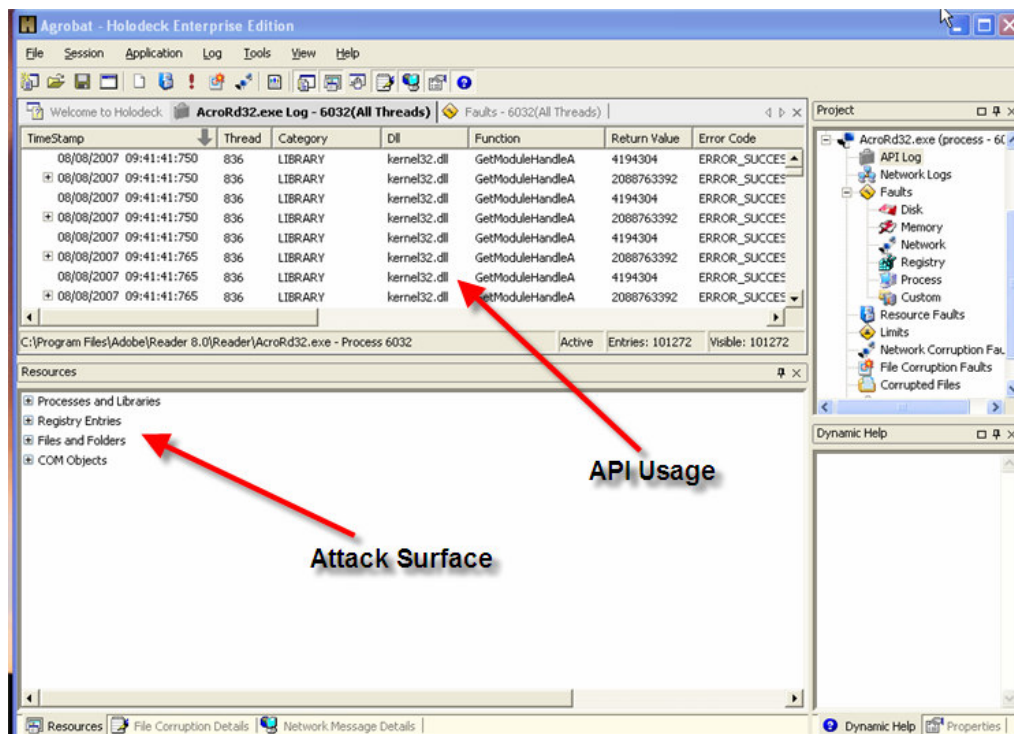


IDENTIFY AN APPLICATION'S ATTACK SURFACE

An application's "Attack Surface" is comprised of all the areas where an application may be exploited. Holodeck provides a view of all these points through its "Resource" pane where it lists all of the resources that an application consumes, hence all of the places it may be vulnerable. Using the Resource pane users can explore all of these areas looking for opportunities to attack and exploit.



