# Transcript: IT Podcast - Ep 115 - C836 Lesson 12 - with Arthur Moore and Jessica Galterio

*The following transcript is a verbatim account of the video or audio file accompanying this transcript.*

Speaker # 1: Arthur Moore

WGU's IT audio series, flexible, portable, profound.

Hey, everybody. This is Arthur with WGU. I'm one of the Course Instructors on C836 fundamentals of information security. As a quick note, this series is meant to enhance the learning material, not meant to replace it. Let's go on ahead and get our notes prepared for this particular chapter because it is very technology acronym driven and heavy. Let's go ahead and jump into Chapter 12. Software development vulnerabilities.

A number of common software development vulnerabilities can lead to security issues in our application. These issues are all well known as being problematic from a security perspective. The reasons the development practices that lead to them should not be used is a frequent topic of discussion in both the information security and software engineering communities.

The main categories of software development vulnerabilities include buffer overflows, race conditions, input validation attacks, authentication attacks, authorization attacks, and cryptographic attacks. We got tongue-tied there. Buffer overflows, also referred to as buffer overruns, occur when we do not properly account for the size of the data that we input into our applications.

If we are taking data into an application, most programming languages will require that you specify an amount of data we can expect to receive and set aside storage for that. Race conditions occur when we have multiple processes or threads within a process control or share access to a particular resource. The correct handling of that resource depends on the proper order or timing of transactions.

Input validations, if we are not careful to validate the input into our applications, we may find ourselves on the bad side of a number of issues, depending upon the particular environment and language being used. A good example of input validation is a problem of the format string attacks.

Authentication attacks, when we plan the authentication mechanisms our applications we will make sure to use strong mechanisms. We will help to ensure that we can react to a reasonable amount in the face of attacks. There are a number of common factors across various mechanisms we choose that can help make them stronger. For example, making sure you're using more than just a single factor authentication let's you use dual factor and multi-factor authentication. Authorization, just as we discussed where we looked at authentication, placing authorization mechanisms on the client side is a bad idea as well. Any such process that is performed in the space where might be subject to direct attack or manipulation by users is almost guaranteed to be a security issue at some point. We should instead authenticate against a remote server or on the hardware of the device. If we have a portable device, we are considerably in more control. Cryptographic attacks, we leave ourselves open to failure if we do not pay close attention to designing our security mechanisms. While we implement cryptographic controls in our application, cryptography is easy to implement badly, and it can give us a false sense of security. Client-side attacks. Client-side attacks take advantage of the weakness in the software loaded onto our clients, or those attacks that use social engineering to trick us into going along with the attack. There are a large number of such attacks, but we focus specifically on some that use web style or web attack vehicle. Cross-site scripting, cross-site request forgery, and click jacking. Now, we've been over cross-site request foraging and click jacking before. I will just read cross-site scripting. Cross-site scripting, or XS, is an attack carried out by placing code in the form of a scripting language into a web page or on other media that is interpreted by the client browser, including Adobe Flash animation and some other file videos. When another person views the webpage or media, he or she executes the code automatically and the attack is carried out. We have security server-side attacks, just as we discussed, the client-side attacks, now we're going to take a quick look at the server-side attacks. Just as on the client side, the server side can be riddled with vulnerabilities depending upon the software, operating system, various scripting languages and many other factors that we're using. Here are some of the examples. Lack of input validation, Structured Query Language, SQL injection gives us a strong example of what might happen if we do not properly validate the input of our web application. SQL is a language that we use to communicate with many common databases in the market today.

Improper or inadequate permissions prompt, particularly with web applications and pages, there are often sensitive files and directories that will cause security issues if they are exposed to general users. One area that might cause us trouble is the exposure of configuration files, extemporaneous files where we move a web server from development to production, one of the tasks often missed is the process of cleaning up any files not directly related to the running of the site or the application, or that might be artifacts of the developmental build process.

Database security. As we discussed and went over web security issues, the vast majority of websites and applications today make use of databases in order to store the information they display in process. In some cases, such applications may hold very sensitive data such as tax returns, medical data, or legal records. Or they might contain only the contents of the discussion forum on knitting.

A number of issues can cause us trouble in ensuring the security of our databases. The chronological list includes the following. On authenticated flaws and network protocols, authenticated flaws and network protocols, flaws and authentication protocols on authenticated access to functionality, arbitrary code execution, and intrinsic, arbitrary code execution in securable SQL elements.

Privilege escalation via SQL injection and local privilege escalation issues. Protocol issues, we can take a look at the network protocols used to communicate with the database, some of which we'll need to have a set of credentials in order to make sure that we're properly authenticated. There is often a stream of vulnerabilities for major database products inversions that we might need to take care of to examine.

Such vulnerabilities often involves some of the more common software development issues such as buffer overflows, which we discussed at the beginning of this lesson. On authenticated access, we give a user or process the opportunity to interact to our data place without supplying a set of credentials, we create the possibility for security issues. Arbitrary code execution. We can find a number of areas for security flaws in the languages we use to talk with databases.

Generally these are concentrated on SQL, as it is the most common database language in use, the default SQL language, a number of built-in elements are possible security risks, some of which we can control and others we cannot. Privilege escalation. In each presence, privilege escalation is the category in which we make use of any number of methods to increase the level of access beyond what we are authorized to have or managed to gain and to use system for check. Generally speaking, privilege escalation is aimed at gaining administrative access to the software in order to carry out other attacks without needing to worry about having the required access. With that, ladies and gentlemen, I bring this web series to a close. Congratulations, you have finished all 12 of the lessons for C836, Fundamentals of information security. If you need any help with anything else, I've listed my contact information at the beginning of each lesson. Feel free to get in contact with me. Good luck to you all and I wish you the best educational career here at WGU.

Schedule time with your course instructor to explore more deeply. WGU, A New Kind of U.