```
[Identification]
    OptionType = NetTransport
[Options]
    TC
[FileConstants]
Manufacturer    = "Microsoft"
ProductMajorVersion     = "4"
ProductMinorVersion     = "0"
ProductVersion  = $(ProductMajorVersion)"."$(ProductMinorVersion)
NetBTEventDLL   = "%SystemRoot%\System32\netevent.dll"
NetEventDLL     = "%SystemRoot%\System32\netevent.dll"
IoLogMsgDLL     = "%SystemRoot%\System32\IoLogMsg.dll"
Sockmaxlength           = 16
Sockminlength           = 16
ProductDHCPName          = "DHCP"
ProductDHCPImagePath     = "%SystemRoot%\System32\services.exe"
ProductDHCPSvcType       = "autoserviceshare"
ProductNETBTName          = "NetBT"
ProductNETBTImagePath     = "%SystemRoot%\System32\drivers\netbt.sys"
ProductNETBTSvcType       = "kernel"
NetRuleNETBTType          = "netbt netBiosTransport netbtTransport"
NetRuleNETBTClass         = {"netbtTransport netBiosTransport"}
NetRuleNETBTUse           = $(SoftwareType)" none none"
NetRuleNETBTBindForm      = """NetBT"" yes yes simple"
ProductOpSupport     = 134
ProductLMHOSTSName      = "LmHosts"
ProductLMHOSTSImagePath  = "%SystemRoot%\System32\services.exe"
ProductTCName      = "Tcpip"
ProviderName       = $(ProductTCName)
Winsock20Provider = "%SystemRoot%\System32\rnr20.dll"
Winsock11Provider = "%SystemRoot%\System32\wsock32.dll"
ProductProviderGUID      = "{22059d40-7e9e-11cf-ae5a-00aa00a7112b}"
ProductProviderNameSpaces = 12
ProductTCImagePath   = "\SystemRoot\System32\drivers\tcpip.sys"
ProductTCWshDllPath  = "%SystemRoot%\System32\wshtcpip.dll"
ProductTCSvcType     = "kernel"
NetRuleTCType        = "tcpip tcpipTransport"
NetRuleTCUse         = $(SoftwareType)" none none"
NetRuleTCBindForm    = """Tcpip"" yes yes container"
NetRuleTCBindable    = {"tcpipService tcpipTransport non exclusive
100","tcpipTransport ndisDriver non non 100"}
NetRuleTCClass       = {"tcpipTransport basic","tcpipService basic yes"}
ProductKeyName  = $(!NTN_SoftwareBase)"\"$(Manufacturer)"\"$(Product$
(Option)Name)"\CurrentVersion"
ProductTCKeyName  = $(!NTN_SoftwareBase)"\"$(Manufacturer)"\"$(ProductTCName)"\
CurrentVersion"
TCNetRuleKeyName = $(!NTN_SoftwareBase)"\"$(Manufacturer)"\Tcpip\CurrentVersion\
NetRules"
NetBTNetRuleKeyName = $(!NTN_SoftwareBase)"\"$(Manufacturer)"\NetBT\CurrentVersion\
NetRules"
OldProductKeyName  = $(!NTN_ServiceBase)"\Tcpip\Parameters"
LinkageKeyName  = $(!NTN_ServiceBase)"\Tcpip\Linkage"
ParametersKeyName  = $(!NTN_ServiceBase)"\Tcpip\Parameters"
ServiceTCKeyName = $(!NTN_ServiceBase)"\Tcpip"
ServiceKeyName  = $(!NTN_ServiceBase)
NetBTLinkageKeyName  = $(!NTN_ServiceBase)"\NetBT\Linkage"
AdaptersKeyName  = $(!NTN_ServiceBase)"\NetBT\Adapters"
PerformanceName  = $(!NTN_ServiceBase)"\NetBT\Performance"
NetBTParamKeyName  = $(!NTN_ServiceBase)"\NetBT\Parameters"
```

```
DosDevices      = "SYSTEM\CurrentControlSet\Control\Session Manager\DOS Devices"
[GeneralConstants]
UtilityInf       = "UTILITY.INF"
subroutineinf    = "SUBROUTN.INF"
IPINFOINF        = "ipinfo.inf"
SoftwareType     = "transport"
Exit_Code        = 0
BillboardVisible = 0
from      = ""
to        = ""
ExitCodeOk     = 0
ExitCodeCancel = 1
ExitCodeFatal  = 2
KeyNull         = ""
MAXIMUM_ALLOWED   = 33554432
SERVICE_NO_CHANGE = 4294967295
RegistryErrorIndex = NO_ERROR
KeyProduct      = ""
KeyParameters   = ""
TRUE            = 1
FALSE           = 0
NoTitle         = 0
