윈도우 드라이버 코드 사이닝 실용 가이드

역자의 글: 이 문서는 David Grayson이 작성한 http://www.davidegrayson.com/signing/ 문서를 원 저자의 허락을 맡고 한국어로 그대로 번역한 것입니다. 한국어로의 번역 작업은 저 혼자 진행하였습니다. 모든 내용은 원문을 그대로 옮긴 것이며, 번역 시 역자에 의한 추가나 수정은 없었습니다. 원문은 http://www.davidegrayson.com/signing/ 에서 읽으실 수 있습니다. 원문은 계속하여 업데이트 되고 있으며, 번역은 2015-07-03 일 기준의 문서를 기반으로 진행되었습니다. 모든 저작권은 원 저자에게 있습니다.

Note from whom translated this: This document is a translated version of http://www.davidegrayson.com/signing/ which David Grayson originally wrote and shared through this website. Translation into Korean was done by HeeJune Kim voluntarily by myself under permission of original author, David Grayson. This translated result does not contain any additional comment or modification by the translator, and you can still read the original article from http://www.davidegrayson.com/signing/. All rights are reserved to the original author, David Grayson.
2015-07-14.

윈도우 드라이버 코드 사이닝 실용 가이드

윈도우 8, 7, 비스타, XP를 위한 드라이버 코드 사이닝 가이드

SHA-1과 SHA-2에 대하여: 이 문서는 원래 2013년 1월에 최초로 나왔고, SHA-2 해쉬 알고리즘을 사용하는 인증서를 쓰면서 제가 겪었던 문제들을 설명한 문서입니다. 본문을 통해 저는 SHA-1을 계속 쓸것을 권해오고 있습니다.

처음 글을 낸 이후에, 마이크로소프트는 2016년 1월에 SHA-1을 deprecate할 것이라고 발표(Deprecation of SHA-1)했습니다. 그래서, SHA-1은 더 이상 장기적인 방법이 될 수는 없으며, 많은 분들은 SHA-2로 바꿀 것을(이 문서에서는 반대되는 이야기들을 하고는 있지만)고려하기 시작해야 합니다.

이 문서는 제가 2012년 후반부터 SHA-2 인증서들을 사용하려고 했을 때 겪었던 여러 문제들에 대해서 경고하는 내용을 담고 있습니다. 그런데 이들중 많은 부분이 해결된 상태입니다. KB-2763674는 윈도우 비스타에서 SHA-2를 사용해서 실행 파일을 사인했을 때 발생하던 문제점을 해결했다고 합니다. 커널 드라이버를 SHA-2 인증서를 사용해서 사인했을 때 윈도우 7에서 로딩할 수 없었던 문제는 PiXCL Automation Technologies 홈페이지에 있는 윈도우 8 드라이버 사이닝하기(Signing Windows 8 DriversSigning Windows 8 Drivers) 튜토리얼 문서에서 설명하고 있는 것처럼 드라이버를 두번 사이닝을 해서 해결할 수 있는 문제인데 KB-3033929는 이것을 해결했다고 합니다. 그러므로 제가 여기에 SHA-2와 함께 겪었던 문제들을 정리한 내용들이 반드시 해결 불가능한 문제들이 더 이상 아닐 수도 있습니다. 하지만 여러분의 프로젝트를 진행할 때 여전히 조심해야 할 문제들이기는 합니다.

윈도우 비스타 64비트 버전에서는 커널 드라이버를 SHA-2 인증서로 사인하고 로드하는게 거의 불가능 할 것입니다.

SHA-2 인증서로 변경할 때 제가 생각하는 가장 큰 문제점은 대부분의 윈도우 컴퓨터들의 “신뢰할 수 있는 루트 인증서 기관(Trusted Root Certifiation Authorities)” 리스트에 루트 인증서가 이미 포함된 인증서 제공회사를 찾는 일이였습니다. 인증서가 없으면, 인증서를 가져오기 위해서 윈도우 업데이트에 의존해야만 하는데, 제 개인 경험상 믿을 만 하지 못했습니다.

2015년 중반쯤에 SHA-1에서 SHA-2로 옮겨가려는 계획을 갖고 있습니다. 그때에는 단순히 쓰지 말라고 경고만 하는 게 아니라, SHA-2 인증서를 실제로 어떻게 사용하는지를 설명하는 내용과 함께 이 문서를 업데이트 하도록 하겠습니다.

소개

윈도우에서 한번이라도 소프트웨어나 드라이버를 설치해 본 경험이 있으시다면, 그 소프트웨어를 만든 사람이나 회사의 이름을 알려주는 다이얼로그를 본적이 있으실 것입니다. 이것은 개발사가 개발한 결과물을 암호학적으로 **사인** 했다는 것을 뜻합니다. 여러분의 소프트웨어를 사인하는 것은 중요합니다: 최종 사용자들에게 적절한 다이얼로그를 보여줌으로써 그 사용자들이 이상한 바이러스를 설치하는 것이 아니라는 신뢰감을 갖게 하죠. 디바이스 드라이버의 경우에 코드 사인은 사용하는 윈도우 버전에 따라 더 필수적입니다.

이 가이드의 목적은 여러분이 드라이버나 소프트웨어를 어떻게 사인하는지 제대로 알려주는 데 있습니다.

제 이름은 David Grayson이고, Pololu Robotics & Electronics에서 일하고 있습니다. 2012년에 저는 제가 있는 회사의 모든 USB 드라이버들과 윈도우 버전 설치 파일을 사인하는 작업을 진행했었습니다. 저는 그 과정에서 누군가 미리 저에게 말해줬더라면 고생하지 않았어도 될 수많은 문제들을 만났습니다. 만약 여러분이 같은 일을 하려고 한다면, 진심으로 저는 이 문서가 모든 애매함을 없애고 여러분의 **많은 시간을 아낄 수 있기를** 진짜로 희망합니다. 저는 고생길을 거쳤지만, 여러분은 쉽게 배우세요.

이 정보들의 상당수는 MSDN에서 찾으실 수 있는 공식 마이크로소프트 문서를 통해 검증하실 수 있으며, 필요할 때마다 공식 문서를 인용하도록 하겠습니다. 커널 모드 코드 사이닝에 대한 가장 권위있는 문서는 kmsigning.doc와 KMCS_walkthrough.doc입니다. 이 문서들은 매우 훌륭한 자료들입니다만, 2007년에 쓰여졌기 때문에 윈도우 7이나 윈도우8, SHA-2에 대한 내용을 담고 있지는 않습니다. 또한, 그 문서들은 실행 파일의 사인에 대해서는 다루고 있지 않기 때문에 이 문서보다 훨씬 그 범위가 제한적입니다. 그래서 여기에서 제가 이야기하는 상당수의 내용들은 제가 실제로 직접 실험을 통해 이끌어낸 결론들입니다. 실험을 통해 알아낸 것들에 대해서 제가 말할 때에는 “인것 같다” 또는 “제 경험상”이란 말들을 사용할 것입니다. 만약 실험 결과가 공식 문서의 내용과 상반될 때에는 그렇다고 알려드리도록 하겠습니다.

만약에 제가 여기서 제공해 드리는 정보가 틀렸다고 생각하다면, 이곳에 답글(post a comment)을 달아주시고 알려주시면 한번 같이 확인해 보도록 하시죠.

이 문서는 윈도우 XP 32비트, 윈도우 비스타 32비트와 64비트, 윈도우 7 32비트와 64비트, 그리고 윈도우 8 32비트와 64비트만 다룹니다.

이 문서에서 가장 유용한 부분은 signature requirements 섹션에 있습니다.

RSA 암호 시스템

실행파일이나 드라이버 파일을 사인하는 작업들은 1970년대에 Rivest, Shamir, and Adleman 이 발명해 낸 RSA cryptosystem 시스템을 사용하게 됩니다. 저는 그에 대한 수학적인 배경지식까지는 자세히 설명하진 않겠지만, RSA로 무엇을 할 수 있는지만 간략하게 설명하도록 하겠습니다. 이 내용은 디지털 시그니쳐가 도대체 무엇인지, 그리고 어떻게 동작하는지를 이해하는데 도움이 될 것입니다.

RSA가 제공하는 것은 공개 키와 개인 키라는 것으로 구성된 키 쌍(pair)을 생성하는 방법입니다. 그 이름이 뜻하는 것처럼, 개인키는 비밀스럽게 간직해야하고, 공개 키는 누구에게나 줄 수 있습니다.

그 다음으로 RSA는 함수를 제공합니다. 공개 키는 우리가 이하 *f*라고 부를 함수를 제공합니다. 개인 키는 우리가 *g*라고 부를 함수를 제공합니다. 이들 함수들의 정확한 입력값이나 출력값이 무엇인지에 대해서는 걱정하지 마세요. 이들 함수들의 중요한 특성은 다음과 같습니다:

- f와 g는 서로를 뒤집는 성질을 갖고 있습니다: f(g(x)) = x and g(f(x)) = x
- 공개 키(또는 f)로부터 g를 유추해 내기란 진짜 엄청나게 어렵습니다.

메시지를 암호화(**Encrypting)**하고 복호화(**decrypting)** 하는 것은 각각 함수 f와 g를 호출해서 합니다. 누구나 메시지를 받는 사람의 공개키로부터 얻은 f 함수를 통과하도록 해서 암호화를 할 수 있습니다. 그러면 그 받는 사람은 암호화 된 메시지를 읽을 수 있는 유일한 사람이며, g를 적용하여 복호화를 합니다.

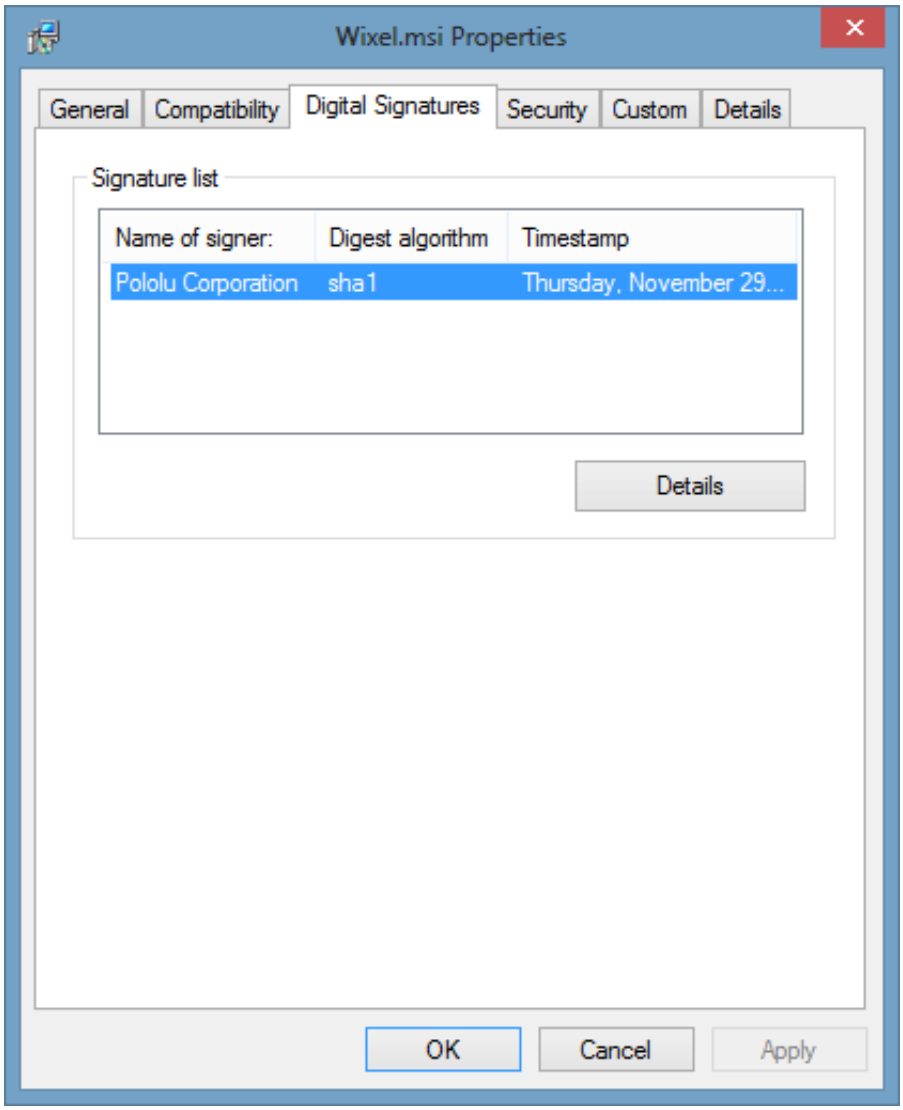
메시지를 사인(**Signing)**하고 검증(**verifying)**하는 것은 함수 g와 f를 각각 적용해서 합니다. 보내는 사람은 메시지에 대한 시그니쳐를 만들기 위해 그의 메시지를(또는 메시지의 해쉬를) 그의 개인키로부터 얻은 g 함수에 넘겨줍니다. 보내는 사람만이 g에 접근할 수 있기 때문에 유일하게 이 작업을 할 수 있습니다. 이 메시지를 받는 모든 이는 공개키로부터 얻은 f 함수를 시그니쳐에 적용하여 모든 것이 일치하는지 검증할 수 있습니다. 이것이 윈도우가 소프트웨어의 시그니쳐를 검증하고 누가 개발사인지를 알려주는 동작들의 숨겨진 원리입니다.

좀더 자세히 설명하자면, RSA 암호화 시스템은 매우 큰 수들을 소수로 인수 분해하기가 엄청나게 어렵기 때문에 동작합니다. 개인 키는 두개의 매우 큰 소수로 이루어져 있으며, 공개키는 그것들의 곱으로 대부분 구성되어 있습니다. 위키피디아서 물론 더 자세한 내용(how RSA works)이 있습니다.

