Lesson Plan

LESSON TITLE: **Module 7: Protection Mechanisms**

SUMMARY:

Topic Outline

- Understanding basic firewall configuration
- Access control and password creation/protection
- System and Security Logging

GRADE BAND:

☐ K-2
☒ 6-8
☐ 3-5
☒ High School

Time Required:

90 minutes

Lesson Learning Objective/Outcomes: Upon completion of this lesson, students will be able to:

To understand cybersecurity protection mechanisms and security design principles.

Materials List:

At least three networked Windows workstations with an updated version of Oracle Java SE installed, and with sufficient permissions to reconfigure the Windows Firewall.

How will you facilitate the learning?

- Describe the Warm-up Activity/Focused Activity/Closure and/or Reflection
- Describe the Teacher Instruction

We adopt the following best instructional practices and strategies to facilitate learning:

- Multimodal presentation of information
- Cooperative active learning
- Team building
- Periodic checking for understanding

This lesson includes:

☒ Mapping to Cyber Security First Principles
☒ Learning Objectives
☒ Assessments

Mapping to Cyber Security First Principles:

☒ Domain Separation
☐ Abstraction
☒ Process Isolation
☒ Data Hiding
☐ Resource Encapsulation
☒ Layering
☐ Modularity
☐ Simplicity
☒ Least Privilege
☐ Minimization

Assessment of Learning:

TYPE (Examples Listed Below)	NAME/DESCRIPTION
Quiz/Test Presentation Oral Questioning	Online pop-quiz/survey using gosoapbox.com will be utilized to check understanding of key indicators. Key Indicators of Understanding • Recognition of Cyber Security Principles involved • Basic knowledge of network protocols • Basic knowledge of firewall rules • Familiarity with characteristics of strong passwords Hands-on exercises will be conducted to reinforce learning. Participants will be asked to present their observations on pertinent case studies.

Accommodations: (Examples may include closed captioning for hearing impaired students; accommodations for students with disabilities.)

We will provide digital, closed-captioned videos to correspond to the hands-on exercises.

Description of Extension Activity(ies):

Background Materials

Protection Mechanisms: Firewalls and Passwords

The purpose of a firewall is to moderate and control network traffic, for the purpose of logging network activity and/or blocking potentially harmful traffic. Firewalls provide a layer of protection between your computer and the network(s) it communicates with. In the case of firewalls which are embedded within network equipment (such as the *packet filtering routers* that many of you probably have in your homes), they also provide a layer of protection between *private* (trusted) networks and *public* (untrusted) networks. This protection has become

increasingly important as more and more systems and applications are connected to the Internet.
One of the most common firewalls is the so-called *software firewall*, a firewall which is installed on an individual's personal computer. (In contrast, firewalls embedded within dedicated network hardware are often referred to as *hardware firewalls*.) Modern computer operating systems typically include a software firewall as one of their standard components, although there are also third-party firewalls that can be installed separately. In both cases, the firewall allows the operating system to monitor incoming and outgoing network traffic in the form of *network packets*, and to determine whether these packets should be forwarded to their intended destination or not. To make this determination, the router consults its set of *rules*; once a rule which matches the packet is found, the corresponding rule action, which usually *allows* or *blocks* the packet, is obeyed.

Modern firewalls are typically shipped with a default set of rules which can provide a reasonable level of security for many applications. However, it is important to remember that a firewall which is not properly configured for the needs of your particular network can be more of a security risk than not having one at all! Relying on a firewall's default configuration can provide a false sense of security, so it is important to be able to analyze and understand this configuration so that it can be strengthened with additional layers of protection as required for your needs.

Another important protection mechanism is the selection of strong passwords. This exercise will explore the characteristics of strong passwords, and the tools and techniques of password circumvention and cracking.

Associated Problem-Based Laboratory Exercises

Part One of this exercise will involve analyzing and expanding the default firewall rule set for the Windows Firewall, the standard software firewall included with Microsoft Windows. As an example of a networked application that can be governed by the firewall, we will use a network chat application called "TEATalk," which you will find inside the "**Module7**" folder on your workstation desktop. TEATalk allows users on a network to create private *chat rooms*, in which they can exchange text messages with each other over the network. It consists of two parts: the *client* (the application that you will use to compose and read the text messages), and the *server* (which acts as a relay, forwarding incoming messages to other users).

The chat rooms are password protected, and the messages within each chat room are protected using the Tiny Encryption Algorithm (TEA), a simple form of encryption. In Part Two, we will explore the tools and techniques of *password circumvention* and *cracking*, which will involve analyzing captured chat traffic, breaking the encryption, uncovering other users' passwords, and accessing these users' chat rooms.

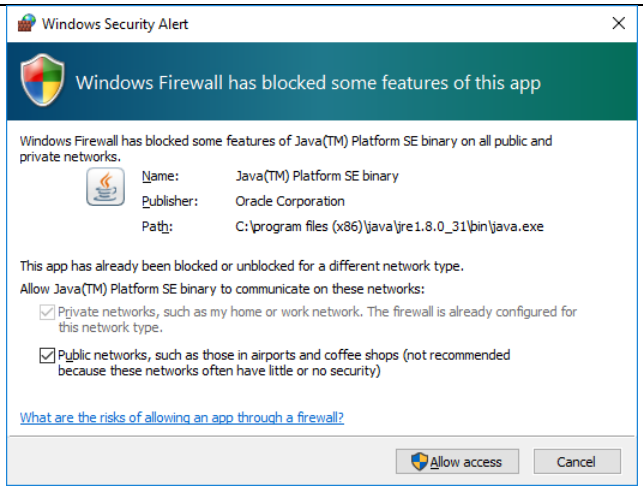
We will begin by dividing the participants in the lab into groups of three or four. In Part One of this exercise, one person in the group will host the chat server on his or her workstation, and will configure the firewall to allow the clients in the group to access it. The others in the group will assume the role of the clients, and will assist with the testing of the firewall configuration. In Part Two, all three individuals in the group will use their chat clients with a shared chat server, and will collaborate in the process of cracking the passwords.

Part One: Firewall Configuration

We will begin by configuring and starting the TEATalk Server on one of the workstations within your group. For the remainder of Part One, we will refer to the workstation that will be hosting the TEATalk Server as "the server", and the other two workstations in the group as "the clients". Within your group, designate one of these as "Client A" and the other as "Client B" (it does not matter which is which).

In order for the clients to be able to join a chat session, the server's firewall must be configured with an Inbound Rule which matches the *TCP port* (the network protocol and communication channel) that is used by TEATalk. Unless this rule is present, the clients' connection requests will not be allowed to pass through the firewall, and the clients will be unable to reach the server.

