Requirements for Web Application Penetration Test

1. While it's not a strict requirement, we prefer conducting the penetration test on a **non-production environment (E.g.: staging, UAT).** This guarantees that our tests cannot have any negative impact on the production systems, such as service disruptions, performance issues, corrupt data, etc.

2. Access to non-production environments is often restricted. If that is the case, we will need **access to the test environment**. If IP restriction is used, our static IP should be added to the whitelist. For the cases where the test environment is in an internal network segment, we should be provided with VPN access. Alternatively, a copy of the test server may be deployed to a server in an internet-accessible network location.

3. Some web applications may not show menu items or action buttons if the relevant data is missing for the user's account. Such cases could prevent us from testing the entire function set of the application. Therefore, we need some **sensible and realistic data to be loaded to the test system**, which would allow us to cover all possible use cases**.**

4. **The test data should also be sanitized**. Sensitive data or PII (personally identifiable information) such as social security numbers, telephone numbers, email addresses, name-surname pairs should either be replaced with random data, be masked with *** or be obscured using alternative means.

   Some of our test scenarios will involve trying to access other users' data by breaching the authorization mechanism implemented within the web application. While we already have internal controls to adequately safeguard the data that we would encounter during the tests, identifying such a security weakness still should not give us unlimited access to your real customer data, even during a security test. Therefore, we strongly suggest the test data be sanitized.

5. **We would need two accounts for each user role** in the web application. If the application has three user roles, such as "Admin", "Data Entry" and "Approver", then we would need a total of six accounts, two from each role. Also, if the web application organizes the user accounts under entities, the six accounts should be from two completely separate (isolated) entities. Below example is presented for clarification:

Entity A	Entity B
Admin 1 Approver 1 Data Entry 1	Admin 2 Approver 2 Data Entry 2

   Accounts under Entity A should not normally be able to access any data belonging to the accounts under Entity B and vice versa.

The goal here will be to see if we can breach the authorization mechanism of the web application and find a way for “Data Entry 1” to access “Data Entry 2”s data (horizontal authorization tests). Also, we will test if we can find a way for the accounts with lower roles to access the screens, functions or data of an account with a higher role (vertical authorization checks - e.g.: "Data Entry" user accesses any of the "Admin" screens, calls any "Admin" functions or sees/changes/edits/deletes any "Admin" data).

6. **The internal IT Security and IT Operation teams should be notified** about our tests and IP addresses. Necessary internal communications should be made to **add our static test IP to the white list of security products**. This would help us perform all the test scenarios within a more manageable time frame.

7. If the web application requires a **specific browser or browser version**, please let us know.

8. If the application is too large or complex, then we may need a **demo, tour or a training video,** to get to know the application.

9. Please **let us know if you have control over the source code of the application**. If you do not own the source code of the application and making changes to the source code is restricted (due to a software license agreement or cost concerns), then where applicable, we would try to make our recommendations in a way that would require as little source code changes as possible.

10. We will consider any **third-party servers are out of scope** for the tests, such as Google Analytics, JavaScript/Font distributors, CDNs, etc. We should not send such servers any attack packets since they are owned by other companies and we are not allowed to test them. Our test scenarios for such servers would be limited to passive tests (examination of the traffic), to identify unintended data leaks from your web application to those servers (E.g.: sensitive data transmitted to a third-party analytics service in the “Referer” header of a request, etc.). If you additionally would like any of your **servers to be marked as out of scope**, please let us know.

11. If you are using **a hosting provider** for some/all of your servers, please note that your hosting provider(s) may require you to **notify them** a few days before the commencement of the penetration test. Please check with your hosting provider if they have such a requirement (Usually referred to as "Penetration testing policy"). Do not hesitate to contact us for any guidance.