Cpts 427 Covert Channel

Shared Resource – File Size

By Reagan Kelley

Introduction:

A covert channel is a means of communication that uses non-traditional and inconspicuous methods. There are two types of covert channels: timing and shared resource. The type of covert channel I decided to implement was one that uses a shared resource, that being the byte size of files.

> **Definition 16–2.** A *covert channel* is a path of communication that was not designed to be used for communication.

Implementation:

My covert channel is set up to be a one-way communication but could easily be revised to make it so the two processes can communicate back and forth. Thus, in my implementation there is a sender and receiver process. Both processes can write to their respective files but cannot read or write to any file that is not their own. The way they communicate is by detecting the changes in file size of the other process' file.

Both processes execute trivial commands such as ping and arp. They use a pipe to redirect these command outputs to their respective files. As they execute these commands, the files increase in byte size. If P1, the sender process, wanted to relay a message to P2, the receiver process, it will run the ping command and redirect the output to its respective text file. It will continue to do this and add small buffer characters until the byte size of the file indicates an ascii value plus a predetermined offset. P2 will watch the ping file until the file size stops changing. It will use that information to record a character value.

The covert channel communicates in a three-way handshake. Upon detecting and translating the covert data, P2 will apply changes to its file by redirecting the output of the arp command. P1 will see that the arp file byte size has changed and use this as an indication that the message is received. Next, P1 will reset the ping file to a byte size lower than the offset value. P2 will take this change in byte size as an indication the P1 has received confirmation that P2 got the package and is about to send another character. This sequence of transactions continue until a null character is relayed, indicating to P2 that no more characters will be sent. It is also important to know that while P1 and P2 are using ping and arp output to fill their files, it doesn't matter what is put into the file, but in this example, these are the commands used to add content to the file.

Noisy or Noiseless?

A covert channel is said to be noiseless if each transmitted character sent by the sender is the same as those received by the receiver with a probability of 1. The means by which my two processes communicate is one that I would argue to more reliable than other covert channels because of little interference or impedance from other programs. My covert channel is more likely to

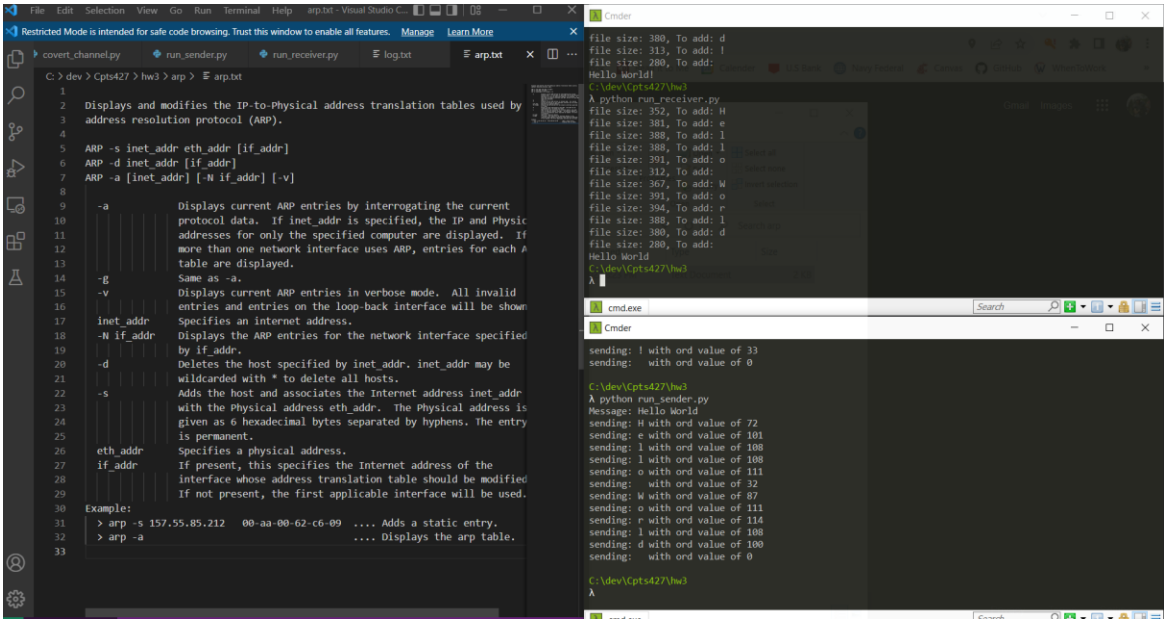
lose information due to timing issues in communication, however I try to mitigate this by using TCP protocol, which does lower latency but nonetheless lowers noise.

Potential Detection

If a defender was looking at the right spot, it would look suspicious for two processes to be looking at file sizes when they ostensibly are unrelated programs, but I think any covert channel is jeopardized if something is looking for that channels' ominous behavior. But, my covert channel would be at risk of being detected if a defender was looking for a pattern in file size changes. If they were to convert the byte size of my file to ascii values, it would see the message. However, this can be easily avoided if the covert channel uses some stenography and adds a shift to the values with an offset. That is what I do in my implementation, as both the sender and receiver agree to add a certain number of bytes to their files not only so the file size can be bigger than 255 (or the maximum readable ascii values) but to remain surreptitious.

Proof of Concept

To show such a concept, I created 2 separate python classes that when run in parallel creates a covert channel.



Demo Video:

https://wsu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=ea7bff91-dfe0-4e4f-87bb-ae8701748ce0&start=0

Assignment Time Log

4/22/2022	**Research** What qualifies as a covert channel? Timing-based vs storage-based
4/25/2022	**Covert Channel Idea** Will use file size to communicate between processes
4/26/2022	**Proof of Concept** Worked on python code to implement my covert channel
4/28/2022	**Proof of Concept** Finished python implementation
4/29/2022	**Reflection** Answering HW3 questions (noise, detection)
4/30/2022	**Assignment Complete** Added needed citations and demo video

~ Citations ~

Bishop, M., n.d. *Introduction to Computer Security*.

Salwan, N., Singh, S., Arora, S. and Singh, A., 2022. *An Insight to Covert Channels*. [online] Arxiv.org. Available at: <https://arxiv.org/ftp/arxiv/papers/1306/1306.2252.pdf#:~:text=Effect%20of%20Noise&text=With%20covert%20channels%2C%20each%20symbol,rates%20in%20a%20communication%20channel.> [Accessed 30 April 2022].