Dude, You Got Dell'd

Publishing Your Privates

Darren Kemp (@privmode)

Mikhail Davidov (@sirus)

Kyle Lady (@kylelady)

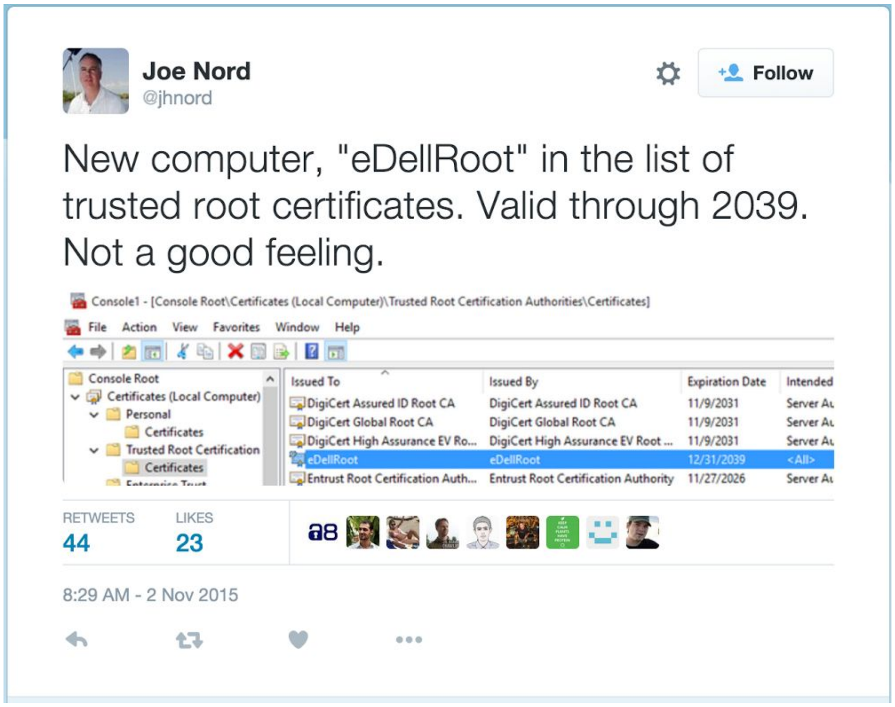
Contents

Questions? Comments? Fan mail?

Reach out to us at labs@duosecurity.com or tweet at @duo_labs on Twitter.

Summary

Recently, Duo Labs security researchers found a few sketchy certificates on a Dell Inspiron 14 laptop we purchased last week to conduct a larger research project. And we weren't the only ones - a reddit thread and some Twitter noise prompted us to share our observations and the real-world impact of our findings.



Joe Nord
@jhnord

New computer, "eDellRoot" in the list of trusted root certificates. Valid through 2039. Not a good feeling.

Console1 - [Console Root\Certificates (Local Computer)\Trusted Root Certification Authorities\Certificates]

Issued To	Issued By	Expiration Date	Intended
DigiCert Assured ID Root CA	DigiCert Assured ID Root CA	11/9/2031	Server Au
DigiCert Global Root CA	DigiCert Global Root CA	11/9/2031	Server Au
DigiCert High Assurance EV Ro...	DigiCert High Assurance EV Root ...	11/9/2031	Server Au
eDellRoot	eDellRoot	12/31/2039	<All>
Entrust Root Certification Auth...	Entrust Root Certification Authority	11/27/2026	Server Au

RETWEETS 44 LIKES 23

8:29 AM - 2 Nov 2015

Notable Findings

- There are two certificates found on Dell machines, including a trusted eDellRoot root certificate
- In the wild, we identified that one of the systems using these certificates for providing web services over HTTPS was a SCADA (supervisory control and data acquisition) system
- eDellRoot is shipped preinstalled with an associated private key, which is a pretty big mistake
- Our research indicates that Dell is intentionally shipping identical private keys in other models
- This means an attacker could sniff a Dell user's web browsing traffic and manipulate their traffic to deliver malware
- We also found another certificate mishap on our Dell machine - an Atheros Authenticode certificate also shipped with the Bluetooth software
- In the interest of full-disclosure, we are including the eDellRoot private key we identified and the entire Atheros certificate bundle here.

Real-World Impact

If a user was using their Dell laptop at a coffee shop, an attacker sitting on the shop's wi-fi network could potentially sniff all of their TLS encrypted traffic, including sensitive data like bank passwords, emails, etc.