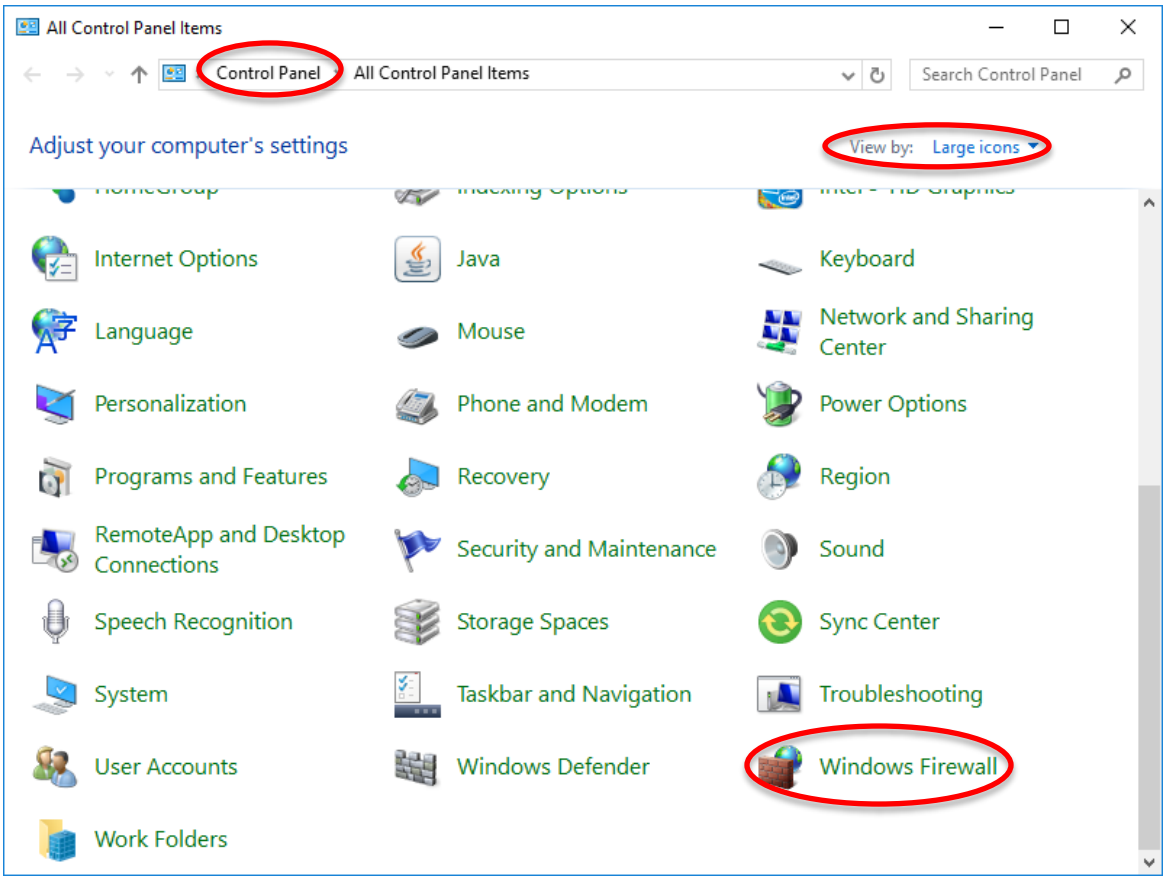
Start the TEATalk Server by double-clicking the "**TEATalk Server**" shortcut in the "**Module7**" folder. You may receive a message from Windows Firewall (see the next page) asking if you would like to create a rule to allow incoming connections; if you do, click "**Cancel**" to skip this step (we will be creating the rule ourselves shortly).



You will know the server is running when a small window appears containing the message:

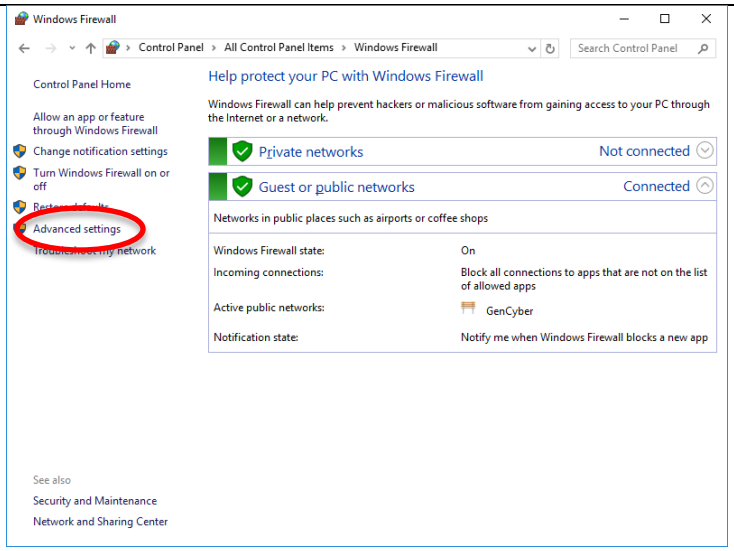
`Server waiting for Clients on port 1500.`

Minimize this window for now and open the Windows Control Panel to begin configuring the firewall. If your workstation is running Windows 7, click the Start Button in the lower-left corner of the screen and choose "Control Panel" from the Start Menu; if your workstation is running Windows 10, right-click the "**This PC**" icon on the Desktop, click "**Control Panel**" in the address bar (circled below), and in the "**View By**" menu, select "**Large Icons**".

