Rethinking Authentication

Steven M. Bellovin
https://www.cs.columbia.edu/~smb

Why Authenticate?

- Many privileges—i.e., access to assorted resources—are based on identity
- Mere assertion of identity is, of course, inadequate
- So—how should we authenticate?

The Usual Trilogy

- Something you know
- Something you have
- Something you are

The Usual Trilogy?

- Something you know
- Something you have
- Something you are

Generally the wrong way to look at it…

*These are not **fundamental** properties!*

Assertions

- Authentication is a security issue
- Authentication is a systems issue
- People are part of the authentication system

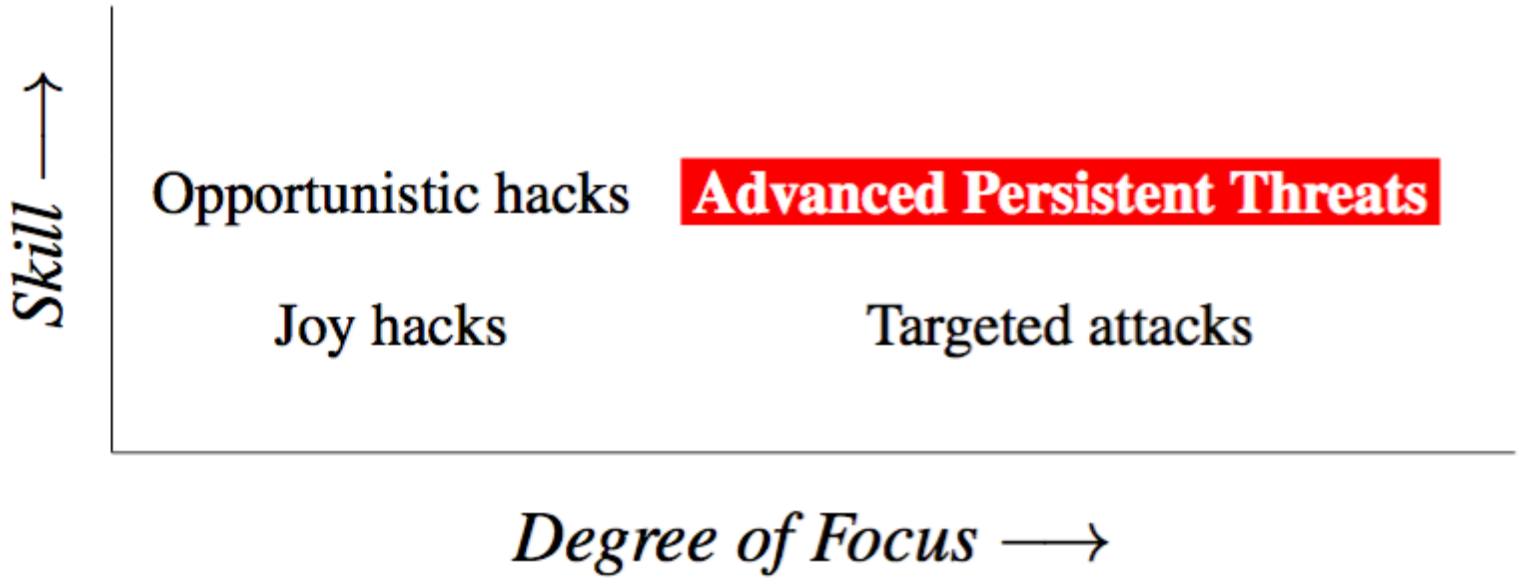
A correct solution must incorporate all of these aspects.

Security Issues

Adversaries

- What are the adversaries' powers?
- What are their goals?
- *Who* isn't important, except as it provides clues to powers and goals.

