SCENARIO

The application is vulnerable to web cache poisoning because the query string is unkeyed. We will try to poison the cache with a response that executes alert(1) in the visitor's browser.

**PROCEDURE**

1. Open the web application and in the BurpSuite’s Proxy tab send the GET request for homepage to BurpSuite’s Repeater and study it.
2. In order to display the cache key in the response we will include the Payload 1 in the request.
3. Try adding any arbitrary query parameters to the request and we see that it does not matter as we still receive the cached response because they are not included in the cache key generation.
4. Try adding Payload 2 in the request and we see that we can successfully use that parameter as a cache buster.
5. When we get a cache miss, notice that our injected parameters are reflected in the response. If the response to our request is cached, we can remove the query parameters and they will still be reflected in the cached response.
6. According to the way our injected query parameter we will craft an exploit string which will break out of that tag and trigger our alert by executing arbitrary JavaScript.
7. Send the malicious request after removing the cache buster parameter and keep replaying the request until we see our exploit server URL being reflected in the response and **X-Cache: hit** in the headers.

**PAYLOAD**

1. Pragma: x-get-cache-key
2. Origin: [www.hacker.com](http://www.hacker.com)
3. ?hacked='/><script>alert(1)</script>

**REMEDIATION**