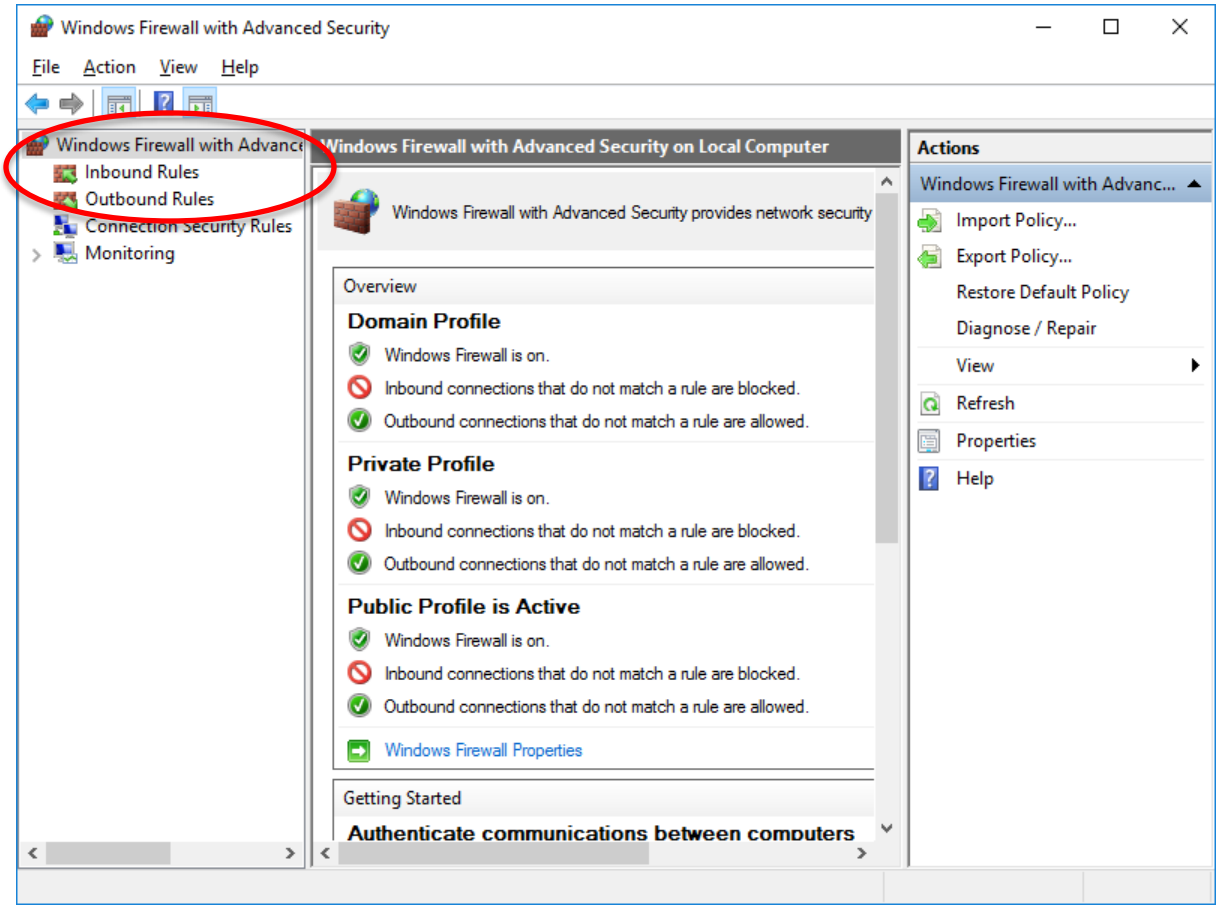


From the Control Panel window, click "**Windows Firewall**". You will see a window indicating the state of the firewall, along with a menu of options on the left-hand side; an example is shown on the next page.

Notice the "**Restore Defaults**" option: if you make a mistake at any time during the configuration, you can restore the firewall to its original state by choosing this option.



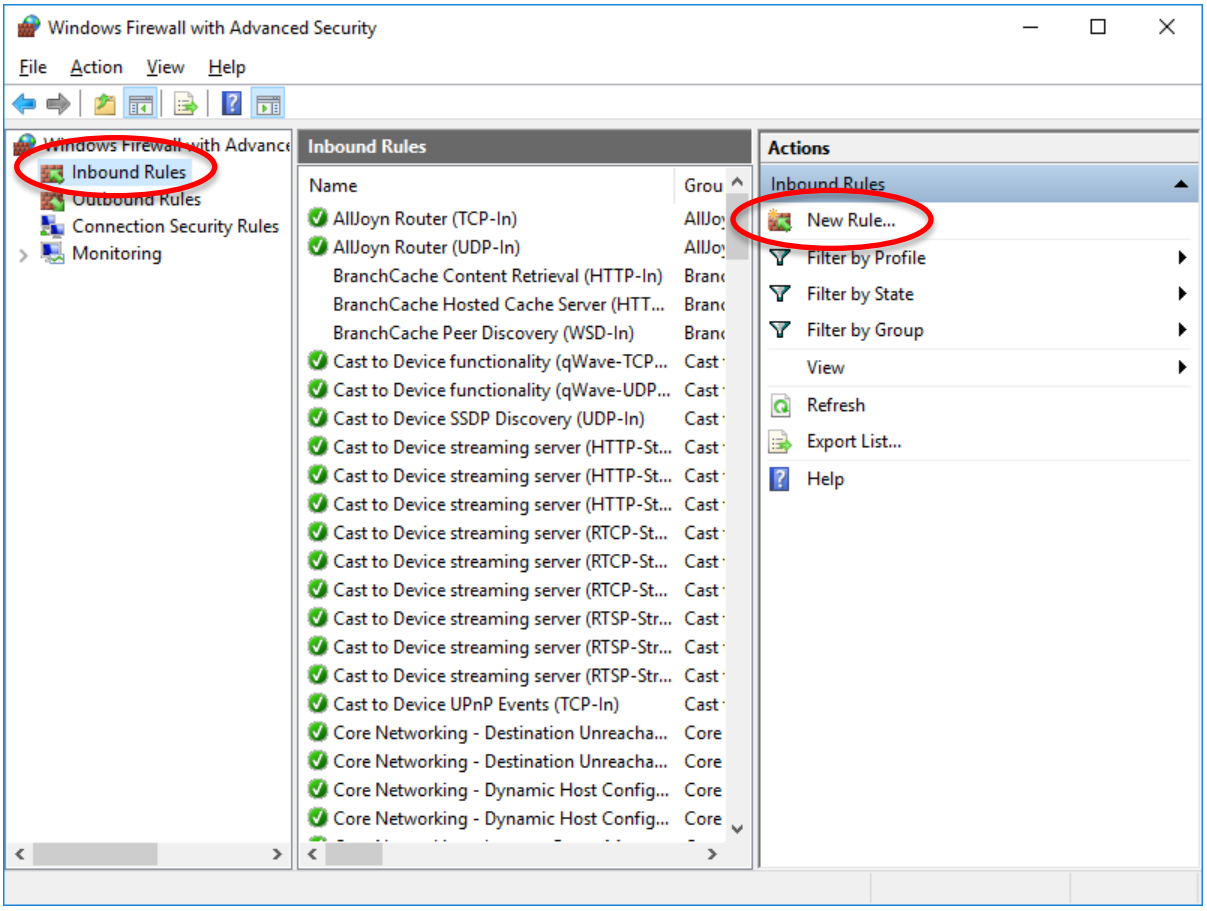
Choose "**Advanced Settings**" from the menu. You will be presented with a "**Windows Firewall with Advanced Security**" window. Click the topmost item in the Console Tree (the leftmost area of the window, circled in the screenshot below) to see an overview of the firewall configuration:



Note that there are several "profiles" listed, including "Public" and "Private" profiles. You may have noticed that, when you connect your computer to a new network for the first time, you are asked to specify whether the network is a "Public Network" or a "Private Network". This choice determines the firewall rules that govern your network connection: "Private" networks are presumed to be trusted networks, and thus enable more network services by default than "Public" networks, which are more strictly protected. The rule we will be adding will be applied to *all* profiles.

Notice also that, in all profiles, inbound connections are *blocked* by default unless they match one of the firewall rules, but outbound connections are *allowed* by default. This is why the client workstations will not require any changes to their firewall configuration.

