Weak private key generation

Moderate **drieseng** published **GHSA-72p8-v4hg-v45p** on May 29

Package

SSH.NET (.NET)

Affected versions

2020.0.0, 2020.0.1

Patched versions

2020.0.2

Description

During an **X25519** key exchange, the client's private is generated with System.Random:

```
var rnd = new Random();
_privateKey = new byte[MontgomeryCurve25519.PrivateKeySizeInBytes];
rnd.NextBytes(_privateKey);
```

Source: KeyExchangeECCurve25519.cs

System.Random is not a cryptographically secure random number generator, it must therefore not be used for cryptographic purposes.

Impact

When establishing an SSH connection to a remote host, during the X25519 key exchange, the private key is generated with
a weak random number generator whose seed can be bruteforced. This allows an attacker able to eavesdrop the
communications to decrypt them.

Workarounds

To ensure you're not affected by this vulnerability, you can disable support for `curve25519-sha256` and `curve25519-sha256@libssh.org` key exchange algorithms by invoking the following method before a connection is established:

```
private static void RemoveUnsecureKEX(BaseClient client)
{
    client.ConnectionInfo.KeyExchangeAlgorithms.Remove("curve25519-sha256");
    client.ConnectionInfo.KeyExchangeAlgorithms.Remove("curve25519-sha256@libssh.org");
}
```

Thanks

This issue was initially reported by **Siemens AG, Digital Industries**, shortly followed by **@yaumn-synacktiv**.

Severity

Moderate

CVE ID

CVE-2022-29245

Weaknesses

No CWEs

Credits

yaumn-synacktiv