

Windows XP/2003 Leak Info



[noexit4237wjhu6v.onion](#) - [gitea server](#)

[yotsuba7pverpfs5.onion](#) - [nt5src.7z/toolset.zip/NTDEV videos](#)

Matrix Chat - <https://matrix.to/#/!gaxpoQqVZuAKwRucaK:privacytools.io>

Best:

[magnet:?xt=urn:btih:1a4e5b67060ff2bc8fe2de36a6c265c77f392a0c&dn=NOTREPACKED](#)

REPACKED (about 40GB of useless shit):

[magnet:?xt=urn:btih:cec41d87d329822ea9b05c4f6c0dff7d29514d59&dn=Microsoft%20leaked%20source%20code%20archive%5F2020-09-20](#)

Direct DL:

<http://yotsuba7pverpfs5.onion/files/nt5src.7z>

Contents

1. [Windows XP/2003 Leak Info](#)
 1. [Best:](#)
 2. [REPACKED \(about 40GB of useless shit\):](#)
 3. [Direct DL:](#)
 4. [Contents](#)
2. [Windows Server 2003 \(NT 5.2 / 3790\) guide](#)
 1. [Preparing Environment](#)
 2. [Building](#)
 1. [Clean build](#)
 2. ["Dirty" build](#)
 3. [Getting ISO files](#)
 4. [Patches](#)
 5. [Additions](#)
 1. [Why build flags are necessary](#)
 2. [Removing timebomb](#)
 3. [Different build options](#)
 4. [Creating fresh build](#)
 5. [Original CD filenames](#)
 6. [Product keys](#)
 7. [64-bit specific](#)
 1. [list of tools that must be recompiled in order to build on x64](#)
3. [TODO:](#)

Windows Server 2003 (NT 5.2 / 3790) guide

Preparing Environment

- Extract source tree, in this guide we will assume `C:\NT` (if you wish for your binaries to match RTM as closely as possible, use `D:\srv03rtm`)
- Unset Read Only on extracted directory (including subfolders and files)
- Create desktop shortcut for `%windir%\system32\cmd.exe /k C:\NT\tools\razzle.cmd free offline` (see below for explanation) and change **Start in** to `C:\NT` (**do not open it yet!**)
- Acquire new certificates and overwrite old files with newly extracted ones (archive also contains private keys in `.der` format if you wish to renew them at any point)
- Manually import `\tools\driver.pfx` certificate and private key (if asked for password, simply press Next without typing anything)
- Apply patches (available below) to NT tree
- Copy these files/directories from XP source tree and copy them into 2K3 tree:
 - `\public\internal\windows\lib\i386\directui.lib`
 - `\public\internal\windows\lib\i386\conlibk.lib`
- Acquire these files/directories and copy them into source tree:
 - `\base\ntos\ex\exinit.c`
 - `\base\ntos\ex\systemtime.c`
 - `\base\ntos\ex\exp.h`
 - `\tools\missing.cmd`
 - `\tools\oscdimg.cmd`
- Copy these files/directories and store them safely outside source tree:
 - `\ds\security\services\ca\tools\certut\obj\`
 - `\inetcore\outlookexpress\external\obj{pd}\`
- Now you can open new razzle window by using your shortcut

Building

You are now ready to attempt first build. First, decide on how you would like to proceed.

Clean build

- `perl tools\timebuild.pl -NOPOSTBUILD -NOSCORCH`
- Copy back the files you've stored outside of source tree in preparation step
- Run `build /ZP` again and ensure `build.err` is empty

"Dirty" build

- `nmake set_builddate set_buildnum set_buildname -f makefil0`
- `perl tools\timebuild.pl -NOCLEANBUILD -NOPOSTBUILD -NOSCORCH`
- Run `build /ZP` again and ensure `build.err` is empty

Rest of instructions are the same for both types:

- Mount RTM CD and execute `tools\missing.cmd` (optionally, perform this step for every SKU)
- Edit `\tools\postbuildscripts\pbuild.dat` and comment out (;) the `sxs_make_asms_cabs` lines
- `tools\postbuild.cmd` (use `-sku:{sku}` if you want to process only specific one, expect errors if you ignore this and have skipped on `missing.cmd` optional step)

Getting ISO files

- Execute `tools\oscdimg.cmd {sku}` where `{sku}` is one of:
 - `srv` - Windows Server 2003 Standard Edition
 - `sbs` - Windows Server 2003 Small Business Edition
 - `ads` - Windows Server 2003 Enterprise Edition
 - `dtc` - Windows Server 2003 Datacenter Edition
 - `bla` - Windows Server 2003 Web Edition