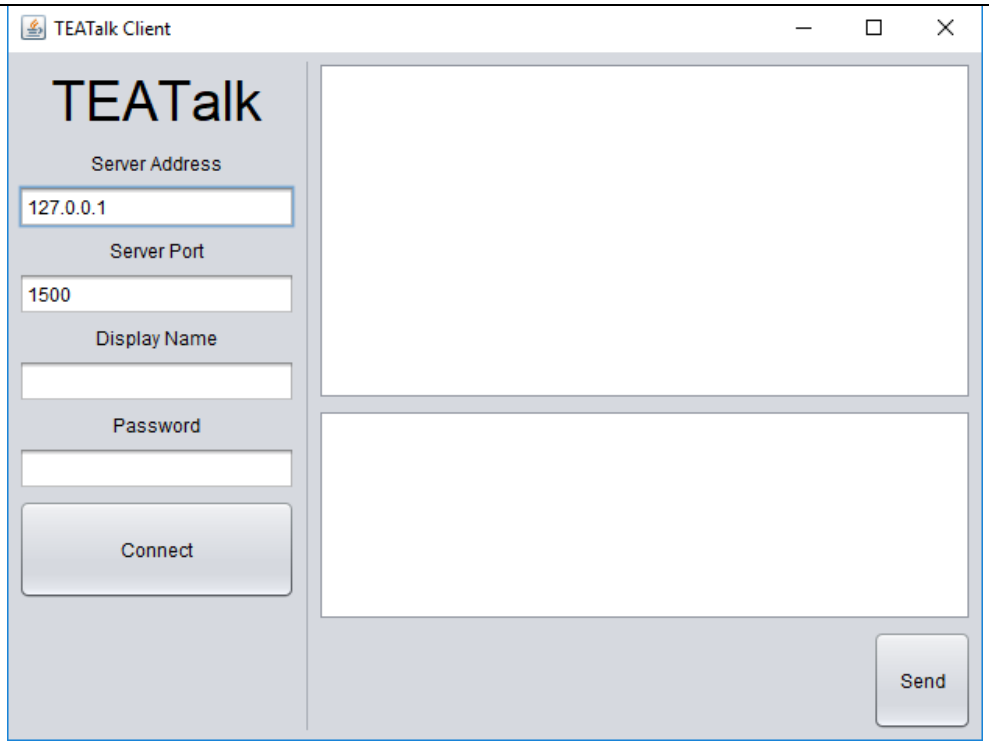
Choose the "Inbound Rules" option in the Console Tree (circled below), and you will be presented with a long list of rules corresponding to the various network services and applications that the firewall already recognizes (your list may be slightly different from the one shown in the sample). We wish to add a new rule for the TEATalk Server, so choose "New Rule" from the Action Pane (the area on the right-hand side of the window).



You will now be guided through the steps of creating a rule by the New Inbound Rule Wizard:

1) First, you will be asked to choose the type of firewall rule to create. Choose "**Port**" from the list, then choose "**Next**."
2) In the "**Protocol and Ports**" window, choose "**TCP**" from the list of protocols and enter "**1500**" (without the quotes) in the "**Specific Local Ports**" field (choose the corresponding radio button if it is not already chosen). Choose "**Next**."
3) From the list of rule actions, choose "**Allow the connection**" if it is not already chosen. Choose "**Next**."
4) When asked to specify the profiles for which this new rule should apply, choose all three options ("Domain", "Private", and "Public"), then choose "**Next**."
5) When asked to enter a name for the new rule, enter "**TEATalk**" in the "**Name**" field, then choose "**Finish**."

The new "**TEATalk**" rule will now appear in the list of Inbound Rules. The rule takes effect immediately, so on the two client workstations, launch the "**TEATalk Client**" shortcut in the "**Module7**" folder. The TEATalk Client window will appear, and should resemble the example shown on the next page.

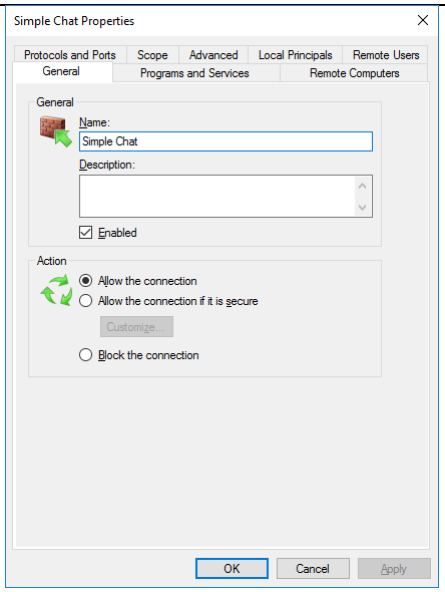


To start a chat session, it will be necessary to specify your display name, a password, and the network (IP) address of the server workstation. Enter your first name as the display name and "**SEAL**" (in capitals, without the quotes) as the password. To find the server workstation's address, open a Command Prompt window on the server, type the command "**ipconfig**", and press ENTER. The IPv4 address of the active network adapter is the address that the clients should use. Enter this information into the corresponding fields on the left-hand side of the TEATalk Client window, then click "**Connect**."

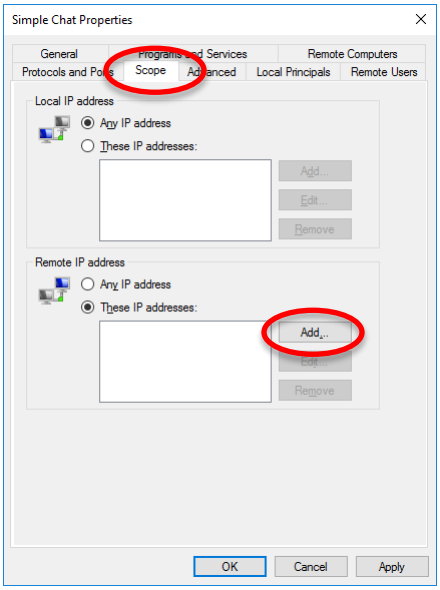
If the new firewall rule is functioning correctly, both clients should be able to connect. After exchanging a few short messages to test the connection, both clients should click "**Disconnect**" to end their chat session. It will not be necessary to re-enter the server information to connect again; instead, simply click the "**Connect**" button to open a new session.

(If the clients cannot connect, another rule in the server's firewall configuration might be interfering. Close the "Windows Firewall with Advanced Security" window, and in the main Windows Firewall window, choose "**Restore Defaults**" from the left-hand menu. After the default settings have been restored, try creating the new rule again.)

