Lucent Technologies
Bell Labs Innovations

Cryptographic  Generation using features in a speaker's voice

Fabian Monrose
Bell Labs, Lucent Technologies

Joint work with Q. (Peter) Li*, Mike Reiter & Susanne Wetzel

Road Map

- Motivation
- High level illustration of our scheme
- Brief look at related work
- Background information and criteria
- Examine proposed technique
- Empirical results
- Conclusion (major hurdles)

What are we trying to do?

Cryptographic
Key for encrypted
communications

Voice signal

Key recovery should be difficult for adversary even if the device is captured.

Why Voice?

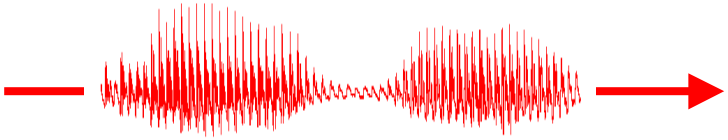
- A natural user interface for many devices

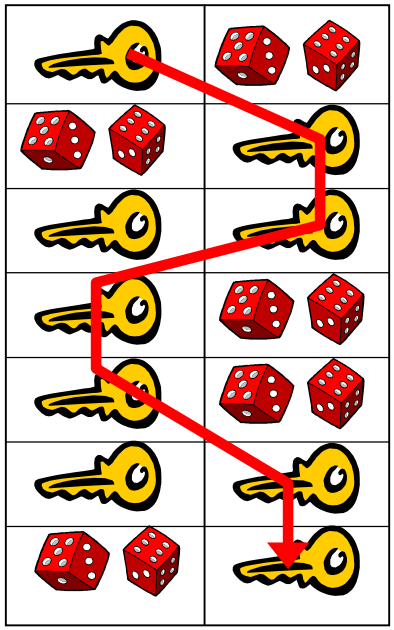
- Known to differentiate between users
  - rich literature in speaker verification

- Unlike static biometrics such as a fingerprint, changing the password changes vocalization of it, so a user can have many keys.

Illustration of our technique

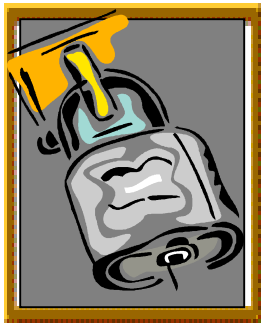
Speaker





Table

recovered key



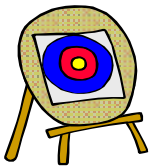
Encrypted file

Yeah, but isn't this similar too … ?

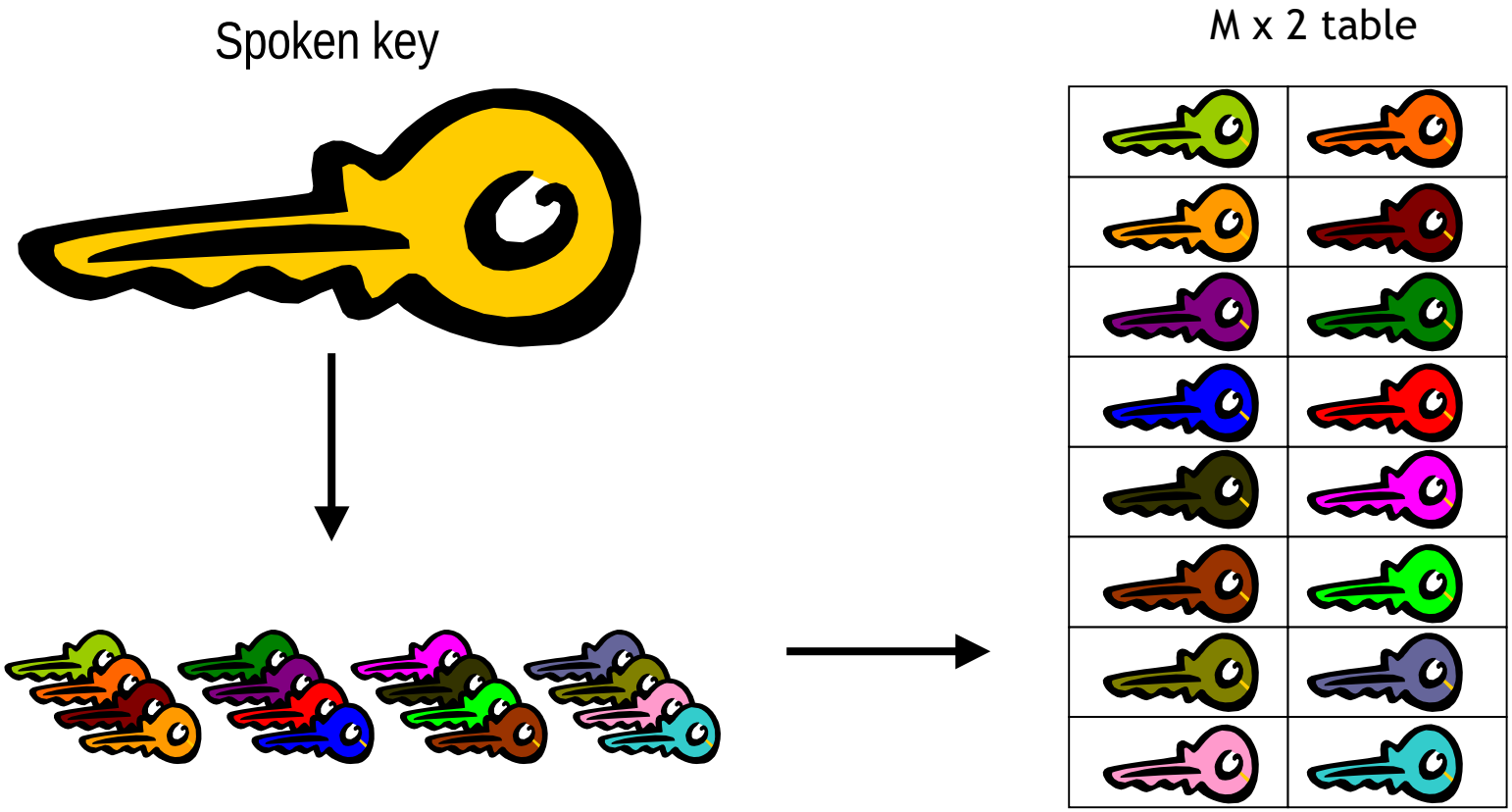
- Voice encryption (e.g., STU-III, PGPFone, etc)
  - ✘ encrypts voice signal, but generates key via other input
- Encryption with spoken password
  - ✘ password entropy is low
  - ✘ even lower for pronounceable passwords
- Speaker verification
  - ✘ compares speech to speaker-dependent, plaintext model
  - ✘ if captured, model leaks keying material

