CS255 Programming Project 2

Programming Project 2

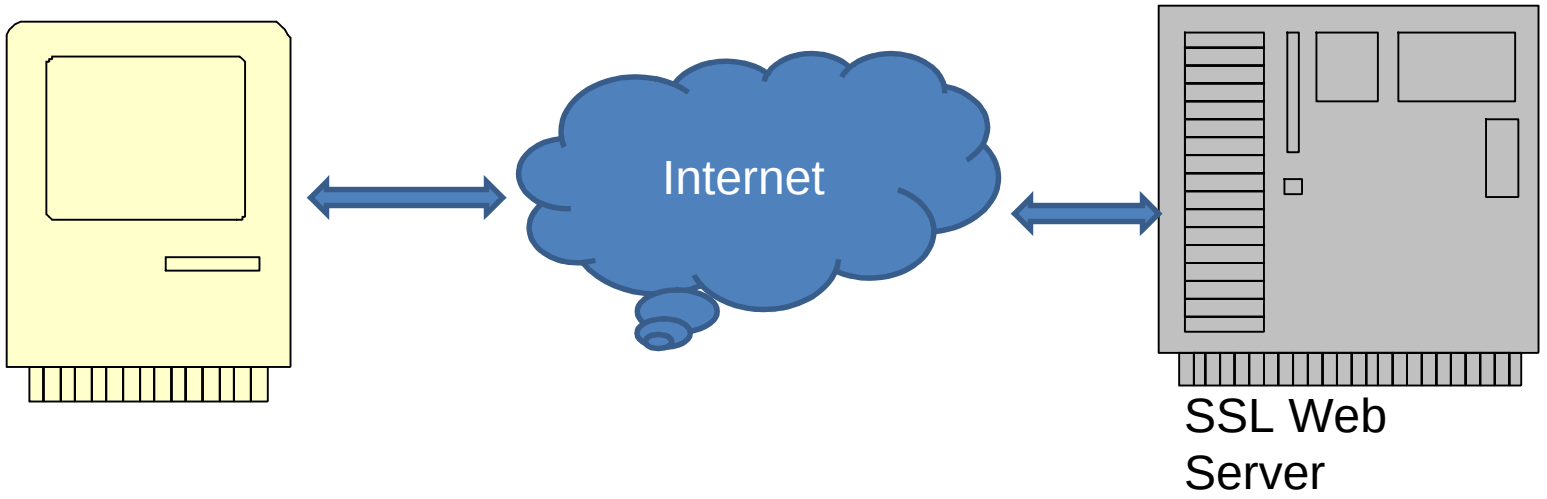
- Due: Wednesday March 14th (11:59pm)
  - Can use extension days
- Can work in pairs
  - One solution per pair
- Test and submit on Leland machines

Overview

- Implement a simple Man In The Middle (MITM) attack on SSL
- Use Java's networking, SSL and Certificate implementations
  - No need for low level packet manipulation
- Also implement a password based authentication system for the MITM server
  - Allows hacker to issue commands to server

