|  |  |
| --- | --- |
| Machine Description | SkyTower is a multi-stage VulnHub boot2root that combines SQL injection, an open HTTP proxy (Squid), and exposed credentials to obtain a shell and escalate to root via local misconfigurations |
| Target IP | 10.22.169.4 |
| Vulnerability Name | Unauthenticated SQL injection with proxy/service exposure leading to credential disclosure and remote code execution (RCE) via proxy-mediated SSH access. |
| Service / Version | TCP / 80 HTTP  TCP / 3128 HTTP-Proxy Squid proxy |
| CVSS Score / Severity | 9.8 Critical |
| Attack Vector | CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H |
| Proof Of Concept | STEP 1 – Discover the IP of the target    STEP 2 – Discover Open Ports Using Nmap    STEP 3 – The scan has revealed three open ports: 80 (HTTP) and 3128 (Squid Proxy) so the next thing to do will be to start enumerating HTTP.    STEP 4 – When adding a ‘ to the password field, the login page would produce an error, indicating it is probably vulnerable to SQL injection. While trying to bypass authentication, it appears that the “‘or 1=1–” does not seem to work as MySQL is not commenting the rest of the query, therefore resulting in the below error    STEP 5 – This means that some characters were probably being filtered out or escaped, as the only part of the payload that was left in the query was “’11”   * In MySQL, there are other ways other than “OR” to create an or statement (using ||) and there are other ways of commenting code other than — (using #), like in many other scripting and query languages. Therefore, the following payload will successfully bypass authentication     STEP 6 – This takes to a page that contains credentials for a “john” user. As it turns out those credentials belong to an actual user on the box.   * Since port 22 was filtered as seen in the Nmap scan, Squid can be used to create a tunnel on the target machine that will open port 22 when port 2222 on the localhost is interacted with  1. run proxytunnel to bind local 2222 and forward to target:22 via squid      1. then on another shell run and force a remote shell allocation     STEP 7 – Privilege Escalation When inspecting the login.php file, found some database credentials     * Authenticating into MySQL        * Enumerate Database MySQL login to find other details      * We Got Credentials here   STEP 8 – Switch user to sara using same method used for john     * Checking privileges     Here we got ROOT |
| Impact | * attacker can obtain a shell as a valid user and run commands on the host. * database and cleartext credentials can be harvested and exfiltrated. * proxy allows access to internal services otherwise unreachable externally. * misconfigured local services and SUID/privilege issues let attacker escalate to root. |
| Remediations | * Fix and parameterize SQL queries; remove verbose SQL error messages. * require authentication for proxy services and restrict proxy to trusted hosts. * rotate exposed credentials and enforce strong password policies and MFA. * harden local configs, remove unnecessary SUID bits, and run periodic privilege audits. |
| Reference | <https://portswigger.net/web-security/sql-injection> |