Clickjacking Lab

We keep getting calls that customers are ordering things that they didn't intend to. And suspiciously they are all accidentally ordering a particular product. Coincidence or hacking? Could it be that some hacker has decided to help out the manufacturer of that product?

We troll the security sites and find out that we're the victim of a clickjacking attack. Let's protect our site from clickjacking.

Testing the vulnerability

1. Log on to .Net Web Goat as a regular user. Do a little shopping. Maybe place an order.
2. In another tab or window, or even in the same window, pretend that the bad guy has tricked you into visiting the evil site.
3. Click on the clickjacking link.
4. Go back to .Net Goat and look at your cart. You should now see a new product in your cart.

Yikes! We've confirmed that our site is not only vulnerable but is an active target of a clickjacker. We've got to fix this!

Protecting with X-Frame-Options

5. Edit site.master.cs. Find the page load event.
6. Add the X-frame-Options header. Give it either of the two values; deny or sameorigin. Do something like this:

```
Response.AddHeader("X-Frame-Options","deny");
```

7. Test the clickjacking page again. It should silenty refuse to put anything in the iframe. Make sure this is the case. Have you fixed the problem? _____ Good!
8. Now we want to learn another technique. So undo your fix. Comment out or just remove the line from the page load event.

Protecting with frame-breaking JavaScript

9. Go into site.master again. This time, edit the page (site.master) and not the code-behind (site.master.cs).
10. Find the head tag and add this inside it:

```
<script>
 if (top != self)
   top.location = location
 </script>
```

11. Let's see how this one works for us. Test the clickjacking page again. It should force the framed page to be the parent. It will take over the evil page and replace it with the legitimate page. Make sure this is the case.

Once you have successfully thwarted the clickjacker, you can be finished.