The attacker could also manipulate the user's traffic, e.g., sending malware in response to requests to download legit software, or install automatic updates - and make it all appear to be signed by a trusted developer.

A Second Suspect Certificate

The Duo Labs research team also found a second certificate discovered on a small handful of systems on the Internet. Skip to the remediation section to learn how to fix this issue immediately.

Technical Overview

What Are Certificate Stores?

Certificate stores are collections of certificate authorities (CAs) that represent organizations. They're always allowed to approve, or sign, other certificates - similar to the role of a notary public that verifies document authenticity.

These certificates can be used for a variety of purposes, such as SSL connections to websites (the lock icons that you're told to look for) to signing programs as legitimate and officially approved.

For example, Apple's OS X intentionally makes it difficult to run programs that aren't properly signed with a security feature called Gatekeeper, since it's unlikely that malicious programmers will use the relatively traceable process of signing their malicious programs.

Unusual Root Certificate: eDellRoot

We found an unusual root certificate named 'eDellRoot' in the certificate store of our Dell laptop, consistent with what others have found on Twitter and reddit.

What was even more disconcerting was that the certificate also shipped with the associated private key; for those with even a modest understanding of cryptography, this is a pretty big mistake.

The thumbprint for the certificate we discovered is:

98 a0 4e 41 63 35 77 90 c4 a7 9e 6d 71 3f f0 af 51 fe 69 27

and a copy of the private key is also included with this post.

Identical Keys Across Different Dell Models

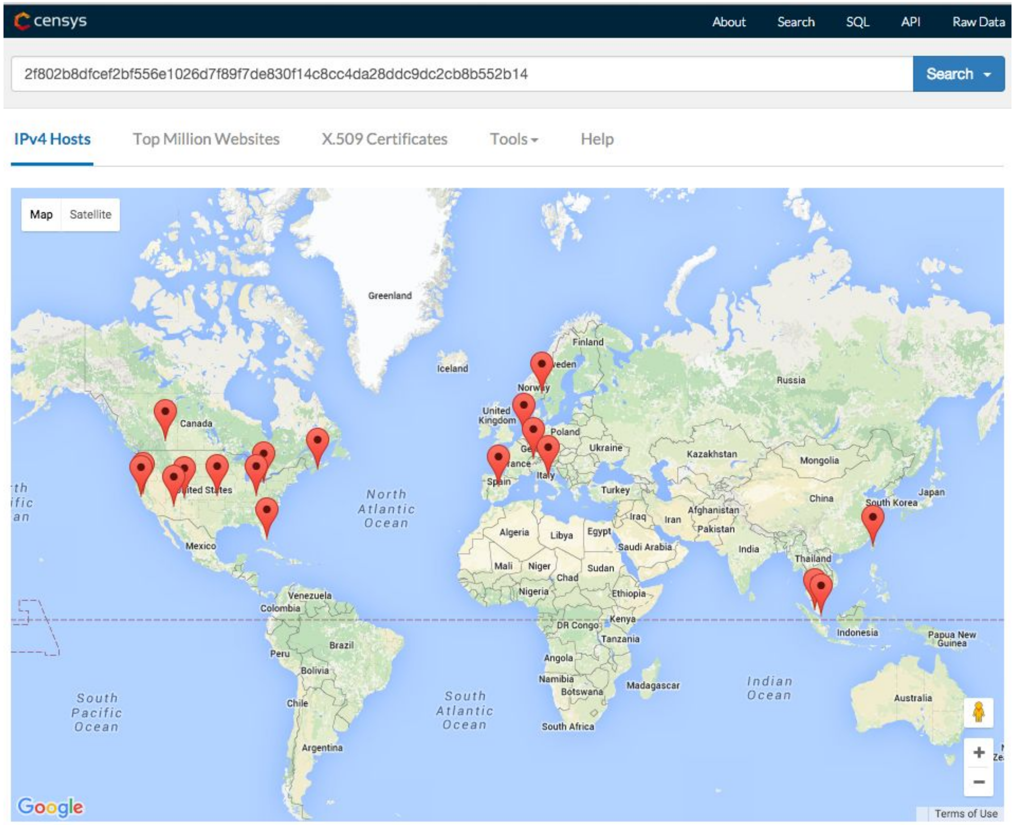
Given that this certificate can be used to sign SSL certificates for secure web communications, we talked to the good people at the Censys project. The Censys project uses zmap to scan the whole IPv4 Internet and archive data, such as the SSL certificate that server sends when the scanner opens a connection. There do not appear to be any servers online that are using the initial eDellRoot CA certificate directly (98:A0:43:[...]).