Classifying the Adversary



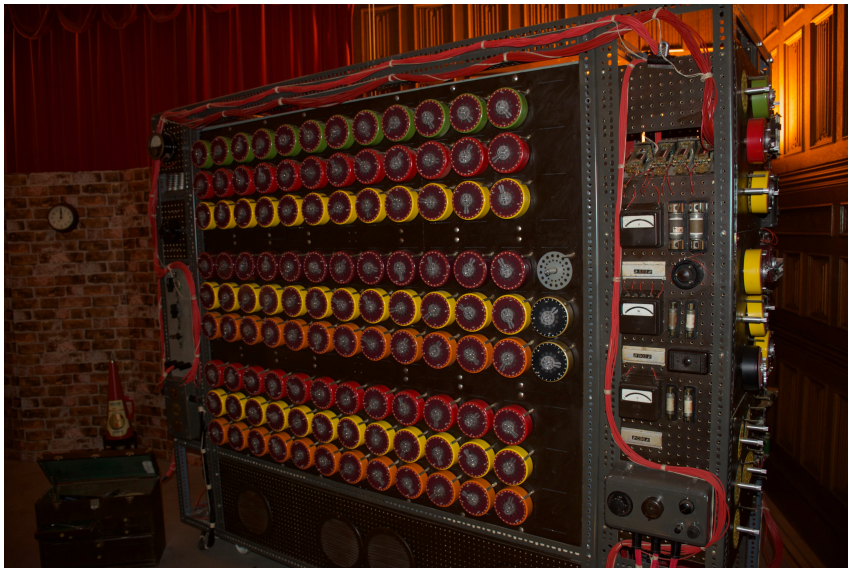
Skill

- Skill is partly pure technical ability
- It's also a measure of resources being expended: will your attacker do what's needed to acquire or buy sufficient capabilities?
  - There's a lot of information available online
- The resources needed for high-grade hacking are much less than those needed for, say, nuclear weapons development

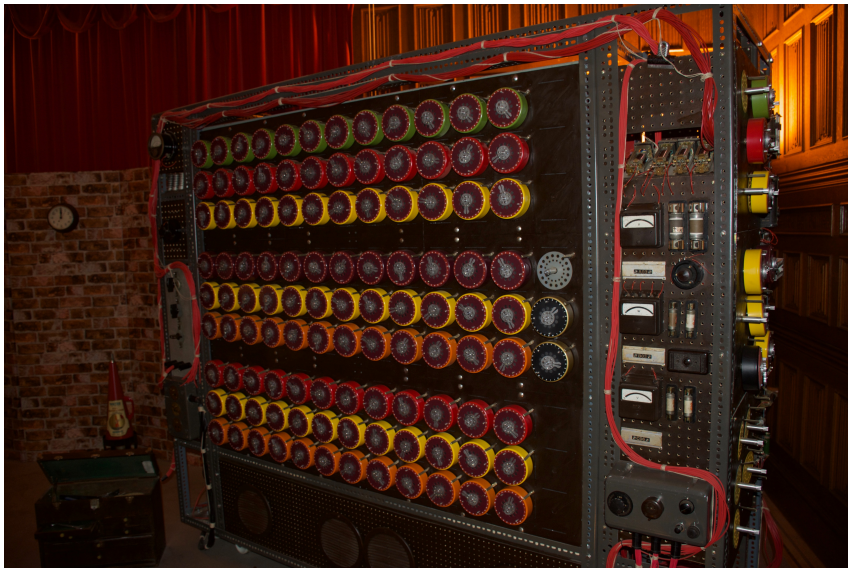
Powers

- Can the attacker tap links?
  - Your LAN?
    - A hotspot used by your users or employees? Their home networks?
  - Your access link?
  - The Internet backbone?
- The "Three Bs": Blackmail, burglary, and bribery?
- Subvert an insider?
- Cryptanalytic skills and computational resources?

Cryptanalysis



Cryptanalysis



Outrunning the Bear

- Does the attacker want to penetrate your system or some random system?
- Does the attacker want a particular user's identity or some random user's identity?
- For opportunistic attackers, you just have to be better than other possible targets
- "Targetiers" can do more

Targetiers

- Background research
  - What is *your* mother's maiden name?
  - Who might send *you* an email?
- Obtain and exploit physical proximity
  - Obviously, this is a question of resources, too
- May be a disgruntled insider or ex-insider
- Obtain fingerprints, photos, even iris close-ups

System Issues

What Can Be Compromised?

- A user's machine?
- A user?
- An authentication server?
- A hardware security module?

Lost Credentials

- Users *will* lose credentials
  - Yes, even biometrics
- How do you recover?
- How serious is a compromise?
- How serious is denying access?

Secondary Authentication

- Mother's maiden name?
- Employee ID number?
- Password reset mail?
- Physical mail?

