

# Vista Least Privilege Lab

## Due

March 6, 2008 in compass by midnight.

## Goal

Use techniques for least privilege to implement a simple client server program.

## Scenario

Bob Co has retained you for another assignment. Bob Co needs to implement a data management package, and they are considering using Vista as the OS, but they are unsure of how well the security mechanisms of Vista would support them. They hired you to prototype the solution on Windows Vista and evaluate the security support.

Their application needs to perform some privileged operations (to access very secret files) regardless of who invokes it. After this initial privileged operation, the program must read and write files that are only accessible to the invoking user.

## Requirements

Bob Co already has a simple client server program implemented that implements the basic communication but minimal security and no core functionality. On Windows, you would install the server as a service. It listens on a named pipe for client requests. The client passes in the full path name of the file it wants the server to access. The server will try to access the file as itself, and then as the user. The server should log the results of these access attempts to a log file. Assuming it is successful; the server will return the first 512 bytes of the file. Instead of actually installing a service, you can use the “runas” utility to invoke the server and client as different users for testing.

Run your Server as a member of the administrative group. The program should examine and disable all unnecessary privileges at the start of the program. It should log the privileges and their original states to a log file.

Bob has also heard about the new mandatory integrity controls implemented in Vista. He is interested in how that mechanism really works and wants you to experiment with that mechanism too.

## Windows Base Code

Very simple client and server Windows code is posted to the web site under the assignments tab. The server can run from the command shell or as a service. If run from the command line (or from the debugger) you must pass it an argument such as -noService. If the server is invoked with no arguments, it assumes it is being invoked as a service.

If you are operating in the lab, you will want to change the name of the pipe to avoid conflicts with other student's services.

The sample code was developed with Visual Studio 2008 professional and that is what is installed in the lab.

## **Windows Documentation**

The list at [http://msdn.microsoft.com/library/default.asp?url=/library/en-us/secauthz/security/authorization\\_functions.asp](http://msdn.microsoft.com/library/default.asp?url=/library/en-us/secauthz/security/authorization_functions.asp) includes the functions you will need for impersonation and privilege manipulation. Specifically, the following functions should be of interest to you:

- ImpersonateNamedPipeClient
- RevertToSelf
- AdjustTokenPrivileges
- GetTokenInformation

Integrity Control documentation is at <http://msdn2.microsoft.com/en-us/library/bb625964.aspx>.

## **Notes on using MS Visual Studio (MSVC)**

In the sample server and client directories you will see .sln and .vcproj files. Double click on these files to open up MSVC.

Look at Build menu to compile. Error messages will appear in the bottom window and compile errors should be highlighted in the code.

Look at debug menu to run your program. Running without debugger will leave the console window up after executing. Running with debugger (F5) will remove window after program has exited. You can of course also run program from cmd shell directly. By default, the executable will be in the Debug subdirectory.

Look at Project->Properties to set arguments for the program to use while debugging. Specifically look at the Configuration Properties->Debugging window. Fill in the Command Arguments. You will need to pass the -noService argument to the sample server so it will not start up as a service.

Consider setting breakpoints to help your debugging. To set a breakpoint, select a line and right-click.

Select the breakpoint menu option.

## **Your tasks**

You will need to perform the following tasks.

1. Create the client and server programs for windows that match the requirements of the scenario. Run the following cases where the service is running as an administrative user (Alice or root) and the client or unprivileged portion of the application is running as an unprivileged user (e.g., Bob):
  - a. Client asks for file that both have access to.
  - b. Client asks for file that the client has access to.

- c. Client asks for file that the administrative user has access to.
2. Explore the mandatory integrity control mechanism.
  - a. Who can assign labels to files?
  - b. Invoke the server at a lower integrity level. How does this affect the client's named pipe communication?
  - c. Can you apply the no read up policy to some files? How does this work?
3. What you would recommend to Bob Co if they decide to proceed with an implementation on Windows Vista? Which, if any, security mechanisms should they apply? What vulnerabilities would remain?

## Hand-in Items

Hand in on compass. Provide the following.

- Server log from running the operations in the first task.
- Code for the client and server
- Description of what you learned about the mandatory integrity mechanisms.
- Final recommendation report.