However, searching Censys for "eDellRoot" turned up another certificate, which was similar to the first one: same name and is signed by the eDellRoot CA. This is the localhost certificate, which is installed beside the eDellRoot CA. Normally, a given certificate would only be associated with one IP address, as it's considered poor practice to share the private component of the certificate across multiple machines. Otherwise, it's impossible to tell which computer actually sent a given message, a property that is often demanded in cryptosystems.

Nevertheless, this second eDellRoot (localhost) certificate (SHA-1 fingerprint: 43:58:95:cc: 86:3e:25:a3: a3:1c:77:9c:a9:35:58:20:d6:80:d9:71) is offered up by 24 different IP addresses, as of the time of this writing. They're scattered across the world and owned by different organizations, which suggests that these are distributed with the hardware. This suggests that Dell is intentionally shipping identical keys in other models. This seems to be a blatant disregard for basic cryptographic security, when the goal of having a cryptographic certificate for Dell software to use could have accomplished by, e.g., shipping a program that generates a unique certificate the first time you boot the computer up.

Note: The above section has been revised to clarify our findings due to a misleading statement.

We don't know which model(s) are shipped with 43:58:95:[...], but this suggests that **Dell is intentionally shipping identical keys in other models.** This seems to be a blatant disregard for basic cryptographic security, when the goal of having a cryptographic certificate for Dell software to use could have accomplished by, e.g., shipping a program that generates a unique certificate the first time you boot the computer up.



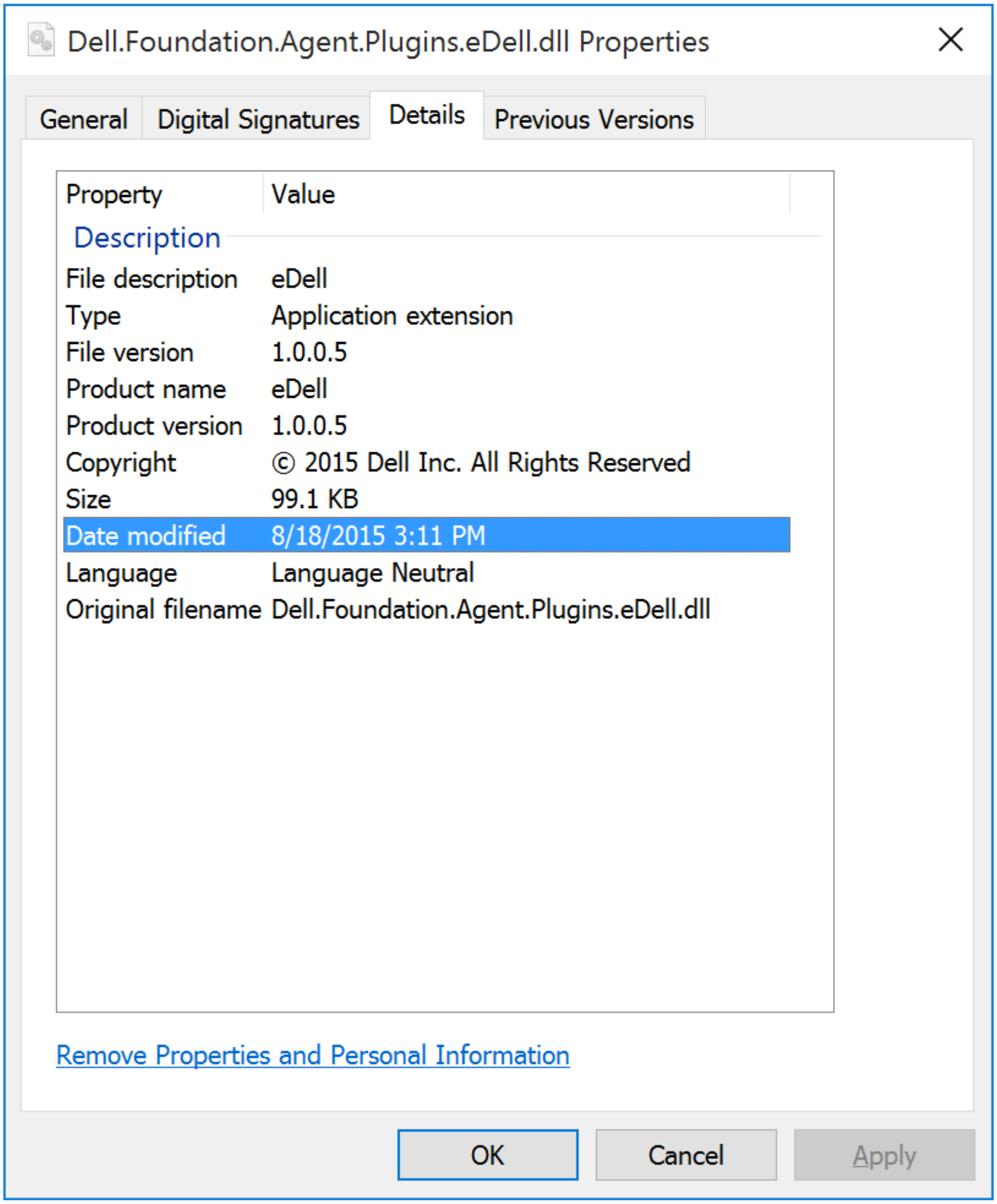
2f802b8dfcef2bf556e1026d7f89f7de830f14c8cc4da28ddc9dc2cb8b552b14