STEPS

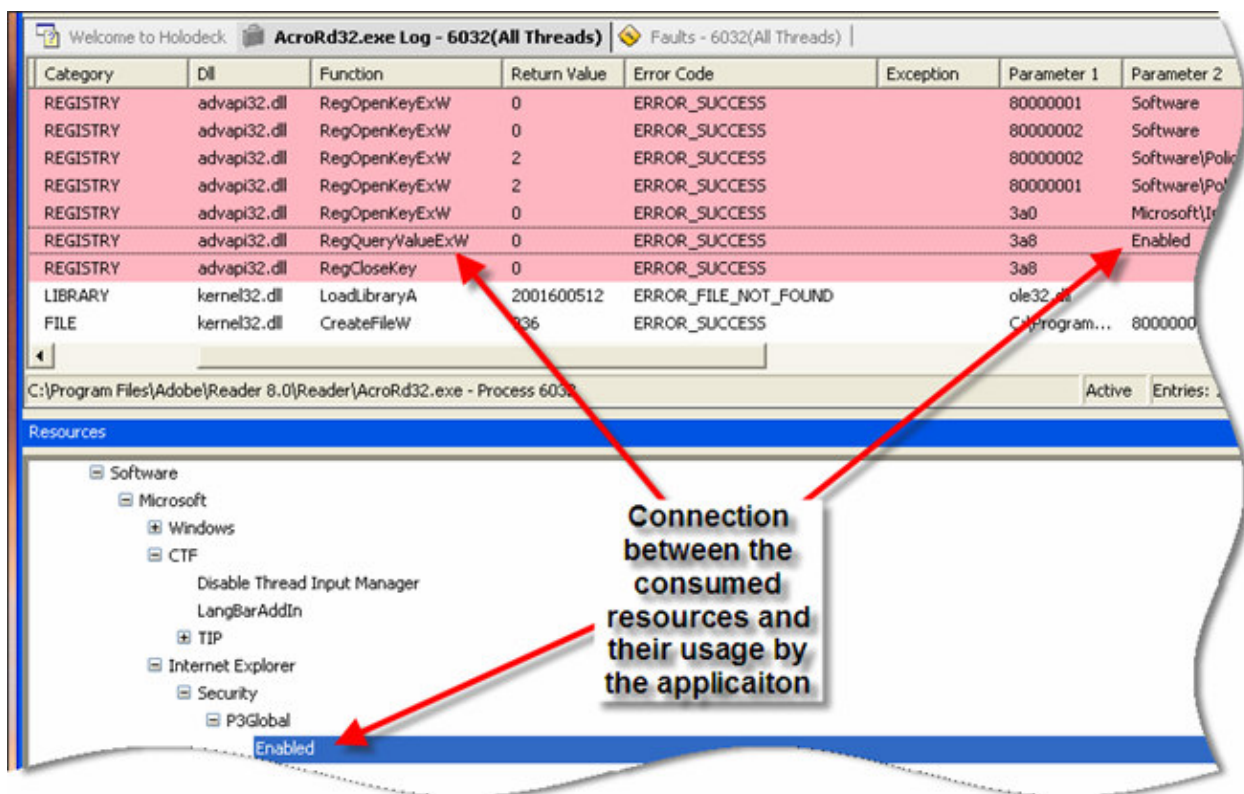
1. Start a new Holodeck project with Adobe Reader 8.x
2. Open a **pdf** file to view
3. Open the Holodeck resource pane
4. Explore resources

All the resources an application consumes are available for perusal by expanding the various nodes, for example, expanding the "Registry Entries" node lays out all of the areas in all the registry hives that an application touches.

TRY THIS (OR SOMETHING SIMILAR)

1. Open "*Resources/HKEY_CURRENT_USER/Software/Microsoft/Internet Explorer/Security/P3Global*"
2. Right Click **Enabled**
3. Click "*Select most recent log entry*"

Notice that the Log view now highlights the last usage of that specific key. Scrolling the log to the right provides all of the parameters used by the call including data and return values. Try this with other resources and see how they are used by the application



The fact that a resource is consumed by an application may be significant, and it may not. One way to find out is to mess around with it a bit and observe how an application behaves when it's missing or changed.

TRY THIS (OR SOMETHING SIMILAR)

1. Select COM Objects/Microsoft Web Browser
2. Right Click and Click **Select most recent log entry**
3. Right Click and Select **Create a Fault**
4. Click **Next** to accept the objects resource path
5. Select **COM Object Does Not Exist**
6. Click **Next** then **Finish**
7. In Reader select **Help/How To/Adobe Reader Essentials**

Reader will now proceed to crash. If you click on **Application/Restart** in the Holodeck main menu you'll find that the application will not start again, and will not until you release the fault by un-checking the **COM Object Does Not Exist** fault in the Holodeck Resource pane.

From this quick exercise we have learned two things. First that Acrobat does in fact depend on the selected COM object and crashes when it is taken away mid-run, and that it may do the "right" thing and check to make sure the object is available and ready for use before calling it at startup, hence the failure to start when the fault is enabled. This is appropriate behavior for a secure application and marks it an unlikely target for any type of runtime attack, but there may be other opportunities.

