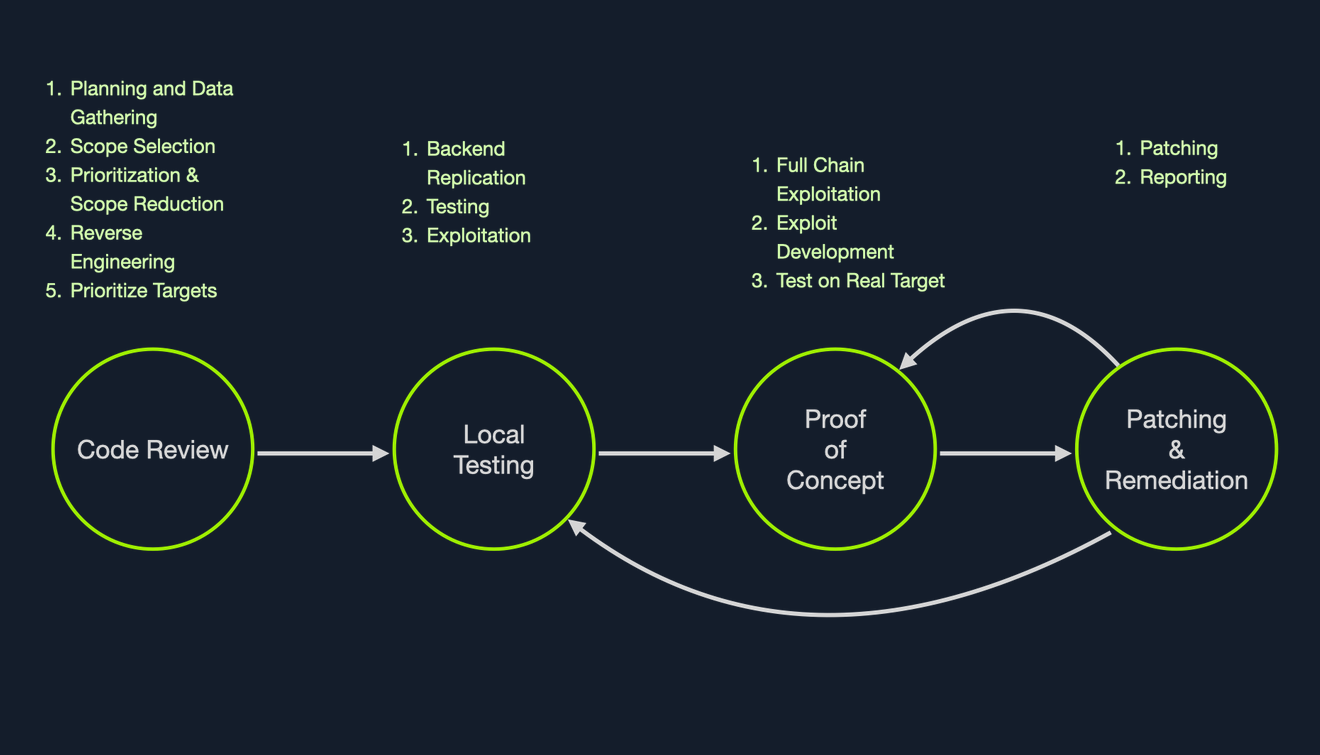
## The Process

Our whitebox pentesting process consists of 4 main steps, as follows:

|  |  |  |
| --- | --- | --- |
| **Order** | **Step** | **Description** |
| 1. | Code Review | General review of the code to understand its functionality and shortlist potentially vulnerable functions |
| 2. | Local Testing | Testing/Debugging the code locally to test our findings and identify vulnerabilities |
| 3. | Proof of Concept | Writing an exploit to prove the exploitability of the target automatically |
| 4. | Patching & Remediation | Patching the vulnerability and all of its sources/causes |



Before diving into a detailed review of each step in the upcoming sections, let's briefly explain each.

## Code Review

The Code Review step mainly consists of a static analysis of the code by reading it to identify potential vulnerabilities. However, combining that with dynamic analysis through application usage for specific vulnerabilities is also possible.

Code review can take the longest in this process, so learning to quickly identify, shortlist, and prioritize interesting functions is essential. This step also requires knowledge of programming languages and application design, without which we could not understand how the application functions.

## Local Testing

Once we have a list of prioritized functions that may have flaws, we need to start our dynamic analysis by testing the prioritized/shortlisted functions and locally testing them while utilizing the tools and access we have. In this step, we would be able to determine whether a function is vulnerable and exploitable.

We would need a testing environment setup that closely matches the application's production environment. A testing environment allows greater visibility into how our requests are handled and how the application processes them, which may enable us to identify certain hard-to-find vulnerabilities.

## Proof of Concept (PoC)

Once a vulnerability (or multiples) has been confirmed through local testing, we need to create a Proof of Concept (PoC) by writing an exploit that automatically exploits and proves the existence of the vulnerability. Doing so allows us to easily replicate the exploit on the production target after testing it on a test environment.

This step also requires scripting knowledge, as we may need to write the PoC in scripting languages like Python, Bash, and JavaScript. We may reuse the application's code to generate certain keys or other hard-to-script functions. Any tests done on the real production target must be done in a safe way that doesn't cause any downtime or data loss.

**Note:** In secure coding modules, we will be mostly focusing on identifying issues in the code and patching them. That is why writing an exploit would not be necessary in such cases, as we only need the exploitation request or a simplified process to replicate the vulnerability as a proof of concept.

## Patching & Remediation

As the goal of any whitebox pentesting exercises is to identify and patch vulnerabilities, it would not be complete without recommending solutions for the identified vulnerabilities. At this step, we would provide detailed descriptions of exact patches with specific changes to the source code such that developers can easily apply them and test them.

We must always test our patches before reporting them to ensure that they both patch the identified vulnerability and retain the original functionality of the code. It is also recommended to provide secure coding tips on how to avoid introducing such vulnerabilities in the future.

Finally, we must re-run our PoC exploit to confirm the application is no longer vulnerable. We should also go through the Local Testing steps again to ensure that all issues have been rectified at every stage.

In the upcoming sections, we will dedicate an entire section for each of these steps to explain them thoroughly. Throughout the module (and other Academy modules), you will also see practical examples of utilizing this process and these steps in an actual whitebox pentest exercise.

**Note:** In these sections, we will not provide practical examples on how to perform each step, as this will be carried throughout the rest of the module, as well as other whitebox modules on HackTheBox Academy. The main point of these sections is to understand the four steps and what each may include. Then, once we start going through the rest of this module, we should get a much better understanding of this entire process and get much more proficient in it.