At least one of the systems we identified using the certificate for providing web services over HTTPS in the wild was a SCADA system (details withheld to protect the innocent). How this particular misconfiguration happened is unclear, but what is clear, is that this certificate is showing up in some extremely unusual and frankly concerning places.

The full extent of its distribution and usage are an interesting problem which can only be determined through further investigation.

How the eDell Plugin Works

As part of the .NET Dell Foundation Services that ship with every Dell lies a component called the DFSAgent. As of the 8/18/2015 release (Application_77P24_WN32_2.2.65.0_A00.EXE), it has contained a plugin for a telemetry component called eDell, (residing in Dell.Foundation.Agent.Plugins.eDell.dll).


Dell.Foundation.Agent.Plugins.eDell.dll Properties
General
Digital Signatures
Details
Previous Versions
Remove Properties and Personal Information
OK
Cancel
Apply

Property	Value
Description	
File description	eDell
Type	Application extension
File version	1.0.0.5
Product name	eDell
Product version	1.0.0.5
Copyright	© 2015 Dell Inc. All Rights Reserved
Size	99.1 KB
Date modified	8/18/2015 3:11 PM
Language	Language Neutral
Original filename	Dell.Foundation.Agent.Plugins.eDell.dll

This plugin installs two certificates from its resources, the eDellRoot and a localhost cert signed by said root:

```
private void SetupHttpsEnvironment()
{
        X509Store x509Store = new X509Store(StoreName.Root, 
StoreLocation.LocalMachine);
        x509Store.Open(OpenFlags.ReadWrite);
        if (x509Store.Certificates.Find(X509FindType.FindBySubjectName, 
"eDellRoot", false).Cast<X509Certificate2>().FirstOrDefault((X509Certificate2 c) 
=> c.Issuer == "CN=eDellRoot") == null)
        {
                var x509Certificate2Collection = new 
X509Certificate2Collection();
                
x509Certificate2Collection.Import(Dell.Foundation.Plugins.eDell.Properties.Resour
ces.eDellRoot, 
Dell.Foundation.eDell.Configuration.Properties.Resources.CertPassowrd, 
X509KeyStorageFlags.MachineKeySet | X509KeyStorageFlags.PersistKeySet);
                X509Certificate2Enumerator enumerator = 
x509Certificate2Collection.GetEnumerator();
                while (enumerator.MoveNext())
                {
                        X509Certificate2 current = enumerator.Current;
                        Console.WriteLine("Subject is: '{0}'", current.Subject);
                        Console.WriteLine("Issuer is:  '{0}'", current.Issuer);
                        x509Store.Add(current);
                }
        }
        x509Store.Close();
        x509Store = new X509Store(StoreName.My, StoreLocation.LocalMachine);
        x509Store.Open(OpenFlags.ReadWrite);
        X509Certificate2 serverCert = 
x509Store.Certificates.Find(X509FindType.FindBySubjectName, "localhost", 
false).Cast<X509Certificate2>().FirstOrDefault((X509Certificate2 c) => c.Issuer 
== "CN=eDellRoot");
        if (serverCert == null)
        {
                var x509Certificate2Collection2 = new 
X509Certificate2Collection();
                
x509Certificate2Collection2.Import(Dell.Foundation.Plugins.eDell.Properties.Resou
rces.localhost, 
Dell.Foundation.eDell.Configuration.Properties.Resources.CertPassowrd, 
X509KeyStorageFlags.MachineKeySet | X509KeyStorageFlags.PersistKeySet);
                var enumerator2 = x509Certificate2Collection2.GetEnumerator();
                while (enumerator2.MoveNext())
                {
                        X509Certificate2 current2 = enumerator2.Current;
                        Console.WriteLine("Subject is: '{0}'", 
current2.Subject);
                        Console.WriteLine("Issuer is:  '{0}'", current2.Issuer);
                        x509Store.Add(current2);
                        serverCert = current2;
                }
        }
        x509Store.Close();
```

