Final exam

- **Monday, May 10 – Don't miss it!**
  - Open Book, Open Note
  - Exam covers full semester
  - Bring copies of the papers
- **If you can't make exam, speak to me now**
  - If there is demand, will schedule something 2 hrs. earlier

End of semester schedule

- **Lab 2A due tonight, 2B due next Tuesday**
- **Sections:**
  - Antonio will answer review questions in this weeks sections
  - No lecture next Tuesday
  - I will have review sections next Wednesday, Thursday
- **My office hours schedule:**
  - Wednesday April 27, 4:30-5:30pm
  - No office hours Monday, May 2
  - Thursday May 5, 5-6pm
  - Friday May 6, 4-5pm
  - Monday May 9, 4:30-5:30pm

SSH overview

- **Widely-used secure remote login program**
- **MACs/encrypts all data sent over the network**
  - Version 2 of protocol basically gets this right
  - Open to man in the middle attack on first server access
- **Often sends password at start of session**
  - Gets sent encrypted in a single TCP packet
- **Assuming crypto secure (& no MiM), how to attack?**

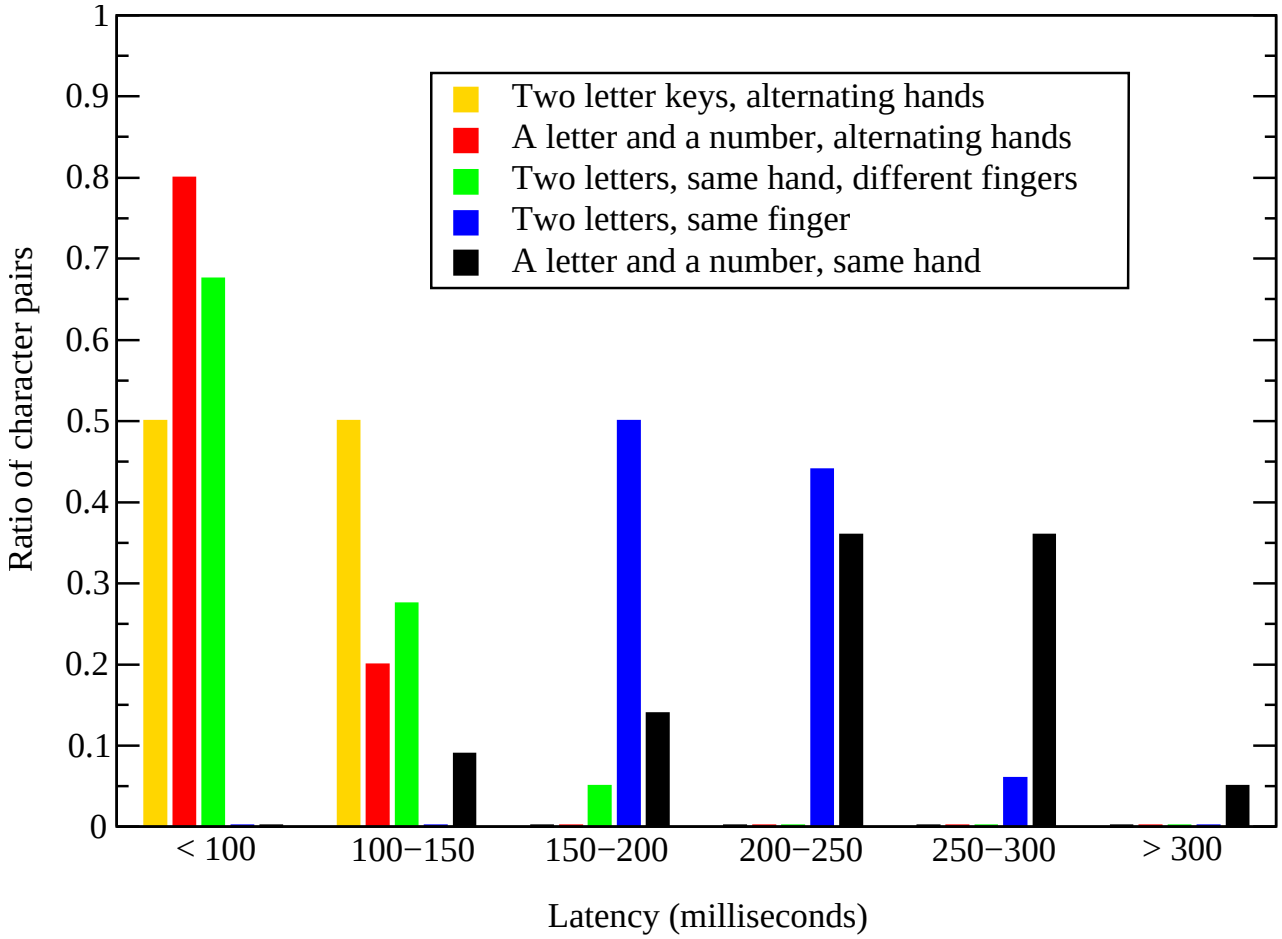
Packet size

- **Transmitted packets rounded to multiple of 8 bytes**
