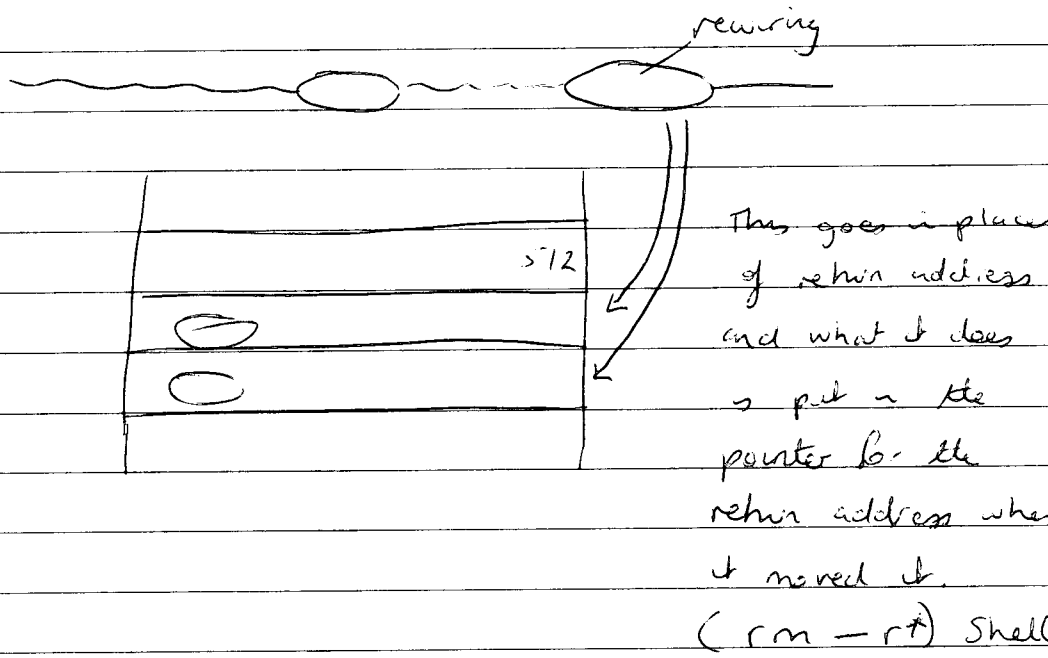
CS4615 Tuesday 7th January 2014

Buffer Overflow.
Less than 1024 but greater than 512.
So if you try and copy 700 bytes then it will copy but when it reaches the 512 it copies over probably every thing else in the stack.

When you finish copying it won't crash, it will crash as soon as you try and go to the return address as it is overwritten.



This is called a stack smashing Attack.

Solution: Balance Checking
Because it overflowed, it corrupted the stack.
This happened because of the dumb mistake of not balance checking.

lib c, was ~~fully~~ full of overflow vulnerabilities. Now there is a new library called SAFE LIB C.

PING OF DEATH

Win 95 was/is vulnerable to this attack.
Send a message from client to server that contains a packet that is larger than 65~~5~~ bytes.

SQL Slammer Worm (Jan 25, 2003), flaw in Microsofts SQL server -- Another example of Stack Smashing. Took only 30 minutes to cause a DoS.

Stack Guard & canary word, if it is being overwritten on the stack, the program will stop as it recognizes it is a Stack Smashing Attack.

Pro Police (IBM option)

You can also change the settings to prevent code from being run ~~of~~ on the stack, downside is that you can't pass programs as pointers, which is problematic.

CS4615 Monday 6th January 2014

System Security

- Java Security Model.

- Register on Moodle - CS4615 System Sec
- 20% - 2 x in-class test + 2 x exercises

- CS4253 old name

Buffer Overflow Vulnerabilities and Stack Smashing

SP = Stack Pointer (top of current top of stack)
FP = Frame Pointer (bottom of current frame, of function that is currently executing)
PC = Program Counter

