Group members: Michael Noeth and Jyothish Varma.

Proposed project: We propose creating a client / server authentication protocol based on the Ipaq H5550's biometric API. Instead of using the traditional user name and password pair to authenticate users, a user name and finger print pair will be used in its place. This is beneficial for the following reasons:

- heightened security – Passwords are susceptible to attacks due to easily guessed passwords, carelessness about passwords (i.e. passwords written down in your desk), etc. A finger print allows each user to have a unique identifier which cannot be easily stolen or reproduced.
- ease of use – Users no longer have to remember a password. They simply need to remember their user name in order to authenticate.

The system will use the Ipaq H5550 as both clients and servers. Servers will store a list of user names and their corresponding finger prints. In order for a client to authenticate itself, it sends a user name and finger print pair to the server. The server would verify that the user name and password were in the database and upon this condition being true, distribute an authentication token to the client. The token can then be used by the client to consume services provided by the server.

The protocol can be used in arbitrary applications to ensure that a legitimate user is interacting with the server. The goal of this project is to provide a simple API that can be used in applications to authenticate remote clients to a central server. A sample code segment of the server side authentication would ideally work like this:

```
if (AuthentiationProtocol::IsAuthorized() )
        provide protected service;
else
        return error code;
```

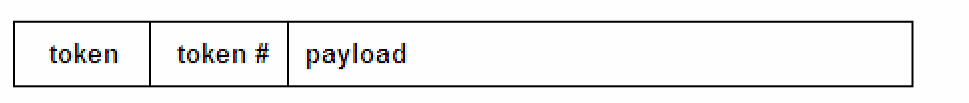
Protocol outline: The protocol will be broken into two sections: (1) biometric authentication and (2) normal communication.

A biometric authentication would proceed as follows:

1. Client enters user name and scans finger.
2. Client sends this data to the server requesting a token.
3. Server looks up user name and finger print pair.
    a. If they match – return a token and token number to the client.
    b. If they do not match – return an error code to the client.

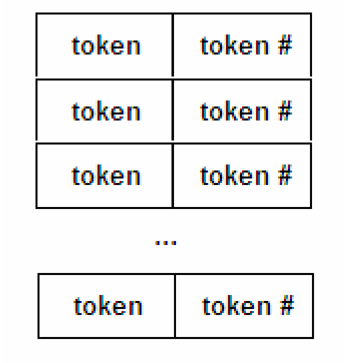
Normal communication would then proceed as follows:

1. Client already has a token at this point.
2. Client sends token and application data to the server.



token	token #	payload

3. Server looks up token by token # in table and determines if the data is paired with the correct token (if they don't match an error code is returned).



token	token #
token	token #
token	token #

...

token	token #

4. Server generates a new token and replaces the old one in the table.
5. Server sends new token back to the client.

Milestones / Division of labor: (note after item 6 – there is no one assigned as this is getting further into the project and we will probably need to re-evaluate at this point).

1. Specify the exact behavior of the new biometric authentication protocol as well as the API that will provide access to the protocol (Jyothish and Mike).
2. Download, install, and practice with Embedded Visual c++ (Jyothish and Mike).
3. Research communication options for the Ipaq H5550 – 802.11, Bluetooth, RF port, etc (Jyothish).
4. Create communication prototype in which a two Ipaqs perform a ping pong communication (Jyothish).
5. Research the Ipaq's biometrics API (Mike).
6. Create biometrics prototype in which an Ipaq requests a user's finger print and compares to a stored finger print. The prototype will print out whether the finger print matched or not (Mike).
7. Create simple prototype using both communications and biometrics. This prototype will have a client and server. The client will request a finger print and send the data to the server. The server will compare the finger print to a stored finger print and report back to the client whether the finger print matched or not.
8. Create a prototype of the authentication prototype in which the server has a database of users and finger prints. Multiple clients should be able to connect to this database by specifying a user name and a finger print.
9. Build the biometric authentication into an existing client / server application for the Pocket PC. Time transactions with biometric authentication versus the same application without biometric authentication. This will give an idea of the overhead the biometric authentication will add to the program.

References:

Title	**Location**	**Description**
Pocket PC Developer Network	www.pocketpcdn.com	Pocket PC programming reference guide.
Windows Mobile Developers	www.microsoft.com/windowsmobile/developers/default.mspx	Another reference for programming the pocket PC.
Biometrics API	http://devresource.hp.com/drc/technical_papers/Bioapi.pdf	Reference for the finger print scanning API on the Pocket PC
Sample of Biometrics	http://pbdj.sys-con.com/read/42623.htm	Useful sample to build biometric verification off of.
Pocket PC Network Programming	QA76.5.M19145 2004 6th floor (in NCSU library)	Guide to help with network aspect of this project