  - Version 1 even had exact packet-size in the clear
- **Can tell if user's password is less than 7 chars**
  - Password sent in one packet of initial exchange
- **Why do we care?**
  - Might tell you which account to try to crack

Inter-keystroke timings

- **Each character typed causes a packet to be sent**
  - Typical inter-character times 10–300 msec
  - Typical network round-trip time 10 of msec
  - Can get very accurate timing information by eavesdropping
- **What can you learn from this?**
  - Some character sequences harder to type than others
  - E.g., v–b is much slower to type than v–o
  - In general, characters with different hands faster
  - Two characters typed with same finger are much slower
  - Digits, special chars also slower
- **Idea: Use timing to learn about passwords**

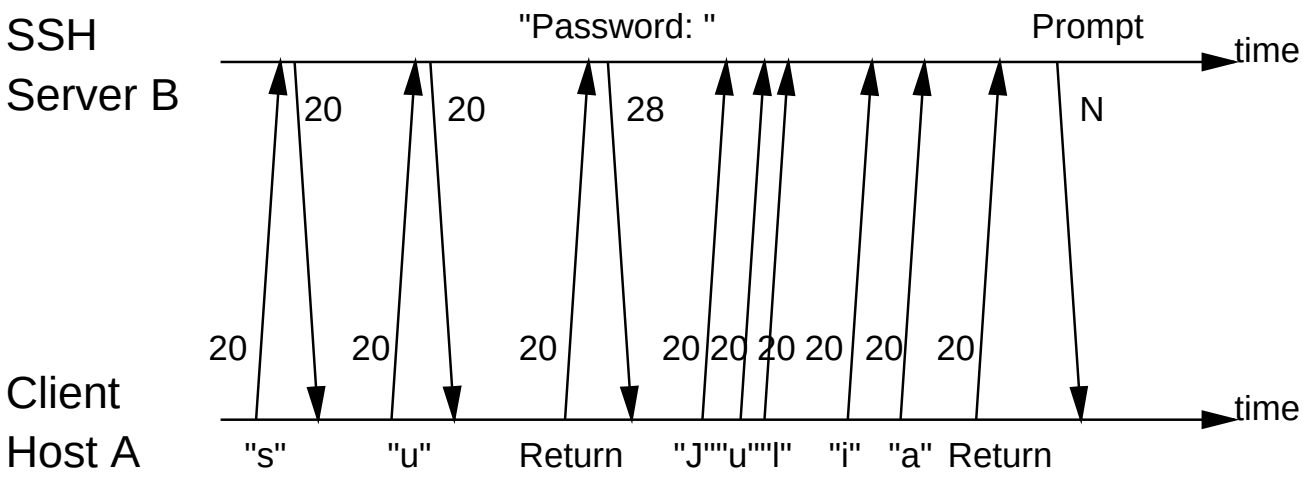
Character latency



How to know password is being typed

- **Traffic signature**
  - E.g., echo turned off when password typed
- **Multi-user attack**
  - E.g., run ps on machine to see when victim runs pgp
- **Nested ssh attack**
  - See remote host open SSH connection to another host