What we aimed for…

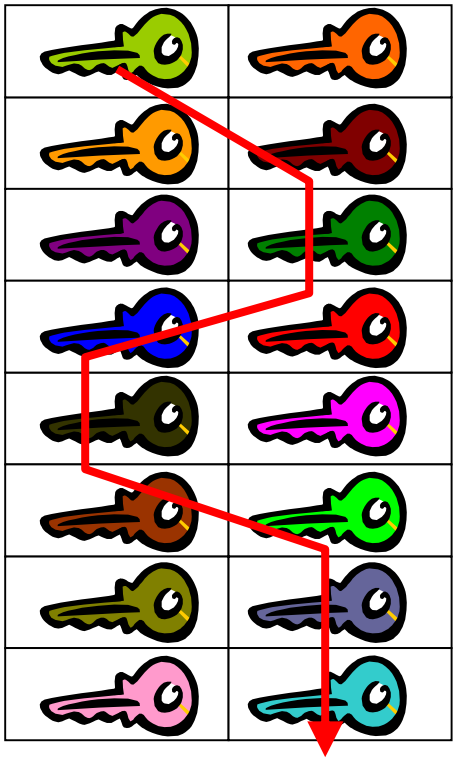
- Sufficient entropy – key should be resilient to offline attacks. No speaker specific information must be stored in plain text on the device.
- False negative/positive rates should be negligible – legitimate speakers should repeatedly generate same spoken key in an environment with low signal/noise ratio.
- Efficient implementation on resource constrained devices, e.g., 3G phones and PDAs.