Now that we have confirmed that our rule is functioning correctly, let us experiment with adding some restrictions by blocking one or the other of the two clients. On the server, right-click the "TEATalk" rule in the list of inbound rules, and choose "**Properties**" from the context menu. You will be presented with a property window for the connection, divided into various tabs:

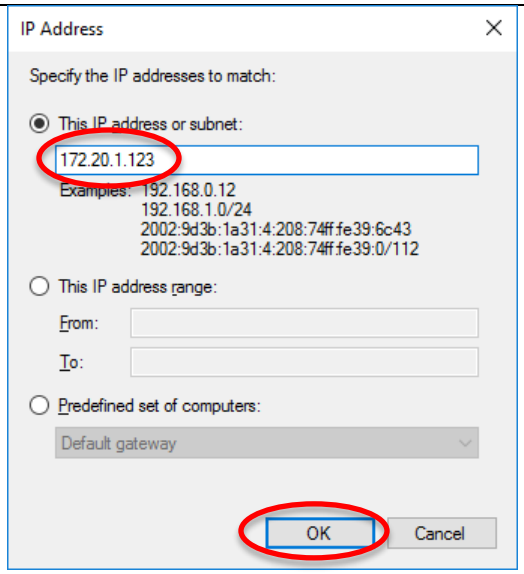


There are several ways to block specific computers from accessing a given network service. The simplest way is to specify the network address(es) corresponding to the machine(s) that you wish to block. From the list of tabs, choose the "Scope" tab, and in the "**Remote IP Address**" area of this tab, choose the radio button labeled "**These IP addresses**:"



If you have not already done so, identify the network addresses of the Client A and Client B workstations. At both workstations, enter "**ipconfig**" at the command prompt and press ENTER, and write down the current "IPv4" address of the active network adapter, just as you did earlier for the server computer. We will block the Client A address from connecting to the server.

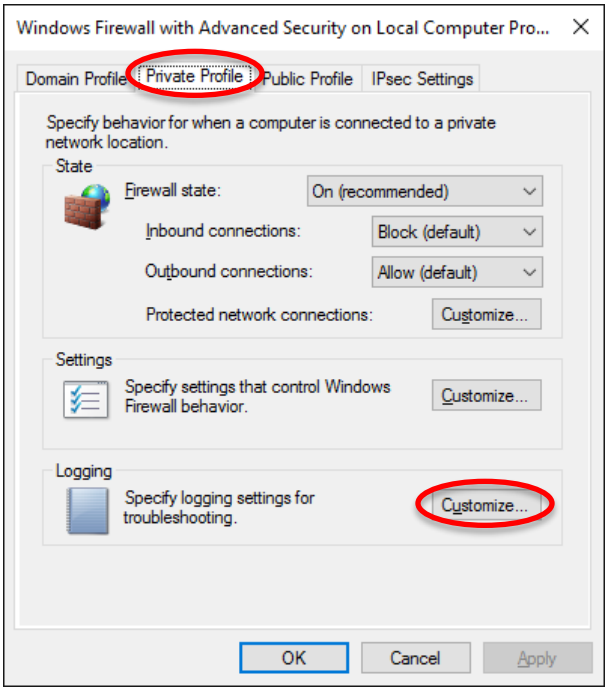
Click the "Add" button in the "Scope" tab, which will open an "IP Address" window. In the "This IP address or subset" field, enter the IP address of Client A (this will be different from the sample shown below), then click "OK":



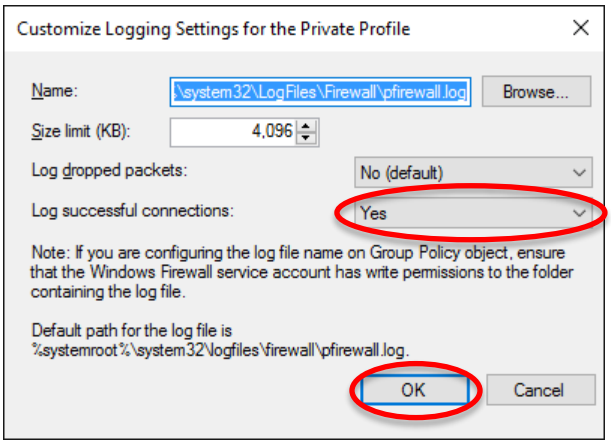
Click "**Apply**" at the properties window, then click "**OK**" to close it. Again, the new rule takes effect immediately, so Client A and Client B should attempt to connect again. As stated earlier, it is not necessary to re-enter the server information. At this point, Client B should be able to connect, but the connection attempt from Client A should fail (it will automatically time out within a few seconds). Client B should disconnect at this point to exit the chat session.

Next, remove the restrictions so that both clients can connect again. Open the properties of the "**TEATalk**" rule again, select the "**Scope**" tab, select Client A's address in the "**Remote IP address**" area and choose "**Remove**", then select the "**Any IP address**" radio button. Click "**Apply**" and "**OK**" to close the window. We will test this with both clients in a moment.

Windows Firewall also provides the ability to log network activity, including *dropped packets* (network traffic forbidden by the firewall) and successful connection attempts. This is useful for *security auditing* and for *intrusion detection*, and for troubleshooting new firewall rules. To enable this feature of the firewall, reopen the "Windows Firewall with Advanced Security" window (if you do not already have it open) by choosing "Advanced Settings" from the Windows Firewall menu, then choose "**Properties**" from the Action Pane on the right. A properties window for the firewall will appear:



Open the "**Private Profile**" tab in this window, then click "**Customize**" in the "**Logging**" area. This will open another window containing a variety of logging options, including the log size and location, and the types of events that should be logged by the firewall (see the screenshot on the next page):



Change the "**Log Successful Connections**" option to "**Yes**", then click "**OK**". After you are returned to the properties window, open the "**Public Profile**" tab and repeat the previous step, enabling logging on both profiles. Click "**Apply**" and then click "**OK**" to commit your changes.

Now, both clients should test the connection. Both should be able to connect, and the firewall should log their connections. To view the log, select "**Monitoring**" from the Console Tree of the "Windows Firewall with Advanced Security" window, and look in the "**Logging Settings**" area for the log file name. The name should be displayed as a clickable link. Clicking the link will open the log; toward the top of the log file, you should see an entry similar to the following:

```
2016-07-10 14:23:12 ALLOW TCP 192.168.1.50 192.168.1.51 25592 1500 0 - 0 0 0 - - - RECEIVE
```