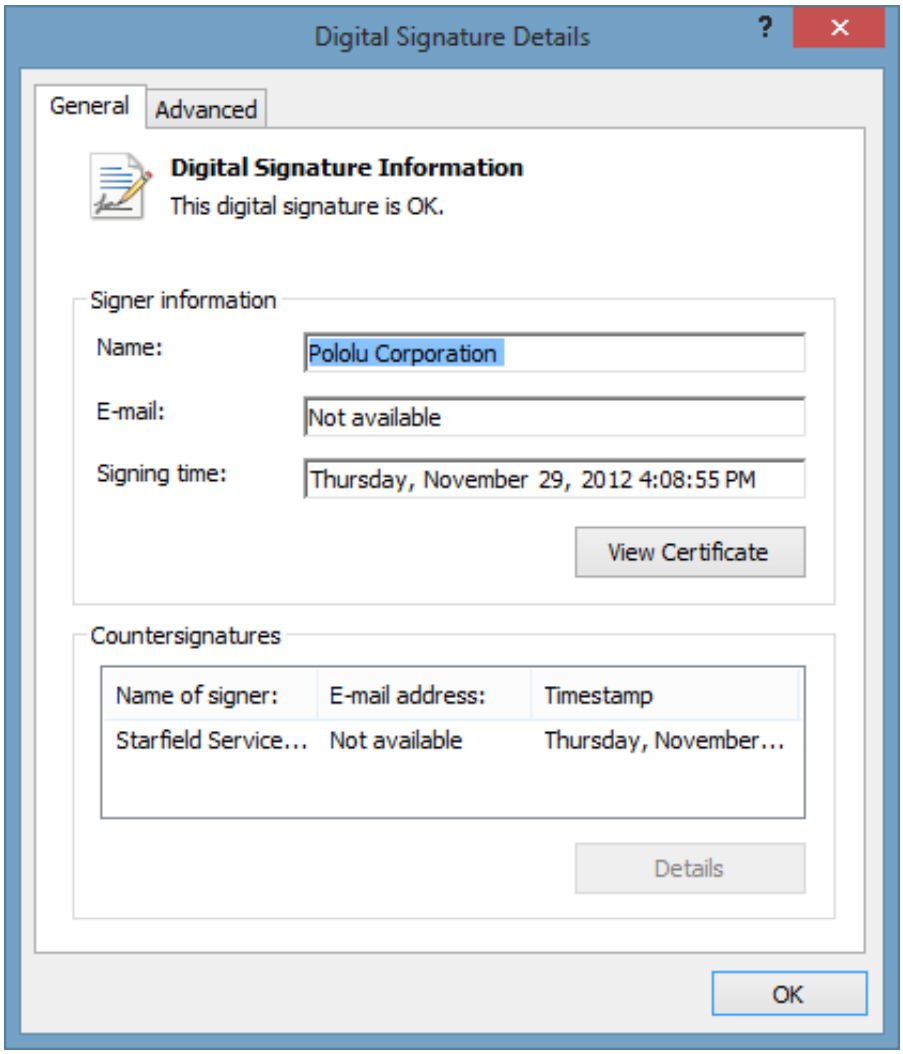
시그니쳐 파해치기

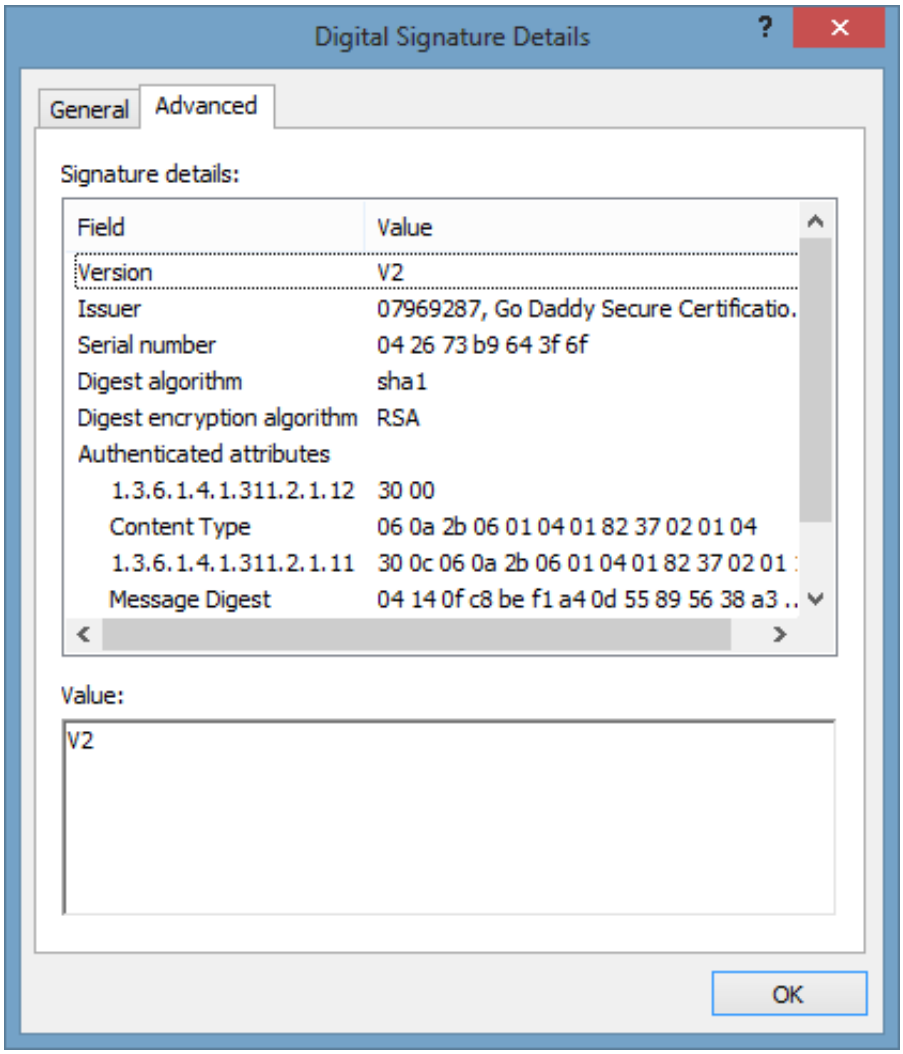
윈도우에는 파일에 임베딩 된 시그니쳐의 세부 내용을 보는 다이얼로그들이 있습니다 . 이들 다이얼로그 상자를 보는 법을 잘 아셔야 합니다.

사인된 파일에 마우스 우클릭을 한 다음에, 등록정보로 가 보세요, 그러면 디지털 시그니쳐 탭이 있을 겁니다.

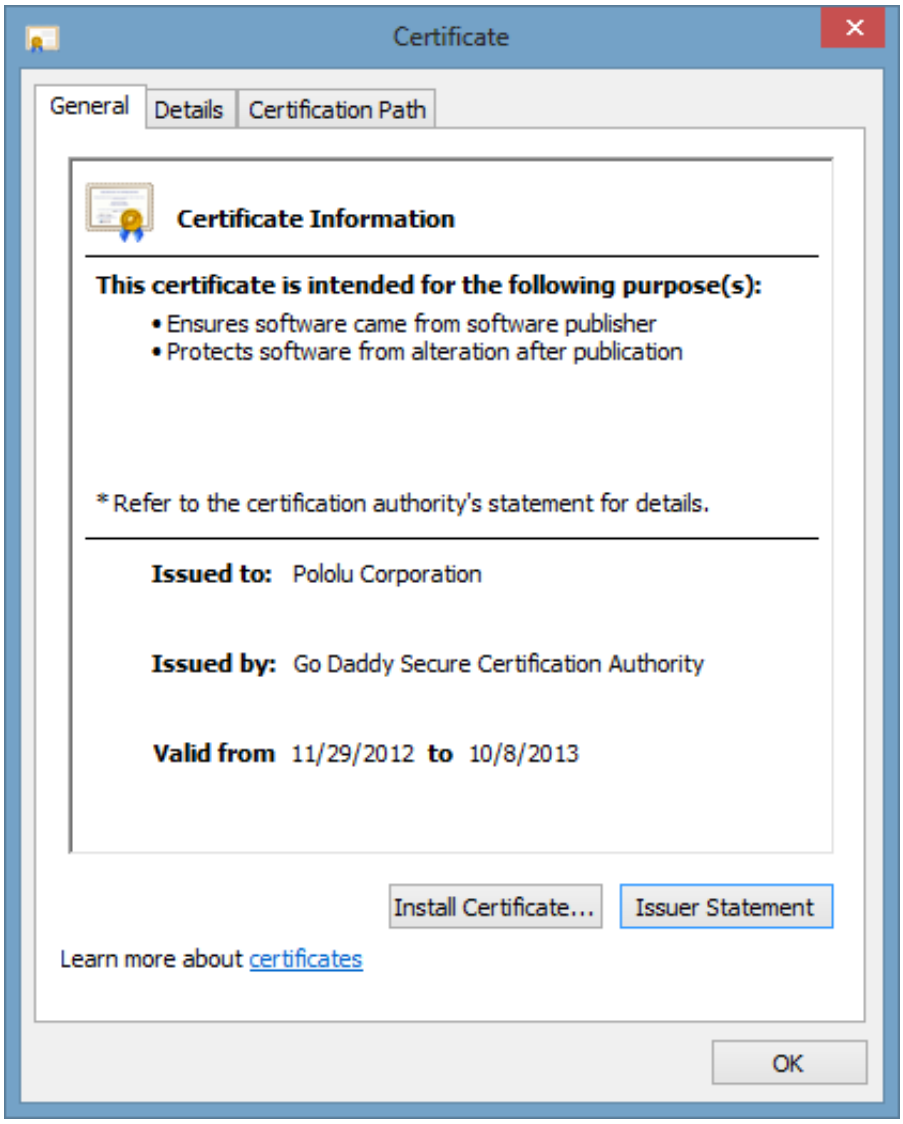


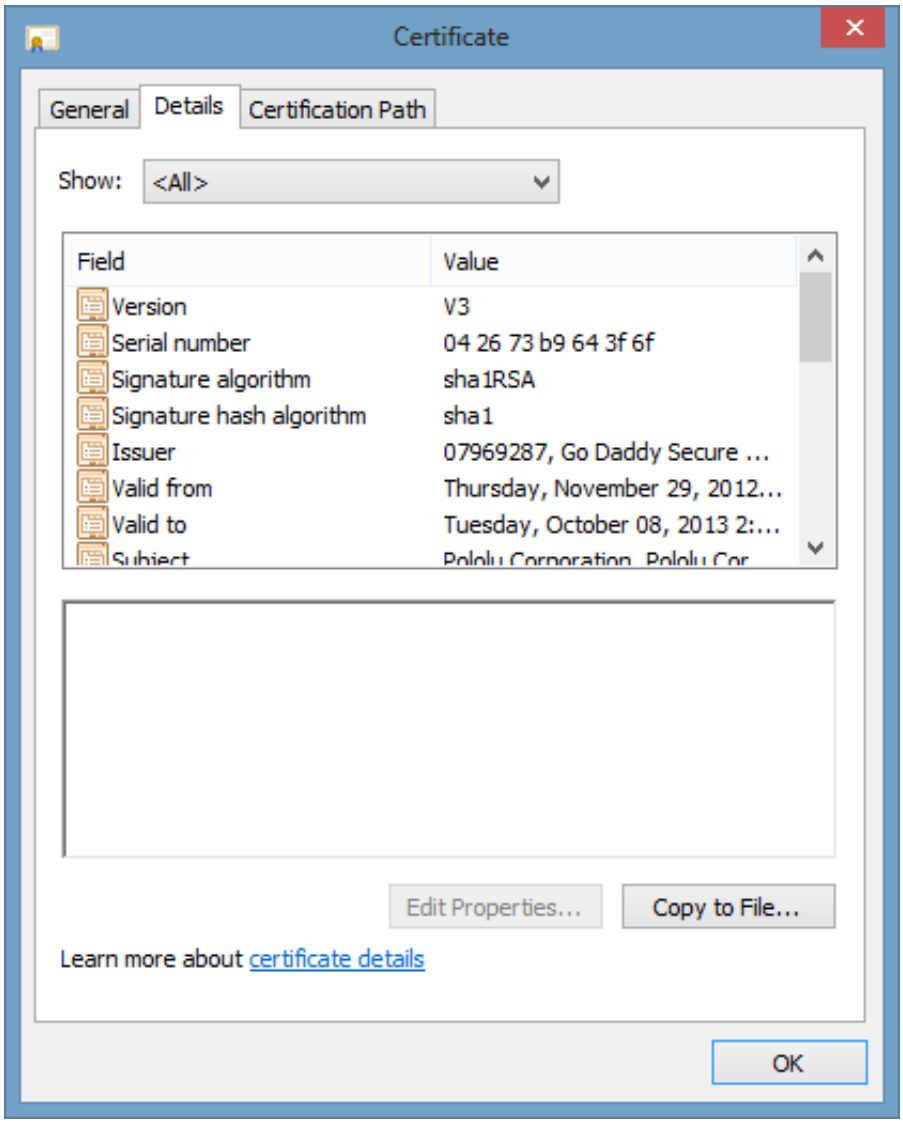
디지털 시그니쳐(Digital Signatures) 탭에서, 세부사항 버튼을 클릭하세요.


