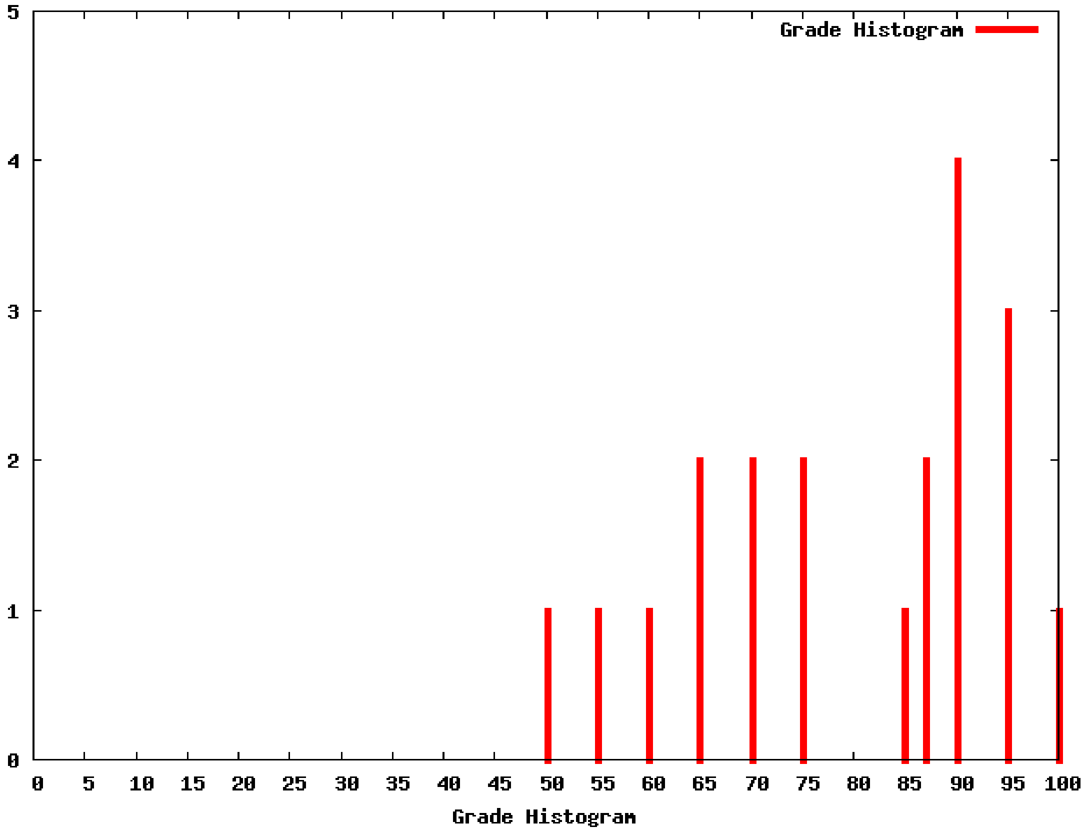
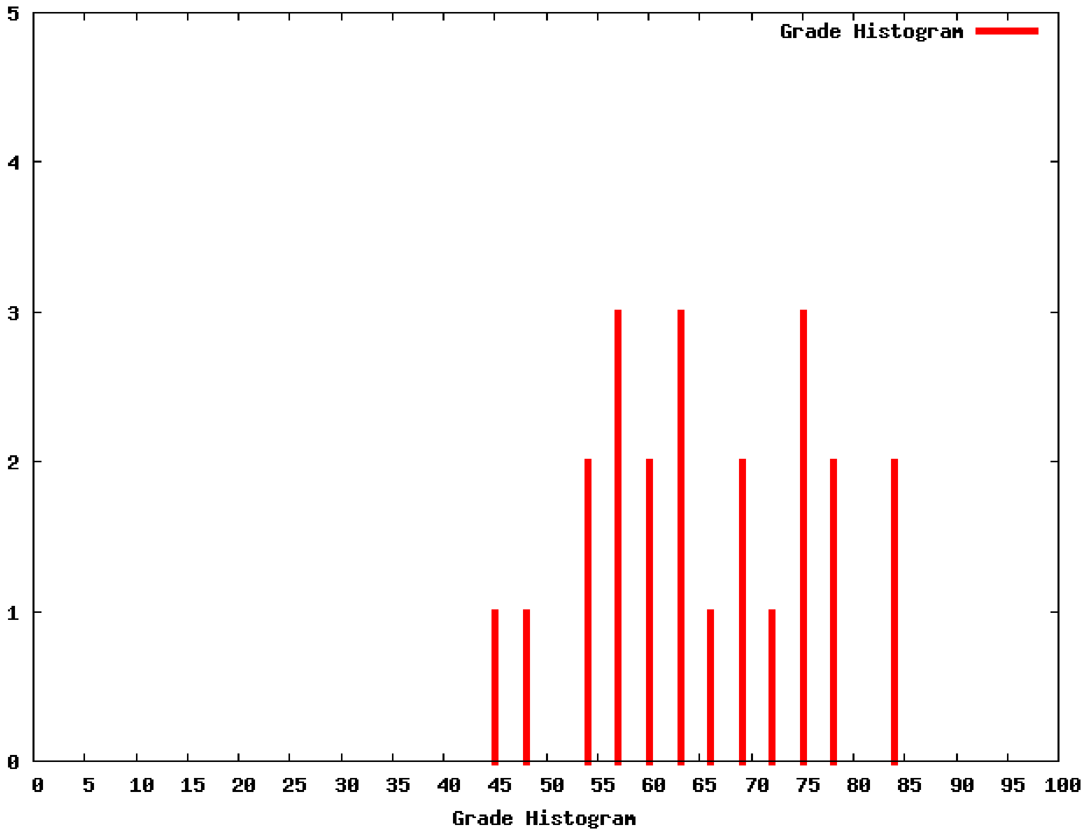
Homework 1