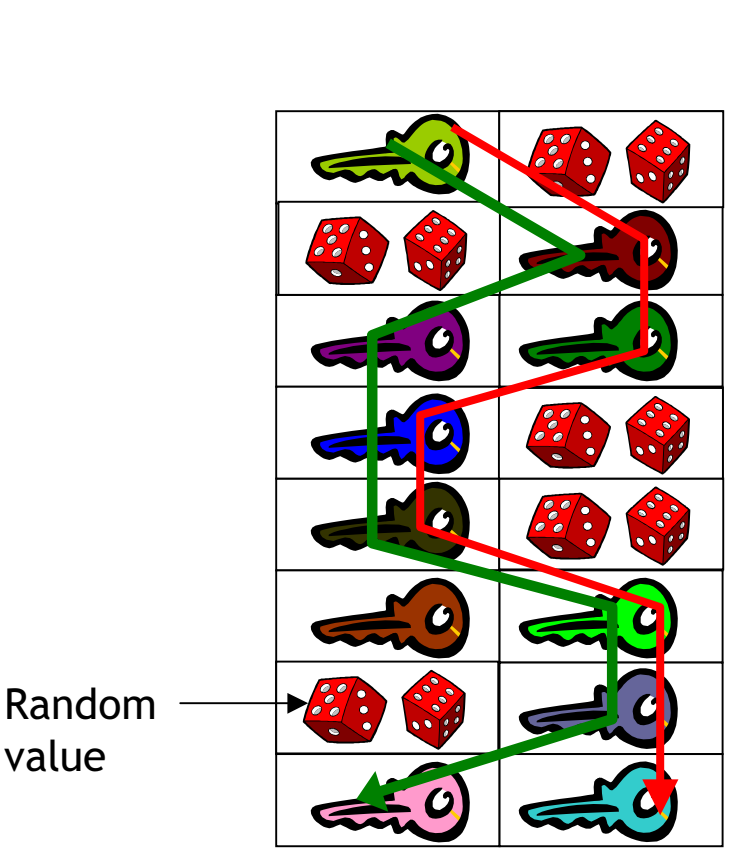
The basic idea – dispersing the secret

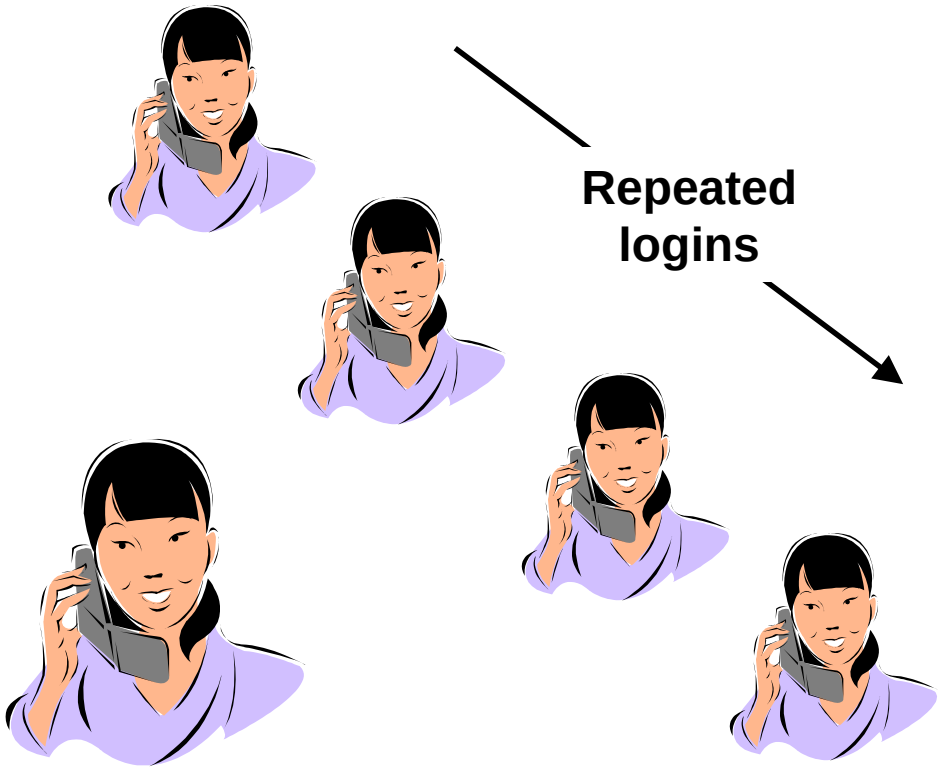


The basic idea - reconstruction



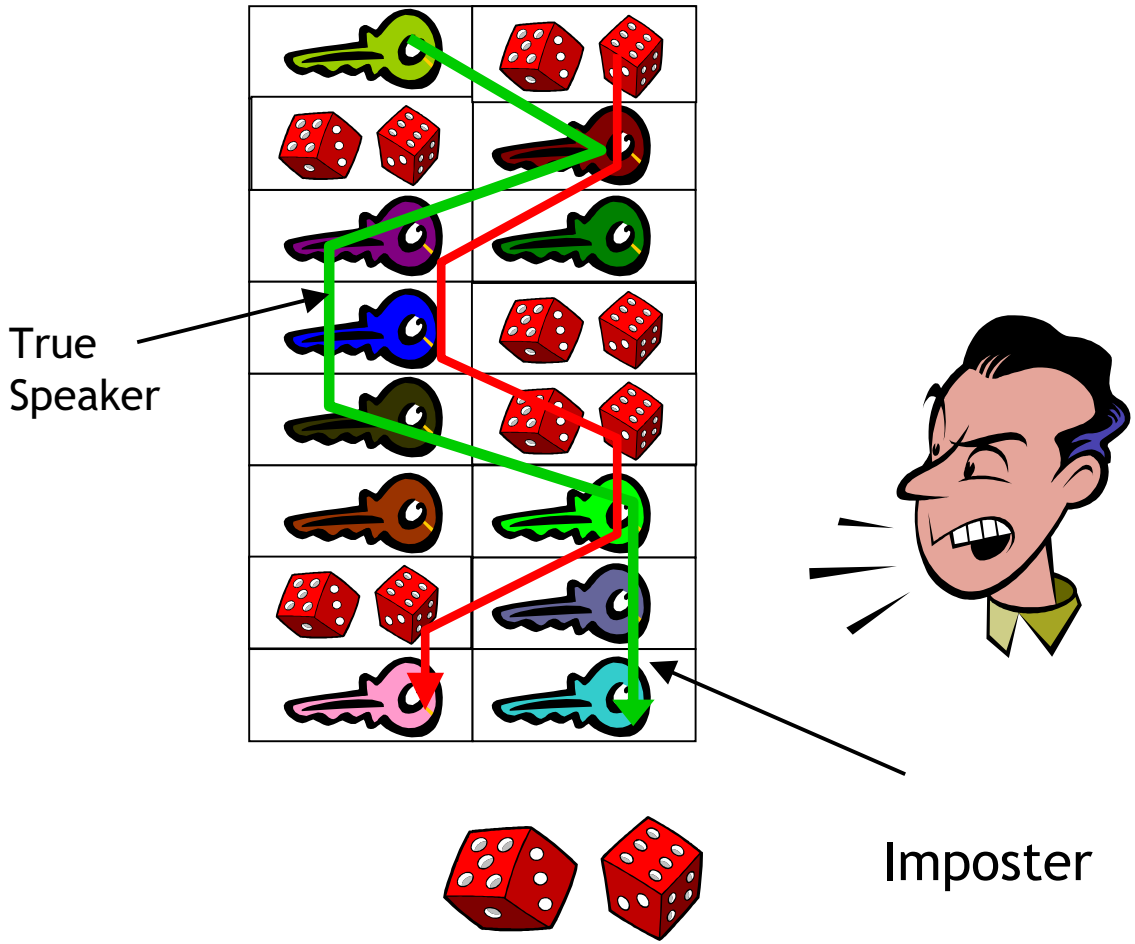
The basic idea - how it works





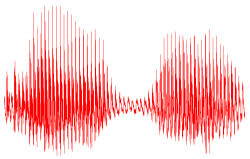
Destroy pieces the correct user doesn't "need"

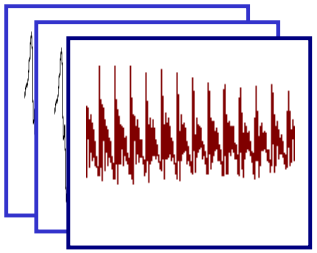
The basic idea - how it works

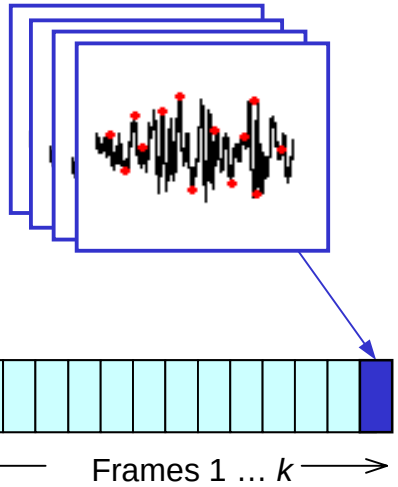


Okay, so what is this good for?

- Conceptually, can be used in any context were traditional passwords are used.
