User emulation

What is it?

When we talk about user emulation, we want to exactly as the words state. We want to emulate normal user behaviour without trying to hack much if anything at all. First we need to get to know our application before we can know how to hack it so we need to get to know all of the functionality.

What is the goal?

Enumeration is the key to a successful hack. If we don't have the right information, we will never be able to see our avenues of attack and that's exactly why we need to spend as much time as possible in getting to know our application like any user would. We first need to know what normal behaviour is so we can recognise all the details that are not normal.

We want to test all different use levels that are available to us and pay close attention to what functionality can be executed by what different user type. It will serve us well later on when we are testing for Broken Access Control issues.

A great help in this battle can the manual and it might not be linked directly on the webpage we are attacking, but nothing is stopping us from googling a good manual. We might not always find it but we will surely learn something new about our targets.

Are there any pre-requisites?

It is best to perform this kind of testing with a MiTM proxy such as burp suite or OWASP zap open in the background and capturing traffic. We do this so we can already make a good sitemap for when we want to perform our actual hacks later on.

How do we do it?

- Start up your MiTM proxy
- Register to the application as the lowest privilege user possible
- Use an XSS attack vector as any input field you see ('"><img src=x>${7*7}). This will allow us to passively test for XSS at integration points
- Get to know the application by playing with it for a couple of hours before hacking
- Register as a higher privilege user
- Repeat until all users have been tried
- While you test, make an excel sheet displaying which users can do what functionality

When do we do it?

- As early as possible
- Can be combined with other testing as time crunch is an issue
- Always have a scan running in the background (like nmap or nikto but also passive vulnerability scanning in burp)