- **per** - Windows XP Home Edition
- **pro** - Windows XP Professional

Patches

Following patches should be applied before building:

```
--- b/windows/core/cslpk/lpk/sources.inc
+++ a/windows/core/cslpk/lpk/sources.inc
@@ -58,6 +58,9 @@

    C_DEFINES=$(C_DEFINES) $(USER_C_DEFINES) -D_USER32_ -DUNICODE -DUSE_MIRRORING -DLANGPACK

+### usp debugging interface missing
+C_DEFINES=$(C_DEFINES) -DDBG=0
+
    DLLDEF=..\lpk.def

    SOURCES= ..\lpk.rc          \

--- a/tools/checktestroot.cmd
+++ b/checktestroot.cmd
@@ -3,7 +3,7 @@

    echo Verifying that the Testroot Certificate is installed...
    set __certinstalled=
    -for /f %%i in ('tfindcer -a"Microsoft Test Root Authority" -s root -S ^| findstr /c:"2BD63D28 D7BCD0E2 51195AEB 519243C1
3142EBC3"') do (
    +for /f %%i in ('tfindcer -a"Microsoft Test Root Authority" -s root -S ^| findstr /c:"A4CAECFC 40A44BB7 3E3BBF69 477BC68D
07B0C7AB"') do (
        set __certinstalled=1
    )

@@ -15,7 +15,7 @@

    echo Check again to see if Testroot is installed...
    set __certinstalled=
    -for /f %%i in ('tfindcer -a"Microsoft Test Root Authority" -s root -S ^| findstr /c:"2BD63D28 D7BCD0E2 51195AEB 519243C1
3142EBC3"') do (
    +for /f %%i in ('tfindcer -a"Microsoft Test Root Authority" -s root -S ^| findstr /c:"A4CAECFC 40A44BB7 3E3BBF69 477BC68D
07B0C7AB"') do (
        set __certinstalled=1
    )

--- a/tools/checktestpca.cmd
+++ b/tools/checktestpca.cmd
@@ -3,7 +3,7 @@

    echo Verifying that the Test PCA Certificate is installed...
    set __certinstalled=
    -for /f %%i in ('tfindcer -a"Microsoft Test PCA" -s ca -S ^| findstr /c:"B1705970 38A12D9B 4E3CFA46 A6BD2454 9BB432D6"')
do (
    +for /f %%i in ('tfindcer -a"Microsoft Test PCA" -s ca -S ^| findstr /c:"52871BBC 6CAAEF1F EB45D478 DDC7517C 06D7A08D"')
do (
        set __certinstalled=1
    )

@@ -15,7 +15,7 @@

    echo Check again to see if Test PCA Certificate is installed...
    set __certinstalled=
    -for /f %%i in ('tfindcer -a"Microsoft Test PCA" -s ca -S ^| findstr /c:"B1705970 38A12D9B 4E3CFA46 A6BD2454 9BB432D6"')
do (
    +for /f %%i in ('tfindcer -a"Microsoft Test PCA" -s ca -S ^| findstr /c:"52871BBC 6CAAEF1F EB45D478 DDC7517C 06D7A08D"')
do (
        set __certinstalled=1
    )

--- a/base/ntsetup/syssetup/crypto.c
+++ b/base/ntsetup/syssetup/crypto.c
@@ -544,8 +544,7 @@
    // test root(s) sha1 hash
    //
    DWORD i;
    -    BYTE arHashData[2][20] = { {0x30, 0x0B, 0x97, 0x1A, 0x74, 0xF9, 0x7E, 0x09, 0x8B, 0x67, 0xA4, 0xFC, 0xEB,
0xBB, 0xF6, 0xB9, 0xAE, 0x2F, 0x40, 0x4C}, // old beta testroot.cer (used until just prior to RC3)
    -    {0x2B, 0xD6, 0x3D, 0x28, 0xD7, 0xBC, 0xD0, 0xE2, 0x51, 0x19, 0x5A, 0xEB, 0x51, 0x92,
0x43, 0xC1, 0x31, 0x42, 0xEB, 0xC3} }; // current beta testroot.cer (also used for OEM testsigning)
    +    BYTE arHashData[1][20] = { {0xA4, 0xCA, 0xEC, 0xFC, 0x40, 0xA4, 0x4B, 0xB7, 0x3E, 0x3B, 0xBF, 0x69, 0x47,
0x7B, 0xC6, 0x8D, 0x07, 0xB0, 0xC7, 0xAB} }; // current beta testroot.cer (also used for OEM testsigning)
    CRYPT_HASH_BLOB hash;

    hash.cbData = sizeof(arHashData[0]);
```

```

--- a/base/win32/fusion/sxs/strongname.cpp
+++ b/base/win32/fusion/sxs/strongname.cpp
@@ -436,7 +436,7 @@
    PCCERT_CONTEXT    pCertContext = NULL;

    BYTE bTestRootHashList[][20] = {
-        {0x2B, 0xD6, 0x3D, 0x28, 0xD7, 0xBC, 0xD0, 0xE2, 0x51, 0x19, 0x5A, 0xEB, 0x51, 0x92, 0x43, 0xC1, 0x31, 0x42, 0xEB,
0xC3}
+        {0xA4, 0xCA, 0xEC, 0xFC, 0x40, 0xA4, 0x4B, 0xB7, 0x3E, 0x3B, 0xBF, 0x69, 0x47, 0x7B, 0xC6, 0x8D, 0x07, 0xB0, 0xC7,