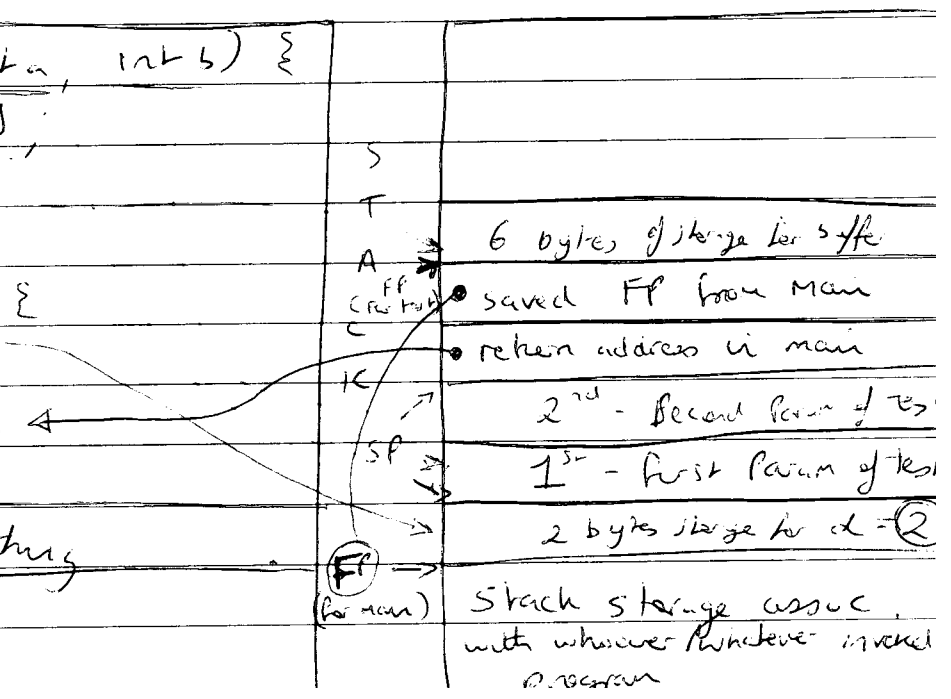
PC →

```
Void test (int a, int b) {
    char buff[5];
}

Void main () {
    int d = 2;
    test (1, d);
}
```

Text

data - nothing



When you are at test you need to remember after execution where I have been (PC) so we need to push the return address onto the stack

We also need to save the FP from main, so this also gets pushed onto the stack.

Compiler is mapping locations to their relevant location in memory!

When complete we need to update the stack + FP. Then we set the PC to return address and get rid of all the storage associated with it. Then we change the value of $d = 4$.

ie. we store the value of PC. (ie. $d = 4$)

This is where the exploit begins!

Buffer Overflow

Try to store a long string to a buff which is only 4 characters long. buff [4].

strcpy (buff, str)

4 LONG

4 STRI = overwritten stack from route

4 NG = overwritten return address

strcpy just blindly copies / increments address / copy... until is completed (ie. reached NULL)

In other words we have corrupted the return address so the program does not know where to go next because return address now is different.

CS4615 Monday 13th January 2014

Trojan Horses

./ls = look in cured directory for a command called ls and execute it.
The trojan horse could be in this file and contain script to change ~~w~~ rights such as chmod a +rwx $HOME.

△ Fix Don't put '.' in in command query.
This is considered a dangerous action

BOTNET

△ Torpig installs a trojan horse in browser software (eg infect Firefox) so when you think you're setting up a secure connection, there is a man-in-the-middle.

△ MS Word - Revisions - (Track Changes) - Customer contracts change something but if revisions are enabled then the user maybe able to see previous versions.
Also we have MS Word fast save which appends all changes to the bottom of the original file.

$a \leq b$ → Reflexive

a may flow to b

Reflexive, Antisymmetric, transitive.

unclassified ≥ secret ≥ topSecret.

CS4615 Tuesday 28th January 2014

2 x InClass Tests
2 x Labs

InClass test 1 - 4pm Monday 3rd
Security Model, reference model
Past 3 weeks

2nd test week (8)

Reference Monitor - Delegation Certificates (& authorisation state rules)

△ A validity Period (dates) for authorisation
△ A flag specifying whether the recipient of the permission may further delegate to others

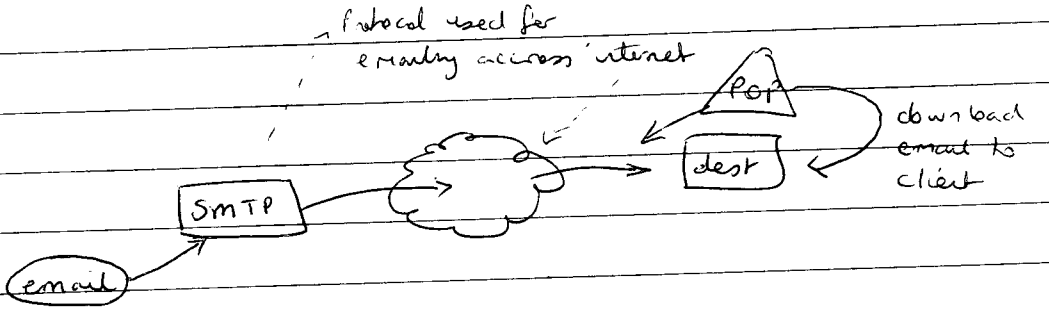
To get around people having certificates that they can't revoke, we should put a time on the certificate for example one day.