ExitState   = "Active"
OldVersionExisted = $(FALSE)
DriverPath      = $(!STF_NTPATH)\drivers
[date]
    Now = {} ? $(!LIBHANDLE) GetSystemDate
[Identify]
    read-syms Identification
    set Status     = STATUS_SUCCESSFUL
    set Identifier = $(OptionType)
    set Media      = #("Source Media Descriptions", 1, 1)
    Return $(Status) $(Identifier) $(Media)
[ReturnOptions]
    set Status        = STATUS_FAILED
    set OptionList     = {}
    set OptionTextList = {}
    set LanguageList = ^(LanguagesSupported, 1)
    Ifcontains(i) $($0) in $(LanguageList)
        goto returnoptions
    else
        set Status = STATUS_NOLANGUAGE
        goto finish_ReturnOptions
    endif
returnoptions = +
    set OptionList     = ^(Options, 1)
    set OptionTextList = ^(OptionsText$($0), 1)
    set Status         = STATUS_SUCCESSFUL
finish_ReturnOptions = +
    Return $(Status) $(OptionList) $(OptionTextList)
[InstallOption]
    set Option   = $($1)
    set SrcDir   = $($2)
    set AddCopy  = $($3)
    set DoCopy   = $($4)
    set DoConfig = $($5)
    set InstallFromRas = $($6)
    set EnableDHCPFlag = $($7)
    set InstallList = $($8)
```

```
    ifstr(i) $(EnableDHCPFlag) == ""
        set EnableDHCPFlag = 0
    endif
    ifstr(i) $(InstallList) == ""
        set InstallList = {"1","1","0","0","0","0","0","0"}
    endif
    set LanguageList = ^(LanguagesSupported, 1)
    Ifcontains(i) $($0) NOT-IN $(LanguageList)
        Return STATUS_NOLANGUAGE
    endif
    Debug-Output "OEMNXPTC.INF: STF_CWDIR is: "$(!STF_CWDIR)
    Debug-Output "OEMNXPTC.INF: STF_LANGUAGE is: "$(!STF_LANGUAGE)
    set-subst LF = "\n"
    read-syms GeneralConstants
    read-syms FileConstants
    read-syms DialogConstants$(!STF_LANGUAGE)
    LoadLibrary "x" $(!STF_CWDDIR)\tcpcfg.dll !TCPCFG_HANDLE
    ifstr(i) $(!NTN_Origination) == "NCPA"
        set Continue = $(OK)
    endif
    read-syms FileConstants$(!STF_LANGUAGE)
    detect date
    set-title  $(FunctionTitle)
    set to   = Begin
    set from = Begin
    set CommonStatus = STATUS_SUCCESSFUL
    EndWait
Begin = +
    Ifstr(i) $(!NTN_InstallMode) == deinstall
        set StartLabel = removeadapter
        set OEM_ABANDON_OPTIONS = {+
            $(ProductDHCPName),+
            $(ProductLMHOSTSName)}
        set OEM_ABANDON_SOFTWARE = {+
            $(ProductTCName),+
            $(ProductNETBTName)}
    else-Ifstr(i) $(!NTN_InstallMode) == Update
        set StartLabel = UpgradeSoftware
    else-Ifstr(i) $(!NTN_InstallMode) == configure
        set IPAddressList  = {}
        set SubnetMaskList = {}
        set StartLabel = configureadapter
    else-Ifstr(i) $(!NTN_InstallMode) == bind
        set StartLabel = bindingadapter
    else
        set StartLabel = installadapter
        set OEM_ABANDON_SOFTWARE = {}
        set OEM_ABANDON_OPTIONS = {}
    endif
    set from = $(fatal)
    set to = $(fatal)
    goto $(StartLabel)
installadapter = +
    OpenRegKey $(!REG_H_LOCAL) "" $(ProductKeyName) $(MAXIMUM_ALLOWED) KeyProduct
    Debug-Output "OEMNXPTC.INF: Back from opening key for the product"
    Ifstr $(KeyProduct) != $(KeyNull)
        CloseRegKey $(KeyProduct)
        Shell $(UtilityInf), VerExistedDlg, $(Product$(Option)Title),+
            $(ProductVersion)
```

```
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "OEMNXPTC.INF: ShellCode error: cannot get an error
string."