0xAB}
    };

    bAllowed = FALSE;

--- a/tools/postbuildscripts/crypto.cmd
+++ b/tools/postbuildscripts/crypto.cmd
@@ -50,7 +50,7 @@
)

set CERT_FILE=%RazzleToolPath%\driver.pfx
-set CSP_SIGN_CMD=signtool sign /sha1 83E06454938FC9248845CB2C4E4E73CF8CCC6E65
+set CSP_SIGN_CMD=signtool sign /sha1 5B8962DC21A68507196A158F8F687F232706CDBC

REM MS Software CSPs

--- a/shell/shell32/defview.cpp
+++ b/shell/shell32/defview.cpp
@@ -442,7 +442,7 @@
#define REGSTR_PATH_GPO_ROOTCERTIFICATES \
    TEXT("SOFTWARE\\Policies\\Microsoft\\SystemCertificates\\Root\\Certificates")
#define REGSTR_KEY_TESTCERTIFICATE \
-    TEXT("2BD63D28D7BCD0E251195AEB519243C13142EBC3")
+    TEXT("A4CAECFC40A44BB73E3BBF69477BC68D07B0C7AB")

    dwPaintVersion = (0 != USER_SHARED_DATA->SystemExpirationDate.QuadPart) ||
        SHRegSubKeyExists(HKEY_LOCAL_MACHINE, REGSTR_PATH_LM_ROOTCERTIFICATES TEXT("\\")
REGSTR_KEY_TESTCERTIFICATE) ||

--- a/ds/win32/ntcrypto/mincrypt/vercert.cpp
+++ b/ds/win32/ntcrypto/mincrypt/vercert.cpp
@@ -234,55 +234,46 @@
// Test Roots
//-----

-// Name:: <CN=Microsoft Test Root Authority, OU=Microsoft Corporation, OU=Copyright (c) 1999 Microsoft Corp.>
+// Name:: <CN=Microsoft Test Root Authority, OU=Copyright (c) 1999 Microsoft Corp., OU=Microsoft Corporation>
const BYTE rgbTestRoot0_Name[] = {
-    0x30, 0x75, 0x31, 0x2B, 0x30, 0x29, 0x06, 0x03,
-    0x55, 0x04, 0x0B, 0x13, 0x22, 0x43, 0x6F, 0x70,
-    0x79, 0x72, 0x69, 0x67, 0x68, 0x74, 0x20, 0x28,
-    0x63, 0x29, 0x20, 0x31, 0x39, 0x39, 0x39, 0x20,
-    0x4D, 0x69, 0x63, 0x72, 0x6F, 0x73, 0x6F, 0x66,
-    0x74, 0x20, 0x43, 0x6F, 0x72, 0x70, 0x2E, 0x31,
-    0x1E, 0x30, 0x1C, 0x06, 0x03, 0x55, 0x04, 0x0B,
-    0x13, 0x15, 0x4D, 0x69, 0x63, 0x72, 0x6F, 0x73,
-    0x6F, 0x66, 0x74, 0x20, 0x43, 0x6F, 0x72, 0x70,
-    0x6F, 0x72, 0x61, 0x74, 0x69, 0x6F, 0x6E, 0x31,
-    0x26, 0x30, 0x24, 0x06, 0x03, 0x55, 0x04, 0x03,
-    0x13, 0x1D, 0x4D, 0x69, 0x63, 0x72, 0x6F, 0x73,
-    0x6F, 0x66, 0x74, 0x20, 0x54, 0x65, 0x73, 0x74,
-    0x20, 0x52, 0x6F, 0x6F, 0x74, 0x20, 0x41, 0x75,
-    0x74, 0x68, 0x6F, 0x72, 0x69, 0x74, 0x79
+    0x30, 0x75, 0x31, 0x2B, 0x30, 0x29, 0x06, 0x03, 0x55, 0x04,
+    0x0B, 0x13, 0x22, 0x43, 0x6F, 0x70, 0x79, 0x72, 0x69, 0x67,
+    0x68, 0x74, 0x20, 0x28, 0x63, 0x29, 0x20, 0x31, 0x39, 0x39,
+    0x39, 0x20, 0x4D, 0x69, 0x63, 0x72, 0x6F, 0x73, 0x6F, 0x66,
+    0x74, 0x20, 0x43, 0x6F, 0x72, 0x70, 0x2E, 0x31, 0x26, 0x30,
+    0x24, 0x06, 0x03, 0x55, 0x04, 0x03, 0x13, 0x1D, 0x4D, 0x69,
+    0x63, 0x72, 0x6F, 0x73, 0x6F, 0x66, 0x74, 0x20, 0x54, 0x65,
+    0x73, 0x74, 0x20, 0x52, 0x6F, 0x6F, 0x74, 0x20, 0x41, 0x75,
+    0x74, 0x68, 0x6F, 0x72, 0x69, 0x74, 0x79, 0x31, 0x1E, 0x30,
+    0x1C, 0x06, 0x03, 0x55, 0x04, 0x0B, 0x13, 0x15, 0x4D, 0x69,
+    0x63, 0x72, 0x6F, 0x73, 0x6F, 0x66, 0x74, 0x20, 0x43, 0x6F,
+    0x72, 0x70, 0x6F, 0x72, 0x61, 0x74, 0x69, 0x6F, 0x6E
};

const BYTE rgbTestRoot0_PubKeyInfo[] = {
-    0x30, 0x81, 0xDF, 0x30, 0x0D, 0x06, 0x09, 0x2A,
-    0x86, 0x48, 0x86, 0xF7, 0x0D, 0x01, 0x01, 0x01,
-    0x05, 0x00, 0x03, 0x81, 0xCD, 0x00, 0x30, 0x81,
-    0xC9, 0x02, 0x81, 0xC1, 0x00, 0xA9, 0xAA, 0x83,
-    0x58, 0x6D, 0xB5, 0xD3, 0x0C, 0x4B, 0x5B, 0x80,
-    0x90, 0xE5, 0xC3, 0x0F, 0x28, 0x0C, 0x7E, 0x3D,
-    0x3C, 0x24, 0xC5, 0x29, 0x56, 0x63, 0x8C, 0xEE,
-    0xC7, 0x83, 0x4A, 0xD8, 0x8C, 0x25, 0xD3, 0x0E,
-    0xD3, 0x12, 0xB7, 0xE1, 0x86, 0x72, 0x74, 0xA7,
-    0x8B, 0xFB, 0x0F, 0x05, 0xE9, 0x65, 0xC1, 0x9B,