- Encrypted email.
- Used to generate keying material for private/public key generation, e.g, VPN access.
- File encryption.





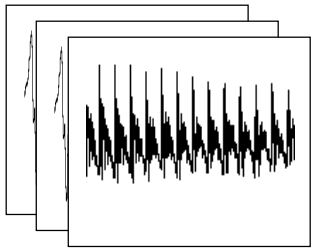


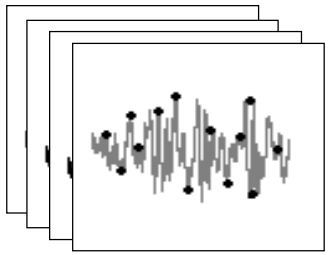
Capture Signal:
8000 samples/sec.

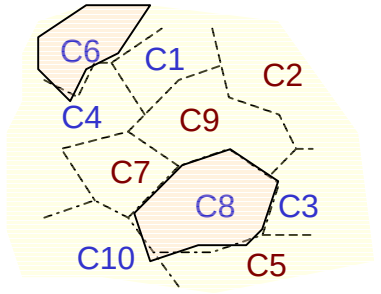
Analyze frames:
Windowing, endpoint detection, silence removal.

Frames are 30ms long, overlapping by 10ms.

Extract features:
12 dimensional vectors of cepstral coefficients.
Intuitively, these features model the vocal tract.

