Using Components with Known Vulnerabilities

Description:

In what we believe, it seems that it is very easy to find out or figure out if you are currently experiencing any components or libraries that is vulnerable. But I some cases, reports or issues on vulnerability in a commercial or into an open source software does not always tell he exact of which type is are standardly vulnerable.

Furthermore, not all of the libraries are using a version that is simple or understandable by most of using the system of numbering. The worst case would be, not every single one of the vulnerabilities are being told or reported to a center of house clearing that is not hard to search.

Prevention:

Many of the complex projects does not make or create patches that are vulnerable for the pas versions. But, will surely fix the problem in the future versions. So in upgrading to the newer versions is very important. Project software processes should have the following:

1. You must know all the types and all the versions that you are currently using which includes all the dependencies.
2. Must always monitor the security and types on databases that are public.
3. You must create security policies about the use of the components.
4. Disable all unused functionalities and must always check every aspect of the component.

XML External Entity (XXE)

Description:

The XML External Entity (XXE) Attack is considered to be a type of server-side request forgery. Basically, it means that attackers can use xml files as a way to bypass the firewall in order to access the files within the server

Attackers will issue a request to the server that contains the xml commands within it, and as long as the server accepts the request or has the capability to parse xml files the attack will be successful. The XXE attack is often used for DDOS attacks (Billion Laughs Attack) or it can be to just steal or tamper with the files in the server.

Insecure and vulnerable web applications are often exposed to these kinds of attacks. As long as the attackers know sufficient information about the server like the directory structure of the website, they will be able to use the attack to achieve their purpose or they can just wreak havoc in the server.

Prevention:

* As long as it is possible, do not use so many complex data formats, for example is JSON, and you must not encounter serializing the data especially when they are sensitive.
* All the XML processors and the libraries must be up to date in use by the application or in the operating system.

Insecure Deserialization

Description:

Insecure Deserialization is a vulnerability which usually happens when an untrusted data is being processed or is being used for the misused of the applications logic, which is commonly inflicts service denials (DoS) attack, or even process or execute arbitrary code upon being deserialized.

Prevention:

1. Yu must not accept those objects that are serialized.
2. The process of serialization must always be encrypted for the hostile object creation and data will not run.
3. You must also run the deserializaion code.