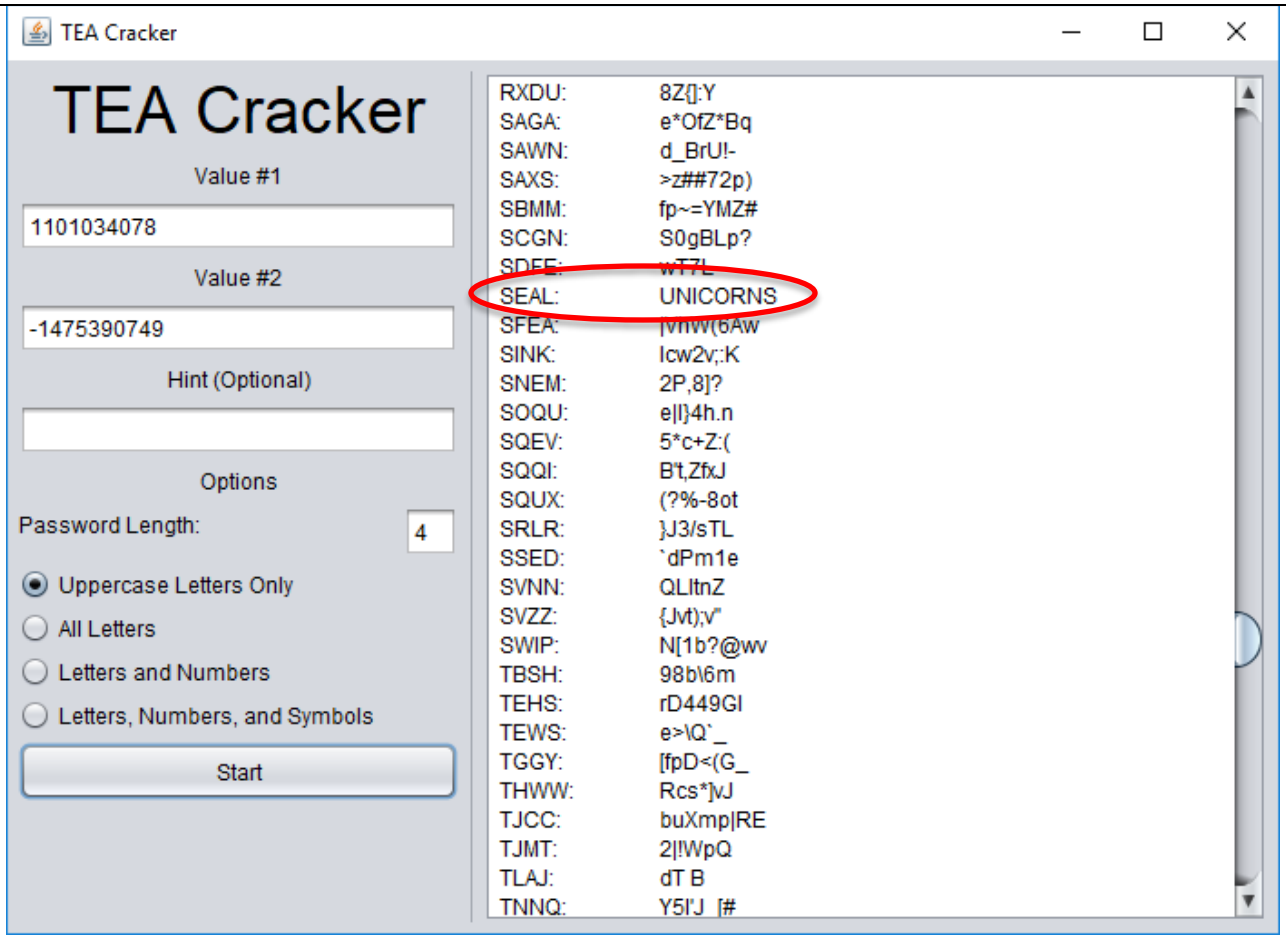
This indicates the *source* and *destination network addresses* of the chat connection; in this example, these addresses are **192.168.1.50** and **192.168.1.51**, respectively. The TEATalk client and server windows should now be *closed* on all three workstations before proceeding.

Part Two: Password Circumvention and Cracking

Now that we have some experience with firewall configuration, all three computers in your group will now assume the role of chat clients, connecting to a shared chat server in the lab. This server will be configured in the same way as the chat server from Part One. The IP address of the shared chat server is **172.20.0.131** (if this has changed, it will be announced in the lab). Open the chat client as described earlier on all three workstations, specifying your name, the password that was assigned to your group, and the shared chat server's IP address.

In order to crack the chat server's passwords, it will be necessary to capture a sample of active chat traffic. During this exercise, all groups in the lab will be directed to log in and begin chatting at the same time, while their chat traffic is recorded using the Wireshark *protocol analyzer* (described in more detail in a moment). This recording, in the form of a Wireshark *capture file*, will be posted to the GenCyber website immediately afterward for your use.

The chat program uses TEA (the Tiny Encryption Algorithm, from which it gets its name) to encrypt your messages by breaking up the message text into eight-character *blocks*. Each block is encrypted, using the chat password as the key, and *encoded* as two 32-bit numbers (*signed integers*). If the block length is not evenly divisible by eight, it is padded with spaces; these spaces are then trimmed from the block after it is decrypted. The integer pairs are transmitted over the network as plain text; this is intended to make them easier for you to recognize.



The password cracker in the "**Module7**" folder, **TEA Cracker**, decrypts samples of TEA data eight characters at a time using a simple *brute-force attack*. It requires the two integer values containing the encrypted data in a single eight-character TEA block, which you can enter into the corresponding fields (just remember to enter them in the correct order!). The password cracker simply generates every possible permutation of every possible password, then attempts to *decrypt* the data using these generated passwords. The results (and the corresponding password) are printed in the text pane on the right-hand side of the window, *if* the decrypted plaintext was found to contain only printable characters.

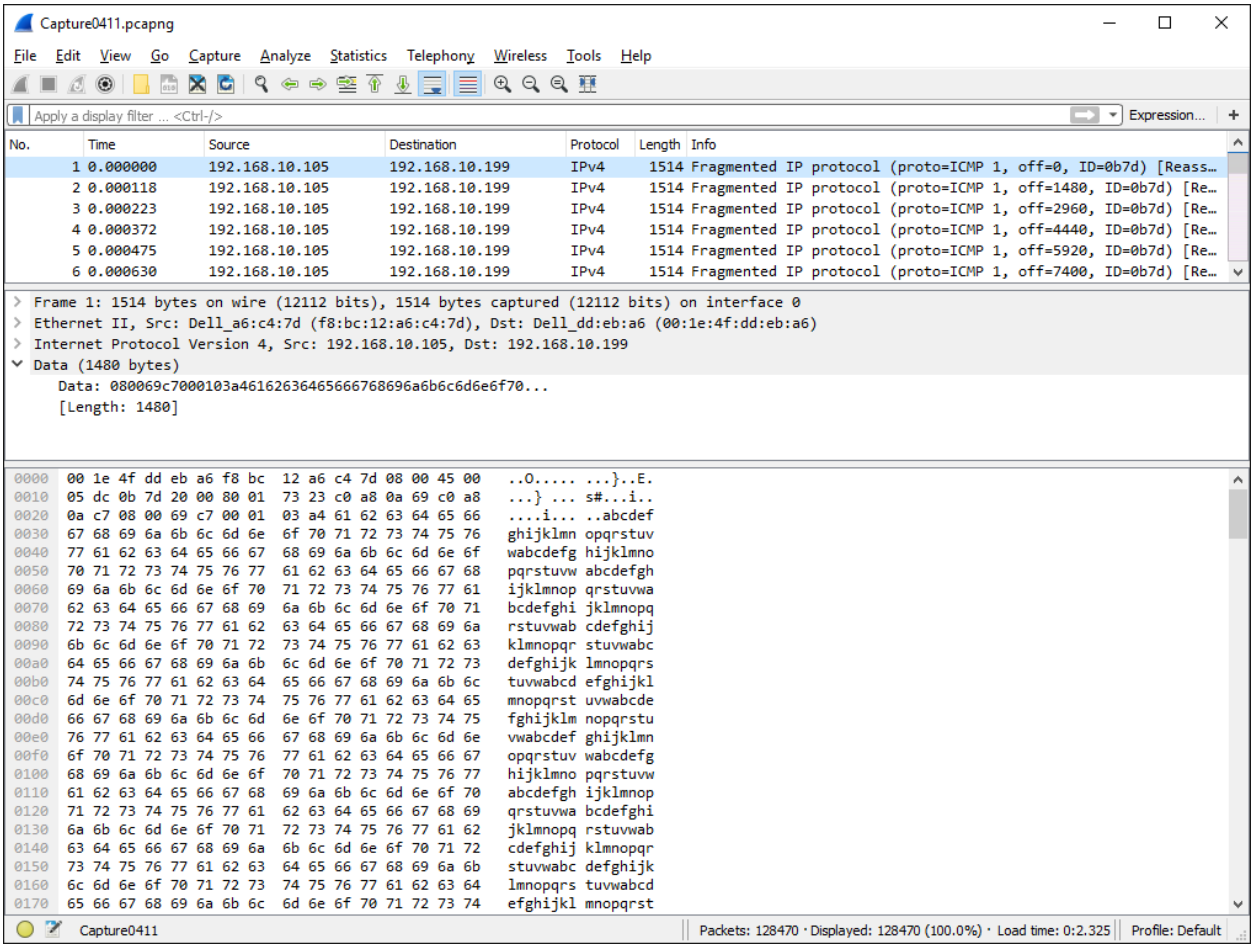
This method results in a lot of false positives, but it still considerably reduces the number of results that you will have to manually inspect. The decision to limit the passwords to four upper-case letters in our chat system was intended to reduce the number of results, and the time required to check them all, even further.