○ Permission - authority to carry out some action

CS4615 Tuesday 11th February 2014

Assignment - KEYNOTE

Email.



No authentication of email (ie xyz@xyz.ie email - SMTP doesn't check) it only - the right to use that SMTP server.

POP : Post office Protocol
A service running on a machine, point client to port on the machine that has the POP service.

Assignment : Keynote credntels / trust management to validate sender of email message.

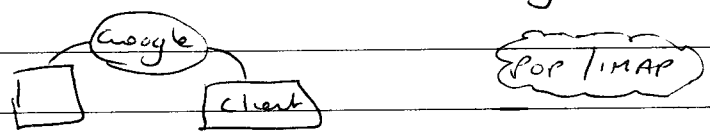
1. Current setup above, the email message will now be signed by KJ
2. Inside body using keynote so credentials are used to confirm it is from KJ. Certificate Body (CA) to confirm identity of sender.
3. After the email is pulled down using POP, the email is examined (signed, certificate) we are going to do a keynote check to confirm.
4. we don't have to write email clients, but instead we are going to put together an access control policy. (i.e. FROM)

5. Package – JavaMail API for access SMTP programatically + POP server "

6. MIME encoding

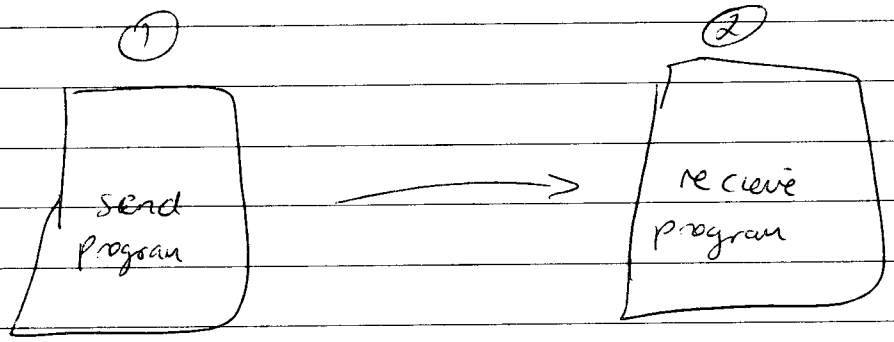
7. Putting the whole thing together?

8. POP
   -> retrieve email message
   -> extract MIME encoding (certificate)
   -> trust management query
   -> ISSUE :- No access to POP server ~~POP~~
   This has all been handed over to Google.



In Google you can enable POP, so JAVAMAIL client can access it.

Figure out what is going on and what needs to be done, there is very little programming required.



sign / CA check / confirm

Marking : 10, 5 marks for sample key note credentials.
Deadline : 2nd March.

Mandatory Access Control (MAC)
Domain and Type Enforcement (DTE)

Found in SELinux (Security Enhanced Linux)

- What are the protection domains do I need.
- What type of data is contained.

~~objects~~ / Types.

Domain				

DDT :- Domain Definition Table

Example : DTE Policy for Tetris High Scores (Slide 6/6)

Domain = {user, game}
Types = {user, highscore}

DDT :

	user	highscore
user	RWX	R
game		RW

- program tetris runs in the domain,
- user shell (~~SE~~CSH) runs in the user
- Highscores file (etc/highscores) is of type highscore
- ~/data is of type user.

CS4615 Tuesday 18th February 2014

Java Security Manager

Access Controller :- Enforcing particular Policies
Code centric policies, grant permissions to code.

The policy is another object inside your program.
At any one time you can only have one policy enforced.

△ CodeBase - a base of code, (Slide 20) except will be thrown because it didn't have permission to open that particular socket.
- Some Piece of Code = CodeBase.