Digital Signature Details
General
Advanced
Digital Signature Information
This digital signature is OK.
Signer information
Name:
Pololu Corporation
E-mail:
Not available
Signing time:
Thursday, November 29, 2012 4:08:55 PM
View Certificate
Countersignatures
Name of signer:
E-mail address:
Timestamp
Starfield Service...
Not available
Thursday, November...
Details
OK

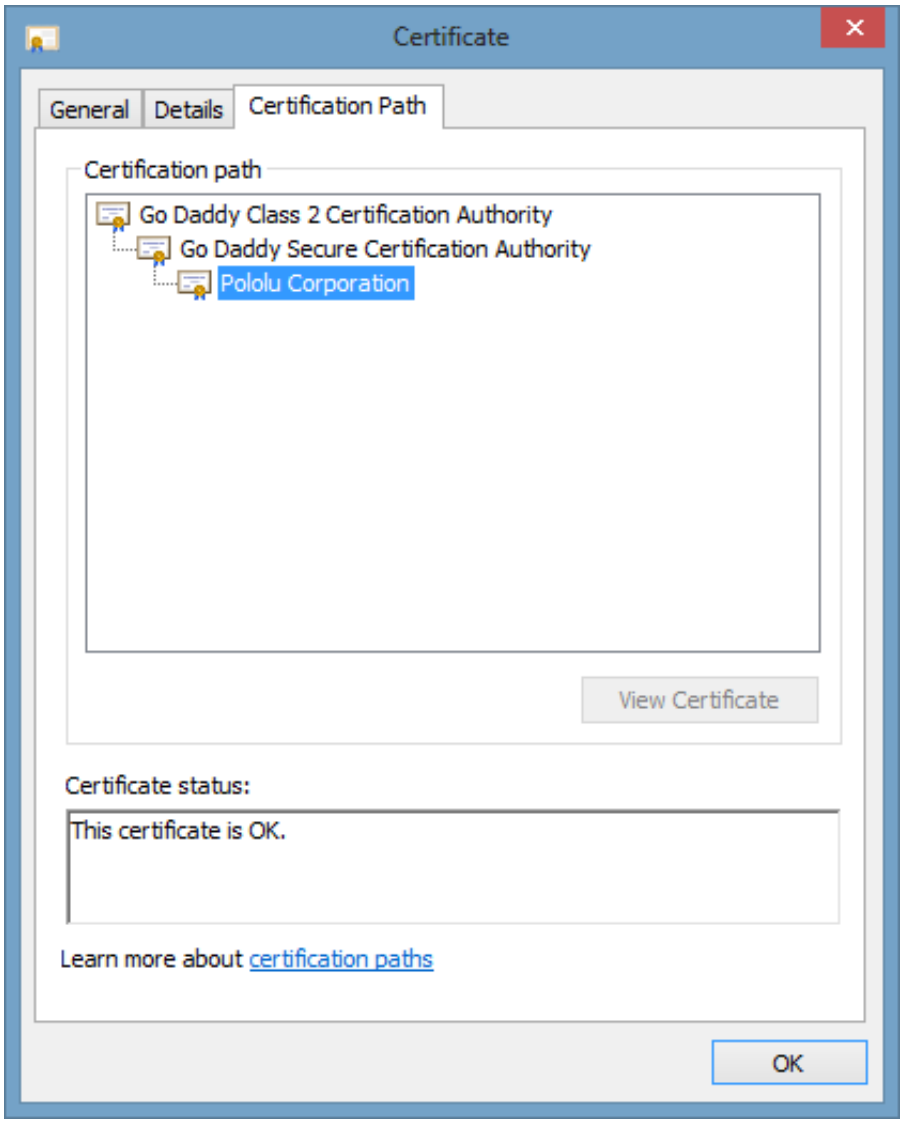


파일의 시그니쳐에 임베딩 된 인증서를 보려면 인증서 보기 버튼을 클릭해 보세요. 인증서는 Verisign과 같은 인증 기관에서 구입합니다.


Certificate
General
Details
Certification Path
Certificate Information
This certificate is intended for the following purpose(s):
• Ensures software came from software publisher
• Protects software from alteration after publication
* Refer to the certification authority's statement for details.
Issued to: Pololu Corporation
Issued by: Go Daddy Secure Certification Authority
Valid from 11/29/2012 to 10/8/2013
Install Certificate...
Issuer Statement
Learn more about certificates
OK



신뢰 체인에 있는 인증서를 보려면 인증서 경로(Certification Path)를 클릭합니다. 여러분의 인증서가 믿을만한 회사에 의해 발급되었다는 것을 증명하기 위해서 인증서 경로가 필요합니다. 어떤 경우엔 인증서 경로가 크로스 인증서(cross-certificate)로부터 왔다고 보여주는데, 이 크로스 인증서란 신뢰 체인을 더 확장하여 연결되도록 사인할 때 시그니쳐에 포함하는 특별한 인증서입니다. 추가적인 정보를 보려면 인증서 경로에 있는 모든 인증서를 더블클릭하여 살펴봅니다.



나중에 이 섹션에 설명을 더 추가해서 확장하고 싶지만, 일단 지금은 여러분들이 직접 시그니쳐를 클릭해서 살펴보세요, 한번 wixel-signing-experiment.zip 파일들도 직접 다운로드 하셔서 파일에 있는 시그니쳐들도 살펴보시기 바랍니다.

그리고 임베딩 된 시그니쳐를 자세히 보려면 WinHex 같은 헥스 에디터를 사용하셔도 됩니다. 그렇게 하면 인증서 경로에 있는 조직과 사인한 사람의 이름을 쉽게 보실 수 있습니다. WinHex의 기능 중에서 두개의 파일들을 비교해서 그 파일 간의 다른 점들만 하일라이트해서 보여주는 기능이 있는데, 시그니쳐가 추가되었을 때 끝에 어떤 바이트들이 추가되고, 헤더에 어떤 바이트가 수정되었는지를 정확히 바이트 단위로 볼 수 있게 해 줍니다.

시그니쳐 요구사항들

여러분들은 개발하시면서 반드시 사용하시려는 디지털 시그니쳐가 아래 정리한 모든 필요한 요구사항들을 모두 만족하는지를 꼭 확인하셔야 합니다. 이들 요구사항들은 아래 테이블에 정리되어 있

으며, **테이블에 사용된 용어들은 테이블 뒤에 설명되어 있습니다.** 이 리스트에 빠진 부분이 있다면 반드시 커멘트를(post a comment) 주세요.

소프트웨어가 잘 실행되기만 하면 되고 경고 메시지를 사용자들이 보는 것은 별로 신경쓰이지 않으신다면, 제 경험상 시그니쳐가 아래 요구사항들을 만족시키면 됩니다.

돌아가기만 하면 될 경우 시그니쳐 요구사항들

Signature requirements for it to just work

	Running an executable	Installing a driver package	Loading a kernel module
on Windows XP	none	none	none
on Windows Vista 32-bit	No SHA-2	none	none
on Windows Vista 64-bit	No SHA-2	none	MCVR & SHA-1
on Windows 7	none	none	MCVR & SHA-1
on Windows 8	none	TRCA	MCVR

소프트웨어가 잘 동작해야 할 뿐만 아니라, 동시에 경고 박스를 비롯해서 모든 등록정보 다이얼로그에 검증된 개발사라고 나오게 하고 싶다면 아래 시그니쳐 요구사항들을 지키셔야 합니다:

보기에도 좋게 하기 위한 시그니쳐 요구사항들

Signature requirements for it to look good

	Running an executable	Installing a driver package	Loading a kernel module
on Windows XP	TRCA	WHQL	?
on Windows Vista 32-bit	TRCA & No SHA-2	TRCA	?
on Windows Vista 64-bit	TRCA & No SHA-2	TRCA	?
on Windows 7	TRCA	TRCA	?
on Windows 8	TRCA	TRCA	?

TRCA

시그니쳐의 신뢰 체인(chain of trust)은 반드시 사용자의 신뢰할 수 있는 루트 인증서 기관(Trusted Root Certifiation Authorities, 이하 TRCA)에 있는 인증서로 다시 돌아가야 합니다. certmgr.msc에서 볼 수 있는 것처럼, TRCA 리스트는 Verisign, Globalsign, Digicert, and Go Daddy와 같이 잘 알려진 몇몇 회사들의 인증서들을 가지고 있습니다. 그리고 또 다른 몇개의 인증서들은 최초에는 그 리스트에는 없지만, 시그니쳐 검증을 하는 동안 필요하다면 윈도우 업데이트로부터 자동으로 다운로드 됩니다. 이것이 Microsoft Root Cerificate Program입니다. 그러나, 저라면 자동 업데이트에 의존하지 않겠습니다. 이론적으로 보자면 컴퓨터가 인터넷에 연결되어 있다면 당연히 동작해야 겠지만 제 경험상 인터넷에 연결되어 있어도 항상 제대로 동작하는 것은 아니었기 때문입니다(does not always work reliably). 그러므로 최선의 방법은 여러분의 신뢰 체인이 윈도우를 갓 인스톨 했을 때 리스트에 포함되어 있는 인증서로 갈 수 있도록 확실히 해 두는 것입니다. 아쉽지만 이에 해당하는 인증기관 리스트를 갖고 있지는 않습니다. TRCA 요구사항은 kmsigning.doc에 문서화 되어 있습니다. 크로스 인증서는 보통 이 경우를 위해서 필요

한 것은 아닙니다. 그리고 사용자들에게 인증서를 “신뢰하는 퍼블리셔” 또는 “신뢰하는 사람” 에 설치하도록 부탁하는 경우에도 잘 동작할 거라고 생각합니다.

MCVR
: 시그니쳐의 신뢰 체인은 마이크로소프트 코드 검증 루트 인증서(Microsoft Code Verification Root certificate) 또는 커널이 신뢰하는 다른 인증서로 연결되어야 합니다. 모든 WHQL 시그니쳐들은 이 요구사항을 지켜야 합니다. kmsigning.doc 문서에서 윈도우 커널은 신뢰할 수 있는 루트 인증기관 리스트(Trusted Root Certification Authorities list) 리스트에 접근 권한이 없다고 명확하게 설명하고 있습니다. 이 요구사항을 따르기 위해 보통 크로스 인증서를 사용합니다. 마이크로소프트는 Cross-Certificates for Kernel Mode Code Signing에 전체 리스트를 공개해 두고 있습니다.

WHQL
: 이 시그니쳐는 WHQL program에서 얻습니다. 제가 알기로는 드라이버를 테스트 하기 위해서 마이크로소프트나 서드파티에게 많은 돈을 내야 하는 것으로 알고 있습니다. 만약 드라이버가 정상 동작한다면, 여러분의 드라이버를 사인해 줄 것이고 판매 제품에 윈도우 로고를 사용할 수 있는 법적인 허가도 줍니다. 그러나 소프트웨어나 드라이버가 동작하는데 있어서 WHQL은 **꼭 필요한 것은 아니며** 표준 코드 사이닝 인증서를 사용하는 것에 비해 더 비싸기만 합니다. 저는 WHQL 사인된 드라이버를 접해보지 못했기 때문에 제 경험은 매우 제한적입니다.

SHA-1
: 시그니쳐를 사용해야만 하며, 그 시그니쳐는 SHA-2를 사용해서는 안됩니다. 오직 SHA-1 입니다.

No SHA-2
: *만약* 시그니쳐가 존재한다면 SHA-2를 절대 써서는 안됩니다.

인터넷 익스플로러 요구사항: 만약 인터넷에서 다운로드하게 할 실행 파일에 사인을 한다면 signtool을 실행하실 때 **/t** 옵션 말고 **/tr** 옵션을 꼭 사용하세요.

마이크로소프트가 SHA-1를 그만 쓰려고 합니다: 2013-11-12일에 발표된 Microsoft Security Advisory (2880823)에서, 마이크로소프트는 2016-01-01 이후에는 루트 인증기관이 SHA-1 코드 사이닝 인증서를 발급하는 것을 더 이상 허락하지 않겠다고 발표했습니다. 2014-09-12일 글을 쓰는 지금, 2016년까지 계속하여 GoDaddy는 SHA-1 인증서를 발급하지 않을 것이라고 했고, (웨비나 동안에 제가 봤던 그쪽의 사용자 인터페이스로 추측하건데) GlobalSign도 동일하게 갈 것입니다. 2016년까지는 여기 문서에 기록한 SHA-2 관련된 모든 문제들 모두 마이크르소프트가 고쳤으면 좋겠습니다.

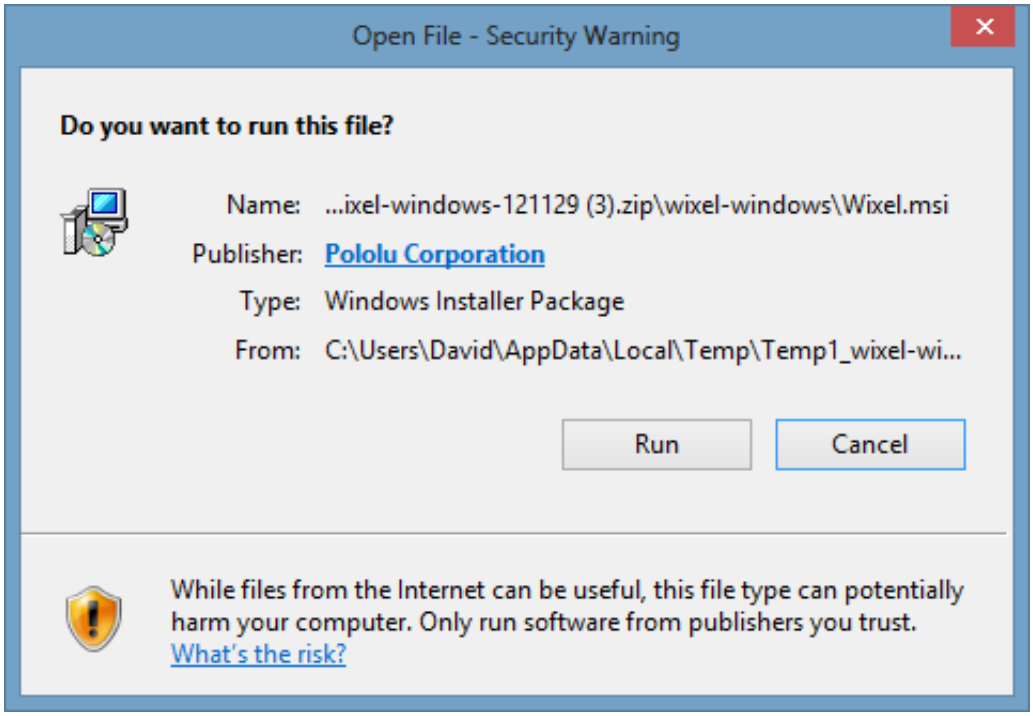
INF 파일 이름 중간에 공백을 사용하지 마세요. 이 내용은 Jimmy Kaz가 알려주었는데 드라이버 패키지 설치 파일에 대한 추가적인 요구사항입니다. 제가 직접 테스트 해보진 않았지만, 만약 INF 파일 이름 중간에 공백이 있으면 윈도우 7에서 드라이버 패키지가 사인되지 않았다고 나온다고 합니다.