When you find an eight-character plaintext string which appears to be a snippet of a recorded chat conversation, the corresponding password is the password that you will be able to use to "break in" to that person's chat session in the lab.

For example, suppose that the captured TEA data is **`[1101034078, -1475390749]`**. Entering these two integer values into the corresponding fields, *making sure to include the minus sign in the second value*, yields the results depicted in the screenshot shown above. Most of the attempted passwords shown in the right-hand pane resulted in "garbage," but if you look carefully, you should be able to spot an eight-character snippet of readable text. In this case, the decrypted plaintext is "**UNICORNS**", and the password that was used to encrypt it was "**SEAL**", the same sample password that we used earlier. We now know that any other chat traffic generated during this session can be decrypted using the same password. Another password, using a real capture file, is given on the next page.

To get the encrypted data from the recorded chat session, download and extract the capture file from the GenCyber website, which will be called **"Capture2017.zip"** (note: this file will extract to a few hundred megabytes in size, so the extraction may take a few minutes). The traffic captured in this file will include several simultaneous chat sessions and is stored in the form of *network packets.* You will be opening the capture file and analyzing these packets at your own workstation using Wireshark. When you find a packet which appears to contain chat data, you can copy this data to a file and run the password cracker on it to determine the author's password.

When you open the file in Wireshark, you will see a window similar to the following:



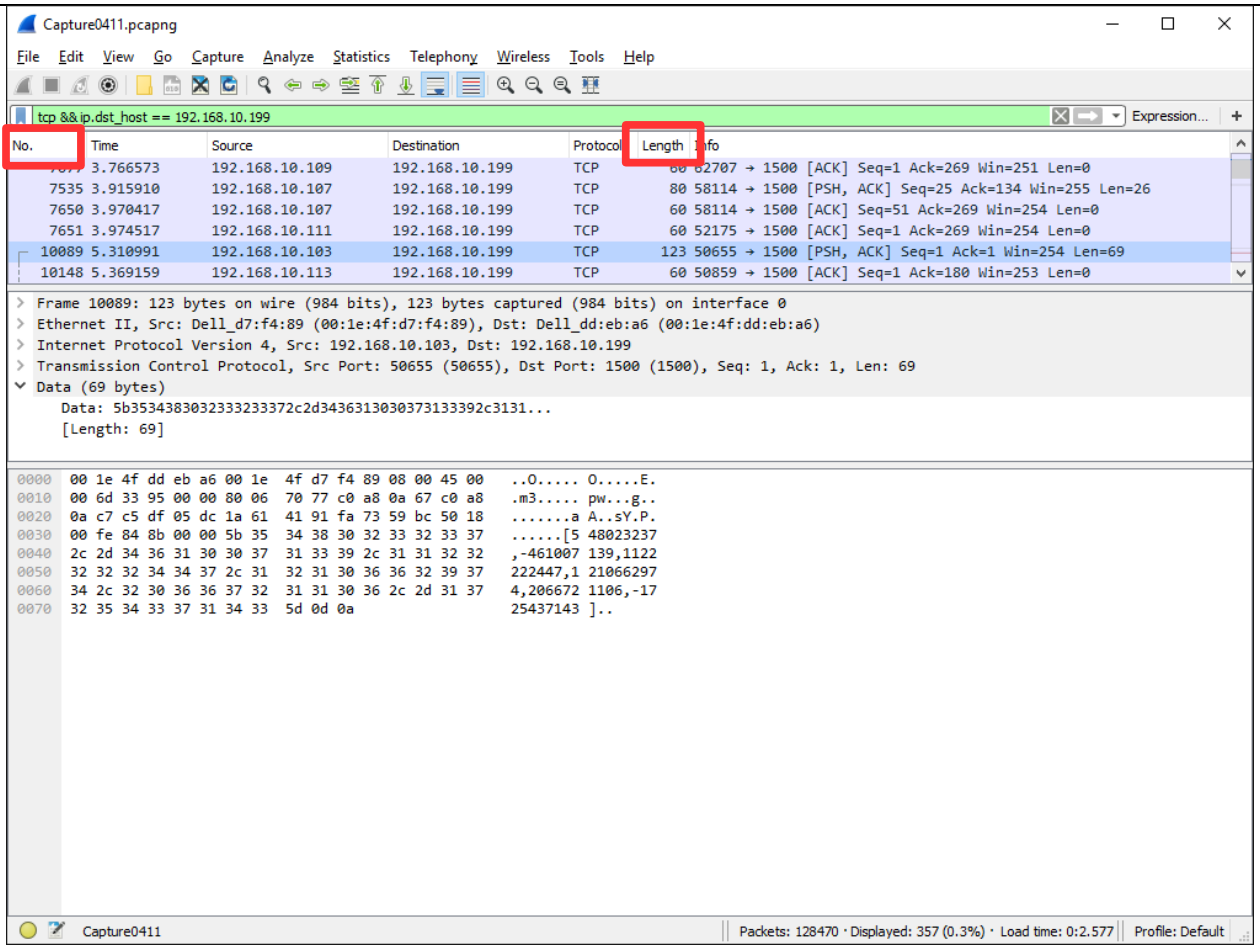
Notice that this window is divided up into three areas: the top area contains a list of individually numbered packets, the middle area contains information about the selected packet, and the bottom area shows the raw data contained within this packet.

