Handling Long-Term Keys

- Where do cryptographic keys come from?
- How should they be handled?
- What are the risks?
- As always, there are tradeoffs
Public/Private Keys

- Who generates the private key for a certificate?
- The server may have better random number generators
- Only the client needs the key
- (Does the corporation need a copy of the key?)
- If the server generates the key, how does it get to the client securely?
- (How does the public key get to the CA securely?)
Secret Keys

- Who generates secret keys?
- The problem is harder — both parties need to know them
- Again, how are they communicated securely?
Communication Options

- Channel authenticated by other means
- Public-key protected channel
- Hard-wired contact
- Out-of-band communications
- Note: process matters!
Out-of-Band Communications

- Telephone
- SMS text message
- Postal mail
What are the Enemy’s Powers?

- Steal letters from a mailbox?
- Fake CallerID with an Asterisk PBX?
- Burglary, bribery, blackmail?
Tamper Resistance

- Keys are safer in tamper-resistant containers — they can’t be stolen
- See “the three Bs” above
- Note well: tamper-*resistant*, not tamper-*proof*
- The availability of tamper-resistant hardware changes the tradeoffs
Online vs. Offline

- Does the key generator need to be online?
- A CA can be offline, and accept public keys via, say, CD
- That may be riskier than having it generate the private key — what if there’s a buffer overflow in the read routine?
- For secret keys, the server can’t be offline; rather, some copy of the key has to be online, to use it
Putting it All Together

- Let’s look at some relatively simple privileged programs
- How do they combine the different mechanisms we’ve seen?
- What are the threats? The defenses?
The “Passwd” Command

- Permits users to change their own passwords
- In other words, controls system access
- Very security-sensitive!
- How does it work?
Necessary Files

- `/etc/passwd` — must be world-readable, for historical reasons
  - Maps numeric UID to/from username
- Historical format:
  
  `root:8.KxUJ8mGHCwq:0:0:Root:/root:/bin/sh`
- Fields: username, hashed password, numeric uid, numeric gid, name, home directory, shell
- Numeric uid/gid is what is stored for files
- Password is two bytes of salt, 11 bytes of encryption output
- Encoded in base 64 format: A-Za-z0-9./
Storing the Hashed Password

- Better not make it world-readable
- Store in a *shadow password* file
- That file can be read-protected
File Permissions

$ ls -l /etc/passwd /etc/shadow
-rw-r--r-- 1 root root 671 Oct 3 10:42 /etc/passwd
-r-------- 1 root root 312 Oct 3 10:42 /etc/shadow
Must Be Owned by Root!

- Ownership of that file is equivalent to root permissions
- Anyone who can rewrite it can give themselves root permissions
- Cannot use lesser permissions
- Note: adding a line to that file (often with a text editor) is the first step in adding a user login to the system
Implications of the Numeric UID/GUID

- Assigning a UID to a username grants access to that UID’s files
- In other words, anyone with write permission on `/etc/passwd` has access to all files on the system
- Consequence: even if we changed the kernel so that root didn’t have direct access to all files, this mechanism provides indirect access to all files
- Conclusion: Cannot give root control over UID assignment on secure systems
What Else Shouldn’t Root Be Able to Change?

- The user’s password!
- Attack: change the user’s password to something you know
- Windows XP does not give Administrator either of these powers
The Passwd Command

- Clearly, must be setUID to root
- Must be carefully written...
Authenticating the User

- Passwd program has real UID
- Demand old password — why?
  Guard against someone doing permanent damage with minimal access
- Root can change other user’s passwords
Where Does the Salt Come From?

- Passwd command generates random number
- Need this be true-random?
- No — “probably different” will suffice.
- Seed ordinary pseudo-random number generator with time and PID
Restricting Access

• Suppose only a few people were allowed to change their own passwords

• Take away other-execute permission; put those people in the same group as “passwd”
Front Ends

- What about the help desk, for forgotten passwords?
- Have a setUID root front end that invokes passwd
- Validate: make sure they can only change certain users’ passwords
- Log it! (Much more later in the semester on logging)
Making a Temporary Copy

• Must copy password file to temporary location and back to change a password
• Watch out for race condition attacks!
• Actual solution: put temporary file in `/etc` instead of `/tmp`; avoid whole problem
• Secondary benefit: use temporary file as lock file, and as recovery location in case of crash
Update in Place

- Password changes could overwrite the file in place
- Doesn’t work for use add/delete or name change
- Still need locking
Passwords on the Command Line?

- Bad idea — `ps` shows it
- Bad idea — may be in shell history file
  ```
  $ history 12
  12  date
  13  man setuid
  14  ls -l 'tty'
  ```
- Your terminal isn’t readable by others:
  ```
  $ ls -l `tty`
  crw--w----  1 smb  tty  136,  5 Oct 26 14:24 /dev/pts/5
  ```
Changing Your Name

• Chsh is like passwd, but it lets you change other fields

• Ordinary users can change shell and human-readable name; root can change other fields

• *Much* more dangerous than passwd
Input Filtering

• What if user supplies new shell or name with embedded colons? Embedded newlines? Both?

• Could create fake entries!

• Must filter for such things
Features Used

- Access control
- Locking/race prevention
- Authentication
- Privilege (setUID)
- Filtering
Security Analysis: Internet Thermostats

- I recently decided to investigate Internet thermostats
- Control and monitor my house temperature remotely
- Are there security risks?
One Popular Brand

- Thermostats have built-in web servers
- Simplest mode: direct connection to thermostat
- Alternate mode: thermostat and user connect to company’s web site; company can generate alert emails
What’s at Risk?

• Turning off someone’s heat in the middle of winter?
• Turning on the heat in the summer?
• Run heat and air conditioning simultaneously?
Local Management
Local Problems

- No https — people can eavesdrop
- Uses “Basic Authentication”:
  
  “The most serious flaw in Basic authentication is that it results in the essentially cleartext transmission of the user’s password over the physical network . . .

  “Because Basic authentication involves the cleartext transmission of passwords it SHOULD NOT be used (without enhancements) to protect sensitive or valuable information.”

- No read-only mode
Remote Management

![Thermostat Management Interface]

- **Temperature**
  - Zone Temperature: 70.0°F
  - Local: 70.0°F
  - Override: 70.0°F
- **Cool Setting**: 78.0°F
- **Heat Setting**: 66.0°F
- **Hold Mode**: Off
- **Schedule Settings**
  - Day Class / Period: Out / Day
- **HVAC Settings**
  - HVAC State: Off
  - HVAC Mode: Auto
  - Fan Mode: Auto
- **Alarm Status**
  - Low Temperature: OK
  - High Temperature: OK
  - Filter change: OK

Remote Management System

End of Document
Remote Problems

- Https — but only to the server
- Unencrypted traffic from the server to the thermostats
- Passwords are sent in the clear across the Internet
- Passwords are stored in bulk on the server
Privacy Issues

- Energy consumption patterns
- Al Gore’s thermostat setting? Japanese office thermostat settings?
- Vacation schedules (burglary risk?)
Defenses

- Can’t touch thermostat software
- Add layering — access controls on top of built-in controls
- Use crypto tunnels
- Filter setting change requests
Last-Ditch Defenses

- Add a low-limit heat switch in parallel
- Add a high-limit heat switch in series
- These are hardware devices, not software
- Protect against bugs
- What if they fail?
- Independent failure modes; protect against each other
How to Analyze This?

- Hard to \textit{know} all the threats
- Approach: see what is made available, and ask who might want it
- Reason by analogy and effect
- Check the “gold standard” (Au): \textbf{Authentication}, \textbf{Authorization}, \textbf{Audit}