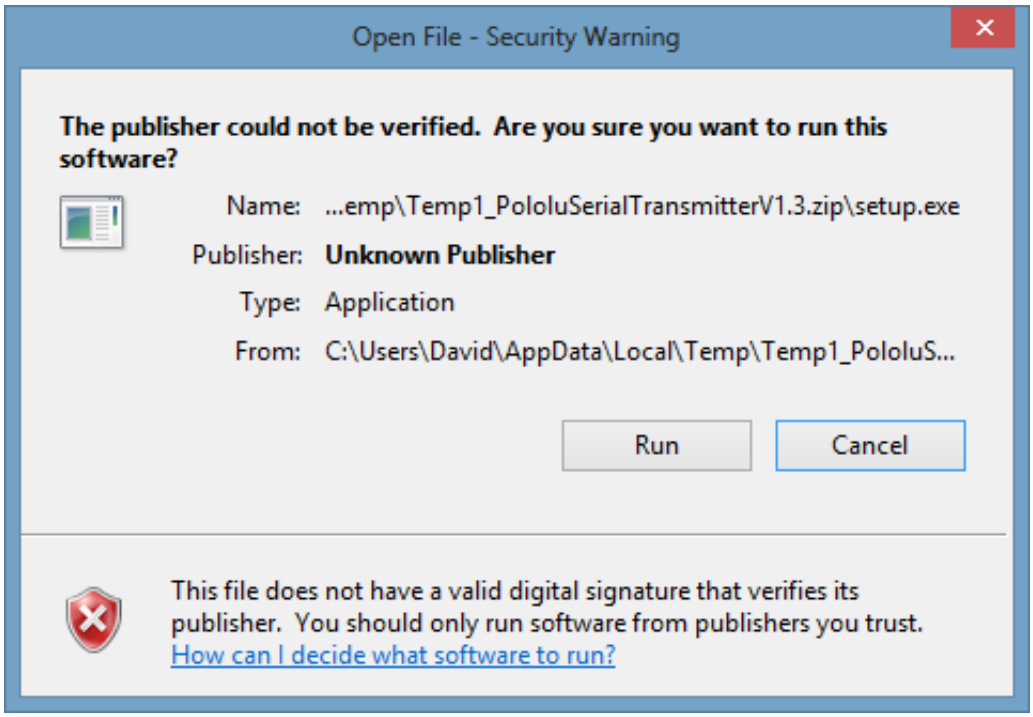
위 내용은 제가 알고 있는 모든 코드 및 드라이버 사인 요구사항들을 요약 정리한 것입니다. 다음 섹션에서는, 제가 요구사항들의 각각의 내용에 대해서 추가로 설명하고, 소프트웨어가 그 내용을 따르지 않았을 때 어떻게 되는지를 살펴보겠습니다.

실행 파일 실행하기

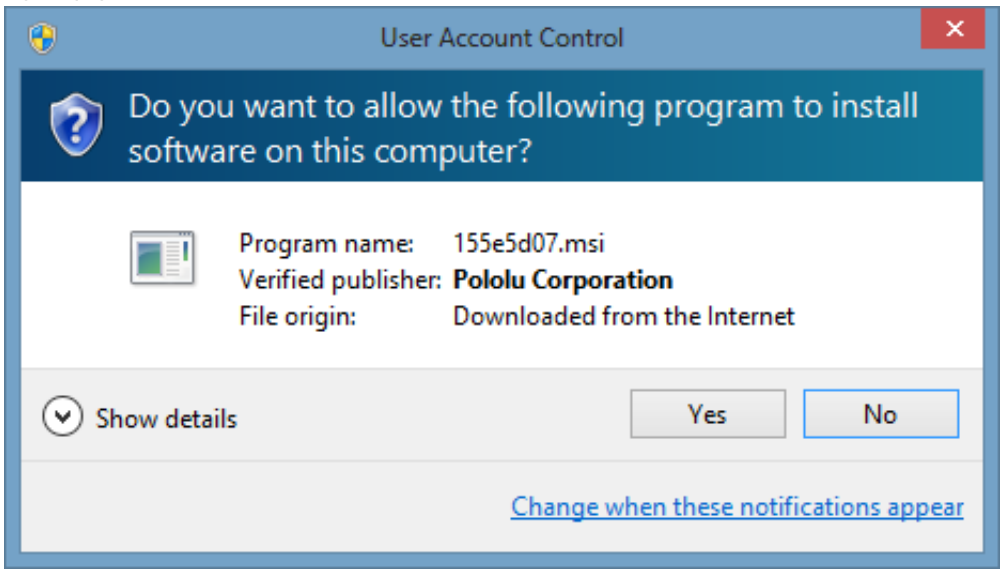
디지털 시그니쳐를 실행 파일(EXE 파일들)과 윈도우 인스톨러 파일(MSI 파일들) 내부에 직접 임베딩할 수 있습니다. 제가 이후에 실행 파일에 대해서 이야기 할 내용들은 전부 윈도우 인스톨러 파일에도 똑같이 적용됩니다. 윈도우는 실행 파일 안에 있는 시그니쳐를 두 가지 상황으로 구분해 검증합니다:

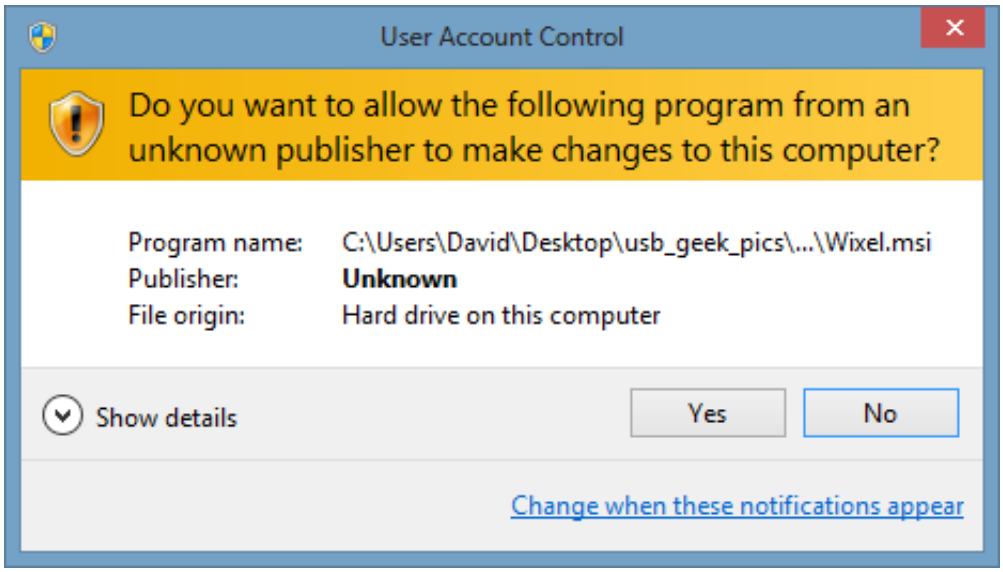
- 만약 파일이 인터넷(네트워크 드라이브 포함)에서 다운로드 되었다면, 윈도우는 사용자가 그 파일을 실행하려고 할 때 "Open File - Security Warning" 경고창을 보여줍니다. 그 경고창에 나와있는 퍼블리셔 정보는 파일 안에 임베딩 된 시그니쳐에서 가지고 옵니다. 보통 경고창은 간단한 다이얼로그 박스 형태지만, 윈도우 8에서는 화면 전체를 차지해 버리는 SmartScreen 경고창이 뜨기도 합니다. 아래 두가지 예제가 있습니다:





만약 그 실행 파일이 관리자 권한을 요구한다면, 윈도우는 UAC prompt 프롬프트를 보여줍니다. 프롬프트의 퍼블리셔 정보는 파일에 임베딩 된 시그니쳐에서 가지고 옵니다. 다음은 그 두 종류의 예입니다:





윈도우는 실행파일에 시그니쳐가 필수라고 말한 적은 없습니다. 그러나, 실행 파일에 사인을 해서 사용자가 퍼블리셔 알수없음 이라고 보는 것 보다 거기에 이름을 보이게 한다면 괜찮은 일이겠죠.

제 경험상, 실행 파일에 여러분의 시그니쳐를 제대로 사인하려면, certmgr.msc를 실행했을 때 나오는 신뢰할 수 있는 루트 인증 기관 리스트에 여러분의 인증서가 다시 연결되게 끔 신뢰 체인을 구성하셔야만 합니다. 또한, 특정한 경우 SHA-2를 쓰지 않는게 무척이나 중요합니다.

실행 파일에는 SHA-2 사용하지 마세요

만약 여러분들이 실행 파일의 시그니쳐에 SHA-2를 사용한다면, 고생을 좀 할 겁니다. **윈도우 비스타에서, 사용자가 SHA-2를 사용하는 시그니쳐를 가진 실행 파일을 다운로드 하고 더블클릭하면, 아무런 일도 일어나지 않습니다! 에러 메시지도 안뜨고 그냥 아무런 일도 발생하지 않습니다.**

혹시 제 말이 안 믿기신다면, 윈도우 비스타 컴퓨터에 이 wixel-signing-experiment.zip 파일을 직접 다운로드하고 실행해 보세요:

제가 생각하기에는 윈도우 비스타에서 다운로드한 실행파일의 경고창(warning dialog for downloaded executables)을 띄울 때 실행파일의 시그니쳐를 체크하는 어떤 코드에 문제가 있는게 아닌가 하고 생각됩니다. 그게 아마 SHA-2를 처리하지 못해서 조용하게 끝나버리는게 아닐까요. 한가지 해결책은 명령창에서 실행파일을 실행해서 경고창을 바이패스해서 시그니쳐 체크하는 부분도 건너 뛰도록 하는 것입니다.

이건 정말로 짜증나는 일입니다. 왜냐하면 signtool의 *verify* 기능을 사용하거나 윈도우 비스타에서 여러분의 인스톨러를 미리 테스트 해 본다고 하더라도, 문제점을 잡아내지 못할 것이기 때문이죠. 이 문제는 사인한 실행파일을 웹사이트에 실제로 올리고 윈도우 비스타 사용자들이 그걸 다운로드 해서 실행할 때에만 나타납니다.

또한, 이것은 시그니쳐를 추가해서 최소한 나쁜 일은 없을 것이라는 기본 믿음조차도 깨버립니다. 최소한 시그니쳐가 없는 것보다는 나아야 하는데요.

이 문제는 NSIS로 만든 인스톨러에는 영향을 끼치지 않는 것으로 보이는데, 그 이유를 전 알것 같

습니다. NSIS 인스톨러들은 항상 윈도우에 의해 pre-emptively elevated 되는데, 그래서 여러분이 NSIS 인스톨러를 실행할 때 여러분은 보통의 UAC 경고창만 보고 다운로드 된 실행파일을 실행해서 나오는 그 특별한 경고창을 보지 못하게 되는 것입니다. 그렇기 때문에 문제있는 윈도우 비스타 코드는 바이패스 됩니다.

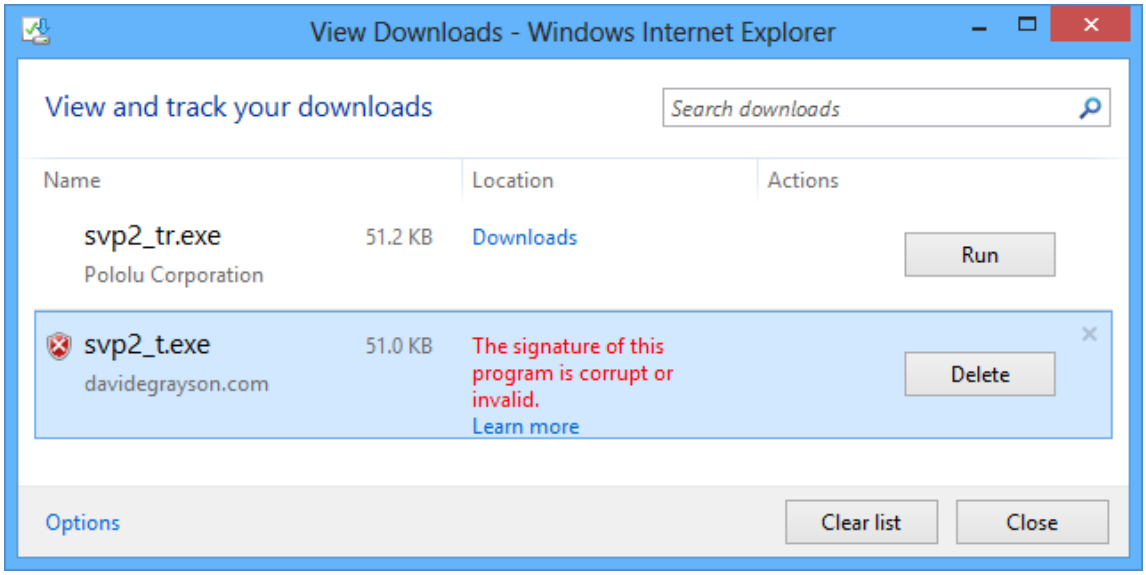
여러분의 시그니쳐가 SHA-2가 아닌 SHA-1를 사용하도록 확실히 해 두려면, 시그니처의 "Digest algorithm" 뿐만 아니라, 여러분의 인증서 경로에 위치한 모든 인증서들의 "Signature algorithm"과 "Thumbprint algorithm"도 확실하게 체크해 두셔야 합니다.

2014-09-12 업데이트: 제가 아직 테스트 해 보진 않았지만, KB2763674 업데이트가 SHA-2 실행파일로 발생하는 문제들을 고쳤다고 합니다. 이 업데이트는 윈도우 업데이트를 통해 설치할 수 있긴 하지만, 얼마나 많은 윈도우 비스타 유저들이 실제로 이걸 설치할 수 있을지는 모르겠습니다. 그러므로 SHA-1를 계속 쓰는 것이 최선입니다(MSDN 스레드에 이 내용을 제보해 주신 Yuhong Bao 감사합니다)

타임 스탬프 서버를 지정하려면 /tr 옵션을 쓰세요

인터넷 익스플로러는 실행 파일을 다운로드 하자마자 바로 실행 파일의 시그니쳐를 체크하는 보안 스캔 기능을 기본으로 내장하고 있습니다. 만약 여러분이 실행 파일을 사인하기 위해 signtool 옵션에 **/t** 옵션을 사용해서 인터넷에 올려둔다면(zip 파일이나 인스톨러에 넣어서 배포하는 것이 아니고), 인터넷 익스플로러는 그 파일을 다운로드한 사람들에게 "이 프로그램의 시그니쳐가 깨졌거나 손상되었습니다(The signature of this program is corrupt or invalid)."라고 알려주기 때문에, 그 프로그램을 실행하기 어렵게 만들어 버립니다(아예 불가능한 것은 아닙니다). 해결책으로는 signtool을 실행할 때 타임스탬프 서버 지정에 **/tr** 옵션을 사용하는 것입니다.