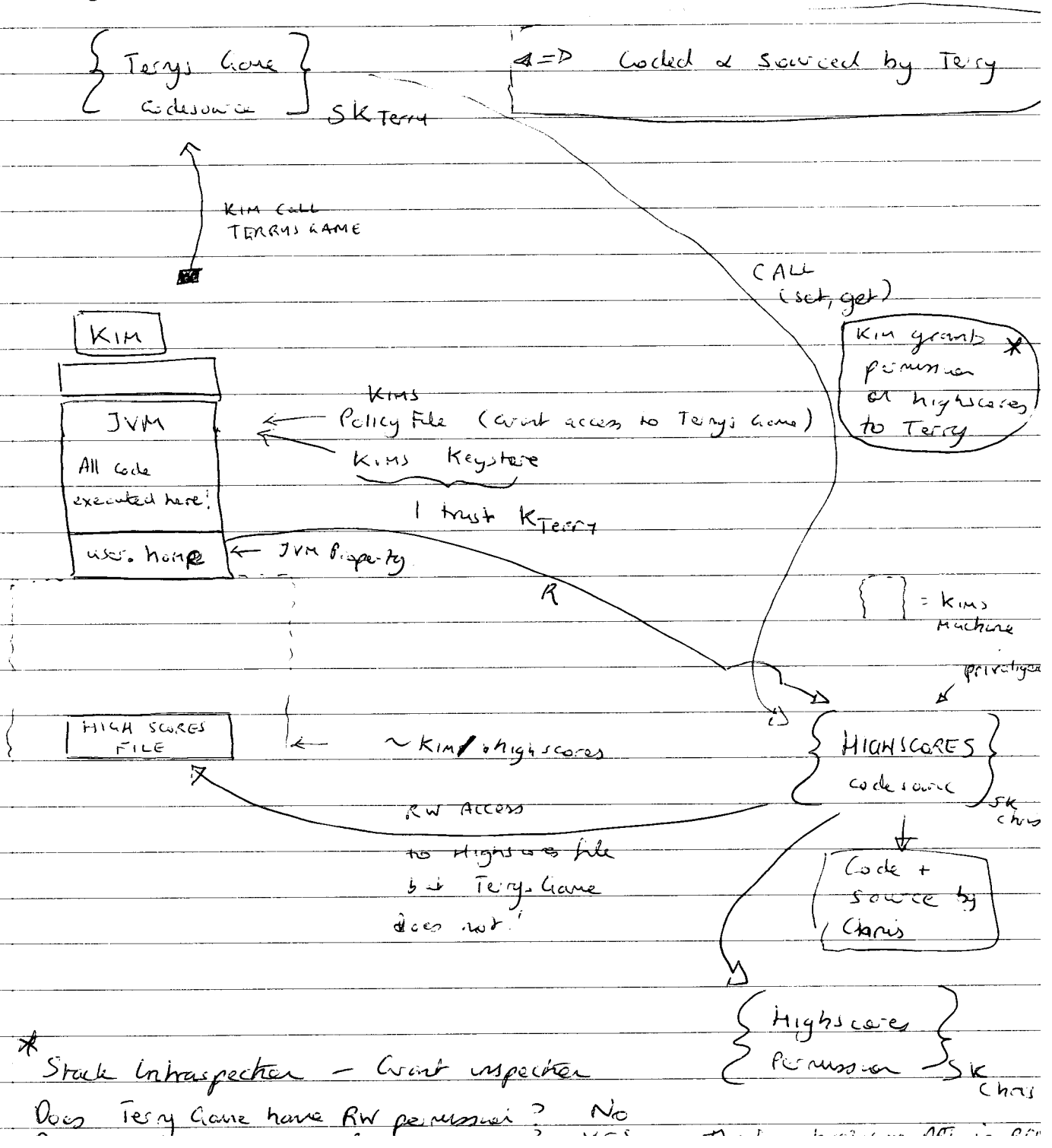
△ Code Centric :- Permissions are granted to code

CS4615 Monday 24th February 2014

Java Security Model (from last week)

Code Centric - assigning permissions to code not people

Terry's Game (Code Source from SVN/ORACLE)