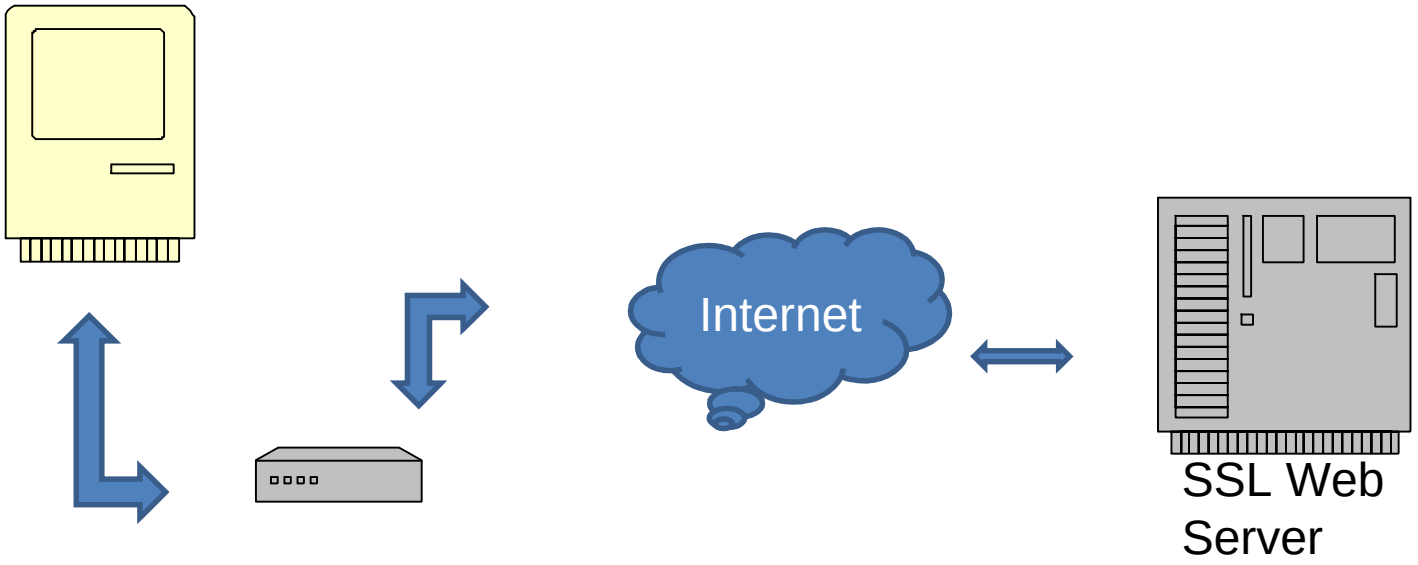
Overview

- Normal SSL
  - SSL encrypted data routed like normal TCP/IP data over the internet



Proxy Server

- Browser connects to proxy
- Proxy connects to web server and forwards between the two



Man in the Middle

- Instead of forwarding encrypted data between the two hosts, our proxy will set up two DIFFERENT SSL connections between the two.
- Proxy<->Remote Server
  - Sets up a normal SSL client connection to requested remote site
- Proxy<->Browser
  - Sets up a SSL server connection to the browser, using its own certificate, generated as a copy of the remote host's cert
- If the browser accepts this fake cert, the proxy has access to the data in the clear!

What is provided?

- Basic Proxy Server setup
  - Parses CONNECT request and sets up a connection between client and remote server
- Basic Admin Server/Client
  - Server listens for connections on a PLAIN socket and parses out username/password/command that the client sends

Security Features

- Secure connection between admin client and proxy server using SSL
- Password based authentication for client
  - Secure storage of password file
  - Passwords stored hashed using public and private salt
- Extra Credit: Challenge / Response authentication
  - This is IN ADDITION TO password authentication

Proxy Server

- Already listens for the browser CONNECT request and sets up the needed SSL connections
- You need to
  - Understand the connections being made
  - Obtain the remote server cert from the remote SSL conn
  - Copy the relevant fields and sign the forged cert using your CA cert (from your keystore) (use IAIK)
  - Modify the code creating the client SSL conn to use the newly forged cert

Signing Certificate

- Build a self signed cert for the proxy server using keytool
  - `keytool –genkey –keyalg RSA`
  - Store this in a JKS keystore for use by your proxy server
  - Use it for signing your programmatically generated certs
  - You pretend to be a CA e.g. Verisign
- Submit a keystore with your project

Generating Certs “On the Fly”

- Not easy to generate certs programmatically using standard Java libs
- Use the IAIK-JCE library
  - iaik.x509.X509Certificate

iaik.x509.X509Certificate

- To convert from a java cert:
  - new X509Certficate(javaCert.getEncoded());
- Signing
  - cert.sign(AlgorithmID.sha256withRSAEncryption, issuerPk);
- See iaik.asn1.structures.Name
  - For extracting info (e.g. common name) from the cert's DN (cert.getSubjectDN())

