Information Leakage and Improper Error Handling Lab

Attackers can get information they'll use against us by purposely causing our application to throw exceptions and then examining the error messages we expose. Let's get hands-on with anonymizing our error messages.

Mounting the attack

1. Log in on our web site and add some things to your cart. Go to the check out page.

On the check out page, look at the credit card date. Notice that an option for the month and year is "-". This is obviously intended for the user to be forced to choose a date, but what if they don't? Let's try it.

2. Set your credit card's expiration to "-" and click check out.
3. Oh noes! An error has occurred. Let it pass through to the browser. Look at it and see what the user might see.

Turning specific errors on

Obviously we don't want a typical user to see details at this level.

4. Now change web.config to set customErrors mode="On"
5. Retest. You should see the generic error message.

Create custom error pages

Frankly, this is still too much to show the user. We should reveal much less by not even showing that we're using ASP.NET.

6. Create a new web page called Error.html. It should say something like "An unspecified error occurred."
7. Edit web.config. Add some custom pages like these to the customErrors tag:

```
<customErrors mode="On" defaultRedirect="Error.html">
      <error statusCode="403" redirect="NoAccess.html" />
      <error statusCode="404" redirect="FileNotFound.html" />
      <error statusCode="500" redirect="ServerError.html" />
      Etc. etc.
</customErrors>
```

8. Re-run and re-test. You should see your generic error page but no error message. We're no longer exposing any data to attackers. Way to go!

When you've got your site showing a generic error page, you can be finished.

Bonus!! Capturing the error

Note that since we're not reporting the error message to the user, we don't know when they experience an error nor what the problem is. We can't fix our problems. If you're finished early and are up to the challenge, do the following:

9. Open your global.asax file. Find the Application_Error() event handler.
10. In it, add:

```
System.Diagnostics.EventLog.WriteEntry("WebGoat for .Net",
  Server.GetLastError().ToString(),
  System.Diagnostics.EventLogEntryType.Error);
```

11. Run and test your app.
12. In Windows, find the EventViewer under the Control Panel and find your error.