

RSASSA-PSS – Why Your Certificate Can't Be Validated

PKI pkisolutions.com/pkcs1v2-1rsassa-pss

Mark B. Cooper

January 24, 2024

Blog February 1, 2017 Certificate Templates, Certificate Validation, Hash Algorithms, Known Issues, PKI

A common theme has been arriving in my email box lately as well as many online forums. Consistently people are reporting error with certificates issued by their internal Microsoft ADCS based CAs. Problems range from Apple devices, Firefox, appliances and many other systems. When people examine their certificates they see that their certificates are SHA based, including many or all of their CAs in the hierarchy. So what is causing the problem?



Expand Your PKI Visibility

Discover why seeing is securing with revolutionary PKI monitoring and alerting.

[Learn More About PKI Spotlight®](#)

[PKCS #1 Version 2.1](#) was published as part of [RFC 3447](#) in 2002 so it has been around for quite some time. Microsoft first implemented support for this in Server 2008 as part of the Crypto Next Generation revision for Windows. This standard defines an alternate and slightly improved method for CAs to sign certificates to eliminate some limited weaknesses. Despite its now 15 year availability, the VAST majority of systems and

applications have failed/neglected/ignored the revision. As a result, certificates signed using this process are often unusable. However, the windows API (CAPI/CAPIC2) and applications that rely on those API have support for the standard. Outside of that, you are likely to run into a wide range of support issues.

So how are these certificates getting signed with PKCS #1 V2.1 signatures? Microsoft defined the attribute AlternateSignatureAlgorithm=1 in the CAPolicy.inf file as the method to indicate to a CA that you want it to sign certificates created by that CA to be signed with PKCS #1 v2.1 – at least on standalone CAs such as Root and Policy CAs. It is important to note that references to DiscreteSignatureAlgorithm as the value name are incorrect. Sources such as the MS Press book and any other online sources are incorrect. During the beta release of Windows Server 2008, the value was called DiscreteSignatureAlgorithm. But with the full release and subsequent OS releases, using this value in your CAPolicy will result in no action as it is an unrecognized value.

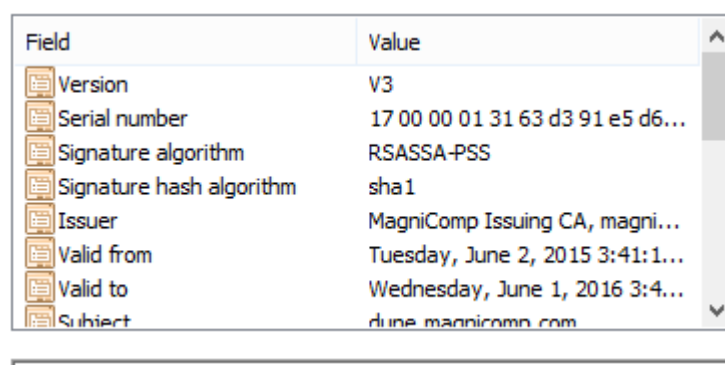
Online Enterprise CAs have the ability to define this for end-entity certificates on a per-template basis. When a template is defined as a V3 or V4 schema version, the Cryptography tab will enable an option for the template administrator to enable **use alternate signature format**. An example template:

The screenshot shows the 'Properties of New Template' dialog box with the 'Cryptography' tab selected. The dialog has a title bar with a close button (X). Below the title bar is a tabbed interface with tabs for 'Superseded Templates', 'Extensions', 'Security', 'Compatibility', 'General', 'Request Handling', 'Cryptography', and 'Key Attestation'. The 'Cryptography' tab is active, showing the following settings:

- Provider Category:** Key Storage Provider (dropdown)
- Algorithm name:** RSA (dropdown)
- Minimum key size:** 2048 (text box)
- Choose which cryptographic providers can be used for requests:**
 - ☒ Requests can use any provider available on the subject's computer
 - ☐ Requests must use one of the following providers:
- Providers:** A list box containing three providers, each with an unchecked checkbox:
 - Microsoft Software Key Storage Provider
 - Microsoft Platform Crypto Provider
 - Microsoft Smart Card Key Storage ProviderThere are up and down arrow buttons to the right of the list box.
- Request hash:** SHA1 (dropdown)
- ☒ Use alternate signature format

At the bottom of the dialog are four buttons: 'OK', 'Cancel', 'Apply', and 'Help'.

When a certificate has been signed with PKCS #1 v2.1 signatures, you will see the Signature Algorithm listed as RSASSA-PSS. You can refer to the Schemes section of the [Wiki doc for PKCS #1](#) for more details. An example certificate:



Field	Value
Version	V3
Serial number	17 00 00 01 31 63 d3 91 e5 d6...
Signature algorithm	RSASSA-PSS
Signature hash algorithm	sha1
Issuer	MagniComp Issuing CA, magni...
Valid from	Tuesday, June 2, 2015 3:41:1...
Valid to	Wednesday, June 1, 2016 3:4...
Subject	dune.magnicomp.com

In general, I recommend avoiding the use of PKCS #1 v2.1 signatures due to the diversity in most environments when it comes to support. To that end, the small increased security gain is greatly offset by support issues. If you have encountered this issue, you will need to determine how to remediate the signature compatibility. More often than not, people deploy this setting not realizing the support issues. If an application rejects an end-entity certificate due to the RSASSA-PSS encoding, then the certificate will need to be reissued. This can be turned off on the template that is being used to issue the certificate. There is also a VERY strong likelihood that if an application fails validation on end-entity certificates, it will fail the PKI chain if also signed with PKCS #1 v2.1. In this case, your CAs will need to modify their signature configuration in the registry and subordinate CAs reissued as needed. If the Root CA was signed this way, it too will need to be renewed. Start at the top of the chain and renew your chain downwards.