can include our current implementation for online password guessing defense

- Other platforms can use it as an authentication service
Conclusion

• We build a online brute forcing detection system that is
  ➢ Robust against online attacks
  ➢ Don’t disclose more information under offline attacks
  ➢ Stop attacks at much earlier stages
  ➢ Lightweight and scalable for website/app developers
Thanks!

Yuan Tian
yuant@virginia.edu