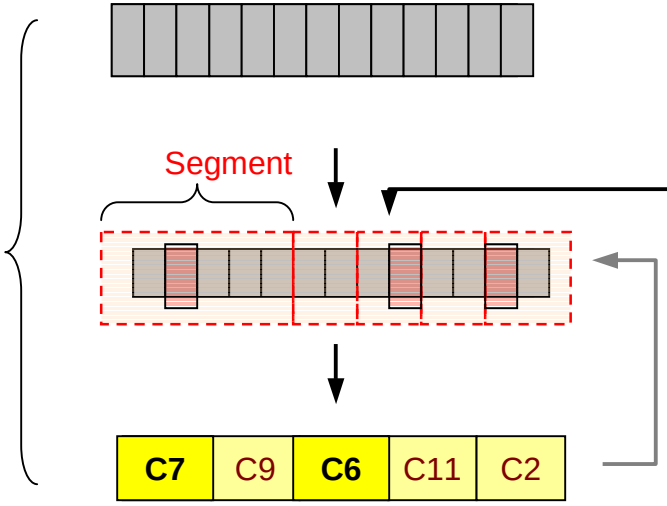






Acoustic Model

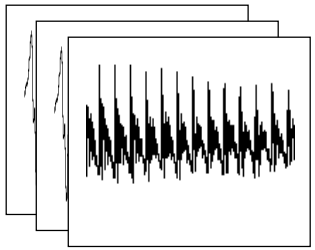
Speaker independent/text independent

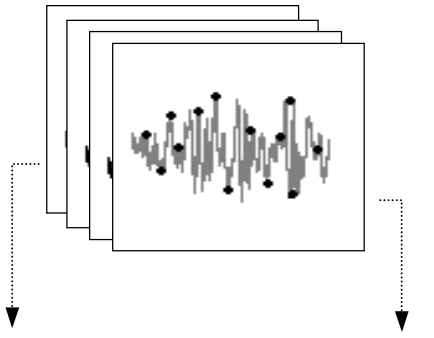


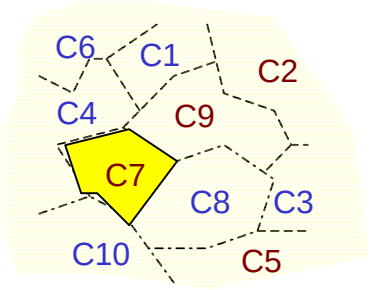
Segment frames into component sounds using Viterbi decoding

Chosen centroids from acoustic space

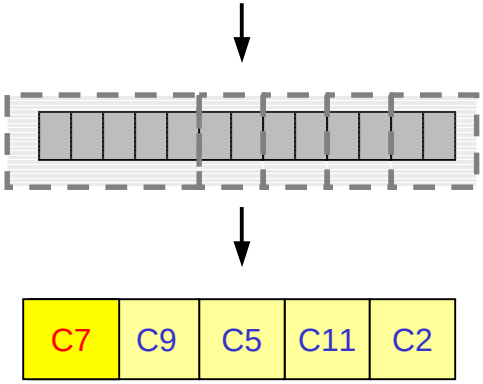
The segmentation technique we apply is similar to Segmental Vector Quantization where segments are mapped using maximum likelihood.

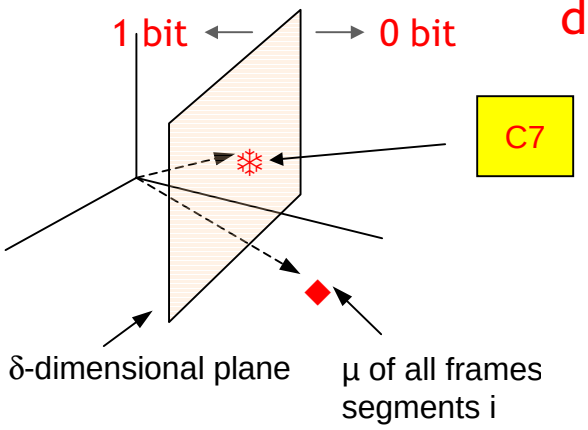






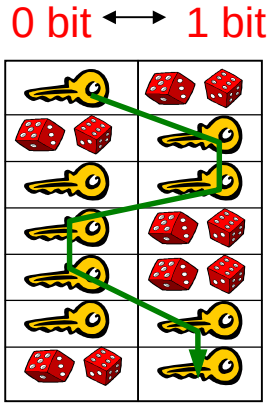
Generate feature descriptor





0 1 1 0 0 0 1 1 1 0 1 0 1

Use descriptor to index into table



Reconstruct & test key

Tradeoffs

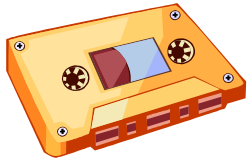
Reliability ?

How many attempts does a legitimate user need in order to successfully regenerate his key?

Security?

How difficult is it for an attacker to impersonate a user?