Grade Histogram
5
4
3
2
1
0
0
5
10
15
20
25
30
35
40
45
50
55
60
65
70
75
80
85
90
95
100
Grade Histogram

Midterm



Grade Histogram
5
4
3
2
1
0
0
5
10
15
20
25
30
35
40
45
50
55
60
65
70
75
80
85
90
95
100
Grade Histogram

More Architecture — Email Security

- We want to secure email
- Generally, that requires crypto, which in turn requires protecting keys
- How shall we do that?

Standard Techniques

- Encrypt the private key with a user-typed passphrase
- Use special-purpose crypto hardware
- The latter is rarely available; we need to use the former, at least in some cases

Where are Decryption and Signing Done?

- Gateway machine?
- End-user's machine?

Signing at the Gateway

- Tempting target
- Hard for user to supply the key or the passphrase
- How does the gateway *know* who sent the mail?
- Best for *organizational* signatures

Decrypting at the Gateway

- Again, how are keys supplied?
- When is decryption done?
- Is the mail stored internally in the clear?

Signing Every Message

- Suppose we want to sign every message
- Do we prompt users for a passphrase on each email sent?
- Rather annoying — can we cache passphrases?

(Why Sign Everything?)

- Principle?
- Prevent false attribution?
- Anti-spam?

Caching Keys

- If we cache keys, they’re exposed to bugs in the mailer
- How risky are mailers?
- (How big are they?)

Some Mailer Sizes

Mailer	*KLOC*
Thunderbird	**6000**
Evolution	**2500**
(extras)	**2200**
Claws-Mail	**840**
Pine	**530**
Mutt	**288**

Numbers are *very* imprecise. All of these mailers require many libraries.

(Why are Mailers So Big?)

- Mail formats are complex
  - MIME
  - Multilingual
  - GUIs
- HTML rendering
- Other stuff bundled in (calendar, vCard, etc)
- Frequently include an editor

Why are Mailers Insecure?

- Size
- Accept untrusted input
- Plenty of room for user error

Entrust our Keys to Mailers?

- They're big and complicated
- They interact with lots of other programs
- They have long histories of security problems
- Handing them keys doesn't sound like a great idea. . .

Outboard Key Manager

- Should we have a separate application to handle keys?
- How big are such applications?
- Can we trust them?

