CS694 Spring 2012 Final Project

Ocelot	Matt Banzhaf	banzhamj@mail.uc.edu
Pangolin	Josh Burbrink	jwbk31@gmail.com
Quetzal	Jonathan Elchison	elchisjm@mail.uc.edu
	Brett Kizer	brett.t.kizer@gmail.com

What the software does, particularly any novel parts that were not required

Our strategy is to provide a solid automated defense, with strategic human-initiated offense.

We plan to utilize port hopping for our server every 5 minutes. The GUI will stop the server, tear down the necessary tunnels, and restart on a new port. Coupled with our encryption, this hopping will enable us to avoid being on the same port for very long, hopefully before anyone is able to determine our server's location.

We have chosen to make our client fairly plain-Jane, in favor of utilizing human-initiated (as opposed to automated) attacks. The client allows us to send/receive commands and utilize skills learned in the class (such as GUI design, threading, and RMI). We plan on using the built-in "cracking" commands to learn of our opponents' whereabouts. We also plan on attempting to sniff our opponents' packets, hoping to find some clients using no (or bad) encryption. It may not be possible, but we plan on attempting to exploit helios to gain root privileges (and therefore steal as many credentials as possible). These credentials can be used to steal wealth, etc.

Instructions for setting up and using the code

1. Setup SSH with your client's public key in the authorized_keys file on the server.
2. Configure the definitions at the top of ./scripts/start.sh.
3. Run ./scripts/start.sh. Ensure that all four SSH tunnels establish properly.
4. Run the GameBoard applet.
5. Enter username and password.
6. Enter the server port number in the upper left box. If you didn't specify one as a argument to ./scripts/start.sh, then it defaulted to port 20000.
7. Click the Server "Connect" button. It turns green.
8. Click the Client "Connect" button. It turns green.
9. Click the "IDENT" button.
10. Click the "ALIVE" button.
11. Type the host (likely "localhost") next to the "HOST_PORT" button.
12. Click the "HOST_PORT" button.

The "Auto Run" button begins the port hopping, and requires setting up SSH with your client's public key in the authorized_keys file on the server. "Auto Run" should be started with a disconnected server and disconnected client.

Efforts of each person working on the project

Breakdown by class (names taken from revision control)

- ActiveClient - Matt, Josh, Jonathan, Brett
- AutoRunThread - Jonathan

- BetterStringTokenizer - Matt
- CertRemote - Dr. Franco
- CommandHandler - Matt
- ConnectionHandler - Matt, Jonathan
- DbgSub - [original from NetSec final]
- DebugConfig - [original from NetSec final]
- GameBoard - Brett, Jonathan, Matt, Josh
- GlobalData - Matt
- Karn - [original from NetSec final]
- KarnBufferedReader - Dr. Franco, Josh, Jonathan
- KarnPrintWriter - Dr. Franco, Josh, Jonathan
- MessageParser - Matt, Jonathan, Brett, Josh
- PermanentStorage - Matt, Jonathan, Josh
- PlayerCertificate - Dr. Franco, Matt, Jonathan
- PubRSA - Dr. Franco, Matt, Jonathan
- RSA - Dr. Franco, Matt, Jonathan
- Server - Matt, Jonathan, Brett
- Util - Matt, Jonathan

Significant development areas

- Provided lots of code snippets - Dr. Franco
- Initial project skeleton - Jonathan
- Provided code from Network Security final - Matt
- github setup - Matt
- Netbeans project - Matt, Josh
- Eclipse project - Jonathan
- GUI - Brett
- Encryption, first stab at RMI - Josh
- Updating RMI to utilize Dr. Franco's new stub, port number, tunnels - Jonathan
- Streamlining login - Matt
- Scripts to automate tunnels and port rotation - Jonathan
- make_certificate - Josh
- This write-up, submission - Jonathan
- First login, debugging, setup - Jonathan, Matt, Josh

Revision Control

Complete source can be seen at https://github.com/banzhamj/CS694-Final-MB-JB-JE-BK