Two types of attacks:

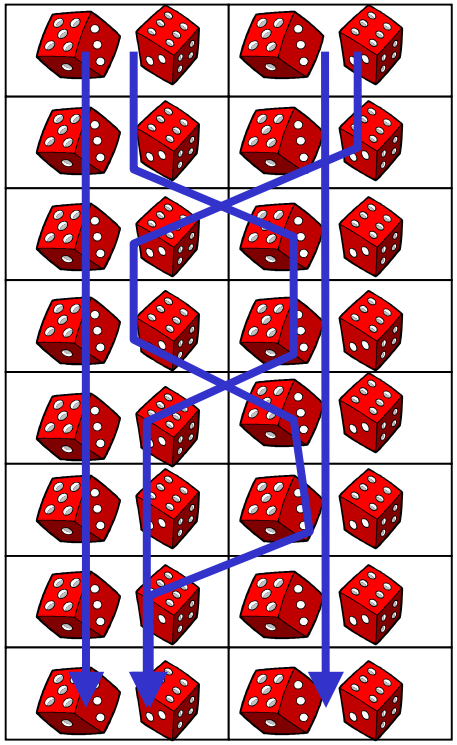


Online attacker:
captures voice, and replays

Offline attacker:
steals device, attempts to recover key @home

Offline attack

What does the attacker do?



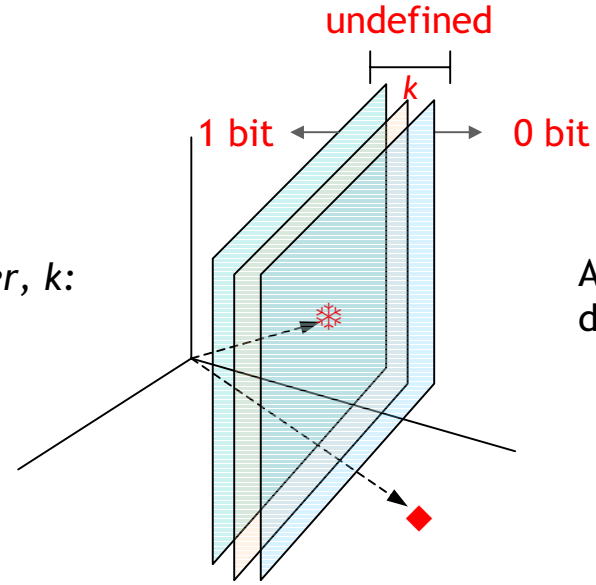
Resident Hacker

Can the attacker efficiently recover key in less than 2^m tries? . . . It depends.

Empirical Results

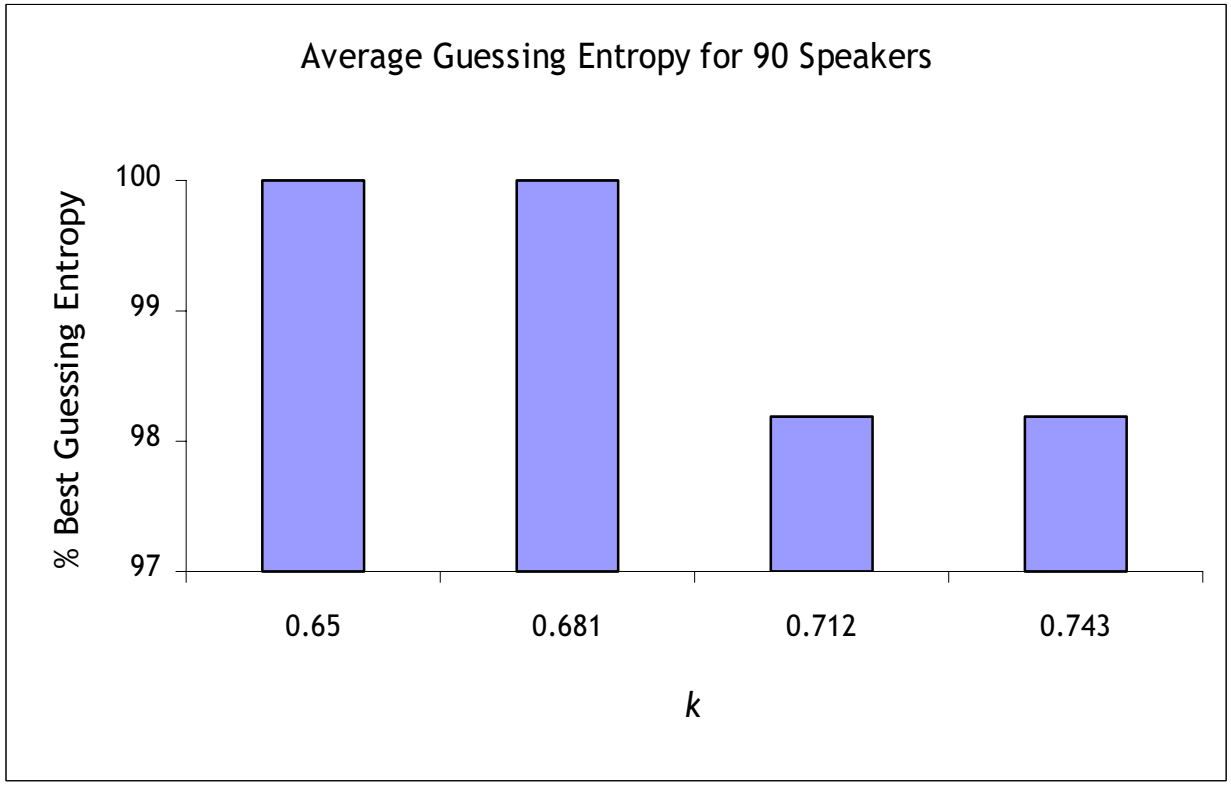
- We used a pre-existing AT&T dataset. 5 samples were collected over the phone from 90 speakers of both sexes, and of varied nationalities, saying a phone number.
- *Guessing Entropy*: The expected number of feature descriptors the attacker must examine before finding the key.
- *False Negatives*: For each user, 4 samples used for training, 1 for testing. Results averaged across all users.
- *For Positives*: For each speaker, 20 other speakers were chosen at random, each saying i) the same 800 number as true speaker and ii) a different 800 number as true speaker.