Secondary Authentication

- Mother's maiden name?
  - Often learnable from (online!) public records
- Employee ID number?
  - What of insider attacks?
- Password reset mail?
  - Is the email account secure (enough)?
- Physical mail?
  - Is the user's mailbox staked out? (Powerful targetiers!)

People

People

- People are part of the system, too
  - Users
  - Your employees
- Systems must be designed for human behavior
  - Homo sapiens 2.0 isn't even in beta trials yet
- (If you must, think of humans as external systems with an odd API—but *think* about the humans who will use your system!)

Exception Handling

- People *will* forget passwords and PINs
- People *will* lose devices
- Biometrics can change because of illness, injury, or simply the passage of time

Any real-world authentication system must be designed to handle such situations properly

Trickery

- Attackers will often try to trick people into doing the wrong thing
  - Phishing, spear-phishing
- This applies to employees, too
- Exception-handling—by definition a rate situation that people aren't accustomed to—is a very fruitful place for attackers to engage in social engineering

Authentication Systems

Authentication Types

- Passwords
- Challenge/response
  - SMS-based
- Time-based
- Cryptographic
- Biometric
- Federated

Password Advice

- Pick a strong password
  - At least 17 characters, including five letters, three special characters, two emojis, two characters from a non-Latin alphabet, and one from a non-human alphabet
- Never reuse it
- Never write it down

Ancient Advice!

- This advice dates to 1978!
- No local storage
- No local computing capability
- A power user might have three logins and passwords

Not much of that is true today...

(Photo courtesy Perry Metzger)

Strong Passwords?

- Useful against password-guessing attacks
- Online? Perhaps rate-limit instead?
- Offline? That follows a server compromise.
  - That's also why we salt and hash passwords
- Useless against phishing attacks or keystroke loggers
- If we can protect the server, do users need strong passwords?

Challenge/Response and Cryptographic Authentication

- Server sends *x*; user returns *f(x)*
- If there's a separate communication channel, *f(x)* can simply be *x*
- If there isn't, the user needs a key *k* and local computing power to calculate *f(k, x)*

Attacking Challenge/Response

- Subvert the communications channel
- Trick the user into sending *f(x)* to the wrong place
- Steal *k*
  - Can be stolen from the user or the server

Note well: these failure modes are very similar to those for passwords—and k isn't hashed!

Subverting the Channel

- The easiest challenge/response mechanism is SMS messages
- But—SMS routing is controlled by the Signaling System 7 network—and it's easy to become an SS7 speaker
- No longer just for governments!

Cryptographic Authentication: Symmetric Algorithm

- *f(k, x)* is the encryption of *x* with some per-user key *k*
  - May be done by the user with an external device
- Server does the same calculation
- User can return only a few bits of *f(k, x)*
- Note well: the server must know *k*, which exposes *k* to theft if the server is hacked

Cryptographic Authentication: Asymmetric Algorithm

- *f(k, x)* is the signature of *x* with some per-user private key *k* (or, more likely, the signature of *H(x)* for some hash function)
- Server verifies the signature
- The server does not know *k*
- User must return all of *f(k, x)*—that's too much to type, which means authentication is device-to-device, not user-to-device

Time-Based Authentication

(Photo by Alexander Klink, via Wikimedia Commons)

- Device contains a clock *t* and a secret key *k*
- The screen displays *F(k, t)*, truncated to six digits
- The server contains a file of per-user *k*

Attacking Time-Based Authentication

- Steal the key file from the server
- Spoof the server's idea of the time of day
- Where does the key file come from? Keys are manufactured into the devices; evidence suggests that they're either generated algorithmically or are retained by the vendor.