            goto ShellCodeError
        endif
        set CommonStatus = STATUS_USERCANCEL
        goto end
    endif
    Shell "" GetFilesSize
    set SizeList = $($R0)
    set OptionFileList = { "oemnsvcu.inf"}
    set TcpOptionList  = { "TCPIPCU"}
    ForListDo $(OptionFileList)
        Shell $($) GetFilesSize
        ifstr(i) $($R0) == ""
            set Size = "0"
        else
            set Size = $($R0)
        endif
        set SizeList = $(SizeList)"@"$(Size)
    EndForListDo
    set ErrorCode = "0"
    set fCancel = 1
    set AllInstalledFlag = "0"
    set InstallList = "1"
    Ifstr(i) $(!STF_GUI_UNATTENDED) == YES
        OpenRegKey $(!REG_H_LOCAL) "" "System\Setup" $(MAXIMUM_ALLOWED) KeySetup
        ifstr(i) $(KeySetup) != ""
            DeleteRegTree $(KeySetup) "Tcpip"
            CreateRegKey $(KeySetup) {"Tcpip",$(NoTitle),GenericClass} "" $
(MAXIMUM_ALLOWED) "" KeyTcpip
            ifstr(i) $(KeyTcpip) != ""
                SetRegValue $(KeyTcpip) {"GuiUnattended",$(NoTitle),$(!REG_VT_SZ),$
(!STF_GUI_UNATTENDED)}
                SetRegValue $(KeyTcpip) {"Unattended",$(NoTitle),$(!REG_VT_SZ),$(!
STF_UNATTENDED)}
                SetRegValue $(KeyTcpip) {"UnattendedSection",$(NoTitle),$(!
REG_VT_SZ),$(!STF_UNATTENDED_SECTION)}
                CloseRegKey $(KeyTcpip)
            endif
        CloseRegKey $(KeySetup)
        endif
    else-ifstr(i) $(InstallFromRas) == "YES"
    else
        read-syms DHCPDialog
        ui start "Warning"
        ifstr(i) $(DLGEVENT) == "YES"
          set EnableDHCPFlag = "1"
        else
          set EnableDHCPFlag = "0"
        endif
    endif
    Ifcontains(i) "1" in $(InstallList)
        ifstr(i) $(!NTN_InstallMode) == "install"
            Ifstr(i) $(DoCopy) == "YES"
               Shell $(UtilityInf), DoAskSource, $(!STF_CWDDIR), $(SrcDir) YES
               Ifint $($ShellCode) != $(!SHELL_CODE_OK)
                   Goto ShellCodeError
               Else-Ifstr(i) $($R0) == STATUS_FAILED
```

```
                Shell $(UtilityInf) RegistryErrorString "ASK_SOURCE_FAIL"
                ifint $($ShellCode) != $(!SHELL_CODE_OK)
                    goto ShellCodeError
                endif
                set Error = $($R0)
                Goto fatal
            Else-Ifstr(i) $($R0) == STATUS_USERCANCEL
                Goto successful
            Endif
            Set SrcDir = $($R1)
        Endif
    EndIf
Endif
Shell "oemnsvcu.inf" "InstallOption" +
                $(!STF_LANGUAGE) "TCPIPCU" +
                $(SrcDir) $(AddCopy) $(DoCopy) $(DoConfig)
ifstr(i) $(!NTN_InstallMode) == "install"
    Debug-Output "OEMNXPTC.INF: installadapter: installing [Install-Option]"
    install "Install-Option"
    ifstr(i) $(STF_INSTALL_OUTCOME) != STF_SUCCESS
       Shell $(UtilityInf) RegistryErrorString "UNABLE_COPY_FILE"
       ifint $($ShellCode) != $(!SHELL_CODE_OK)
           goto ShellCodeError
       endif
       set Error = $($R0)
       goto fatal
    endif
endif
set OEM_ABANDON_ON  = TRUE
read-syms Billboard1$(!STF_LANGUAGE)
Shell "subroutn.inf" PushBillboard NETSTATUSDLG $(Status)