Managing Certs and SSL Sockets

- Use the KeyStore class for
  - Loading certs from file (e.g. your CA cert)
  - Storing programmatically generated certs
- Use SSLContext class for setting up certs to be used with an SSLServerSocket
  - Create a cert
  - Load into new KeyStore
  - Init a KeyFactoryManager with new KeyStore
  - Init SSLContext with new KeyFactoryManager and provided “TrustEveryone” TrustManager
- Use SSLContext for creating SSLSocketFactories

Admin Server

- Already listens for client connections and parses the data sent, using plain sockets
- You need to
  - Modify the code to use SSL sockets (see the proxy server code for examples)
  - Implement authentication for the transmitted username and password
  - Implement the required admin commands
    - Shutdown – the proxy server to stops accepting connections and exit
    - Stats – the proxy server returns a summary of the number of connections it has processed. Add code to record these

Password Authentication

- Proxy server listens for SSL connections from admin client too
- On connection client transmits a username and password
- Server verifies these from its local password file, and executes command if the client is authenticated

Password File

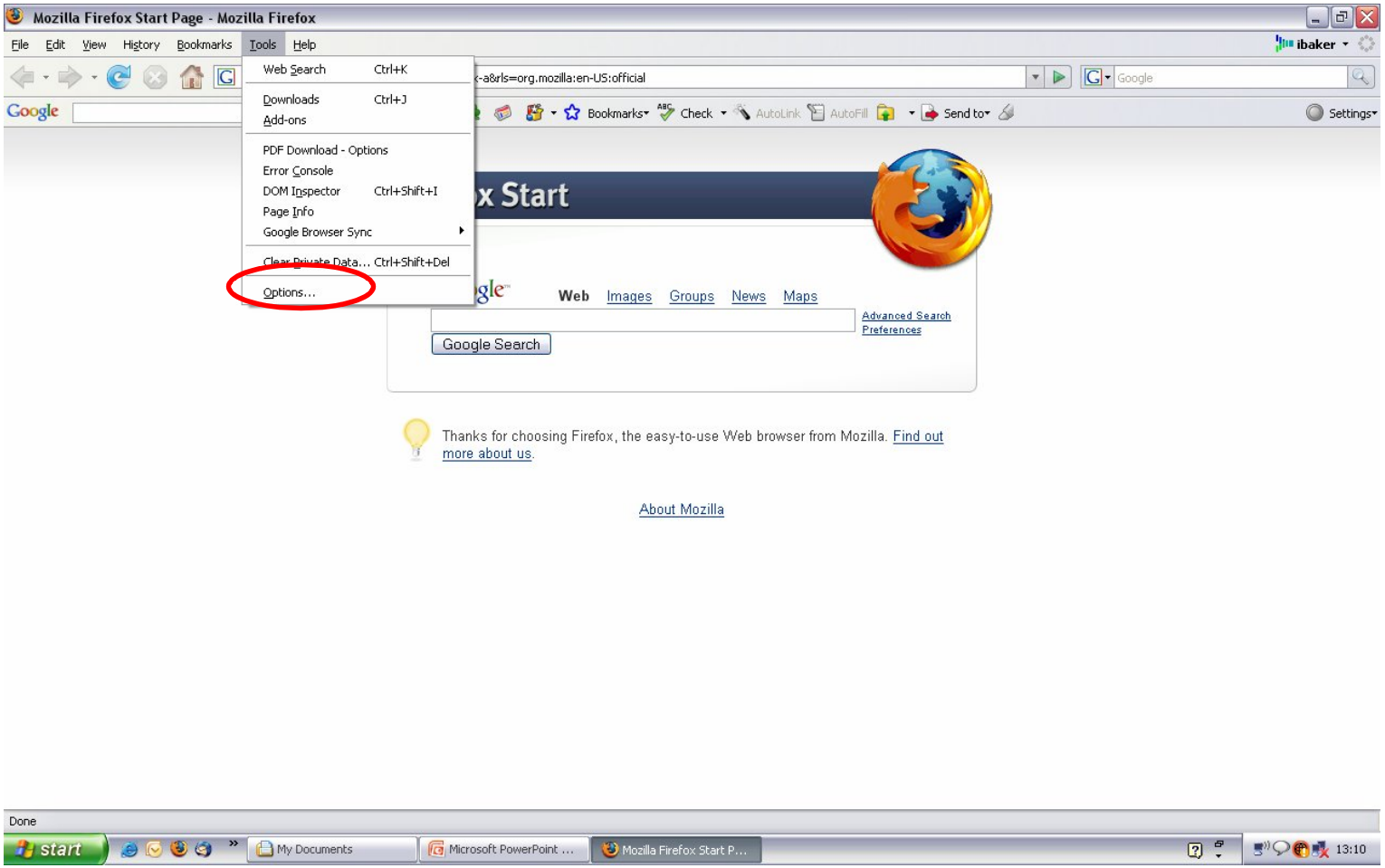
- Need to store a file containing usernames, salts, and hashed passwords
  - Use BOTH public and secret salts (AKA pepper)
- Should be stored encrypted/MACed
  - Similar to how keyfile is stored in project 1
  - Can use built in CTR mode

