***1. My Report On Bugs and Vulnerabilities for (*** [***https://dare2compete.com***](https://dare2compete.com) ***)***

My report is completely based on the security point of view.

This report is prepared on the data by static pen-testing of

the website [***https://dare2compete.com***](https://dare2compete.com).

Before beginning, i would like to inform that the dare2compete

website contains following categories of vulnerabilities:

1.Confirmed - 6

2.Informational - 10

**The 3 which i would like to mention based on their severity and importance with regards to Dare2Compete are:**

***1.[Possible] Phishing by Navigating Browser Tabs***

I observed that dare2compete openes windows through normal hrefs with target="\_blank" which can modify window.opener.location and replace the parent webpage with something else, even on a different origin. While this doesn't allow script execution, it does allow phishing attacks that silently replace the parent.

**Impact**

If the links lack of rel="noopener noreferrer" attribute, third party site can change the URL of source tab using window.opener.location.assign and trick the user as if he is still in a trusted page and lead him to enter his secret information or credentials to this malicious code.

For example this site : <https://reliancetup.in/>

***This is what i requested from browser* :**

GET /amp HTTP/1.1

Host: dare2compete.com

Cache-Control: no-cache

Referer: https://dare2compete.com/76-es5.f1b3219e400eb4b008bf.js

Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,\*/\*;q=0.5

User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36 Accept-Language: en-us,en;q=0.5

X-Scanner: Netsparker

Accept-Encoding: gzip, deflate

***This the response* :**

**</amp-img> </a> </li> <li>**

**<a href="https://reliancetup.in/" target="\_blank"> <amp-img**

**alt="service-banner"**

**src="**[**https://dare2compete.co**](https://dare2compete.co)

**…**

**Conclusion :**

Hence we can use a **Reverse Tabnabbing** attack and create a hologram of that page ( phishing page), and get some credentials to play around and break the privacy.

**2. Robots.txt Detected :**

***Impact***

Depending on the content of the file, an attacker might discover hidden directories and files like the path of an administration panel.

Link of robot file for dare2compete : <https://dare2compete.com/robots.txt>

It is not necessary that robot.txt file may contain desirable information but it shows lack of security measures.

So we should be always active.

This is how it looks like :



**3. Missing X-XSS Protection Header** **:**

a missing X-XSS-Protection header which means that this website could be at risk of Cross-site Scripting (XSS) attacks.

Link to the file : <https://dare2compete.com/runtime-es2015.d0e31b794685186052f7.js>

**Impact :**

Vulnerable to XSS and CSRF(Reverse Forgery)

Apart from this, a lot of other bugs are there like 505 error, poor tls certification algorithm, “option” method is enabled in header of requests, out of version php and CSP,SRI not implemented.

***2. If I found a non-reproducible bug in an application, then I would report it in the following way:***

I am working on a project ( of an application), and I found a bug which isn’t appearing very frequently due to several reasons.

**For example**, the bug is **“ Application crashes upon clicking the SAVE button while creating a new user”.**

But i don't know how it appears and the path of the bug, so i have to report it then:

**SAMPLE BUG REPORT**

**Bug Name:** Application crashes upon clicking the SAVE button while creating a new user.

**Bug ID:** (It will be automatically created by the BUG Tracking tool once you save this bug).

**Area Path:** USERS menu -> New Users

**Build Number:** Version Number 5.0.1

**Severity:** HIGH (High/Medium/Low) or 1

**Priority:** HIGH (High/Medium/Low) or 1

**Assigned to:** Developer-X

**Reported By:** Myself

**Reported On:** Date

**Reason:** Defect

**Status:** New/Open/Active (Depends on the Tool I am Using)

**Environment:** Windows 2003/SQL Server 2005

**Description:** Application crashes upon clicking the SAVE button while creating a new user, hence being unable to create a new user in the application.

**Steps to Reproduce:**

* **If you can’t reproduce a bug, you first document the steps and repeat them under a different environment to find it again.**
* **And if the bug is unable to reappear, then you should ask for more information from the customer/client regarding the bug reproducibility.**
* **If the bug has stopped reappearing at the client side, you can close the bug as “unreproducible” or similar flag in your defect tracking tool.**
* **Capture as much information as you can to write new test cases for the same test steps with different environment, user error and data corruption scenarios.**
* **Check the logging software in the background to identify the system sources when the bug was found.**
* **Ask the client or the test executor to use a screenshot recording tool like Camtasia (or any other tool specific to that operating system) to record the bug.**
* **Compare the environment logs and the software debugger logs before and after noticing such bugs.**
* **Examine the test execution results and evaluate the changes in the tests.**
* **Examine the test data and modify as per the client test inputs or adjust to reproduce the bug.**
* **Add the test steps that get the execution closer to the bug.**
* **Do keep a screenshot or documentation of the error message that appeared during the bug occurrence.**
* **Need more patience (they won’t occur easily)**

**Conclusion :** After following these series of steps, finally we will be able to

Reproduce bugs in an efficient way.