Sensitivity parameter, k:



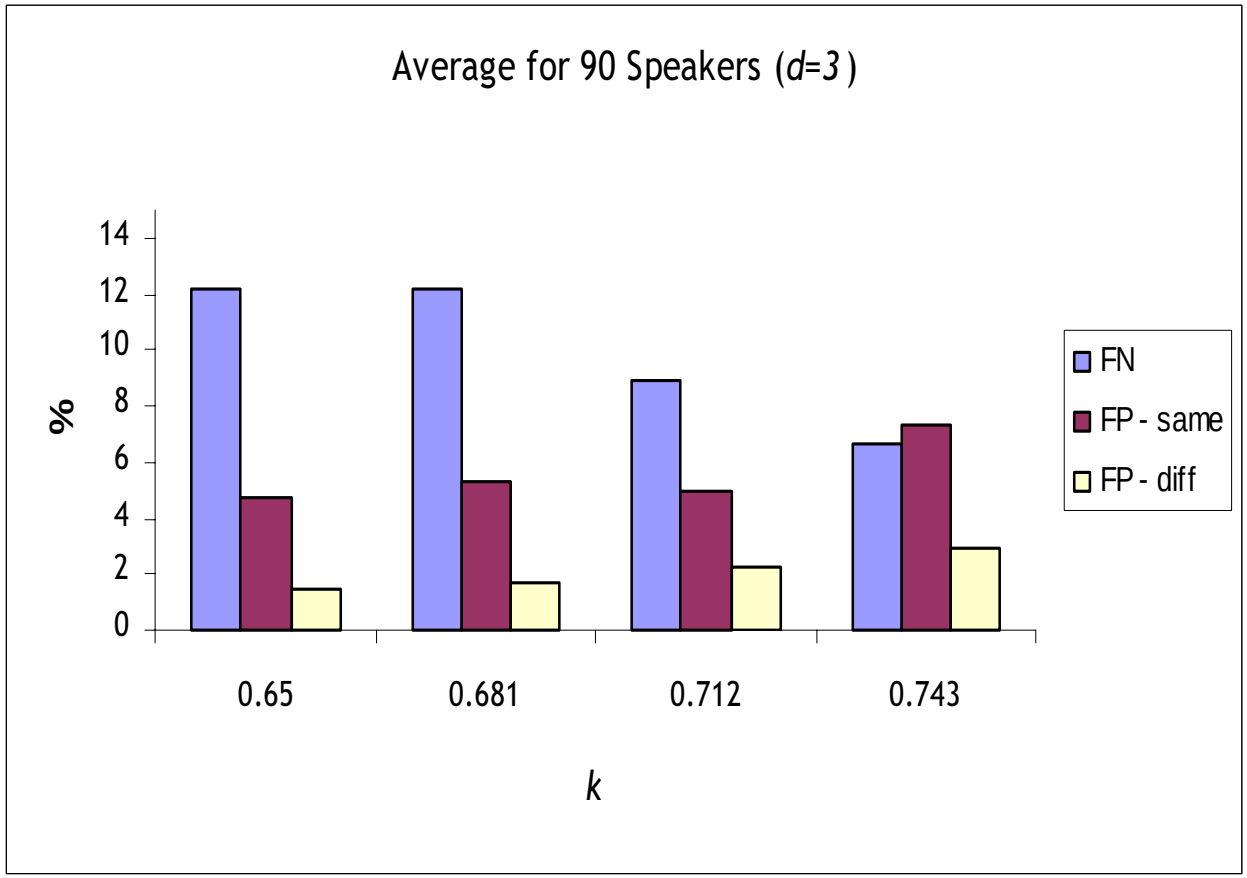
As *k* increases, the number of distinguishing features decreases