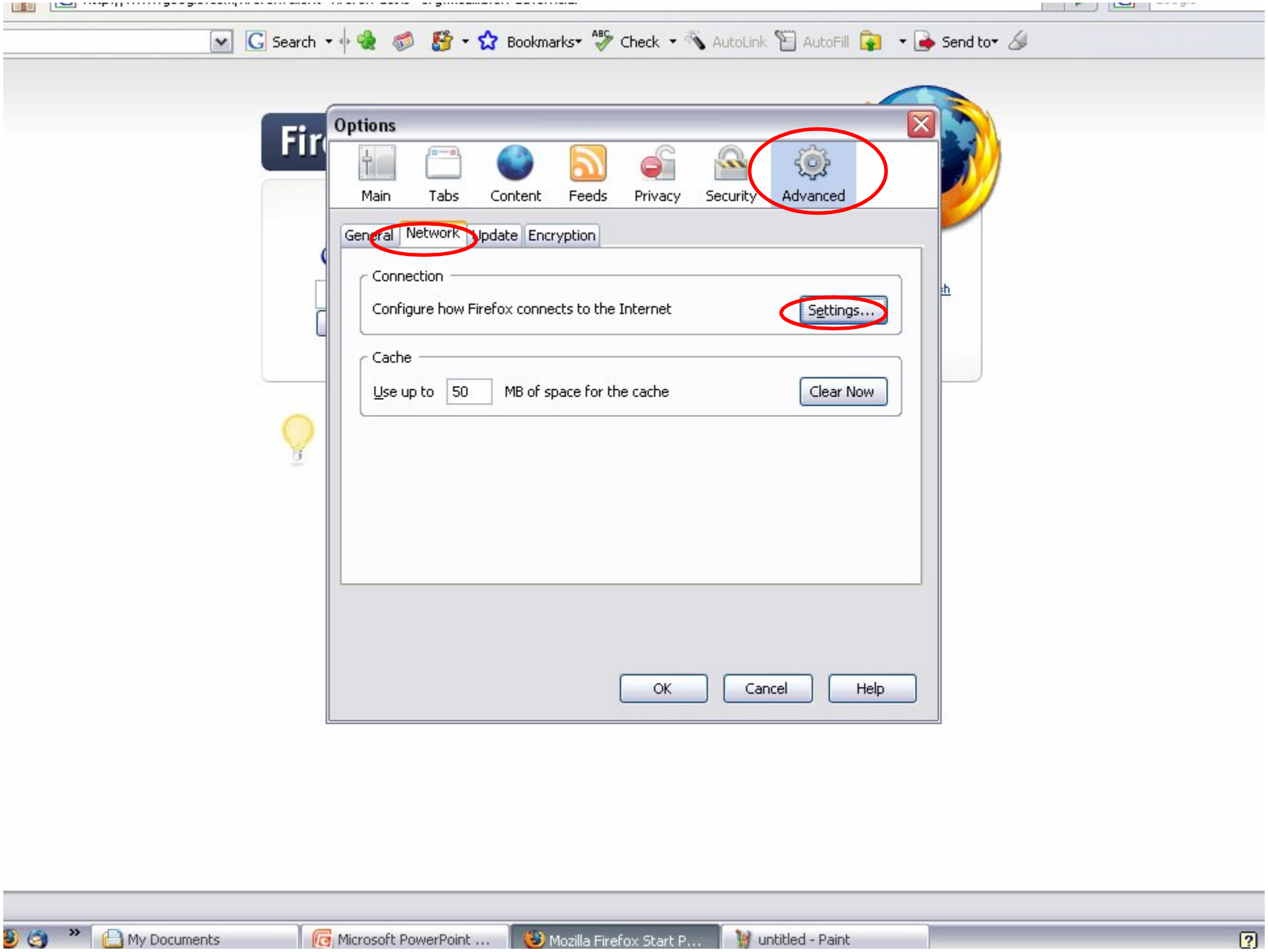
Username	Salt	Password
ibaker	S	H(Pwd||S||P)
singuva	…	…
dabo		
…		