```



```
- 0xD8, 0x56, 0xC2, 0x93, 0xF0, 0xFB, 0xE9, 0x5A,
- 0x48, 0x85, 0x7D, 0x95, 0xAA, 0xDF, 0x01, 0x86,
- 0xB7, 0x33, 0x33, 0x46, 0x56, 0xCB, 0x5B, 0x7A,
- 0xC4, 0xAF, 0xA0, 0x96, 0x53, 0x3A, 0xE9, 0xFB,
- 0x3B, 0x78, 0xC1, 0x43, 0x0C, 0xC7, 0x6E, 0x1C,
- 0x2F, 0xD1, 0x55, 0xF1, 0x19, 0xB2, 0x3F, 0xF8,
- 0xD6, 0xA0, 0xC7, 0x24, 0x95, 0x3B, 0xC8, 0x45,
- 0x25, 0x6F, 0x45, 0x3A, 0x46, 0x4F, 0xD2, 0x27,
- 0x8B, 0xC7, 0x50, 0x75, 0xC6, 0x80, 0x5E, 0x0D,
- 0x99, 0x78, 0x61, 0x77, 0x39, 0xC1, 0xB3, 0x0F,
- 0x9D, 0x12, 0x9C, 0xC4, 0xBB, 0x32, 0x7B, 0xB2,
- 0x4B, 0x26, 0xAA, 0x4E, 0xC0, 0x32, 0xB0, 0x2A,
- 0x13, 0x21, 0xBE, 0xED, 0x24, 0xF4, 0x7D, 0x0D,
- 0xEA, 0xAA, 0x8A, 0x7A, 0xD2, 0x8B, 0x4D, 0x97,
- 0xB5, 0x4D, 0x64, 0xBA, 0xFB, 0x46, 0xDD, 0x69,
- 0x6F, 0x9A, 0x0E, 0xCC, 0x53, 0x77, 0xAA, 0x6E,
- 0xAE, 0x20, 0xD6, 0x21, 0x98, 0x69, 0xD9, 0x46,
- 0xB9, 0x64, 0x32, 0xD4, 0x17, 0x02, 0x03, 0x01,
- 0x00, 0x01
+ 0x30, 0x81, 0xDF, 0x30, 0x0D, 0x06, 0x09, 0x2A, 0x86, 0x48,
+ 0x86, 0xF7, 0x0D, 0x01, 0x01, 0x01, 0x05, 0x00, 0x03, 0x81,
+ 0xCD, 0x00, 0x30, 0x81, 0xC9, 0x02, 0x81, 0xC1, 0x00, 0xA9,
+ 0x6B, 0xB7, 0xAE, 0x1E, 0x27, 0x36, 0xED, 0x90, 0xD5, 0x7B,
+ 0xE2, 0x59, 0xBB, 0x7F, 0x77, 0x23, 0x57, 0x16, 0x8F, 0x6E,
+ 0x5D, 0x93, 0x9F, 0x87, 0xAE, 0x89, 0x23, 0x4A, 0x7C, 0x9B,
+ 0xA8, 0xB2, 0x6F, 0x33, 0x62, 0x81, 0x5B, 0x64, 0xBC, 0x2D,
+ 0x8B, 0xC1, 0xB2, 0x19, 0x26, 0x76, 0x68, 0x6E, 0x82, 0x2A,
+ 0x18, 0x0D, 0x95, 0xFE, 0x00, 0x3E, 0xEB, 0x86, 0xB6, 0xE3,
+ 0x13, 0x5E, 0xB8, 0x63, 0x2A, 0x79, 0x0F, 0x97, 0x94, 0xF7,
+ 0x34, 0x87, 0xD1, 0x6A, 0xDE, 0x0A, 0xFB, 0x5D, 0x1B, 0xA8,
+ 0x31, 0xE1, 0x66, 0x5E, 0x7A, 0x07, 0x3C, 0x1E, 0x13, 0xBC,
+ 0xFE, 0x44, 0xCF, 0x22, 0x06, 0xC3, 0xA0, 0x8A, 0xA1, 0x76,
+ 0x46, 0xDB, 0x53, 0xBC, 0x7D, 0xD2, 0xFD, 0x4B, 0xE1, 0x0C,
+ 0x4F, 0x41, 0x33, 0x55, 0xA6, 0xCD, 0xD5, 0x60, 0xCD, 0xCE,
+ 0xA8, 0x5A, 0x00, 0xE7, 0xB7, 0x0A, 0xAE, 0x75, 0xD8, 0x8E,
+ 0xD4, 0x71, 0xCF, 0xE7, 0xEE, 0x4B, 0xC5, 0x89, 0x85, 0x0B,
+ 0x38, 0xB1, 0x4C, 0x6C, 0x4E, 0xBE, 0x63, 0xA6, 0x5D, 0x40,
+ 0x2E, 0x1C, 0xFF, 0x80, 0x9A, 0x10, 0xCE, 0x82, 0x41, 0x79,
+ 0x39, 0xAD, 0xAD, 0xE2, 0x97, 0xE1, 0xAE, 0x1B, 0x20, 0x47,
+ 0x3D, 0x1C, 0xBF, 0xCD, 0x25, 0xB1, 0x73, 0x49, 0x16, 0xEE,