Guessing entropy



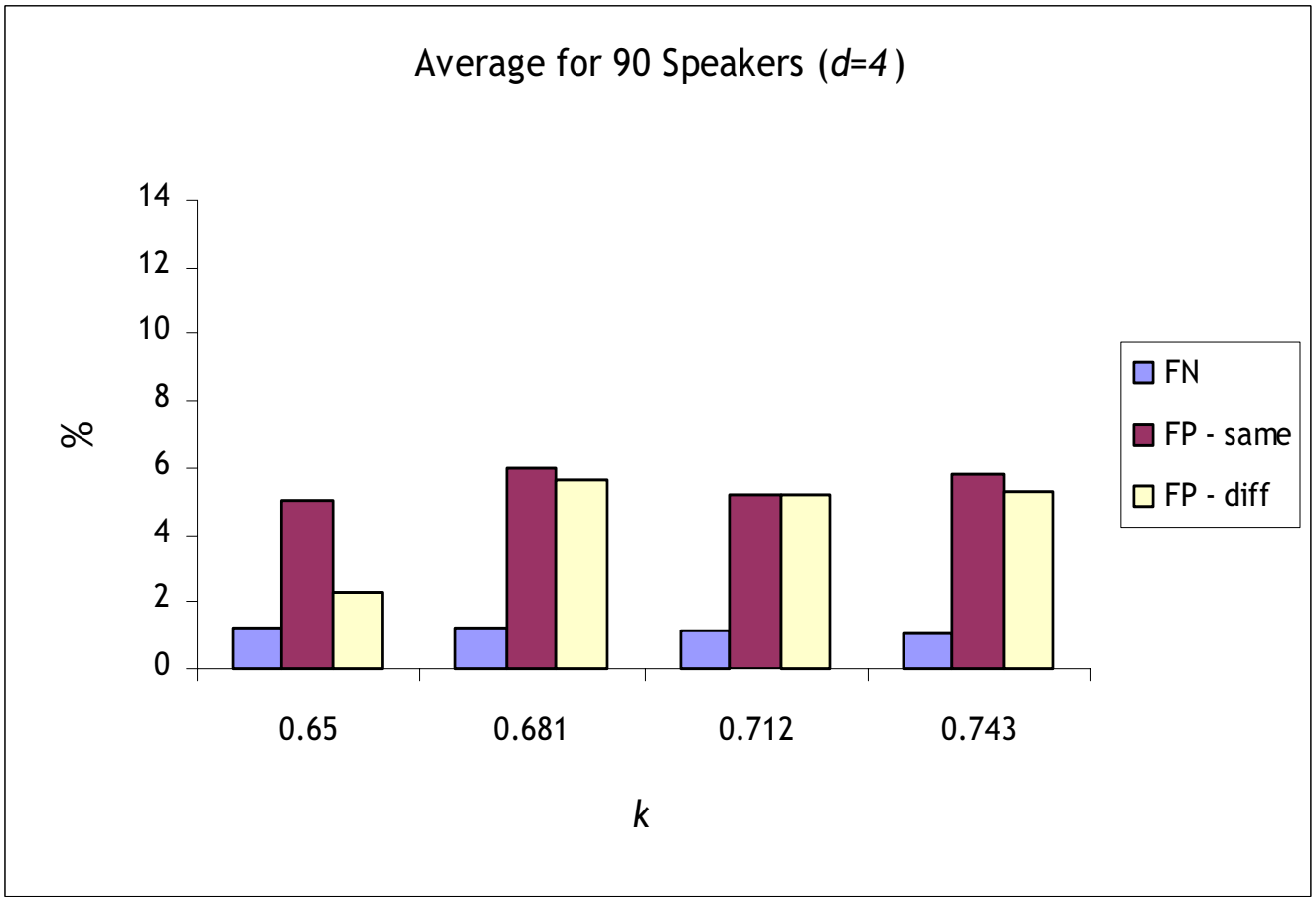
Guessing entropy for feature descriptors of 46 bits. Due to computational complexity we analyzed windows of 10 users at a time, and averaged results across all 90 users.

3 error corrections



- K = 0.712 – False negatives under 9%, false positives (same password) under 6%, and false positives of 2% (different password), while attaining 98% of best guessing entropy.

4 error corrections



- K = 0.65 – False negatives under 2%, false positives (different password) under 2%, while attaining 100% of the best guessing entropy.

Wrapping up

- Showed how to reliably generate at least 46 bits from 2 seconds of voice. Achieved respectable FP/FN rates.

- Current research:
  - Original secret sharing schemes too inefficient for target devices (Psion Netbook, Nokia 9210). More efficient key reconstruction schemes are being implemented/analyzed (Daniel Bleichenbacher).
  - Error correction (tough). Given 60 bit feature descriptors we need to correct for up to 4/5 errors. Can we apply traditional error correcting techniques? How efficiently can we do error correction?
  - Larger key sizes (60+ bits). Explore additional datasets.
  - See http://www.bell-labs.com/user/fabian for more info.