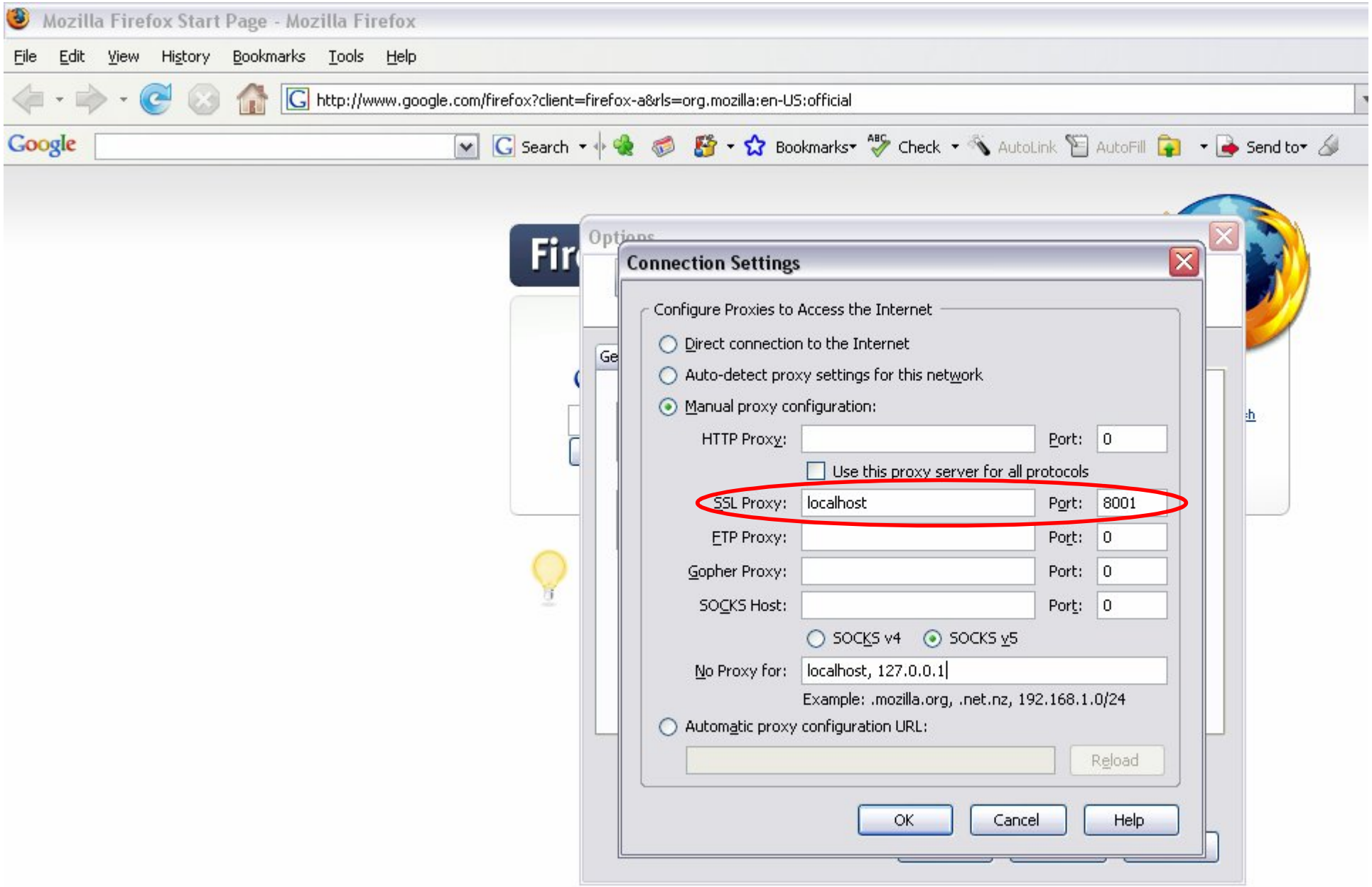
Password File Utility

- You need to add a utility for creating these password files
- Simple method:
  - Make a class to take a file with a list of usernames and passwords and convert it to a password file

Configuring Mozilla




Search
Bookmarks
Check
AutoLink
AutoFill
Send to
Options
Main
Tabs
Content
Feeds
Privacy
Security
Advanced
General
Network
Update
Encryption
Connection
Configure how Firefox connects to the Internet
Settings...
Cache
Use up to
50
MB of space for the cache
Clear Now
OK
Cancel
Help
My Documents
Microsoft PowerPoint ...
Mozilla Firefox Start P...
untitled - Paint


Mozilla Firefox Start Page - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://www.google.com/firefox?client=firefox-a&rls=org.mozilla:en-US:official
Google
Search
Bookmarks
Check
AutoLink
AutoFill
Send to
Options
Connection Settings
Configure Proxies to Access the Internet
Direct connection to the Internet
Auto-detect proxy settings for this network
Manual proxy configuration:
HTTP Proxy:
Port: 0
Use this proxy server for all protocols
SSL Proxy: localhost
Port: 8001
FTP Proxy:
Port: 0
Gopher Proxy:
Port: 0
SOCKS Host:
Port: 0
SOCKS v4
SOCKS v5
No Proxy for: localhost, 127.0.0.1
Example: .mozilla.org, .net.nz, 192.168.1.0/24
Automatic proxy configuration URL:
Reload
OK
Cancel
Help

Possible Problems

- You should be able to start up the proxy server and connect to it “out of the box”
- If you are having problems
  - Is someone else using the port? (default 8001)
    - Try a different port on the command line
  - Firewall problems?
    - Try opening the needed ports 8001/8002 (or whatever)
  - Try running your browser on the same machine and setting the proxy as localhost
  - We can’t debug your local network setup

Grading

- Security comes first
  - Design choices
  - Correctness of the implementation
- Did you implement all required parts?
- Secondary
  - Cosmetics
  - Coding style
  - Efficiency

Submitting

- README file
  - Names, student IDs
  - Describe your design choices
  - How to run your system (e.g. create passwords)
  - Answer to discussion question
- Your sources
- A sample of data recorded from your proxy
- Use `/usr/class/cs255/bin/submit` from a Leland machine

Stuck?

- Use the newsgroup (su.class.cs255)
  - Best way to have your questions answered quickly
- TAs cannot:
  - Debug your code
  - Troubleshoot your local Java installation
  - Troubleshoot your local network