It then stands up an HTTPS and WCF endpoints and configures it through netsh.exe:

```
        Func<bool> func = delegate
        {
                Process process = new Process();
                process.StartInfo.FileName = "netsh.exe";
                process.StartInfo.Arguments = string.Format("http add
sslcert ipport=0.0.0.0:{0} certhash={1} appid={{{2}}}",
eDellAgentPlugin.HttpsPort, serverCert.Thumbprint, Guid.NewGuid());
                process.StartInfo.UseShellExecute = false;
                process.StartInfo.RedirectStandardOutput = true;
                process.Start();
                string text = process.StandardOutput.ReadToEnd();
                this.logger.Debug("bindToPort output: {0}", new object[]
{ text });
                return text.Contains("success") && !
text.Contains("unsuccess");
        };
        Action action = delegate
        {
                Process process = new Process();
                process.StartInfo.FileName = "netsh.exe";
                process.StartInfo.Arguments = string.Format("http delete
sslcert ipport=0.0.0.0:{0}", 7800);
                process.StartInfo.UseShellExecute = false;
                process.StartInfo.RedirectStandardOutput = true;
                process.Start();
                string text = process.StandardOutput.ReadToEnd();
                this.logger.Debug("unbind port output: {0}", new object[]
{ text });
        };
        if (!func())
        {
                action();
                if (!func())
                {
                        this.logger.Info("bindToPort port fail.");
                }
        }
```

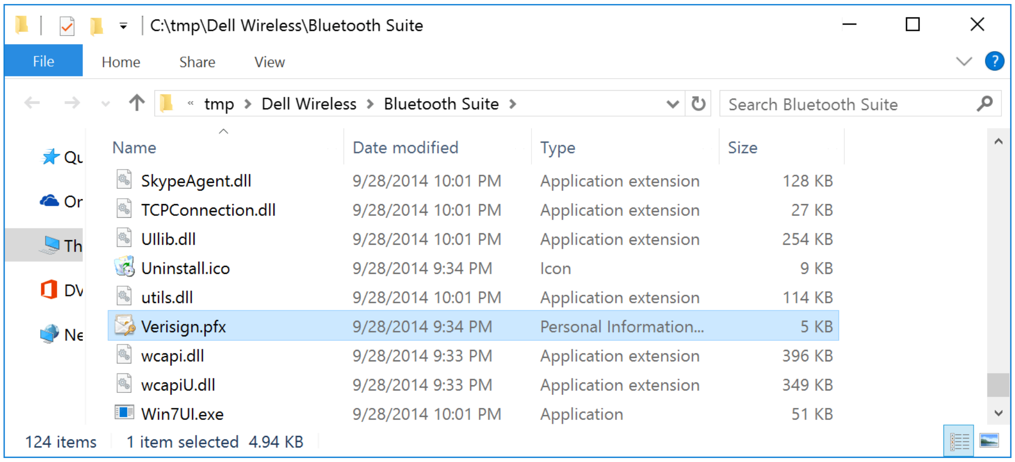
The purpose of these endpoints is to receive signed telemetry queries for the following interface:

```
namespace Dell.Foundation.Plugins.eDell
{
        [ServiceContract]
        public interface IeDellCapabilitiesApi
        {
                [OperationContract]
                string GetServiceTag();

                [OperationContract]
                bool GetDataCollectionStatus();

                [OperationContract, WebGet(ResponseFormat =
WebMessageFormat.Json)]
                bool DataCollectionStatus();

                [OperationContract, WebGet(ResponseFormat =
WebMessageFormat.Json)]
                string ServiceTag();
        }
}
```

The endpoints expect the the API requests are validated against an RSA signature for a key that is (thankfully) not shipped.

Bonus Round: Leaked Atheros Authenticode Certificates

This was not the only certificate mishap we identified with our Dell machine. The Bluetooth management tools which ship with the machine included a file called 'Verisign.pfx' the name alone was pretty ominous.