Example: su command

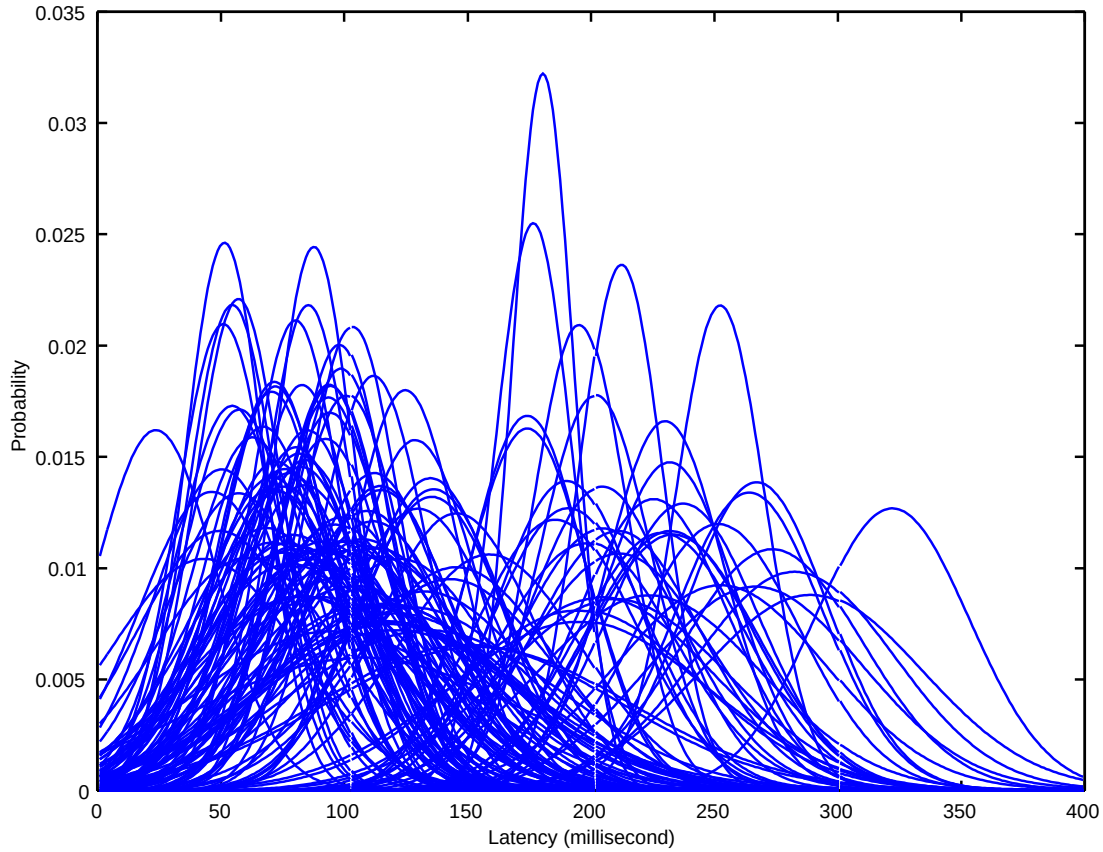


- **"Password:" prompt – 28 char packet**
- **Echo turned off for password, no return packets**

Modeling keystroke timings

- **Assume Gaussian-like distribution of timings**
  - For each key pair q, mean time μ_q, stdev σ_q
  - Prob. of timing y $\Pr[y|q] = \frac{1}{\sqrt{2\pi}\sigma_q} e^{-\frac{(y-\mu_q)^2}{2\sigma_q^2}}$
  - Significant but far from complete overlap between key pairs
- **Model keystrokes as HMM**
  - Each key pair is a state, timing an observation
  - AI techniques allow you to get n best choices

Latency vs. probability of key pairs



Results

- **Experiment: Assign users random passwords**
  - Picked from a reduced set of characters
  - Users practice typing the password before experiments
- **Train on users typing individual key pairs**
- **Ignore pause in the middle of passwords**
- **Output most likely password**
- **Bottom line:** $50\times$ **reduction in brute-force cracking**
  - Half the time password shows up in top 1% output

How to work around the problem

- **Send dummy packets when in echo mode**
  - Foils traffic signature detection of passwords
- **Adding random delays to packets?**
  - Latencies in 100s of msec, so need big random delays
  - Can still get info by averaging many sessions
  - Delay might get seriously annoying
- **Constant bit-rate traffic**
  - Practical for *one session* over a modem

Discussion

- **How serious is this vulnerability?**
  - Would this matter in a system like TAOS?
- **What else could this technique be applied to?**
- **Other possible solutions to the problem?**

Related attacks

- **Timing attacks on crypto**
  - Can infer information about private keys by time to decrypt
- **RSA solution: "blinding"**
  - Pub key $\langle n, e \rangle$, priv key $\langle n, d \rangle$, want to compute $c^d \bmod n$
  - Chose random r, compute $b = \left(r^{-1}\right)^e \bmod n$
  - Compute $(bm)^d\, r \bmod n$
- **Other similar attacks**
  - Power consumption
  - Fault-based attacks (on tamper-proof hardware)

Review

- Why chosen ciphertext security matters
- Prudent Engineering Practice
- Client Authentication on the Web
- Kerberos
- Terra
- Inferring errors in systems code
- Address space randomization

Review 2

- Software fault isolation
- Security Architectures for Java
- Byzantine fault tolerance
- Listen and Whisper
- Dynamic Taint Analysis
- Crowds
- Electronic Voting