+ 0xCD, 0xBB, 0xC9, 0x8F, 0xEC, 0x57, 0x6E, 0x24, 0xF8, 0xCF,
+ 0xE5, 0x02, 0x03, 0x01, 0x00, 0x01
};
```

```
@@ -316,7 +307,7 @@
#define wszSYSTEM_SETUP_REG_VALUE \
    L"SystemSetupInProgress"
#define wszTEST_ROOT_REG \
-    L"SOFTWARE\\Microsoft\\SystemCertificates\\Root\\Certificates\\2BD63D28D7BCD0E251195AEB519243C13142EBC3"
+    L"SOFTWARE\\Microsoft\\SystemCertificates\\Root\\Certificates\\A4CAECFC40A44BB73E3BBF69477BC68D07B0C7AB"

// Check the CERT_STORE_PROV_SYSTEM_REGISTRY store for the Test Root.
//
```

```
--- a/security/cryptoapi/mincrypt/lib/vercert.cpp
+++ b/security/cryptoapi/mincrypt/lib/vercert.cpp
@@ -239,55 +239,46 @@
// Test Roots
//-----

-// Name:: <CN=Microsoft Test Root Authority, OU=Microsoft Corporation, OU=Copyright (c) 1999 Microsoft Corp.>
+// Name:: <CN=Microsoft Test Root Authority, OU=Copyright (c) 1999 Microsoft Corp., OU=Microsoft Corporation>
const BYTE rgbTestRoot0_Name[] = {
- 0x30, 0x75, 0x31, 0x2B, 0x30, 0x29, 0x06, 0x03,
- 0x55, 0x04, 0x0B, 0x13, 0x22, 0x43, 0x6F, 0x70,
- 0x79, 0x72, 0x69, 0x67, 0x68, 0x74, 0x20, 0x28,
- 0x63, 0x29, 0x20, 0x31, 0x39, 0x39, 0x39, 0x20,
- 0x4D, 0x69, 0x63, 0x72, 0x6F, 0x73, 0x6F, 0x66,
- 0x74, 0x20, 0x43, 0x6F, 0x72, 0x70, 0x2E, 0x31,
- 0x1E, 0x30, 0x1C, 0x06, 0x03, 0x55, 0x04, 0x0B,
- 0x13, 0x15, 0x4D, 0x69, 0x63, 0x72, 0x6F, 0x73,
- 0x6F, 0x66, 0x74, 0x20, 0x43, 0x6F, 0x72, 0x70,
- 0x6F, 0x72, 0x61, 0x74, 0x69, 0x6F, 0x6E, 0x31,
- 0x26, 0x30, 0x24, 0x06, 0x03, 0x55, 0x04, 0x03,
- 0x13, 0x1D, 0x4D, 0x69, 0x63, 0x72, 0x6F, 0x73,
- 0x6F, 0x66, 0x74, 0x20, 0x54, 0x65, 0x73, 0x74,
- 0x20, 0x52, 0x6F, 0x6F, 0x74, 0x20, 0x41, 0x75,
- 0x74, 0x68, 0x6F, 0x72, 0x69, 0x74, 0x79
+ 0x30, 0x75, 0x31, 0x2B, 0x30, 0x29, 0x06, 0x03, 0x55, 0x04,
+ 0x0B, 0x13, 0x22, 0x43, 0x6F, 0x70, 0x79, 0x72, 0x69, 0x67,
+ 0x68, 0x74, 0x20, 0x28, 0x63, 0x29, 0x20, 0x31, 0x39, 0x39,
+ 0x39, 0x20, 0x4D, 0x69, 0x63, 0x72, 0x6F, 0x73, 0x6F, 0x66,
+ 0x74, 0x20, 0x43, 0x6F, 0x72, 0x70, 0x2E, 0x31, 0x26, 0x30,
+ 0x24, 0x06, 0x03, 0x55, 0x04, 0x03, 0x13, 0x1D, 0x4D, 0x69,
+ 0x63, 0x72, 0x6F, 0x73, 0x6F, 0x66, 0x74, 0x20, 0x54, 0x65,
+ 0x73, 0x74, 0x20, 0x52, 0x6F, 0x6F, 0x74, 0x20, 0x41, 0x75,
+ 0x74, 0x68, 0x6F, 0x72, 0x69, 0x74, 0x79, 0x31, 0x1E, 0x30,
+ 0x1C, 0x06, 0x03, 0x55, 0x04, 0x0B, 0x13, 0x15, 0x4D, 0x69,
```

```

+ 0x63, 0x72, 0x6F, 0x73, 0x6F, 0x66, 0x74, 0x20, 0x43, 0x6F,
+ 0x72, 0x70, 0x6F, 0x72, 0x61, 0x74, 0x69, 0x6F, 0x6E
};

const BYTE rgbTestRoot0_PubKeyInfo[] = {
- 0x30, 0x81, 0xDF, 0x30, 0x0D, 0x06, 0x09, 0x2A,
- 0x86, 0x48, 0x86, 0xF7, 0x0D, 0x01, 0x01, 0x01,
- 0x05, 0x00, 0x03, 0x81, 0xCD, 0x00, 0x30, 0x81,
- 0xC9, 0x02, 0x81, 0xC1, 0x00, 0xA9, 0xAA, 0x83,
- 0x58, 0x6D, 0xB5, 0xD3, 0x0C, 0x4B, 0x5B, 0x80,
- 0x90, 0xE5, 0xC3, 0x0F, 0x28, 0x0C, 0x7E, 0x3D,
- 0x3C, 0x24, 0xC5, 0x29, 0x56, 0x63, 0x8C, 0xEE,
- 0xC7, 0x83, 0x4A, 0xD8, 0x8C, 0x25, 0xD3, 0x0E,
- 0xD3, 0x12, 0xB7, 0xE1, 0x86, 0x72, 0x74, 0xA7,
- 0x8B, 0xFB, 0x0F, 0x05, 0xE9, 0x65, 0xC1, 0x9B,
- 0xD8, 0x56, 0xC2, 0x93, 0xF0, 0xFB, 0xE9, 0x5A,
- 0x48, 0x85, 0x7D, 0x95, 0xAA, 0xDF, 0x01, 0x86,
- 0xB7, 0x33, 0x33, 0x46, 0x56, 0xCB, 0x5B, 0x7A,
- 0xC4, 0xAF, 0xA0, 0x96, 0x53, 0x3A, 0xE9, 0xFB,
- 0x3B, 0x78, 0xC1, 0x43, 0x0C, 0xC7, 0x6E, 0x1C,
- 0x2F, 0xD1, 0x55, 0xF1, 0x19, 0xB2, 0x3F, 0xF8,
- 0xD6, 0xA0, 0xC7, 0x24, 0x95, 0x3B, 0xC8, 0x45,
- 0x25, 0x6F, 0x45, 0x3A, 0x46, 0x4F, 0xD2, 0x27,
- 0x8B, 0xC7, 0x50, 0x75, 0xC6, 0x80, 0x5E, 0x0D,
- 0x99, 0x78, 0x61, 0x77, 0x39, 0xC1, 0xB3, 0x0F,
- 0x9D, 0x12, 0x9C, 0xC4, 0xBB, 0x32, 0x7B, 0xB2,
- 0x4B, 0x26, 0xAA, 0x4E, 0xC0, 0x32, 0xB0, 0x2A,
- 0x13, 0x21, 0xBE, 0xED, 0x24, 0xF4, 0x7D, 0x0D,
- 0xEA, 0xAA, 0x8A, 0x7A, 0xD2, 0x8B, 0x4D, 0x97,
- 0xB5, 0x4D, 0x64, 0xBA, 0xFB, 0x46, 0xDD, 0x69,
- 0x6F, 0x9A, 0x0E, 0xCC, 0x53, 0x77, 0xAA, 0x6E,
- 0xAE, 0x20, 0xD6, 0x21, 0x98, 0x69, 0xD9, 0x46,
- 0xB9, 0x64, 0x32, 0xD4, 0x17, 0x02, 0x03, 0x01,
- 0x00, 0x01
+ 0x30, 0x81, 0xDF, 0x30, 0x0D, 0x06, 0x09, 0x2A, 0x86, 0x48,
+ 0x86, 0xF7, 0x0D, 0x01, 0x01, 0x01, 0x05, 0x00, 0x03, 0x81,
+ 0xCD, 0x00, 0x30, 0x81, 0xC9, 0x02, 0x81, 0xC1, 0x00, 0xA9,
+ 0x6B, 0xB7, 0xAE, 0x1E, 0x27, 0x36, 0xED, 0x90, 0xD5, 0x7B,
+ 0xE2, 0x59, 0xBB, 0x7F, 0x77, 0x23, 0x57, 0x16, 0x8F, 0x6E,
+ 0x5D, 0x93, 0x9F, 0x87, 0xAE, 0x89, 0x23, 0x4A, 0x7C, 0x9B,
+ 0xA8, 0xB2, 0x6F, 0x33, 0x62, 0x81, 0x5B, 0x64, 0xBC, 0x2D,
+ 0x8B, 0xC1, 0xB2, 0x19, 0x26, 0x76, 0x68, 0x6E, 0x82, 0x2A,
+ 0x18, 0x0D, 0x95, 0xFE, 0x00, 0x3E, 0xEB, 0x86, 0xB6, 0xE3,
+ 0x13, 0x5E, 0xB8, 0x63, 0x2A, 0x79, 0x0F, 0x97, 0x94, 0xF7,
+ 0x34, 0x87, 0xD1, 0x6A, 0xDE, 0x0A, 0xFB, 0x5D, 0x1B, 0xA8,
+ 0x31, 0xE1, 0x66, 0x5E, 0x7A, 0x07, 0x3C, 0x1E, 0x13, 0xBC,
+ 0xFE, 0x44, 0xCF, 0x22, 0x06, 0xC3, 0xA0, 0x8A, 0xA1, 0x76,
+ 0x46, 0xDB, 0x53, 0xBC, 0x7D, 0xD2, 0xFD, 0x4B, 0xE1, 0x0C,
+ 0x4F, 0x41, 0x33, 0x55, 0xA6, 0xCD, 0xD5, 0x60, 0xCD, 0xCE,
+ 0xA8, 0x5A, 0x00, 0xE7, 0xB7, 0x0A, 0xAE, 0x75, 0xD8, 0x8E,
+ 0xD4, 0x71, 0xCF, 0xE7, 0xEE, 0x4B, 0xC5, 0x89, 0x85, 0x0B,
+ 0x38, 0xB1, 0x4C, 0x6C, 0x4E, 0xBE, 0x63, 0xA6, 0x5D, 0x40,
+ 0x2E, 0x1C, 0xFF, 0x80, 0x9A, 0x10, 0xCE, 0x82, 0x41, 0x79,
+ 0x39, 0xAD, 0xAD, 0xE2, 0x97, 0xE1, 0xAE, 0x1B, 0x20, 0x47,
+ 0x3D, 0x1C, 0xBF, 0xCD, 0x25, 0xB1, 0x73, 0x49, 0x16, 0xEE,
+ 0xCD, 0xBB, 0xC9, 0x8F, 0xEC, 0x57, 0x6E, 0x24, 0xF8, 0xCF,
+ 0xE5, 0x02, 0x03, 0x01, 0x00, 0x01
};

//+=====

--- a/windows/core/ntuser/kernel/server.c
+++ b/windows/core/ntuser/kernel/server.c
@@ -2787,12 +2787,11 @@
/*
 * Determine if we have unsigned drivers installed
- * Use 2BD63D28D7BCD0E251195AEB519243C13142EBC3 as current key to check.
- * Old key: 300B971A74F97E098B67A4FCEBBBF6B9AE2F404C
+ * Use A4CAECFC40A44BB73E3BBF69477BC68D07B0C7AB as current key to check.
 */
- if (NT_SUCCESS(RtlCheckRegistryKey(RTL_REGISTRY_ABSOLUTE,
L"\\Registry\\Machine\\SOFTWARE\\Policies\\Microsoft\\SystemCertificates\\Root\\Certificates\\2BD63D28D7BCD0E251195AEB519243C13142EBC3"))) {
- NT_SUCCESS(RtlCheckRegistryKey(RTL_REGISTRY_ABSOLUTE,
L"\\Registry\\Machine\\SOFTWARE\\Microsoft\\SystemCertificates\\Root\\Certificates\\2BD63D28D7BCD0E251195AEB519243C13142EBC3"))
- NT_SUCCESS(RtlCheckRegistryKey(RTL_REGISTRY_USER,
L"\\SOFTWARE\\Microsoft\\SystemCertificates\\Root\\Certificates\\2BD63D28D7BCD0E251195AEB519243C13142EBC3")))) {
+ if (NT_SUCCESS(RtlCheckRegistryKey(RTL_REGISTRY_ABSOLUTE,
L"\\Registry\\Machine\\SOFTWARE\\Policies\\Microsoft\\SystemCertificates\\Root\\Certificates\\A4CAECFC40A44BB73E3BBF69477BC68D07B0C7AB"))) {
+ NT_SUCCESS(RtlCheckRegistryKey(RTL_REGISTRY_ABSOLUTE,
L"\\Registry\\Machine\\SOFTWARE\\Microsoft\\SystemCertificates\\Root\\Certificates\\A4CAECFC40A44BB73E3BBF69477BC68D07B0C7AB")))
+ NT_SUCCESS(RtlCheckRegistryKey(RTL_REGISTRY_USER,
L"\\SOFTWARE\\Microsoft\\SystemCertificates\\Root\\Certificates\\A4CAECFC40A44BB73E3BBF69477BC68D07B0C7AB")))) {
    gfUnsignedDrivers = TRUE;
}