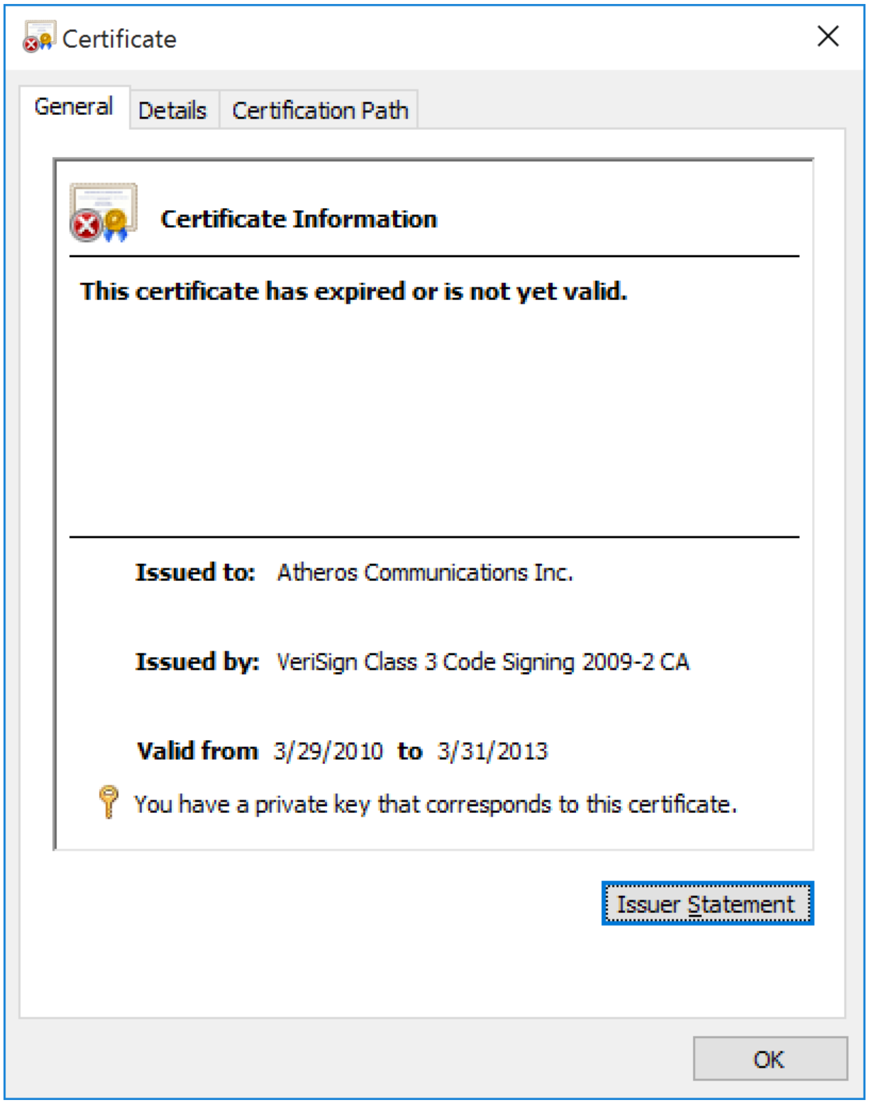


C:\tmp\Dell Wireless\Bluetooth Suite

Name	Date modified	Type	Size
SkypeAgent.dll	9/28/2014 10:01 PM	Application extension	128 KB
TCPConnection.dll	9/28/2014 10:01 PM	Application extension	27 KB
UIlib.dll	9/28/2014 10:01 PM	Application extension	254 KB
Uninstall.ico	9/28/2014 9:34 PM	Icon	9 KB
utils.dll	9/28/2014 10:01 PM	Application extension	114 KB
Verisign.pfx	9/28/2014 9:34 PM	Personal Information...	5 KB
wcapi.dll	9/28/2014 9:33 PM	Application extension	396 KB
wcapiU.dll	9/28/2014 9:33 PM	Application extension	349 KB
Win7UI.exe	9/28/2014 10:01 PM	Application	51 KB

124 items 1 item selected 4.94 KB

The .pfx archive required a password which took us all of 6 hours worth of sub-optimal cloud cracking to recover; the resulting password was: 't-span'.



Certificate Information

This certificate has expired or is not yet valid.

Issued to: Atheros Communications Inc.

Issued by: VeriSign Class 3 Code Signing 2009-2 CA

Valid from 3/29/2010 to 3/31/2013

You have a private key that corresponds to this certificate.