제가 이것을 윈도우 7 64-bit SP1 버전에서 인터넷 익스플로러 10.0.9200.16686 버전으로 직접 실험(experiment)을 통해 결론을 내렸는데, 제 시그니쳐들은 윈도우 8용 Windows Software Development Kit (SDK)에 있는 signtool을 사용해서 만들었습니다. 저는 윈도우 8 64-bit 버전의 인터넷 익스플로러 10.0.9200.16688 버전에서도 똑같은 결과를 얻었습니다.



이 역시 시그니처를 추가하는 것이 안하니만 못하다는 상식을 져버렸습니다. 인터넷 익스플로러는 타임스탬프 값을 아예 안주고 만들었을 때는 완벽하게 돌아갔습니다. 실행파일이 아예 사인이 안되어 있으면, 인터넷 익스플로러는 사용자들에게 그냥 경고만 하고 그 프로그램을 실행할 수 있게는 해 줍니다. 저는 아무나 파일을 조작할 수 있게 실행 파일에 사인을 아예 안하는 것보다 **/t** 옵션으로 실행 파일에 타임스탬프 값을 주는 것이 어째서 더 위험한 일인지를 모르겠습니다.

드라이버 패키지 설치하기

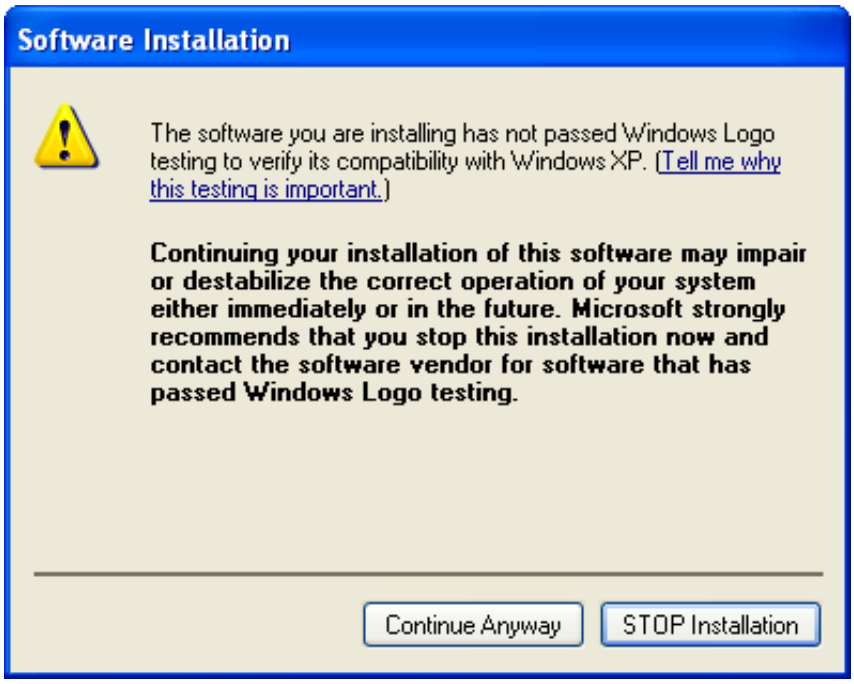
드라이버 패키지는 INF file 파일과 그 파일이 참조하는 파일들로 구성됩니다. 여러분은 같은 디렉토리에 여러 개의 INF 파일들을 포함할 수 있지만, 제 경험상 윈도우는 각각의 INF 파일을 개별적이고 독립적인 드라이버 패키지로 다룹니다. 드라이버 패키지는 모든 파일들의 해쉬 값으로 보안 카탈로그 파일을 우선 생성하고 난 뒤에 그 보안 카탈로그에 시그니쳐를 임베딩 하는 방법으로 사인을 합니다. 그 보안 카탈로그는 파일 이름들의 리스트와 각 파일들의 해쉬를 갖게 됩니다. 이 보안 카탈로그 파일을 간단히 더블클릭 해 보면 담고 있는 정보와 그것들의 시그니쳐를 살펴볼 수 있습니다.

드라이버 패키지를 설치하는 데에는 최소 네 가지의 방법들이 있습니다. 첫재, 만약 그 INF 파일이 DefaultInstall section을 가지고 있다면 사용자가그 INF 파일에 마우스 우클릭을 해서 “Install” 항목을 선택하여 설치할 수 있습니다. 두번째, 프로그램이 SetupCopyOEMInf 함수를 호출하여 설치할 수 있습니다. 세번째, 프로그램이나 사용자가 PnPUtil을 실행할 수 있습니다. 네번째, 사용자가 디바이스 관리자에서 해당하는 디바이스에 마우스 우클릭을 해서 "Update driver software...” 선택한 다음에 윈도우에게 드라이버 패키지가 어디에 저장되어 있는지를 선택하여 설치할 수 있습니다.

드라이버 패키지 설치가 시작될 때, 윈도우 버전마다 시그니쳐는 다르게 체크됩니다.

윈도우 XP에서 드라이버 패키지 설치

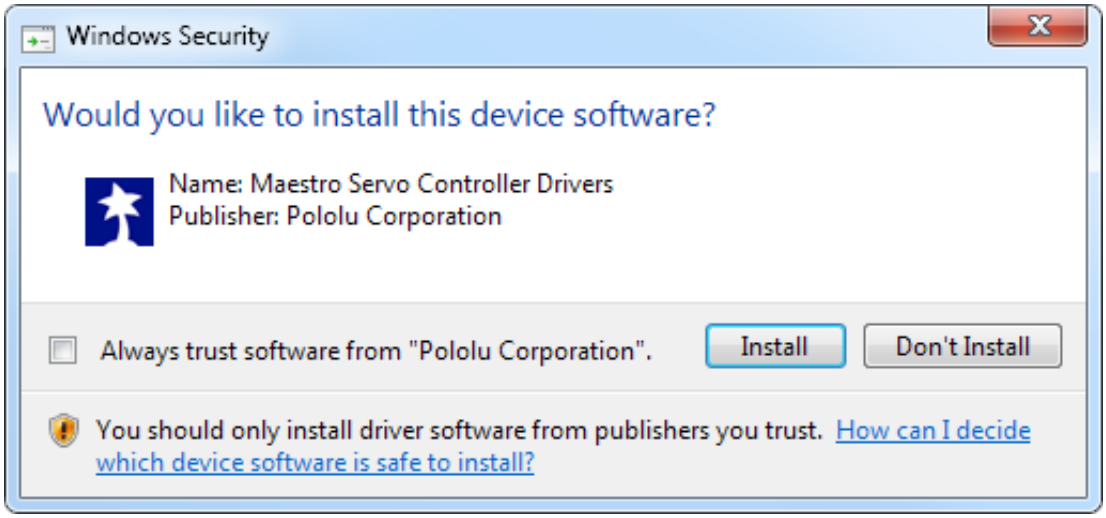
윈도우 XP 때에는, 마이크로소프트는 드라이버 시그니쳐 중에서 WHQL 시그니쳐만 유일하게 신경을 썼던 것 같습니다. 여러분이 윈도우 비스타 이후 버전에서는 설치 프롬프트가 잘 나오는(nice install prompt in Windows Vista and up) 드라이버 패키지를 갖고 있더라도, 그 동일한 드라이버 패키지를 윈도우 XP에 설치를 하면 다음과 같은 경고창이 나옵니다:



윈도우 비스타와 7에서 드라이버 패키지 인스톨

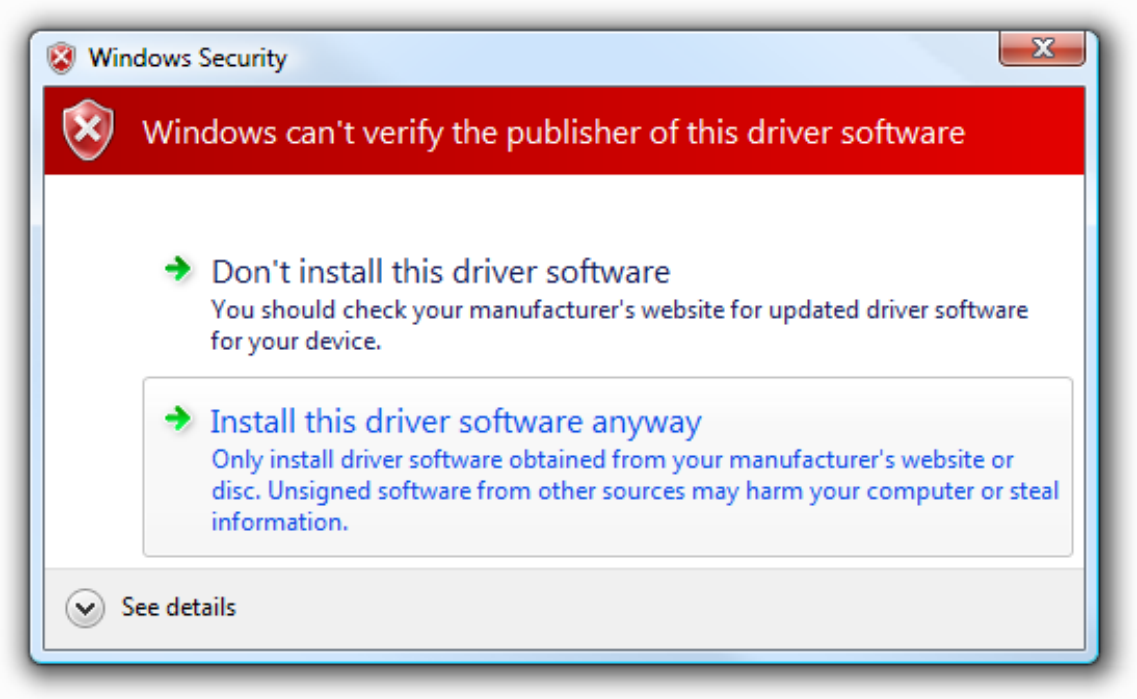
마이크로소프트는 윈도우 비스타 때부터 약간의 조율을 하기 시작했습니다. 드라이버들이 WHQL 테스트를 통과했는지 아닌지를 사용자 경고 메시지로 내보내는 대신에 윈도우 비스타와 윈도우 7은 사용자들에게 퍼블리셔가 검증되었는지 아닌지의 여부를 경고하기 시작했습니다. 검증된 퍼블리셔라고 보여주려면 여러분은 적절한 시그니쳐로 사인한 CAT 파일을 제공해야 합니다. 이 요구사항은 kmsigning.doc 문서와 위 signature requirements 섹션에 설명되어 있습니다.

만약 드라이버 패키지를 제대로 사인했다면, 그 패키지를 인스톨 할 때 윈도우 비스타, 7, 8에서는 아래의 친숙한 프롬프트를 보게 될 것입니다:



프롬프트에 나온 이름은 INF 파일의 DriverPackageDisplayName 디렉티브에서 구해온 것이고, 퍼블리셔 정보는 CAT 파일에 있는 검증된 시그니쳐로부터 온 것입니다.

그러나, 만약 드라이버 패키지를 사인하지 않았다면 사용자들은 윈도우 비스타나 윈도우 7에서 그 패키지를 인스톨 하려할 때 다음의 빨간 경고창을 보게 될 것입니다:



제 생각으론 마이크로소프트가 이런 변경을 결정한 것이 잘한 것이라고 생각됩니다. WHQL 테스트는 비싸고 유연하지 않죠. 만약 여러분이 드라이버 한글자만 바꿔도 여러분은 다시 테스트하고 제출해야 합니다. 그냥 보통 코드 사이닝이 훨씬 쉽고 쌉니다: 여러분은 일년에 몇십만원만 내면 인증서도 얻을 수 있고 그걸로 원하는 만큼의 드라이버 패키지들을 사인할 수 있으니까요. 이렇게 되면 더 많은 회사들이 드라이버를 사인하게 될 것이고, 결국은 맬웨어(그리고 우리같은 작은 회사들도)가 더 눈에 띌 수 있겠죠.

윈도우 8에서 드라이버 패키지 설치

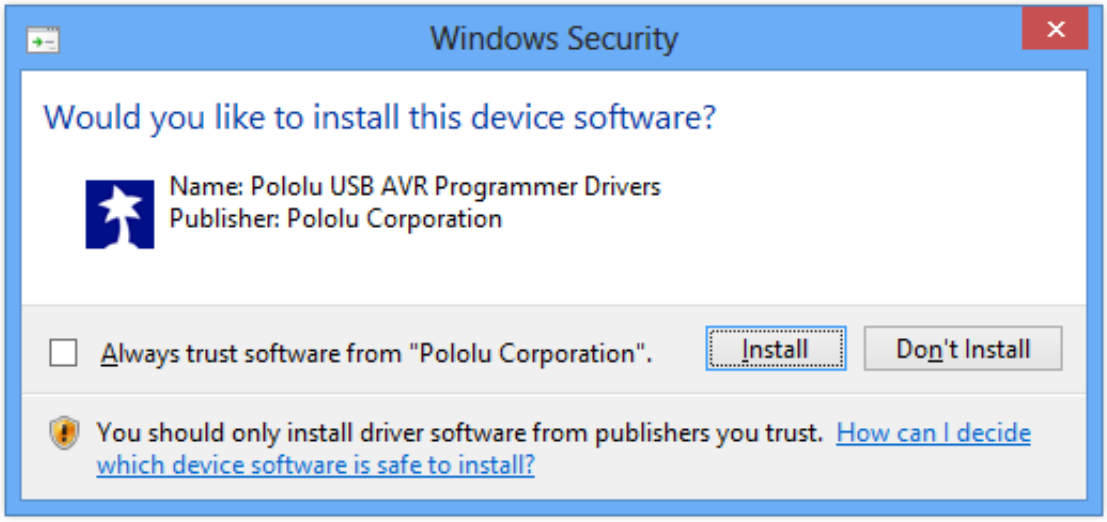
윈도우 8부터는, 모든 드라이버 패키지들은 사인되어 있어야만 합니다. 안타깝게도, 이 변경사항에 대해서 적어놓은 마이크로소프트의 공식 문서를 아직 찾지 못했습니다. 스택오버플로에 질문 (asked about it)도 했지만 말입니다.

