Conquest Web Application Hacker

Daniel Zhang, Abhi Rathod, Tanner Harper, Asad Mahdi

May 5, 2017

Summary

Conquest is a web application attack tool that takes a URL input and automates reconnaissance, vulnerability testing, and exploitation of a target web application. Our script seeks out live web pages belonging to a target address, uncovers vulnerabilities in those pages, and attempts to exploit the pages using known cross-site scripting (XSS) and SQL injection. There is an option for an authenticated execution, where the user provides valid login credentials for further website access.

Example usage:
python3 WebHack.py <website url>
python3 WebHack.py -u <username> -p <password> <website url>

Step 1: Wide Reconnaissance

Conquest uses two different forms of reconnaissance to gather URL addresses of a target's live pages. The first Conquest module runs a crawler that recursively follows all hypertext links beginning at the URL that was provided to the script. In addition to the crawler, the module also utilizes a common files and directories list to forced browse the target website, observing successful responses and adding successes to the list of known live pages.

If username and password flags are passed, Conquest will look for login pages based on observed password fields and attempt to authenticate. Once authenticated, conquest crawls again to discover the pages only available to logged in users.

Step 2: Identifying Vulnerabilities

After all live pages have been gathered, a second module probes each web page to see if the web page has vulnerabilities. At this stage, Conquest parses the victim pages' HTML to identify all input fields. The module then inserts various test scripts and injection strings into the fields and observes how each web page responds. Any pages that appear to be vulnerable are then marked for exploitation.

Step 3: Exploitation

Conquest offers a range of XSS and SQL exploits for users to select in the exploitation step.

- iframe exploit - This stored XSS exploit is set up for comment fields -- typically seen on public forums and blogs. Conquest will scrape the login page of the target website, create an iframe using the scraped login page, and spin-up a minimal server to capture any input submitted into the iframe. The credentials are then saved into an output file on the attacker's system.
- Redirect exploit - A more general exploit, this is another form of stored XSS. This exploit redirects anyone that visits the victim page to a website of the attacker's choosing. The redirect can be customized with the URL of a website that best fits the attacker's needs.
- Table enumeration - After finding a field that is vulnerable to SQL injection, Conquest can attempt to incrementally enumerate the victim's database tables to obtain information such as names of the database, tables, and columns. The user is then then given the option of printing out the contents of the database.
- Authorization exploit - This SQL injection exploit attempts to input attack strings into username and password fields to bypass authorization. If successful, the attacker can then utilize these strings in place of a username and password.
- Cookie exploit - A victim that visits a page compromised by this XSS exploit will have their cookie sent to the attacker. The cookie can be used for stealing victims' browser sessions.

Difficulties Encountered

While the reconnaissance and vulnerability checking modules performed effectively, Team 4 experienced much frustration with SQL exploitation. We knew automating SQL injection would be difficult, but after attempting to integrate it into our tool, we realized exactly how nuanced enumerating just a lightly protected database and accessing information is. Even powerful tools like sqlmap must have output monitored closely by a human and tamper scripts carefully tweaked to outwit protections like filters, whitelists, and blacklists. In the end, we were able to implement only basic SQL injection testing and exploitation.

Coming up with payloads for XSS was also more difficult than expected. It is simple to create specific payloads tailored to our demo website, but making a payload that worked for most vulnerable pages took a lot more time and effort, especially given our team's relative lack of scripting experience.

Future Improvements

- Currently, our SQL exploits only work for SQLite database implementations. The first thing we would attempt is to expand our tool's compatibility with other database systems like MSSQL, MySQL, or postgreSQL.
- The nature of Conquest allows for indefinite expansion. As soon as new web exploits are discovered, they can be readily integrated into Conquest. We can increase Conquest's depth by including more SQL and XSS exploits, and we can expand Conquest's breadth by exploiting a larger variety of vulnerabilities, such as command injection of remote and local file inclusion.
- Currently, the different kinds of exploits use a combination of cluttered terminal print statements and output log files to record information. We feel that our tool would be well suited to a graphical user interface down the road to display information.