It turns out an Atheros signing certificate also shipped with the Bluetooth management software! It's the certificate used to sign four of the Bluetooth drives that shipped with the install:

- btath_hcrp.sys
- btath_lwflt.sys
- btath_pan.sys
- bthathfax.sys

Thankfully, this certificate expired on 3/31/2013 making it less prone to potential abuse. However, it appears that this certificate was in circulation while it was still valid (at least 11 days from what we can tell).

That means anything that was signed and timestamped prior to the certificate expiring could be valid. The earliest driver version that we could identify with this certificate was released on March 19, 2013 with the "Dell Wireless 1601,10.0.0.227, A00 Driver" version 11.22.54.596.

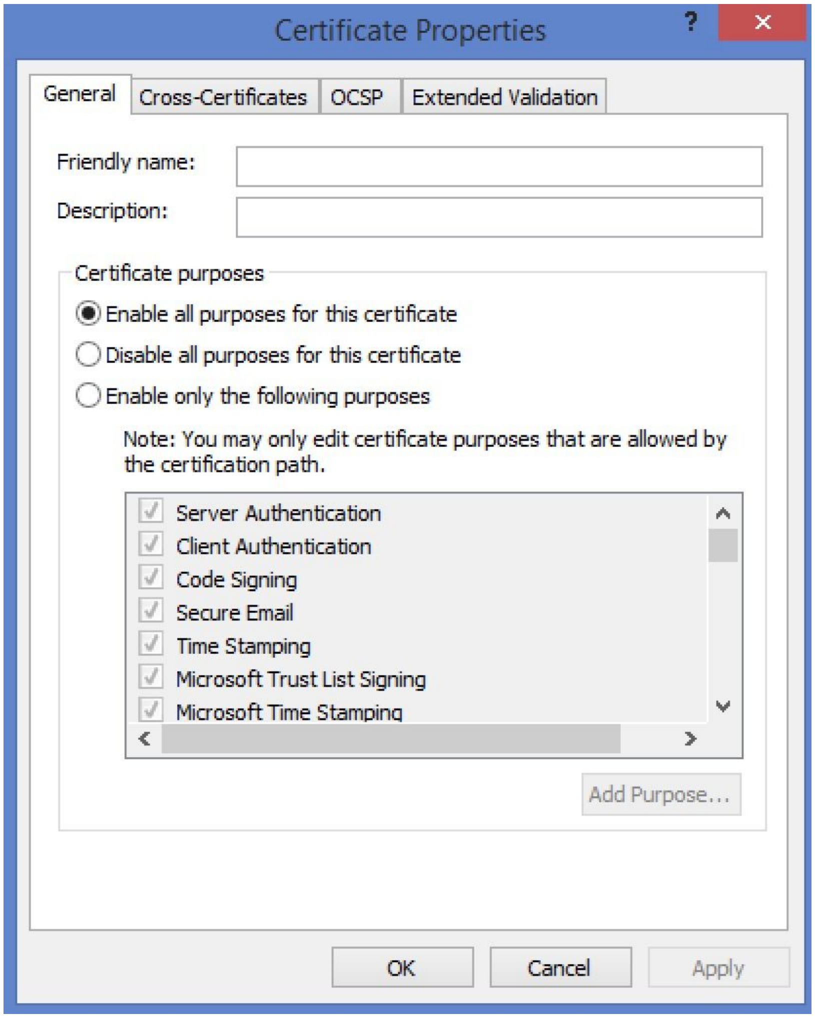
What is a “T-span,” Anyway?

According to this: “In 2011, Qualcomm bought Atheros Communications for $3.2 billion. Qualcomm's strategy in the sale was to take its strong position in broadband wireless networking and combine it with Atheros' considerable existing market share in making Wi-Fi chipsets. Atheros had spent years building that marketshare since its founding as a company named T-span in 1998.”

Security Implications

The eDellRoot certificate is configured for ‘all purposes,’ this includes sensitive things like authenticode (user-mode) and device driver (kernel-mode) code signing capabilities.

That means attackers with access to any of the eDellRoot private keys can do things like sign malicious code, target web browsers with man-in-the-middle attacks, and a whole slew of other evil things, as we described in the summary.



Friendly name:

Description:

Certificate purposes

● Enable all purposes for this certificate

○ Disable all purposes for this certificate

○ Enable only the following purposes

Note: You may only edit certificate purposes that are allowed by the certification path.