만약 윈도우 예전 버전에서 동작했던 사인되지 않은 드라이버 패키지를 인스톨 하려고 하면, 다음과 같은 간단한 에러 메시지를 보게 될 것입니다:

이것은 마이크로소프트 쪽에서 보자면 나쁜 결정이 아니지만, 작은 규모로 USB 디바이스들을 만드는 개인이나 저희 같은 수많은 작은 회사들에게는 나쁜 소식입니다. 몇년간, 저희들은 윈도우 XP와 비스타, 윈도우 7에서 WinUSB와 usbser.sys와 같은 마이크로소프트에서 사인한 커널 드라이버(SYS 파일들)를 사용하는 사인되지 않은 드라이버 패키지들을 배포해 왔고 잘 동작해 오고 있었습니다. 윈도우 8부터는 드라이버 사인 절차를 파악해야 했고, 윈도우 8을 사용하는 고객들에게 드라이버 시그니쳐 검증 옵션을 끄라고 복잡한 과정을 설명해 주어야 했습니다.

만약 이쪽 분야가 처음이고, USB 디바이스를 만들기를 시작하고 싶다면, USB-IF에 vendor ID를 얻기 위해서 우선 $2000 불을 내야 하고, 윈도우 8을 위한 코드 사인 인증서를 구입하는데 일년에 몇백 달러가 또 들 것입니다. 그러나, 사인할 때 타임스탬프를 사용했다면 인증서가 만료되더라도 시그니쳐가 잘 동작합니다.

윈도우 8에서 사인된 드라이버 패키지들의 드라이버 설치 프롬프트는 윈도우 비스타, 윈도우 7에서의 화면과 매우 비슷합니다.



커널 드라이버 로딩하기

드라이버 패키지들은 커널 모드 드라이버 파일(SYS 파일들)을 포함하고 있고 이것들은 보통 디바이스가 컴퓨터에 연결될 때 커널에 로드됩니다.

윈도우 비스타 64비트부터는 커널 드라이버는 반드시 적절하게 사인된 보안 카탈로그 파일(CAT 파일)이 있어야 커널에 로드할 수 있습니다. 2007년 7월, 윈도우 비스타가 출시된지 6개월 후에 마이크로소프트는 새로운 사인 요구사항에 관한 두가
지 kmsigning.doc and KMCS_walkthrough.doc 문서를 내놓았습니다.

만약 여러분의 드라이버가 WinUSB 또는 usbser.sys 드라이버만을 사용한다면, Installing a driver package 섹션에 설명되어 있는 것처럼 여러분의 드라이버 패키지를 설치하는 것만 신경쓰시면 됩니다. 여러분이 사용하고 있는 커널 모듈들은 이미 마이크로소프트에 의해 사인이 되어 있기 때문에 드라이버 패키지가 인스톨 된 후에는 그것들을 커널에 로드해서 쓰는데 문제가 없을 것입니다.

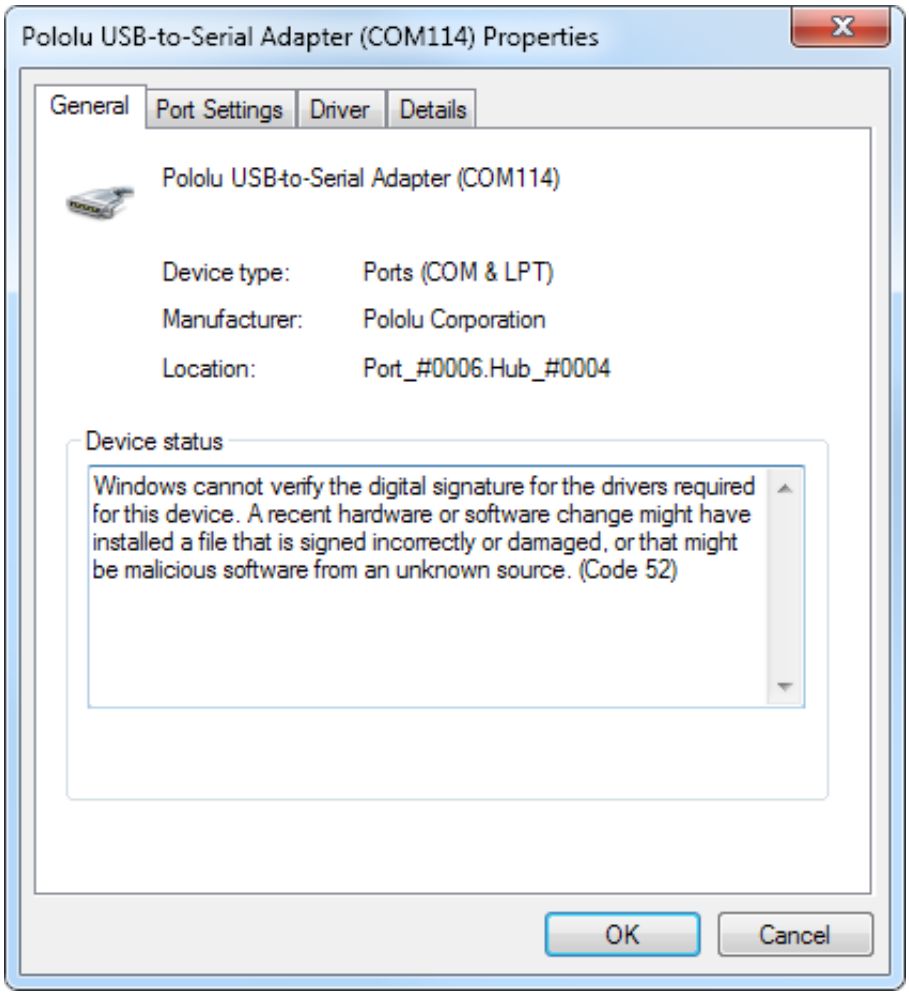
커널 드라이버에 대한 시그니쳐는 드라이버 패키지를 위한 보안 카탈로그 파일(CAT 파일)에 보통 보관되어 있지만, kmsigning.doc에 따르면 부트-스타트 드라이버의 경우에는 SYS 파일 그 자체에 시그니쳐를 임베딩 해야 한다고 설명하고 있습니다.

시그니쳐의 신뢰 체인은 Microsoft Code Verification Root 인증서 또는 커널이 신뢰하는 인증서로 반드시 연결되어야만 합니다. 모든 WHQL 시그니쳐는 이 요구사항을 만족합니다. 이 내용은 kmsigning.doc 문서에 명확하게 문서화 되어 있는데, 그 문서에서는 커널이 Trusted Root Certification Authorities 리스트에 접근할 수 없기 때문이라고 설명하고 있습니다. 그래서 크로스 인증서가 이 요구사항을 만족시키기 위해서 필요합니다. 마이크로소프트는 Cross-Certificates for Kernel Mode Code Signing에 전체 리스트를 올려 놓았습니다.

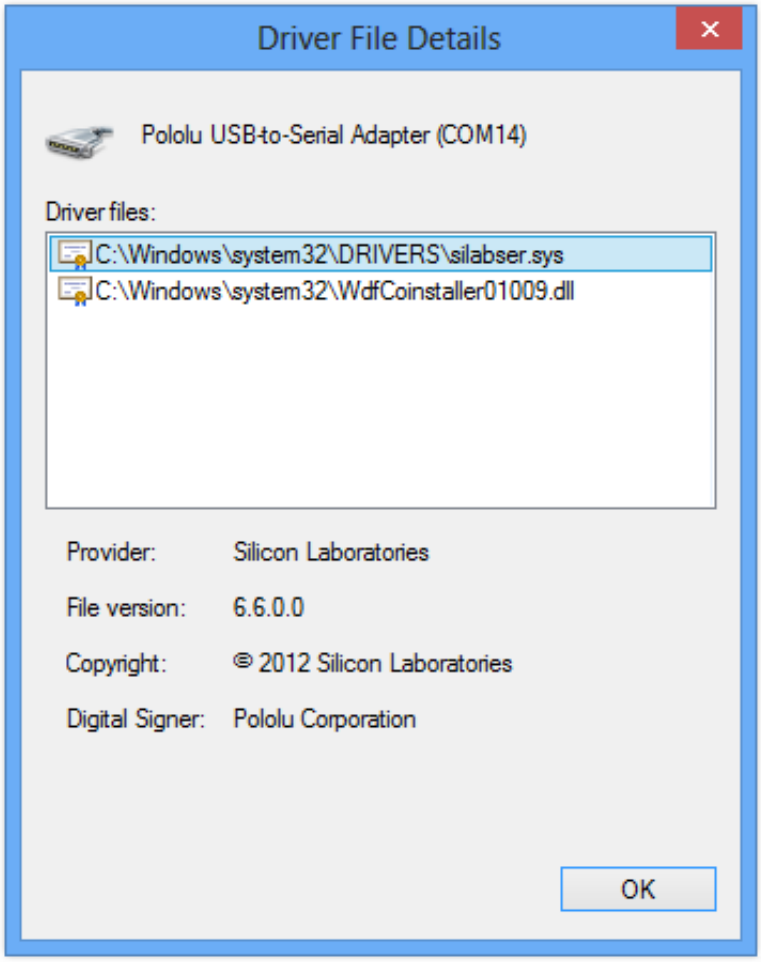
한가지 사소하지만 중요한 점은 시그니쳐는 시스템에 설치된 어떠한 보안 카탈로그 파일에서도 올 수 있다는 점입니다. 시그니쳐는 실제로 꼭 그 디바이스와 매칭하는 INF 파일을 위한 보안 카탈로그 파일 안에 있지 않아도 됩니다. 다른 말로 하자면, 만약 X라는 회사가 윈도우 비스타와 윈도우7을 위해 커널 드라이버를 만들고 적절하게 사인을 했고, Y라는 회사가 그 커널 드라이버를 사용하는 자신들만의 드라이버 패키지를 만든다면 회사 Y는 윈도우 비스타와 윈도우7에서 돌아가게 하기 위해 별도의 시그니쳐를 또 제공할 필요는 없다는 뜻입니다. 곰곰히 생각해 보면 별로 놀라운 일은 아닙니다; 커널에 뭔가를 로딩할 때 위험한 점은 커널 드라이버 그 자체이지, INF파일이나 같이 사용하는 디바이스가 아니기 때문입니다. 실제로, 이러한 이유 때문에 저희들의 제품인 CP2102 USB-to-Serial Bridge Driver가 윈도우 XP/비스타/7/8에서 잘 돌아가고 있습니다. 제가 우리 카탈로그 파일 (plluvcp.cat)에 넣은 시그니쳐는 커널에 silabser.sys 드라이버를 로드하기에는 요구사항을 다 만족시키지 못하지만, Silicon Laboratories가 그들의 카탈로그 파일 (slabvcp.cat)에 넣은건 요구 사항을 다 만족하기 때문에 그것을 사용중인 저희 것도 잘 동작합니다.

SYS 드라이버 파일을 커널에 로드할 때, 윈도우는 보안 카탈로그 스토어 (*C:\Windows\System32\catroot*)에 있는 모든 파일들을 스캔해서 SYS 파일에 대한 해쉬와 그에 해당하는 시그니쳐를 담고 있는 파일이 있는지를 스캔합니다. 만약 찾으면, 로딩이 성공합니다.

만약 윈도우가 여러분의 커널 드라이버를 위해 적절하게 사인된 보안 카탈로그 파일을 찾지 못하면, 여러분 디바이스의 프로퍼티 다이얼로그에는 코드 52 에러가 나올 것입니다:



커널 드라이버에 대한 디지털 시그니쳐는 디바이스 매니저에서 사용자들이 보게 될 내용에도 영향을 미칩니다. 드라이버를 사용하는 디바이스를 더블클릭하고, **자세히 탭**을 선택하고, 드라이버 **세부내용** 클릭하세요. 여러분의 회사 이름이 이 다이얼로그 박스에 나온다면 제일 좋겠지만, 그렇게 하려면 어떤 요구사항을 충족해야 하는지 거기까지는 아직 알아내지 못했습니다. 이것이 제가 왜 "Signature requirements for it to look good" 항목의 "커널 드라이버 로딩하기" 컬럼에 물음표를 달아놓은 이유입니다.



윈도우 8 이전 버전을 대상으로 하는 커널 드라이버에는 SHA-1 사용하세요

SHA-2는 SHA-1보다 더 나은 해쉬 알고리즘입니다. 그러나 불행하게도, 윈도우 8이전 버전에서 SHA-2 지원은 좀 별로입니다.

저는 보안 카탈로그 파일을 Go Daddy에서 받은 SHA-2 인증서로 사인했을 때 윈도우 7에서 커널 드라이버(silabser.sys) 로드가 실패했고, 동일한 드라이버 패키지로 윈도우 8에서 정상적으로 동작하였습니다. 안타깝게도, Go Daddy는 Microsoft Code Verification Root로 신뢰 체인이 연결되는 SHA-1 인증서를 제공하지 않기 때문에, 패키지는 그대로 두고 SHA-1 인증서만 바꿔서 윈도우7에서 돌아가는지 테스트는 해 볼 수 없었습니다. 그리고 윈도우 7에서 동작하는 다른 커널 모드 드라이버들 살펴보았을 때 모두 SHA-1을 사용하는 것을 확인했습니다. 단, 윈도우 비스타에서는 테스트하는 것을 깜빡했습니다.