NETWORK ATTACK SURFACE

An application's attack surface is not limited to local resources, which means that the channels used to communicate with remote resources are vectors for attack as well. Determining if a network stream is part of the application's attack surface using Holodeck involves "fuzzing" the network I/O

channel and observing the results. “Fuzzing” is the practice of corrupting streams in either random or defined patterns and observing the results.

TRY THIS (OR SOMETHING SIMILAR)

1. Start Holodeck with SQLyog (Community Edition)
2. Right Click **Network Corruption Fault** and Click **Create a Network ...**
3. Click **Next, Next, Next, Finish**
4. Log into any Database with SQLyog

SQLyog will start throwing up error dialog boxes that indicate various different types of errors, sometimes it will be a malformed packet error, sometimes an invalid user error and sometimes something else, and sometimes it just hangs or crashes.

What may be happening is that the application is misinterpreting the return values or mistaking the data in the stream for something it's not; in any case it becomes obvious that the behavior of the application can be influenced by the network traffic flowing to and from the database server and therefore, the network stream is a potential attack vector and must be included in the overall application attack surface.

HOLODECK AS AN ATTACK SURFACE ANALYZER

Holodeck places the applications entire potential attack surface at your fingertips and gives you the ability to rapidly expose all the areas where attacks can take place, allowing you to uncover them and mitigate any risk you may discover early in the development process, and before they can cause any trouble in the field.

The screenshot displays the 'SQLyog.exe Log - 1180(All Threads)' window. The log table shows various system events, with one entry highlighted in blue: '08/08/2007 15:54:49:781 3200 ws2_32.dll WSARrecv 0'. A red arrow points from this entry to a callout box labeled 'Random Fuzz'. Below the log, the 'Network Message Details' section shows hex and ASCII data. The ASCII data includes the string 'CHEMATA... Databas' and 'ion_schema... m'. At the bottom, the 'SQLyog Community Edition - MySQL GUI' window is shown with an error dialog box that reads 'Error No. 2027 Malformed packet'. A red arrow points from this dialog to a callout box labeled 'The Application is Actually Hung Now'.

TimeStamp	Thread	Dll	Function	Return Value
08/08/2007 15:54:49:796	3200	kernel32.dll	CloseHandle	True
08/08/2007 15:54:49:796	3200	kernel32.dll	CloseHandle	True
08/08/2007 15:54:49:796	3200	kernel32.dll	CloseHandle	True
08/08/2007 15:54:49:781	3200	ws2_32.dll	setsockopt	0
08/08/2007 15:54:49:781	3200	ws2_32.dll	WSARrecv	0
08/08/2007 15:54:49:718	3200	ws2_32.dll	setsockopt	0
08/08/2007 15:54:49:718	3200	ws2_32.dll	send	19
08/08/2007 15:54:49:718	3200	ws2_32.dll	setsockopt	0
08/08/2007 15:54:49:718	3200	ws2_32.dll	select	0
08/08/2007 15:54:49:718	3200	kernel32.dll	CloseHandle	True
08/08/2007 15:54:49:718	3200	kernel32.dll	CloseHandle	True
08/08/2007 15:54:49:718	3200	kernel32.dll	CloseHandle	True
08/08/2007 15:54:49:718	3200	kernel32.dll	CloseHandle	True
08/08/2007 15:54:49:718	3200	kernel32.dll	CloseHandle	True
08/08/2007 15:54:49:718	3200	kernel32.dll	CloseHandle	True
08/08/2007 15:54:49:718	3200	kernel32.dll	GetModuleHandleW	NULL

Offset	Modified Hex Data	Modified ASCII Data
0x00000000	01 00 00 01 01 31 00 00 l d e f . . S
0x00000010	43 48 45 4d 41 54 41 00	C H E M A T A . . . D a t a b a s
0x00000020	65 0b 53 43 48 45 4d 41	e . S C H E M A _ N A M E . ! .
0x00000030	c0 00 00 00 fd 01 00 00 i n f o r m a t
0x00000040	00 22 00 13 00 00 04 12	" i o n _ s c h e m a m
0x00000050	69 6f 6e 5f 73 63 68 65	y s q l t e s t . . .
0x00000060	79 73 71 6c 05 00 00 06	
0x00000070	07 fe 00 00 22 00	