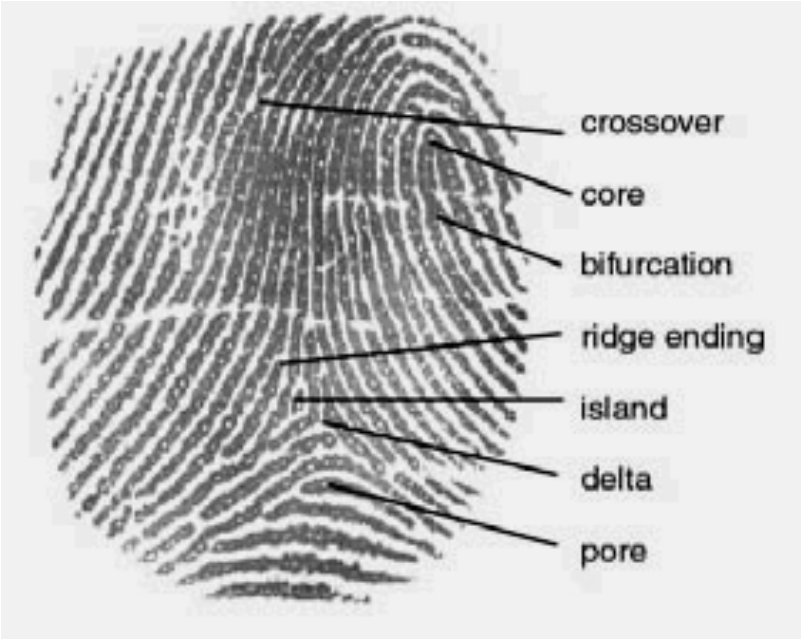
Biometrics

- Seems perfect—can't be forged, can't be guessed, can't be forgotten
- What could possibly go wrong….?

Limitations

- Forging biometrics has often proved possible
  - People leave fingerprints behind everywhere!
  - Mass market fingerprint readers only look at part of the finger
- Body parts can be injured
- Some people don't have usable fingerprints
- You can change your password, but you can't change your iris scan very often…

Stealing Fingerprints



- Biometric recognition is not done by image matching
- Rather, *templates* are stored for key features
- If a server is hacked and a template is stolen, can you make a fake fingerprint to match? Perhaps…

Federated Authentication

- Trust some external party, especially Facebook or Google
- You inherit the weaknesses of the underlying service's authentication mechanism—but they may be better at protecting their servers
- You also have to trust that party
- There may be privacy issues

Attacks

Scenarios

- Let's look at some failure scenarios and see how various mechanisms fare
- No method is perfect
- And: it turns out there are remarkably similar failure modes between the different authentication methods

Guessing

- Obviously, passwords are vulnerable, which is why sites say "pick strong passwords"
- But—is guessing online or offline?
- If online, you can rate-limiting guesses
- If offline, the attack can only occur if the server has been hacked

Password guessing is only a major problem after a server compromise!

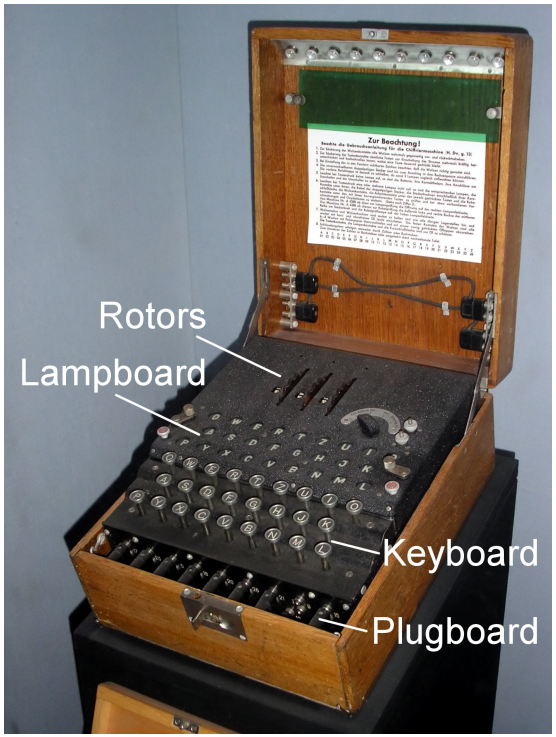
Server Compromise

- Challenge/response—if cryptographic, the attacker gets everyone's keys, and not just hashed passwords
- Biometric templates can often be reversed
- The same is true for time-based authentication and some cryptographic authentication
- Cryptographic authentication might be safe, but *only* if asymmetric crypto is used

Wiretapping

- Is wiretapping a risk in your environment?
- Tapping a LAN is trivial. Tapping Internet access links or the backbone is hard—except for governments that can compel ISP cooperation.
- Some governments, of course, can tap undersea fibers…
- There are also ways to misuse BGP to affect Internet routing

