Security Policy, Risk Analysis, and Safeguard Design (An Example)

Michael P. Zeleznik, Ph.D.

This example demonstrates one aspect of a much larger process of developing a security architecture for a commercial product. For an arbitrary company "CORP" that sells software/hardware that works with client data, this example demonstrates how to (1) define a security policy, (2) perform a qualitative risk analysis that uncovers vulnerabilities, and (3) design and implement a set of safeguards to address the vulnerabilities as required to enforce the security policy. The approach shown here allows one to achieve reasonable initial goals with minimal effort, and to then build upon this in the future. One must keep this process as simple as possible, since it can quickly become unwieldy.

1 Risk Analysis

This will employ a *qualitative*, not quantitative risk analysis; risk probabilities, costs, and losses will not be assigned. That can be added later, if required. The terminology and approaches vary widely in the security field and in implementations. Following are our definitions.

Risk Analysis Term	*Our Definition*
Assets	Our security relevant resources. These are the only things that the security policy deals with.
Security Policy	What we need to enforce regarding our assets.
Threats	Things that can compromise (violate) the security policy.
Vulnerabilities	Specific ways in which threats can be realized.
Distinction between Threat & Vulnerability	A threat is the causal agent, whereas a vulnerability is a specific mechanism by which this agent occurs. For example, the threat of "unauthorized destruction of data" can be realized by vulnerabilities such as "attack by disgruntled employee", "disk failure", "fire", or "flood".
Risks	Probability of threat occurring via vulnerability. N/A since this is not a quantitative analysis.
Costs	The cost of loss associated with each realized threat. N/A since this is not a quantitative analysis.
Losses	Cost*Risk for each realized threat. N/A since this is not a quantitative analysis.
Safeguards	Mechanisms that reduce likelihood of vulnerabilities.

2 Acronyms and Conventions

Acronym	*Our Meaning*
AI	Authentication Information
ACA	Authorized CORP Agent(s)
AS	Authorized Subject
LAI	Login Authentication Information
SA	Superuser Account
SAFE	Safeguard
UA	User Account
UES	Unix Enhanced Security
US	Unauthorized Subject
VUL	Vulnerability

The term **subject** will generally imply a person, but it can also imply a computer process.

3 Assets

Primary Assets
- Client data

Secondary Assets (assets only as far as they impact primary assets)
- CORP software
- Authorized CORP agents (CORP personnel, designated clients, etc.)
- Authorized subjects
- Operating system (security relevant parts)
- Computer hardware (containing client data in memory)
- Disks (containing client data on disk)
- Removable media (MODs, tapes, floppies)
- External network and connection

4 Security Policy

Goal:

1. To protect the privacy and integrity of client data, and to control the ability to run and modify CORP software.

Policy:

1. There are only 2 types of accounts on a CORP machine: User and Superuser.

2. The granularity of accountability is limited to these two types of accounts, and does not extend to specific subjects.

3. A subject who can "access an account" will receive all privileges that accompany that account.

4. A subject is able to "access an account" only by possessing Login Authentication Information (LAI) for that account (e.g., a username/password pair).

5. A subject becomes authorized to "access an account" when provided with LAI for that account by an Authorized CORP Agent (ACA), and is then referred to as an Authorized Subject (AS).

6. A subject that is neither an ACA nor an AS is an Unauthorized Subject (US).

7. ACAs are trusted by CORP to distribute LAI only to subjects who are trusted with the privileges that accompany that account.

8. Only ACAs are allowed to share LAI with other subjects.

9. A Superuser Account (SA) can do anything, including circumventing or changing the security policy implementation.

10. A User Account (UA) can read or modify client data, machine data, and CORP software, can run CORP software, can change their password, and can create/modify files/processes under their ownership. A User Account cannot modify the security policy implementation.

11. Issues not directly addressed in this policy are not our concern (e.g., denial of service attacks).

5 Threats

T:01 US can login to valid User or Superuser account.

T:02 US can create valid User or Superuser account.

T:03 US can access disk data outside of valid accounts.

T:04 US can access memory data outside of valid accounts.

6 Vulnerabilities

For each threat listed above, the vulnerabilities by which it can be realized are indicated by an identifier (e.g., **V:01**) followed by a description. Note that a vulnerability can apply to multiple threats. The numerical IDs of VULs (e.g., **V:13**) are not necessarily sequential since they were added over time while I wanted to keep the concepts grouped.

T:01 can be achieved by:
- **V:01** US gets AI from valid user voluntarily.
- **V:02** US gets AI from valid user accidentally.
- **V:11** US gets AI from valid user by force.
- **V:03** US gets AI by cracking password file.
- **V:04** US gets AI by sniffing network (local or remote).
- **V:12** US gets AI from online or physical storage.
- **V:05** US gets AI by guessing.
- **V:06** US breaks OS security, circumventing account protections.
- **V:07** US exploits net vulnerability (e.g., sendmail, FTP, portmap, NFS).
- **V:08** US boots computer in single user mode.

T:02 can be achieved by:
- **V:07** above
- **V:08** above

T:03 can be achieved by:
- **V:06** above
- **V:07** above
- **V:08** above
- **V:09** US steals disks.
- **V:10** US steals computer.
- **V:13** US steals removable media.

T:04 can be achieved by:
- **V:06** above
- **V:07** above

7 Safeguards

For each vulnerability listed above, the safeguard(s) that address it are indicated by an identifier (e.g., **S:01**) followed by a brief description. Each safeguard is described in later sections. Vulnerabilities are listed in the order presented above. Note that a safeguard can address multiple vulnerabilities. The numerical IDs of SAFEs (e.g., **S:05**) are not necessarily sequential since they were added over time while I wanted to keep the concepts grouped.