Key Managers

Component	*KLOC*
GNOME Keyring	**150**
GNOME Keyring Manager	**97**
GPG	**520**
GPG2	**737**
pinentry	**55**

These aren't exactly tiny, either. . .

Bug Rates

- How many bugs per 1,000 lines of code?
- Hard to measure
- Different types of software have different rates
- We can't count bugs that aren't found!
- That said. . .

Component	*Bugs/KLOC*
Linux 2.6 Kernel	**`.17`**
Commercial code	20–30

Managing the Key Manager

- The mailer still tells the key manager what to decrypt or sign
- If the mailer is buggy, it can fool the key manager
- You don’t know what’s *really* being signed or decrypted
- (This all applies to crypto hardware solutions, too)

Pure Outboard Solution?

- Save inbound mail; manually decrypt it
- Edit outbound mail separately; manually sign, then paste that into mailer buffer
- Does this work?

It’s Too Inconvenient

- Most users *won’t* put up with this
- Result: very few signed messages
- Result: reluctance to receive inbound encrypted messages
- Does this give us worse security?

What Do We Do?

- There are no perfect solutions
- How disciplined are the users?
- How important is secure email?
- Can you have separate grades of keys?
- Who is your enemy?

Outboard Keys

- Despite the risks, outboard keys are still better
- Still simpler than the mailer
- Less risk of key theft
- Easier to add (secure) audit trail

Windows Vista and IE 7

- Web browsers have also been problematic
- Internet Explorer has been worse. . .
- IE 7 on Vista is a lot better
- Why?

Protected Mode

- Run web browswer with fewer privileges (exception: trusted sites can have full privileges)
- Compromise of the browser does not result in compromise of (most) user files

Components

- User Account Control (UAC)
- Mandatory Integrity Control (MIC)
- User Interface Privilege Isolation (UIPI)

User Account Control

- Eliminate need to log in as Administrator
- Even Administrator can run most applications without privilege — they changed the privilege requirements for some operations
- Privilege can be raised as needed, with password entry. (Will users make that decision correctly?)

Mandatory Integrity Control

- Low-privilege processes cannot write to protected files
- Available levels: low, medium, high
- Similar to MAC

Bell-Lapdula and MIC

- Recall how Bell-Lapadula confidentiality mechanisms could be used for integrity protection, by reversing labels
- MIC uses half of it: it’s really “no write down”
- MIC does not provide confidentiality protection

Privilege is Inherited

- The privilege level of a process is inherited by its children
- Children spawned by protected mode IE also run at Low privilege
- This blocks attacks by ActiveX, VBScript, etc.

Virtualization

- A lot of existing code wants to write files (cache, temporary files, cookies, history, registry, etc.)
- A shim layer virtualizes these functions
- Files to be modified in Low mode are copied to the Low area; the changes are made only to the copies

Gaining Privilege

- Sometimes, Low processes need to do things requiring privilege
- Special *broker* processes will perform such operations on request
- Brokers ask user consent before proceeding
- Is that reliable?

Trustinging the User?

- Users can be tricked
- Many of today's dialog boxes are useless
- From a W3C glossary Wiki:

> *Dialog box: A window in which resides a button labeled "OK" and a variety of text and other content that users ignore.*

Lack of Confidentiality Protection

- Low mode malware can still read your files
- It appears possible for Low mode applications to export data
- But — full Bell-Lapadula confidentiality control is impractical
- Cookies are a special case — prevent (some) cross-site scripting attacks

User Interface Privilege Isolation

- Prevents Low mode processes for sending certain messages to higher-mode processes
- Blocks “shatter attack” (inject code into another process via Windows messages)
- In essence, ACL for message-passing

What Has Microsoft Done?

- Separated Internet Explorer from Windows Explorer (i.e., restored the distinction between net and desktop)
- Used OS access controls to isolate browser
- Added more access controls
- *Structural separation*

Does it Work?

- Vista was immune to the **`.ani`** file attack
- More precisely, it couldn’t escape the Low mode jail
- Human interface attacks may still be an issue