To pare down the list, Wireshark allows you to apply a *filter*, which will exclude any packets from view which do not match the criteria you specify; you can enter the filter into the field directly above the packet list (containing the text "**Apply a display filter ...**").

Apply the following filter to exclude all packets which are not TCP packets and which are not directed to the chat server (which is at address **192.168.10.199** in this example):

```
tcp && ip.dst_host == 192.168.10.199
```

After a few moments, the irrelevant packets in the capture file will be hidden. Scroll through the list of remaining packets, selecting each one to see the data inside it. Eventually, you should find a packet similar to the one shown on the following page:



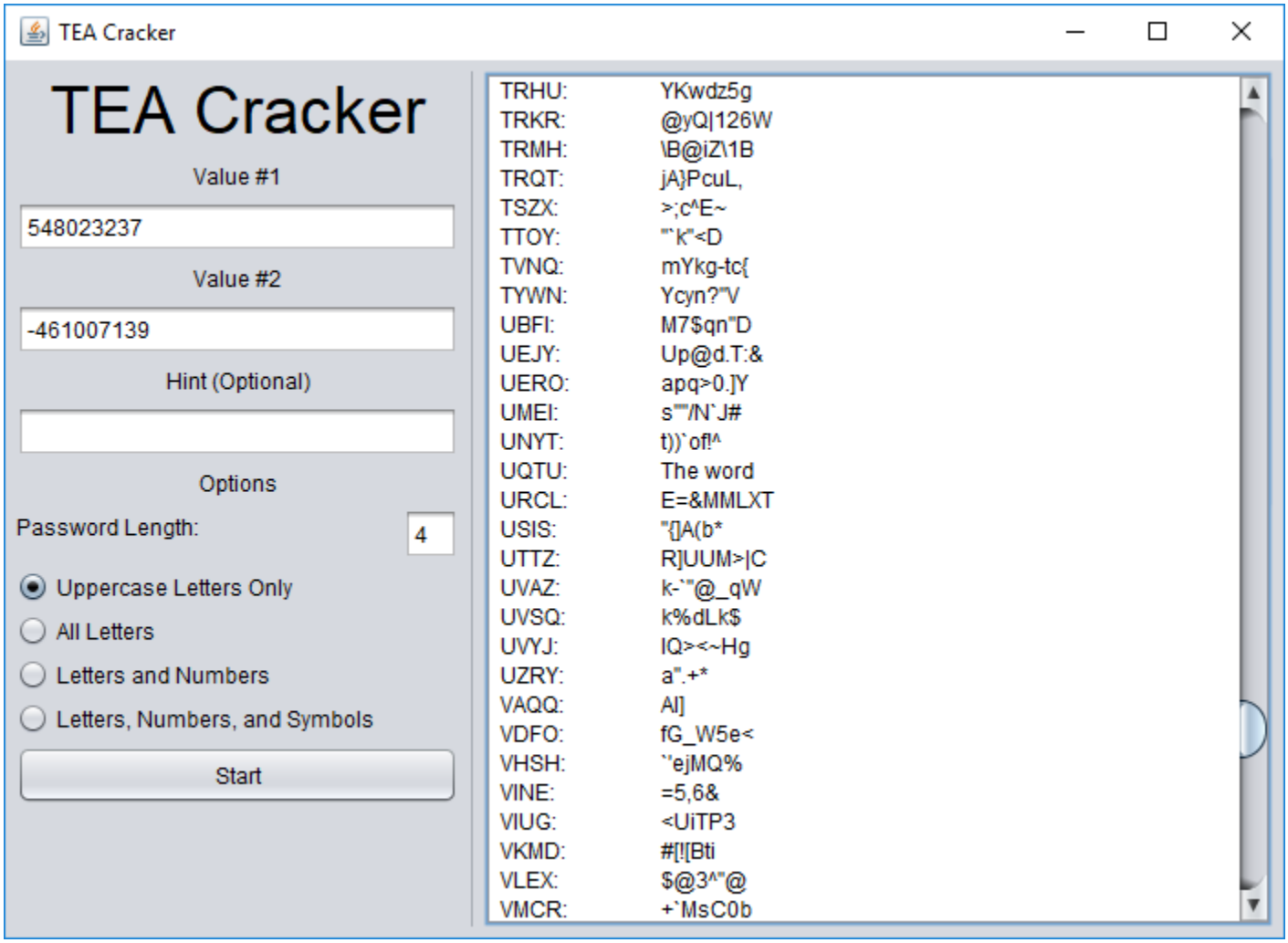
Notice that the data in this packet (shown in the bottom area) includes a *comma-delimited list* of numbers, enclosed in square brackets. This is how the encrypted chat data is transmitted over the network. To more quickly identify packets that contain chat data, check the size of the packets, shown in the "**Length**" column highlighted above; packets with long chat messages will have larger sizes. Also, when you find a chat packet, note the packet number in the "**No.**" column (also highlighted above).

To extract the data from this packet, copy its contents to the clipboard. Right-click the packet in the packet list (the top area), open the "Copy" submenu, and choose "… as Printable Text" from the submenu. You can then paste this data into a text editor, such as Notepad (you will find a shortcut to Notepad on your desktop). For the packet shown in the example, this data will resemble the following:

```
OOEm3pw
g
aAsYP[548023237,-461007139,1122222447,1210662974,2066721106,-1725437143]
```

The "junk" at the beginning of the packet, outside the array of numbers (the portion enclosed in square brackets), can be ignored.

Now, you are in possession of encrypted data that you can crack with the password cracker. Launch the "**TEACracker**" shortcut in the "**Module7**" folder. Enter the first pair of numbers collected from the packet as "**Value #1**" and "**Value #2**", then accept the default options and click the "Start" button. The results are shown on the next page.



At first glance, these results may not seem to contain any valid messages, but remember that they must be searched carefully. Keep scrolling through the remainder of the results, closely checking the second column of the output to look for legible text. When you find it, the corresponding password in the "key" column is the password that you should be able to use to log in to the author's chat session in the lab. Do you see the decrypted plaintext in the sample shown above?

Once you have a password, open a new client window, and log in using the password that you found instead of your assigned password. If you can connect, that means that you have successfully broken into someone's chat room! Send them a quick message to politely let them know they've been invaded, and see how many other passwords your group can discover!

(If you wish to customize the TEA Talk client, server, or password cracker for use in your own classes, the source code is provided for all three programs in the form of a NetBeans project. NetBeans is a popular Integrated Development Environment for the Java programming language. In the "Module7" folder, extract the "Source Code" archive to a new folder, and from NetBeans, open the project(s) within this folder. See the comments provided in the source code for more details.)

Acknowledgements: