Outline:

Intro – 5 min

SQL injection protection via parameterized sql (canonical example)

Output encoding (manual in ASPX, automatic in Razor)

Access control in data access code

Permission checks via attributes

API bearer token check via Attributes (Web API)

Static analysis

* Controllers w/out permission check
* UI testing?

Anti CRSF Tokens

* ASP.NET
* ?

Build security into the process (case template w/ security section)

OAuth token management in the news recently

Conclusion – 5 min

* Access control in data access code (C#)
* Permission checks via attributes (C#)
* API bearer token check via Attributes (Web API)
* Permission checks via Annotations (Java?)
* Static analysis
  + Find all controllers without a permission check

Hello, and welcome to “Don’t Write Secure Code”. I’m Seth Petry-Johnson, and unlike many of the other speakers in this track, I am not a security professional. I’m a normal developer that’s “security conscious”.

And as a developer, I’m here today to talk to you about a big problem in our industry.

**(click)**

That problem is that most developers, myself included, hate writing secure code.

It’s not that I don’t want my code to be secure. It’s not that I don’t *care* if my site is vulnerable, and it’s not like I *want* to expose my user’s sensitive data. But the reality is that many developers operate in environments and cultures that may *claim* to value security, but actually provide disincentives for making secure choices. As result, it’s often harder or less rewarding to write secure code than insecure code. And I firmly believe that every good developer is lazy; our psyche is one that values automation, reuse, doing a lot with minimal effort. Our psyche does not seeks out the hard way, and that’s where lots of security holes come from.