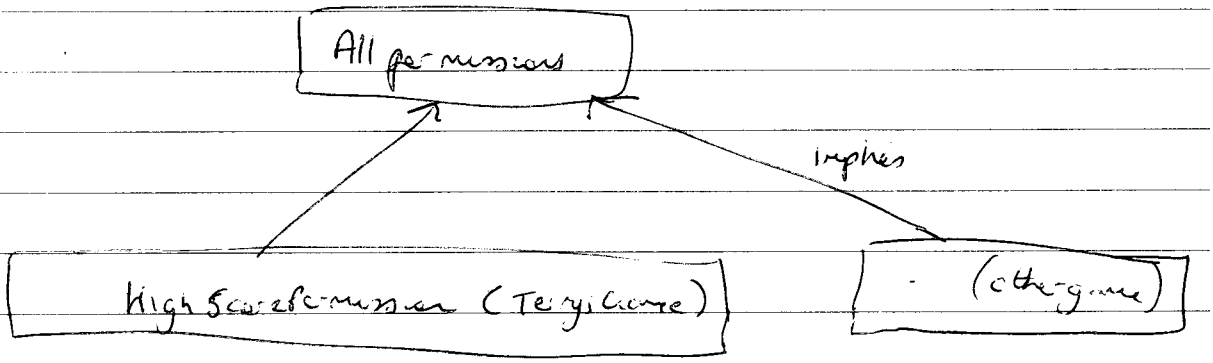


* Stack Introspection - Grant inspection

Does Terry Game have RW permission? No

```
public HighScorePermission (String name)
{
    super(name);
}
```

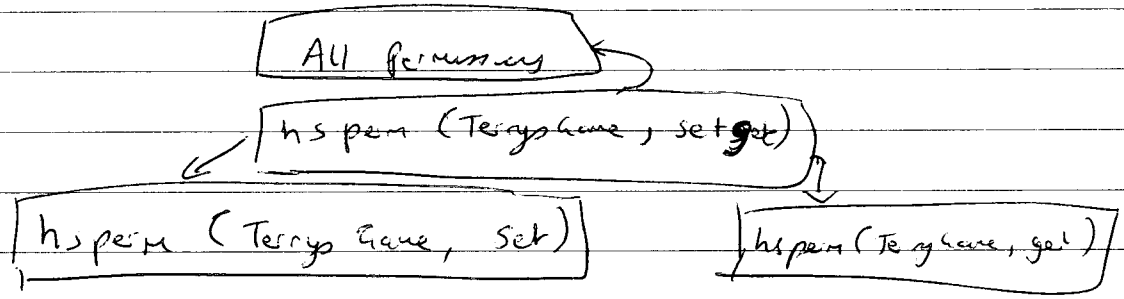
eg



Only allow TerrysGame to access TerrysGame highscores and not other games highscores.

```
public HighScorePermission (String name, String action)
{
    super(name, action);          // super(TerrysGame, set);
}
```

eg



docs.oracle.com/javase/tutorial/security/

Sample Policy File (The Java Tutorials > Security Features in Java SE > Implementing your own permission)

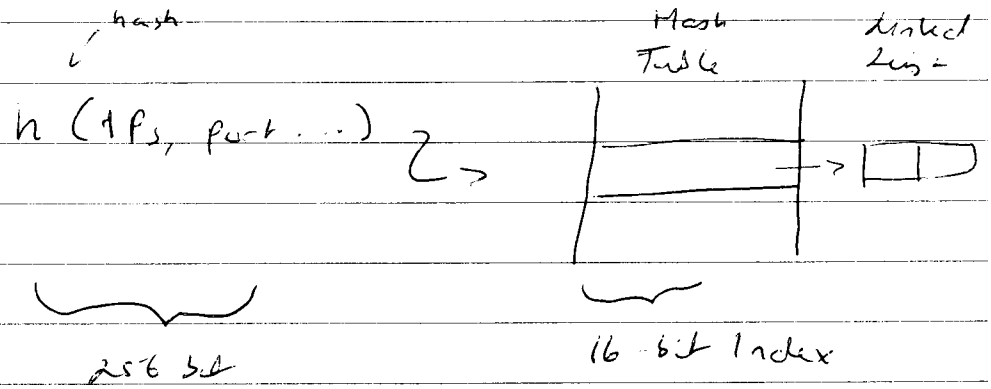
How does this compare to Unix setuid?
They are very similar, run program we use permissions for that program eg exepasswd. - Setuid is quite coarse-grain, the JVM is fine grain - only certain permissions.

CS4615 Monday 3rd March 2014

TCP/IP Protocol Vulnerabilities

Transport layer - fault tolerance
Sort packets in order 1,3,4,2 - 1,2,3,4

SYN-Flood : destination being flooded by SYN packets
$\frac{1}{2}$ open connections no y+1 response
to server port.



To increase security we can use a secret
ie

h_K (IPs, port ...)

↑

keyed one-way hash.

SYN Cache.

:> 4 of 15

Monday 10th March 2013

Firewall - look at packets as they go past and see if they are allowed.

Policies - set of rules - reject / permit packets
Accept / deny.