V:01 is addressed by:
- **S:01** Only ACA is permitted to disclose AI. AS is trusted to not share AI.

V:02 is addressed by:
- **S:02** AS and ACA must insure password typing is not observed.

V:11 is addressed by:
- *NOT ADDRESSED*

V:03 is addressed by:
S:03 Employ shadow passwords (via UES)

V:04 is addressed by:
S:04 Employ secure remote connections (via ssh) into host.
S:11 Restrict/Control remote connections out of host.

V:12 is addressed by:
S:09 Do not record plaintext passwords where US can get them.

V:05 is addressed by:
S:05 Disallow simple passwords (via UES)

V:06 is addressed by:
S:06 Install OS with base level security enabled.

V:07 is addressed by:
S:07 Disable all unnecessary net services, especially known security holes.

V:08 is addressed by:
S:08 Control physical access to computer; only ACA or AS is allowed alone.

V:09 is addressed by:
S:08 above

V:10 is addressed by:
S:08 above

V:13 is addressed by:
S:10 Control physical access to removable media; only ACA or AS is allowed.

8 Safeguard Implementation

For each safeguard listed above, its identifier and description are restated, followed by the method(s) of implementing it. Note that some implementation methods can apply to multiple safeguards. Methods in italics are further described in the Safeguard Implementation Details section.

Safeguards S:01 S:02 S:08 S:10

S:01 Only ACA is permitted to disclose AI. AS is trusted to not share AI.
S:02 AS and ACA must insure password typing is not observed.
S:08 Control physical access to computer; only ACA or AS is allowed alone.
S:10 Control physical access to removable media; only ACA or AS is allowed.

Make ACA and AS aware of these policies and their importance.
Then we must trust them to follow the rules.

Safeguards S:03 S:05

S:03 Employ shadow passwords (via UES).
S:05 Disallow simple passwords (via UES).

Enable Enhanced Security for passwords.
Change existing passwords.

Safeguard S:04

S:04 Employ secure remote login (via ssh) into host.

Install ssh/scp.
Disable telnet/rlogin/ftp servers and clients.

Safeguard S:11

S:11 Restrict/Control remote connections out of host.

If AS can telnet/rlogin/ftp/Netscape out of our host, plaintext passwords can be exposed on net. If these passwords are the same as on our host, password sniffing vulnerability exists. This is difficult to guard against.

Option 1: *Disable telnet/rlogin/ftp clients.* This stops casual user, but any user can drop in their own client and run it. Further, if we provide a web browser for CORP software, then they can use that to authenticate with external sites via non-secure connections.

Option 2: Instruct AS, if they plan to authenticate with external boxes, to not use the same passwords as on our box (and trust that they do so).

Option 3: Configure sshd to limit access by IP address to only ACA sources. Thus, sniffed passwords could not be used remotely. They would have to login directly to the box. This will become a maintenance headache, and we'd have to open up large subnets to support dynamic IP. Not recommended.

Recommended solution: Option 1 (just move client programs to new names), plus Option 2.

Safeguard S:06

S:06 Install OS with base level security enabled.

Do this at system build/install time, or enable on existing systems.

Safeguard S:07

S:07 Disable all unnecessary net services, especially known security holes.

Stop running routed, gated, bind, nfsd.
Stop running and disallow sendmail.
Disable ALL inetd services.

LIMITATION: Sites that require NFS must leave that enabled. Inform them of need to block NFS at their router/firewall.

Safeguard S:09

S:09 Do not record plaintext passwords where US can get them.

This should be obvious, but can be overlooked. For example, some site passwords might be in intranet web pages. If they need to be there, they should be encrypted with a known password that is not kept online. Similarly, posting passwords on whiteboard or wall is bad if adversaries can view them.

9 Safeguard Implementation Details

Change existing passwords

For each account (as root)
    passwd account
    Enter new password twice
For each account (not as root), verify by
    su account
    Enter new password

Enable Enhanced Security for passwords.

TO BE DONE

Install ssh/scp.

Make tarball containing everything.
Drop in tarball, untar, install.
Move init scripts to /sbin/init areas.

Disable telnet/ftp/rlogin servers and clients.

cd /usr/bin
mv telnet telnet.real
mv rlogin rlogin.real
mv ftp ftp.real
Do the same for server programs…

Stop running routed, gated, bind, nfsd.

Edit /etc/rc.config, adding "no" for these lines.
If NFS is required, leave enabled, and inform client that their network administrator needs to disable NFS mounting at their router or firewall.

Stop running and disallow sendmail.

chmod go-rwx /usr/sbin/sendmail
mv /usr/sbin/sendmail /usr/sbin/sendmail.real

Disable ALL inetd services.

Edit /etc/inted.conf, adding # at front of non-# lines.

10 Known Limitations and Pending Issues

Sites that must run NFS are venerable to those known holes. It is not unreasonable to tell the client to block NFS access at their router or firewalls. We have to live with this solution, or move to a more secure network file system (such as AFS or even newer ones). That would be a big hurdle to test and understand, and then to deploy.

We do not try to stop denial of service attacks. Any user can bring the system down by exhausting resources such as disk space, CPU cycles, number of processes, etc.

There is no mechanism to validate the security architecture at any point in time. Any of our required security mechanisms could be obviated at a later time due to software upgrades, development team debugging, client actions, CORP side changes (e.g., someone adds a password field to the website database), etc. This would require additional tools which would have to be run periodically, and can get very complicated.

Must address local network vulnerabilities (e.g., wireless LAN eavesdropping or unauthorized users with hosts on the wired LAN). Although the wireless issues are more visible, the same issues apply to wired LANs and should be addressed in a unified way.

Add something about limits of CORP responsibility, that security is also the end user's responsibility.