Defending Against Wiretapping



(Karsten Sperling, via Wikimedia Commons)

- Encryption is the obvious defense against wiretapping
- However—many encryption mechanisms have very poor human factors; people don't notice if it's on or if the other endpoint is correct
- That check *must* be automatic
- That it isn't is one reason phishing works

Phishing

- Passwords are obviously vulnerable
- Most other methods are also vulnerable to active attacks—capture the credential and (ab)use it in real-time
  - For SMS-based challenge/response, use forged SS7 messages to redirect the challenge
- The attacker doesn't obtain the actual, reusable credential—but in some situations, one access is enough
- Defense: user's device *must* verify remote site's identity—but that protects all methods

Lost or Forgotten Credentials

- Passwords can be forgotten. Most other methods require hardware, which can be lost. Now what?
- Lost hardware is often abusable by a thief.
- If the device isn't tamper-resistant, the credential can be extracted.
  - Phones? They may be unlockable by spoofed fingerprints.
- Regardless, you have to recover

Secondary Authentication

- Secondary authentication is generally *much* weaker—the data is readily available
  - Mother's maiden name was used at least as early as 1882
- If there's a human in the loop, e.g., your help desk, that person can be tricked
  - Btw—note that CallerID on phones can be spoofed
- But you have to have *some* way to recover!

Temporary Access

- It's easy to reset a password
- For lost devices, perhaps provide a temporary password access mechanism—but that will have some of the same problems as regular passwords
  - Temporary or reset passwords are often provided via email—which means that the user's email credential is their most valuable. Is *it* secure?
- How do you reset a fingerprint?

Summary

Summary

- There are *no* perfect authentication methods
- Two-factor authentication is often stronger not because of the other mechanism's strength but because its failure modes are somewhat different
- Server compromise, phishing resistance, and recovery are *very* difficult

Risk Factors

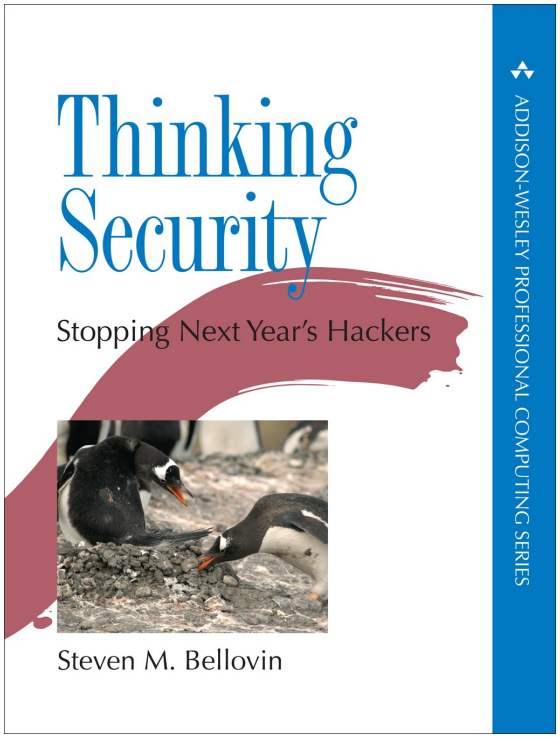
	Guessing	Forgetting	Device loss	Server file compromise	Temp access	External trust
Passwords	✗	✗	✓	✗	✓	✓
Chall/resp	✓	✓	✗	✗✗	✗	✓
SMS	✓	✓	?	✓	✗	?
Time-based	✓	✓	✗	✗✗	?	✗
Crypto	✓	✓	?	✗, ✓	?	✓
Biometric	✓	✓	?	✗	✗	✓
Federated	?	?	✓	✓	?	✗

✓	No particular problem; strength of this mechanism
?	Some trouble or implementation-dependent
✗	Significant risk
✗✗	Very serious risk

What to Do?

- Asymmetric cryptographic authentication is generally the strongest scheme
- However, it requires trusted hardware for people can't type a long signature
  - This hardware can also verify the remote site's identity
- This rules out use from other folks' devices—though that's probably a good thing
- You still have to solve the recovery problem

Disclaimer



Much of the material in this talk comes from Chapter 7 of *Thinking Security*.