여기에서 중요한 점은 제 SHA-2 시그니쳐로 사인한 드라이버 패키지가 설치만 목적으로 했을 때 드라이버 패키지는 윈도우 7에서 정상적으로 설치되었다는 점입니다. 그러나 동일한 시그니쳐가 커널에 코드를 로드하는 것까지는 충분하지 않았나 봅니다. 제가 보기엔 윈도우마다 다 다른 시그니쳐 확인 코드들이 동작해서 윈도우7 같은 버전 안이라도 다 똑같이 동작하는게 아닌 것 같습니다.

2013–03-11 업데이트: 이 섹션은 틀릴 수도 있습니다! PiXCL Automation Technologies에 있는 한 개발자가 저에게 연락해서 이 섹션에 있는 내용이 틀렸다고 주장했습니다: 그는 Go Daddy에서 받은 SHA-2 드라이버 사이닝 인증서를 사용해서 윈도우7에 커널에 드라이버를 로드할 수 있었다고 합니다. 다만, 그는 윈도우 7을 지원하기 위해 `/fd sha1` 옵션을 사용해서 사인하고, 윈도우 8을 지원하기 위해 `/fd sha256` 옵션으로 사인을 두번 했다고 합니다. 저는 그의 주장을 검증할 자료는 없는데, 그 분은 매우 능숙해 해보였고, 설득력도 있었습니다. 여러분은 그의 튜토리얼을 Signing Windows 8 Drivers에서 확인하실 수 있습니다.

인증서 선택하기

여러분은 인증서 발급 회사를 정해서 코드 사이닝 인증서를 구입해야 합니다.

Code Signing Certificate from GlobalSign
GlobalSign에서 파는 코드 사인 인증서

제가 직접 해본것은 아니지만, Globalsign에서 판매하는 code signing certificate는 모든 방면에서 완벽해 보입니다. 마케팅 “데이터시트”에서는 특히 "Vista Kernel Support” 기능도 있다고 주장하고 있습니다. GlobalSign 크로스 인증서는 SHA-1을 사용합니다. "GlobalSign Root CA” 인증서는 아마도 모든 사용자의 컴퓨터에 신뢰할 수 있는 루트 인증서 기관(Trusted Root Certification Authority)으로 설치되어 있을 것 같은데, 완전히 확실한 것은 아닙니다. 그리고 비교적 쌉니다: 2012년 12월 기준으로, 기업용은 일년에 $229, 개인 개발자 용은 $129밖에 안하고 한꺼번에 3년치를 사면 더 쌉니다.

GlobalSign의 인증서를 구입해 보시길 추천합니다. 잘 됐는지 결과를 커멘트로 알려주세요(post a comment)

Go Daddy의 코드 사이닝 인증서

만약 여러분이 실행 파일이나 드라이버 패키지만 사인해야 하고 커널 드라이버 코드(SYS 파일)는 포함하지 않는다면 Go Daddy Code Signing Certificate에서 판매하는 $199.99/year 인증서가 가장 괜찮습니다. 2015년 중반쯤에 GlobalSign에서 SHA-2 인증서로 바꿀 계획이 있긴 하지만 이것이 현재 저희가 사용하는 인증서입니다. 만약 여러분의 USB 드라이버가 usbser.sys나 WinUSB를 사용한다면 여러분의 드라이버 패키지는 커널 드라이버 파일들을 추가로 포함하지 않을 것이므로 이 인증서는 그 상황에서 잘 동작할 것입니다.

여러분의 코드 사이닝 인증서를 위한 개인 키를 생성할 때, SHA-1와 SHA-2(정확히 말해서 SHA-256) 사이에서 선택하라는 프롬프트가 뜰 것입니다; 우리는 SHA-1을 골랐습니다. 또 발급 기관을 고르라고 물어보는데, Starfield와 Go Daddy 중에서 Go Daddy를 골랐습니다. 여러분은 이러한 결정 사항들을 나중에 Go Daddy 웹 인터페이스에서 다시 키발급을 하면서 바꿀 수 있습

니다.

만약 여러분이 Go Daddy에서 SHA-1을 골랐다면, 여러분의 인증서에 대한 신뢰 체인은 Go Daddy Class 2 Certification Authority로 따라서 돌아갈 것입니다. Go Daddy Class 2 Certification Authority는 아마도 거의 대부분의 사용자 컴퓨터에서 신뢰할 수 있는 루트 인증서 기관에 들어 있을 것이기 때문에 이건 좋습니다; certmgr.msc를 사용해서 이를 검증해 볼 수 있습니다.

Go Daddy에서 드라이버 사인 인증서를 구입하진 마세요; 이 제품은 코드 사이닝 인증서와 똑같이 보이고 동일한 가격인데 SHA-2 해쉬 알고리즘만 고를 수 있습니다.

GoDaddy는 2016년까지 유효기간이 있는 SHA-1 인증서를 발급하지 않으므로, SHA-1 인증서를 거기서 구입하길 원한다면 2015년 1월 1일 전에 해야만 할것입니다(다른 대부분의 인증서 기관들도 비슷할 거라고 생각합니다)

좋은 인증서 찾기

다목적의 좋은 인증서를 찾기 위해서, 저는 다음처럼 합니다:

1. 마이크로소프트에 있는 Cross-Certificates for Kernel Mode Code Signing 페이지에서 모든 크로스 인증서를 다운로드합니다. 그 페이지에 리스트에 나와있는 회사들만이 Microsoft Code Verification Root로 연결되는 신뢰 체인을 가진 인증서들을 제공합니다, 그러므로 여러분의 커널 드라이버 모듈을 사인할 수 있는 유일한 인증서들입니다.
2. 다운로드한 크로스 인증서를 모두 열어서 "Signature algorithm" and "Thumbprint algorithm" 항목이 모드 SHA-1인지 확인합니다. 제가 위에서 설명한 것처럼, SHA-2는 윈도우 비스타를 위한 실행 파일들(또는 MSI 파일들), 그리고 윈도우 비스타와 윈도우7을 위한 커널 드라이버를 사인하는데 절대 사용해서는 안됩니다. 그러므로 SHA-2를 사용하는 모든 인증서들을 삭제합니다. 2012년 12월 기준으로 SHA-2를 사용하는 유일한 크로스 인증서는 Go Daddy 것이였습니다.
3. 주변에 있는 여러대의 윈도우 컴퓨터들을 붙잡고 certmgr.msc를 실행해서 신뢰할 수 있는 루트 인증기관 리스트에 이미 어떤 인증서들이 등록되어 있는지 직접 살펴봄으로써 리스트 범위를 더 좁힐 수 있습니다. 예를 들어서, 제 윈도우 8 컴퓨터에서는 신뢰할 수 있는 루트 인증서 기관에 "GlobalSign Root CA"가 있고, 마이크로소프트 웹페이지에 있던 크로스 인증서 중 하나와 "Subject"도 매치합니다. 두개의 인증서들의 "Subject Key Identifiers" 역시 매치합니다. 이것은 Globalsign의 인증서가 윈도우 8에 기본으로 깔려서 온다는 것을 뜻합니다. 그러나 Globalsign의 인증서가 윈도우8에 기본으로 깔려온 것이 아니라 자동으로 윈도우 업데이트 되어서 깔린 것일 수도 있습니다. 여러분의 목적은 여러분 제품을 사용할 모든 최종 사용자의 컴퓨터에 Trusted Root Certification Authority 신뢰할 수 있는 루트 인증서 기관의 인증서로 신뢰 체인이 연결되어 있는 인증서를 사는 것이므로, 윈도우 업데이트에 의존하지는 마세요.
4. 이제 남은 인증서들이 좋은 인증서를 판매할 후보 회사들입니다. 웹사이트에 들어가서 여러 인증서들 제품과 가격들을 비교해 보세요. 특히 여러분이 사려고 하는 제품이 드라이버 사인에 쓸 수 있는지를 확실히 체크해 보고 사세요, 왜냐하면 어떤 회사들은 하나 이상의 루트 인증서를 가지고 있고 여러분의 인증서의 신뢰 체인에 속해 있는지를 확인하기 어려운 경우가 있기 때문입니다. 만약 가능하다면 여러분이 챙겨야 하는 signature requirements를 모두 만족하는지 그들이 팔고 있는 인증서로 사인한 샘플을 구해보세요. 괜찮은 환불 정책이 있는지도 체크하세요.

사인하기

이 섹션에서는 코드 사인 인증서를 구입한 후에 실제로 그것을 어떻게 사용하는지를 설명하도록 하겠습니다. 이 설명은 공식 문서들에서 대부분 찾을 수 있지만, 이런 문서들(예를 들자면 kmsigning.doc)은 대부분 옛날 자료들입니다.

제일 먼저, 여러분의 인증서 제공 회사들의 지시부터 따르세요. 개인 키를 발급하는 절차나, 그것들을 여러분의 컴퓨터에 설치하는 절차와 같은 것들 말입니다. 이들 지시사항을 따랐으면, certmgr.msc를 실행해서 여러분의 인증서가 이상이 없는지를 확인해 보세요. 이들 인증서는 “Personal” 카테고리에 있어야 하고, 그 인증서의 등록정보 다이얼로그에서는 “이 인증서와 관련된 개인키가 있습니다(You have a private key that corresponds to this certificate.")란 메시지가 떠야 합니다.

크로스 인증서

필요한 signature requirements 시그니쳐 요구사항들을 모두 만족하도록 신뢰 체인을 확장하기 위해서는 적절한 크로스 인증서를 다운로드 해야 합니다. 모든 표준 크로스 인증서들은 Microsoft Code Verification Root로 연결되도록 되어 있으며 available for download from Microsoft 에서 다운로드 가능합니다. 인증서 제공사에서 그들의 웹사이트에 크로스 인증서를 다운로드할 수 있도록 제공하는 경우도 있습니다. 크로스 인증서를 사용하기 위해서는 signtool을 실행할 때 `/ac "path-to-your-cross-cert.ct"` 포함시켜 주어야 합니다.

타임스탬프 서버

꼭 타임스탬프를 하셔서 여러분의 시그니쳐가 만료되더라도 계속 동작하도록 하세요. 인증서를 구입한 곳에서 알려준 문서를 찾아보시면 타임스템프 서버의 URL이 있을 겁니다, 아니라면 다른 제공사의 타임스탬프 서버를 무료로 사용할 수도 있습니다. 제가 Verisign 고객이 아니지만 Verisn 타임스탬프 서버를 저는 잘 사용했습니다. 시그니쳐에 타임스탬프를 찍으려면, signtool을 실행할 때 `/tr http://timestampserver.com` 형태로 포함하셔야 합니다.

Signtool과 inf2cat

사인을 하려면 마이크로소프트에서 나온 Signtool.exe (Sign Tool)을 사용합니다. signtool을 구하려면 최신 버전의 Windows SDK을 설치하시면 됩니다.

드라이버 패키지를 사인하기 위해서는, 우선 보안 카탈로그 파일(CAT 파일)을 만들기 위해 Inf2Cat (Inf2Cat.exe)라고 하는 다른 툴을 사용해야 하며, 그 다음에 signtool로 사인할 수 있습니다. inf2cat.exe를 구하려면, 최신 버전의 Windows Driver Kit를 설치하시면 됩니다. 가장 최신 버전의 윈도우까지 지원하려면 가장 최신 버전의 툴을 구해서 실행하도록 하세요.

공식 문서에도 inf2cat과 signtool을 사용하는 방법이 나와있긴 합니다만, 아래에도 간략한 예제

가 있습니다.

배치 스크립트(.bat) 예제입니다. 여러분은 간단히 실행 파일이나 MSI 파일을 그 위에 드래그하여 사인하실 수 있습니다:

```
"C:\Program Files (x86)\Windows Kits\8.0\bin\x86\signtool" sign /
v /ac "your-cross-cert.crt" /n "Your company name" /tr http://tsa
.starfieldtech.com %1
pause
```

다음은 다른 배치 스크립트 예제입니다. 이 내용으로 배치 파일을 만들어서 여러분의 드라이버 패키지와 동일한 디렉토리에 넣어두고 더블 클릭하면 보안 카탈로그를 만들고 사인합니다.

```
"C:\Program Files (x86)\Windows Kits\8.0\bin\x86\inf2cat" /v /dri
ver:%~dp0 /os:XP_X86,Vista_X86,Vista_X64,7_X86,7_X64,8_X86,8_X64
"C:\Program Files (x86)\Windows Kits\8.0\bin\x86\signtool" sign /
v /ac "your-cross-cert.crt" /n "Your company name" /tr http://tsa
.starfieldtech.com *.cat
pause
```

이 두개의 배치 파일과 크로스 인증서를 동일한 디렉토리에 넣고 `/ac` 파라미터와 함께 간단히 실행하시면 됩니다.

검증하기

여러분이 아직 배우는 단계시라면 반드시 여러분의 시그니쳐를 signtool.exe의 검증 옵션을 사용해서 검증해 보아야만 합니다. signtool 공식 문서에서는 검증하는 옵션이 아주 헛갈리게 나와 있기 때문에 요점들만 정리해서 알려드리겠습니다:

- 실행 파일을 실행하기 또는 드라이버 패키지를 설치하기 위한 목적일 때의 시그니쳐를 테스트 하려면, 적절한 옵션은 `/pa` . 입니다. 이 내용은 KMCS_Walkthrough.doc로부터 얻었습니다.
- 커널 드라이버 로딩의 목적으로 시그니쳐를 테스트 하시려면 올바른 옵션은 `/kp` 입니다.

여기 `/pa` 를 사용하는 예제 배치 스크립트가 있는데 파일을 드롭하면 시그니쳐를 검증합니다:

```
"C:\Program Files (x86)\Windows Kits\8.0\bin\x86\signtool" verify
/v /pa %1
```

```
pause
```

안타깝게도, signtool 검증은 제한적으로만 사용될 수 있습니다. 즉 신뢰 체인이 제대로 필요한 곳으로 연결되어 있는지 검증할 수는 있지만, 제가 위에서 문서로 정리한 다른 시그니쳐 요구사항들 signature requirements의 대부분에 대해서는 알려주지 않습니다. 예를 들어서, SHA-2를 사용하는 것에 대해서 경고해 주지 않습니다. 여러분의 시그니쳐에 대한 확신을 실제로 가지려면, 그것들을 적절하게 **테스트** 해보는 수밖에 없습니다, 그러니까 계속 읽어보세요.

