

# Attack Scenarios: How to Get There from Here

Lawrence R. Rogers  
Software Engineering Institute  
Carnegie Mellon University  
Pittsburgh, PA

Remember Star Trek's transporter mechanism? "Beam me up, Scotty" allowed Jim Kirk, Spock, and their friends to be transported from one place to another instantaneously. Well, that is science fiction, but it is an analogy that is useful in describing how computer systems vulnerabilities are exploited.

First, recall that in Star Trek, Spock and Kirk had to be somewhere that could be defined to the transporter. That definition was probably something like coordinates, such as 4 degrees north, 20 degrees west. We'll call that the precondition. Then, after the famous "Beam me up" directive, our explorers magically reappear on the starship Enterprise, ready to move on to the next phase of their mission. We'll call that the impact.

On a computer system, each vulnerability must also have a precondition and, after the vulnerability is exploited, an impact. For example, there is a vulnerability in rlogin that enables users to access their computer accounts from a location other than their desktop computer. The precondition is that the intruder must be able to execute the rlogin program. Usually, that intruder must be logged into a computer system with the vulnerable version of rlogin installed, so we'll call that precondition become a local user. Once the vulnerability is exploited, the impact is that that intruder has now gained the ultimate access to a UNIX system, namely root access (root is the all-privileged user similar to administrator on Windows/NT systems). So, with respect to the rlogin\* vulnerability, we can say that a local user can become root on a vulnerable system.



Now, Star Trek is fiction and, unfortunately, transportation in real life is much more complicated. Say that you are going on a trip and starting from your home. You need to get to the airport, so you get into your car and drive. Eventually, you get to the parking lot at the airport and board a shuttle to the terminal. The shuttle drops you at the curb, and you then walk to security and perhaps to "people mover" inside the airport. At the end of that ride, you walk to the gate, check in, and board the

plane. Once you land, you walk to the local transportation area for a cab, car rental, or public transportation. You eventually get to your hotel and check in. What a trip!

The key to notice here is that each activity dovetails nicely to the next activity. That means that when you parked at the airport, there was a shuttle to take you to the terminal (eventually). When you were in the terminal, there was a way to get to the appropriate gate, and so on. The impact of each activity matched the precondition of the next activity. You can chain the activities together to achieve your goal of “taking a trip.”

There are alternatives. You can build your “take a trip” chain from many different activities. The only requirement is that the impact of one activity matches the precondition of the next activity. For example, you could park at the airport and walk to the terminal. Both the “take the shuttle” and “walk” activities have preconditions that match the impact of “drive to the airport.”

Let’s return to the example of the rlogin vulnerability. Its precondition is to be a local user. To achieve the impact of gaining full privileges, you need to exploit at least one other vulnerability. To determine which one, it’s helpful to have vulnerability catalog in which the preconditions and impacts of vulnerabilities are clearly defined in a language that helps identify the “chain” of preconditions and impacts. After identifying your goal (or ultimate impact) you can chain backwards to whatever precondition suits you, taking into account the operating system version, patches, and so on. This chain of vulnerabilities is an attack scenario.

For all the vulnerabilities in the catalog, you can automatically build all the possible attack scenarios given an initial precondition and an ultimate impact. What is a reasonable initial precondition and ultimate impact? The most general initial precondition is that of an arbitrary Internet user and the ultimate impact is root/administrator privileges. Given those, it is possible to glue together the vulnerabilities according to the precondition and impact directives to construct all possible attack scenarios for achieving the goal.

Now, having done this, imagine that some of the attack scenarios require one vulnerability to achieve their goal. These are home run vulnerabilities-you can achieve your goal in one step. Examples include exploiting vulnerabilities in electronic mail, the web, and FTP servers. Organizations typically support these protocols by passing them through their firewalls and other perimeter defenses and running the associated programs with high privileges. Most vulnerabilities found in these services can be exploited by an arbitrary Internet user to achieve root, resulting in a serious security compromise-the intruder gains complete control of your system.

If your goal is to attack a computer system, it is extremely useful to know all the attack scenarios available to you. If your goal is to defend a computer system, then knowing which vulnerabilities appear in the most attack scenarios helps you to defeat the wily intruders more effectively than by simply fixing each vulnerability that comes to light. If your goal is to search for and describe vulnerabilities, then using a consistent language is crucial to documenting your results.

All vulnerabilities to computer systems are important, either by themselves or in combination with others. Knowing what is possible helps you to defend your systems against those who seek to attack you.

“Shields up!”

“Aye, aye, Captain!”

\* rlogin stands for Remote LOGIN. It enables you to actually login to another computer system someplace else where the rlogin server is running.

The rlogin scheme is intended to connect 2 machines that are “trusted” in that once configured, rlogin does not ask for a password for authentication.

The fact that one computer trusts another means that there is a way for trust to be exploited.

In fact, that is what happened when Kevin Mitnick used this to attack the San Diego Supercomputer Center and Tsutomu Shimomura.