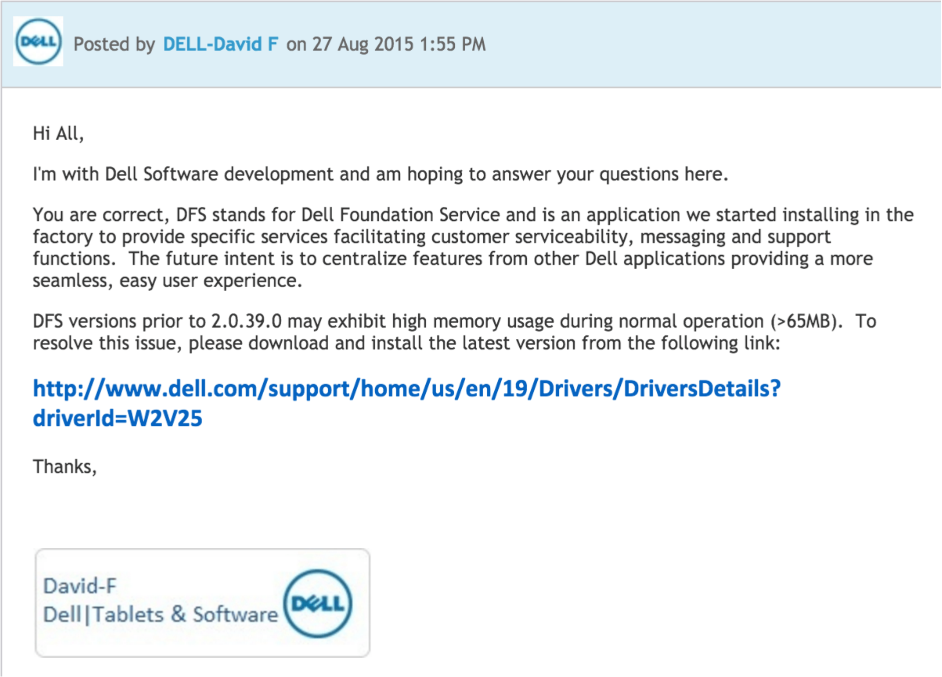
- ☑ Server Authentication
- ☑ Client Authentication
- ☑ Code Signing
- ☑ Secure Email
- ☑ Time Stamping
- ☑ Microsoft Trust List Signing
- ☑ Microsoft Time Stamping

Now, astute readers might point out that some software - perhaps most notably, Google Chrome - includes a defense against rogue certificates, using a technique known as "pinning."

In the past, this has been effective at - for example - detecting fraudulent certificates apparently used to spy on people in Iran. However, Chrome explicitly allows "private" root CAs - which would include the eDellRoot certificate - to bypass its pinning checks.

Remediation for eDellRoot

We are unsure what Dell Foundation Services (DFS) actually does on a Dell systems, however, Dell has a vague description here and a Dell support forum also contains this overly vague and unhelpful response from Dell.



Posted by DELL-David F on 27 Aug 2015 1:55 PM

Hi All,

I'm with Dell Software development and am hoping to answer your questions here.

You are correct, DFS stands for Dell Foundation Service and is an application we started installing in the factory to provide specific services facilitating customer serviceability, messaging and support functions. The future intent is to centralize features from other Dell applications providing a more seamless, easy user experience.

DFS versions prior to 2.0.39.0 may exhibit high memory usage during normal operation (>65MB). To resolve this issue, please download and install the latest version from the following link:

http://www.dell.com/support/home/us/en/19/Drivers/DriversDetails?driverId=W2V25

Thanks,

David-F
Dell|Tablets & Software

Many people have indicated that removing the eDellRoot certificates from the root and personal certificate stores is sufficient to protect users. This is not entirely accurate; **you must remove the eDell plugin entirely or the certificate will be reinstalled whenever it is loaded.** This can be accomplished by deleting the 'Dell.Foundation.Agent.Plugins.eDell.dll' module from the system. Failure to do so may result in continued exposure to this security flaw.

An alternate remediation could also be as follows, however, we recommend that this be only performed by qualified professionals:

Note that if you ever perform a factory reset on your Dell system, this certificate and the eDell plugin will be restored to the system and you will have to manually remove it again.

Conclusion

This highlights a disturbing trend among original equipment manufacturer (OEM) hardware vendors. Tampering with certificate stores exposes users to unnecessary, increased risk.

Tampering with the certificate store is a questionable practice, and OEM's need to be careful when adding new trusted certificates, especially root certificates. Sadly, OEM manufacturers seem to not be learning from historical mistakes and keep making them over and over.

We look forward to engaging with the security community as a whole to get to the bottom of this to help protect affected Dell customers, and we look forward to more care and consideration on the part of OEMs when deciding to customize certificate stores.

Stay tuned for more security research from Duo Labs in the near future.