테스트하기

여러분이 배포하려고 하는 윈도우 모든 버전마다 여러분의 사인된 드라이버/실행파일들을 테스트 한다면 가장 이상적이란건 당연합니다.

그러나 많이들 실수하시는 부분이 실행 파일들이나 MSI 파일들을 테스트 하려고 할 때, 인터넷에서 그것들을 **다운로드 한 직후에 그 파일들을 실행해 보아야 한다는 점**입니다. 제가 Installing a driver package 섹션에서 설명한 것처럼, 윈도우 비스타에서는 만약 파일이 인터넷에서 다운로드 되었다면 그 자신만을 manifest하는 버그가 있고, 그와 유사한 다른 버그들도 많이 있을 것이기 때문입니다. 여러분의 실행 파일을 zip으로 묶어서 나만 아는 URL에 올려둔 후에 테스트 컴퓨터에 다운로드 하세요. 실행 파일을 실행하려고 할때 경고 메시지가 제대로 뜨는지 체크하여 테스트를 해 보실 수 있으실 것입니다.

다운로드 할 파일(ZIP 파일이나 인스톨 파일)을 인터넷 익스플로러에서 다운로드 해서 인터넷 익스플로러가 여러분의 시그니쳐를 체크할 때 문제가 없는지도 반드시 테스트를 해 보셔야 합니다. 제 경험상, 인터넷 익스플로러는 EXE 파일(아마도 MSI도 마찬가지일 것입니다)에 대해서 시그니쳐 체크를 합니다만, 나중에 ZIP 파일 내부에 있는 실행 파일에 대한 시그니쳐를 체크할 수도 있을 것입니다.

백업

만약 여러분 인증서 제공사가 Go Daddy와 유사하게 운영을 한다면, 그쪽에서는 여러분들을 위한 개인 키의 백업본을 따로 저장해두지 않습니다. 개인 키는 여러분 컴퓨터에만 오직 존재하며, 만약 그것을 잃어버리면, 여러분의 인증서를 재발급"re-key" 해야 합니다. 재발급("re-key")이 Go Daddy에서는 큰일은 아닙니다; 그들의 웹 인터페이스에서 무료로 할 수 있도록 하고 있습니다.

그러나 저는 여러분의 인증서와 개인키를 백업하기를 권장하고 있습니다. 그렇게 하시려면, 우선 certmgr.msc를 실행하고 여러분의 인증서를 선택한 다음에 **Action > All Tasks > Export...**를 선택하세요. 이렇게 하면 인증서 내보내기 마법사가 실행됩니다. "Yes, export the private key"를 선택하고, PFX 형식을 선택하고, "Include all certificates in the certification path if possible"선택하고, "Delete the private key if the export is successful"는 선택하지 말고, "Export all extended properties"를 선택하세요. 그러면 패스워드를 선택하라고 물어본 후 출력 파일의 이름을 물어봅니다. 모두 마치고 나면 여러분의 인증서와 개인 키를 포함하는 패스워드로 보호되는 Personal Information Exchange (PFX) 파일이 생성됩니다. 단순히 더블클릭하고 패스워드를 입력하면 다른 컴퓨터에 그 파일을 설치할 수 있습니다.

사인과 관련된 오해들

여러분이 도움을 청했을 때 좋은 의도로 사람들이 답변해 준 내용들이 결국에는 사실이 아니거나, 일부만 맞는 경우일 때는 참 짜증납니다. 여기 제가 들었던 흔한 오해들입니다.

오해: 커널 모드 드라이버들은 WHQL 테스트가 필요합니다

> Let's say that the booby girls at GoDaddy don't give the user much confidence in the a certificate issued by them, promising that the driver always works. Drivers for the 64-bit version of Windows have to be qualified by Microsoft, not them girls. Google "whql labs certifications".Hans Passant, who has 300,000+ reputation on StackOverflow, in response to my question

> A customized installation [generated by our software] does not contain certified drivers for Windows XP/2003/Vista/7. Certification must be performed by Microsoft for the new driver installation. An uncertified installation will not cause any other problems other than the warning message displayed by Windows XP/2003/Vista when installing uncertified drivers. Uncertified drivers cannot be installed in Windows 7 unless they are installed with a testing certificate or the Ignore Serial Signing option is enabled by pressing F8 on start up and selecting the corresponding option.Silicon Labs, makers of the CP2102 serial bridge, in AN220

이건 거짓입니다. 마이크로소프트가 하려는 것은 커널에 로드되는 모든 드라이버들을 완벽하게 통제 하려는 것이 아니라, 사용자들이 문제를 겪을 때 좀더 쉽게 그 문제를 파악하고자 하려는 것입니다..

그래서 비스타 64비트 때 부터는, 윈도우는 커널 드라이버를 로드하려면 시그니쳐를 요구합니다. 윈도우 8부터는 인스톨 할 때부터 이미 드라이버 패키지가 사인되어 있어야 합니다. 드라이버에 무한 루프 코드나 바이러스가 있을 수도 있으므로, 문제가 발생하면 그 소스를 추적할 수 있어야 합니다. 전체 설명을 보시려면 signing requirements 섹션을 보세요.

Silicon Labs와 관련해 위에서 인용한 부분은 좀 안타깝습니다. 만약 그쪽에서 제가 Loading a kernel module 섹션에서 설명한 세부 사항들을 알고 있었다면, Silicon Labs의 고객들이 윈도우 비스타나 윈도우 7에서 CP2102를 기반으로 한 장치들의 사인되지 않은 드라이버 패키지들을 어떻게 만드는 지 따라하기 자료를 쓸 수도 있었을 것입니다. Silicon Labs에서 나온 공식 사인된 드라이버 패키지는 사인되지 않은 드라이버 패키지들이 사용하는 silabser.sys를 위한 제대로 된 디지털 시그니쳐를 제공합니다.

거짓: DefaultInstall은 사인된 드라이버와는 동작하지 않습니다

> The INF file of a driver package must not contain an INF DefaultInstall section if the driver package is to be digitally signed.Microsoft, in the INF Default Install Section documentation

이 문서는 잘못되었습니다. 제가 2012년 11월부터 고객분들께 DefaultInstall 섹션이 포함된 사인된 드라이버들을 배포해오고 있는데, 문제가 없었습니다. 왜 이것이 문제가 되는지 이유를 모르겠습니다. 저는 늘 DefaultInstall 섹션을 드라이버 패키지 설치를 테스트할 때 사용해 오고 있습니다.

DefaultInstall 섹션은 사용자들이 Inf 파일을 마우스 오른쪽 클릭해서 "설치" 선택해서 INF 파일을 설치할 수 있도록 해 줍니다. 드라이버를 설치하는 방법에는 DPInst, SetupCopyOEMInf, PnPUtil 들이 있지만, DefaultInstall은 추가하기도 쉽고 또 잘 쓰고 있는 사용자들도 있으므로, 꼭 챙기도록 하세요.

예를 들어서, 만약 여러분의 드라이버의 이름이 `foo_driver.inf` 라면, 다음 라인을 추가하시면 됩니다:

```
[DefaultInstall]
CopyINF=foo_driver.inf
```

원하신다면 CopyINF 디렉티브에 여러개의 INF 파일들을 참조하게 할 수도 있습니다.

거짓: INF 버전 넘버는 OS 지원을 나타냅니다

> Create an INF file in your driver package directory and edit it for Windows Vista. Specifically, change the build date to 4/1/2006 or greater and the version to 6. For example: `DriverVer=04/01/2006, 6.0.1.0` Microsoft, in kmsigning.doc

kmsigning.doc는 대체로 훌륭한 문서지만, 저 라인은 틀렸습니다.

제 드라이버들을 2006년 이전으로 날짜를 지정 해본 적 없어서 문서에서 저 날짜에 대해서 말하는 내용이 맞는지는 모르겠으나, 저 버전 넘버 이야기는 사실이 아닙니다. 제가 윈도우 XP/비스타/7

을 사용하는 수천명의 고객들에게 드라이버를 성공적으로 배포해 왔는데, 우리 드라이버 버전들은 모두 1.0.0.0 to 3.0.0.0 사이에 있었습니다.

INF DriverVer 디렉티브는 MSDN 여기에 문서화 되어 있습니다. 만약 DriverVer 버전 넘버가 중요하다면, MS에서는 kmsigning.doc 문서의 11페이지에 파묻혀 있는게 아니라 저 페이지에 문서화 시켜 놓았을 것입니다. 사실, 저 페이지에 따르면 DriverVer 버전은 옵션 값입니다.

제 생각에 버전 넘버를 가장 적절하게 사용하는 예는 1.0.0.0부터 시작하는 것이고, 파일을 수정할 때마다 버전 넘버를 올리고 드라이버 날짜를 업데이트하는 것입니다.

반만 진실: 윈도우 7은 SHA-2를 지원하지 않는다

> In some cases, you might want to sign a driver package with two different signatures. For example, suppose you want your driver to run on Windows 7 and Windows 8. Windows 8 supports signatures created with the SHA256 hashing algorithm, but Windows 7 does not. For Windows 7, you need a signature created with the SHA1 hashing algorithm.Microsoft, from Signing a Driver for Public Release on MSDN

이것은 반만 맞는 말입니다. 제 경험상, 드라이버 패키지(CAT 파일들)에 대한 SHA-2 시그니쳐는 윈도우 비스타와 윈도우 7에서 드라이버 패키지 설치(driver package installation)를 목적으로 할 때에는 잘 동작합니다. 그러나, 커널에 드라이버를 로드하려는 목적(loading kernel modules)일 때에는 동작하지 않습니다. 만약 커널 드라이버를 포함한 드라이버 패키지를 사인하려는데 SHA-2를 사용한다면, 여러분의 장치를 꽂고 실제로 드라이버를 사용하려고 할 때 Code 52 에러가 뜰 것입니다. 윈도우 비스타/7의 커널이 파일을 로드할 수 있는지 체크하는 부분이 SHA-2 시그니쳐를 인식하지 못하는 것 같습니다. 만약 여러분의 드라이버 패키지가 새로운 커널 드라이버를 포함하고 있지 않다면(WinUSB나 usbser.sys를 사용한다면), SHA-2 시그니쳐는 잘 동작할 것입니다. 이에 대한 자세한 내용을 보려면, 위에 signature requirements 섹션을 보세요.

때로는 여러분들에게 아예 거짓말을 하는 것보다 반만 맞는 말을 하는 것이 더 안좋은 일일 수 있습니다. 제가 MSDN 문서에서 위에서 인용한 문단을 읽었을 때, 제 경험상 SHA-2 시그니쳐는 제 WinUSB와 usbser.sys 기반의 드라이버 패키지에서는 정상적으로 동작했기 때문에 완전히 틀렸다고 생각했었습니다. 저 문단이 맞는 커널도 있습니다만, 불행하게도 애매하게 틀렸기 때문에 사실로 받아들일 수 없었습니다. 다음부터 여러분이 문서를 쓰거나 누군가에게 어떤 내용을 설명할 때에는 꼭 명심하세요: 듣는 사람의 경험과 반대되는 말을 누군가에게 한다면, 당신이 말하는 걸 믿지 않을 겁니다.

반대로, 저에게 이렇게 말한 분도 있었습니다:

> Signing is perhaps the least suitable area to show off creativity and independent thinking. Just the opposite. Try to follow the instructions precisely. ... No matter what they scribble at Stack Overflow – the WDK documentations says the ultimate truth

> (when updated, of course).Pavel A., in response to my question on MSDN

글쎄요, 이 경우에는 Pavel이 맞았습니다. 제가 그냥 별 생각없이 무조건 문서 내용만을 따른다면, 저 위의 문단을 사실로 받아들일 수 있었을 것이고, 나중에 별로 고통스럽지도 않았겠죠

하지만 저는 생각없이 무조건 내용을 따를 수는 없습니다, 그러나 Pavel과 타협하는 것도 중요하죠. 우리는 공식 문서를 신중하게 받아들여야 하고, 우리 경험과 반대되는 말을 할 때에는 우리가 아직 테스트 하지 않은 어떤 환경에서는 저 문서가 맞을 가능성이 있다고도 생각해야 합니다. 그래서 저 위에 제가 거짓이라고 적어놓은 것은 *실제로는* 절반만 거짓이 될 수도 있는 것입니다.

참고문헌

1. KB2763674. Microsoft.
2. Digital Signatures for Kernel Modules on Windows (kmsigning.doc). Microsoft. 2007-07-25.
3. Kernel-Mode Code Signing Walkthrough (KMCS_walkthrough.doc). Microsoft. 2007-07-25.
4. Windows root certificate program members. Microsoft.
5. Cross-Certificates for Kernel Mode Code Signing. Microsoft.
6. Signtool.exe (Sign Tool). Microsoft.
7. Inf2Cat. Microsoft.
8. Automatic root certificate update problems when verifying my signed INF driver package. David Grayson.2012-10-03.
9. Signing Windows 8 Drivers. PiXCL Automation Technologies. 2013-03-09.
10. Microsoft Security Advisory (2880823). Microsoft. 2013-11-13.

Comments

여러분의 생각을 듣고 싶습니다! 저는 여기에 댓글을 운영하지 않으므로, 말할 내용이 있으시면 제가 만들어 놓은 MSDN thread 이곳에 글을 올려주세요. 이 문서가 유용하다고 생각하신다면 제 원본 글을 추천해 주시고 감사 인사도 남겨주시구요.

Revision History

- 2015-03-20: Added information about KB-3033929 in the note at the top.
- 2015-02-08: Added tip from Jimmy Kaz about avoiding spaces in INF file

names.
- 2015-01-14: Added links to PnPUtil.
- 2014-12-16: Added a note at the top about how SHA-1 is going away so parts of this document will need to be updated.
- 2014-09-12: Added a mention of KB2763674, which should make SHA-2 executables usable on Windows Vista.
- 2014-03-07: Added a mentions of Microsoft Security Advisory (2880823).
- 2013-10-02: Added section Use /tr to specify the timestamp server and changed all examples to use /tr.
- 2013-03-11: Added new info from PiXCL about how to double-sign with SHA1 and SHA2 to the Use SHA-1 for kernel modules prior to Windows 8 section.
- 2013-01-07: Initial release.