```

```
--- a/ds/security/cryptoapi/pki/certstor/policy.cpp
+++ b/ds/security/cryptoapi/pki/certstor/policy.cpp
@@ -1848,11 +1848,11 @@
     //      OU=Microsoft Corporation
     //      OU=Copyright (c) 1999 Microsoft Corp.
     //
-    //  NotBefore:: Sat Jan 09 23:00:00 1999
-    //  NotAfter::  Wed Dec 30 23:00:00 2020
+    //  NotBefore:: 10.01.1999 08:00:00 CET
+    //  NotAfter::  31.12.2020 08:00:00 CET
     {
-        0x22, 0xCD, 0x37, 0xF1, 0xB1, 0x47, 0x50, 0xAE, 0x53, 0x7C,
-        0x8C, 0x6A, 0x03, 0x67, 0x47, 0xE2, 0xB7, 0x1E, 0x17, 0xB7
+        0x8E, 0xFF, 0xD8, 0x6B, 0xC3, 0x8B, 0xC1, 0x95, 0xFA, 0xE7,
+        0xA3, 0x55, 0x1E, 0xE6, 0x37, 0x64, 0x8D, 0xAC, 0xDC, 0x53
     },
};
```

Additions

Why build flags are necessary

Certain parts of code are missing but present in pre-built form, using `-NOCLEANBUILD` and `-NOSCORCH` ensures that they are not removed from the tree

Removing timebomb

Comment out `timebomb` lines from `tools\postbuildscripts\pbbuild.dat` before running `postbuild` for the first time (currently there is no guide to revert it)

Different build options

You can modify your razzle shortcut (or execute command manually inside `C:\NT`) to include (or remove) additional argument(s):

- `free` - build 'free' bits (production, ommiting it will generated checked bits)
- `CkhKernel` - build 'checked' (testing) kernel/hal/ntdll when building 'free' bits
- `no_opts` - disable binary optimization (useful for debugging, but will most likely fail a full build, some code can't be built without optimization)
- `verbose` - enable verbose execution of the build process
- `binaries_dir <basepath>` - specifies custom output directory (default is `binaries`, the suffix added after `.` is non-customizable)

Other options are not described here, see `razzle.cmd /?` for details.

Creating fresh build

You can use `tools\postbuild.cmd -full` to create fresh package, however this has not been properly tested yet, expect the unexpected!

Original CD filenames

- `5.2.3790.0.srv03_rtm.030324-2048_x86fre_server-standard_retail_en-us-NRMSFPP_EN.iso` (SHA1: A600409482A5678EF6AF2B26D3576D6D9894178D)
- `5.2.3790.0.srv03_rtm.030324-2048_x86fre_server-datacenter_retail_en-us-NRMDOEM_EN.iso` (SHA1: E2B47A7CE45C6C6305594CEE4C1B64894805AAF4)
- `5.2.3790.0.srv03_rtm.030324-2048_x86fre_server-enterpriseserver_retail_en-us-NRMEFPP_EN.iso` (SHA1: 0309FFB4181BA5122C692A6E1079E9FC1D53FCE4)
- `5.2.3790.0.srv03_rtm.030324-2048_x86fre_server-webserver_retail_en-us-NRMWFPP_EN.iso` (SHA1: 46C1CCB2CFC96803E304A35BEF50CD71B2C1DE38)
- `5.1.2600.0.xpclient.010817-1148_x86fre_client-home_retail_en-us-WXHFPP_EN.iso` (SHA1: B273C8D41E3844E3E46722F52F5A4CF9F206C8D0)
- `5.1.2600.0.xpclient.010817-1148_x86fre_client-professional_retail_en-us-WXPFPP_EN.iso` (SHA1: 1400DED4402D50F3864ED3D8DCF5CC52BA79A04A)
- `sbs.iso` (coverted from mdf; SHA1: CDB30C80FDE314C16CA11F5CD31650ECBEC7A214)

Product keys

- Standard Edition: M6RJ9-TBJH3-9DDXM-4VX9Q-K8M8M

64-bit specific

list of tools that must be recompiled in order to build on x64

appendtool
bingen
build
cabbench
calchash
cat
chktrust
flist
gennt32t
gidl
hdivide
infparser
infutil2
makecat
mibcc
midlc
muirct
ntrelhash
pdbstr
preprocessor
rmtshare
shimdbc
symchk
symmake
symstore
tlist
updcats

TODO:

get a functional iso, provide a complete buildguide