All this text wasn’t written by me (Tiago a.k.a TigaxMT).

All of it was transcribed by me from HackerOne video lessons.

All credits and thanks go to them.

Threat Modeling

Threat Modeling is a process by which you can determine what threats are important to an application and find points where defenses may be lacking. There are many different types of threat modeling, but we'll be going over two today.

WHY IS IT IMPORTANT?

- Achieve better coverage in testing
  - Make sure you're testing all the entrypoints
- Find more interesting and valuable bugs
  - Have a testing game plan before you start
- Waste less time on dead ends
  - Eliminate entire classes of vulnerabilities before you even start testing

TYPICAL HEAVY-WEIGHT APPROACH

DECOMPOSITION

This is the first step of the typical threat modeling process. The modeler will document each component of the application and its infrastructure, then develop data flow diagrams that show how these components interact. Additionally, privilege boundaries are identified, to ensure that proper controls are in place for any data crossing these boundaries.

THREAT DETERMINATION

Next you develop threats for each portion of the application, e.g. "Attacker may be able to access administration features." You link each of these threats to the components that would be affected in the case of such an attack.
Finally, you rank them by means of an objective measure of severity.

COUNTERMEASURES

You then determine and document any countermeasures currently in place to prevent an attack, along with identifying new locations where countermeasures may be installed to prevent threats you have assessed.

THIS IS USELESS (FOR ATTACKERS)

While this can be a valuable tool for developers and internal security teams, this kind of heavy-weight threat modeling approach is completely unsuitable for bug bounty hunters and most software security consultants.
It requires too much time, too much access to code and internal documentation, and doesn't give you a real game plan for testing.

LIGHTWEIGHT THREAT MODELING

ENUMERATE ENTRYPOINTS

First you make a list of every entrypoint you can find. One simple approach is to enable Burp Proxy and then use every function of the application which you can find, fro every access level you have.

DOCUMENT TARGET ASSETS

Think through and write down every asset in which an attacker may be interested, along with the business impact of its compromise.

- User PII and passwords
- Admin panel access
- Transaction histories
- Source Code
- Database credentials

DEVELOP GAME PLAN

Once you have all of this, you can rank the entrypoints in order of perceived value. For instance, any entrypoint which takes little to no user data will likely be less valuable than one which takes a substantial amount of data. Additionally, you have enough information to eliminate entire vulnerability classes from your tests. E.g. If it's a support knowledge base application, CSRF on any page for end-users will likely give you nothing.

EXAMPLE

Below is an example threat model, breaking down the unauthenticated parts of the HackerOne site.
Using this as a basis, try logging into an account and expanding this threat model to include the authenticated portions of the site.
Example: https://www.hacker101.com/resources/hackerone_threat_model

RESTRICT YOURSELF

When doing this kind of lightweight threat modeling, you should be able to handle the vast majority of applications in under an hour.
If you find you’re going over this, you may be overthinking it and might want to reconsider your approach.