Set BillboardVisible = 1
set DoTC        = FALSE
Set DoLMHOSTS   = FALSE
Set DoNbt       = FALSE
Set DoAFD       = FALSE
Set DoDHCP      = FALSE
Ifstr(i) $(Option) == TC
    set DoTC       = TRUE
    Set DoLMHOSTS  = TRUE
    Set DoNbt      = TRUE
    Set DoAFD      = TRUE
    Set DoDHCP      = TRUE
Else
    Debug-Output "OEMNXPTC.INF: Unrecognized option"
Endif
StartWait
Ifstr(i) $(DoLMHOSTS) == TRUE
    Debug-Output "OEMNXPTC.INF: Install LmHosts registry"
    Set OEM_ABANDON_OPTIONS = >($(OEM_ABANDON_OPTIONS), $(ProductLMHOSTSName))
    Shell $(UtilityInf), CreateService, $(ProductLMHOSTSName), +
        $(ProductLMHOSTSDisplayName), $(ProductLMHOSTSImagePath), +
        "autoserviceshare", "", {"+NetworkProvider"}, "LocalSystem",+
        $(NetEventDLL)
    ifint $($ShellCode) != $(!SHELL_CODE_OK)
        Debug-Output "OEMNXPTC.INF: ShellCode error"
        goto ShellCodeError
    endif
    set RegistryErrorIndex = $($R0)
```

```
        CloseRegKey $($R1)
        CloseRegKey $($R2)
        CloseRegKey $($R3)
        Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
            EndWait
            Debug-Output "OEMNXPTC.INF: Registry error: add software components"
            goto fatalregistry
        endif
    endif
    ifstr(i) $(DoDHCP) == TRUE
        Set ThisOption = DHCP
        Set OEM_ABANDON_OPTIONS = >($(OEM_ABANDON_OPTIONS), $(Product$
(ThisOption)Name))
        Debug-Output "OEMNXPTC.INF: installing DHCP..."
        ifint $(EnableDHCPFlag) == 0
            set ProductDHCPSvcType = serviceshare
        endif
        Shell $(UtilityInf), CreateService, $(Product$(ThisOption)Name),+
            $(Product$(ThisOption)DisplayName),+
            $(Product$(ThisOption)ImagePath),+
            $(Product$(ThisOption)SvcType), "TDI", {"Tcpip","Afd","NetBT"}, "",+
            "%SystemRoot%\System32\dhcpcsvc.dll", 7, "", "", "",+
            "%SystemRoot%\System32\kernel32.dll"
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "OEMNXPTC.INF: ShellCode error"
            goto ShellCodeError
        endif
        set RegistryErrorIndex = $($R0)
        Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
            EndWait
            Debug-Output "OEMNXPTC.INF: Registry error: Create Service components"
            CloseRegKey $($R1)
            CloseRegKey $($R2)
            CloseRegKey $($R3)
            goto fatalregistry
        endif
        CloseRegKey $($R1)
        Set DhcpParameterKey = $($R2)
        CloseRegKey $($R3)
        CreateRegKey $(DhcpParameterKey) {"Options",$(NoTitle),GenericClass} "" $
(MAXIMUM_ALLOWED) "" KeyOptions
        CreateRegKey $(KeyOptions) {"1",$(NoTitle),GenericClass} "" $
(MAXIMUM_ALLOWED) "" Key1
        Shell $(UtilityInf) AddValueList, $(Key1),+
            {{"RegLocation", $(NoTitle), $(!REG_VT_SZ), "System\CurrentControlSet\
Services\?\Parameters\Tcpip\DhcpSubnetMaskOpt"},+
            {"KeyType", $(NoTitle), $(!REG_VT_DWORD), $(!REG_VT_MULTI_SZ)}}
        CloseRegKey $(Key1)
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "OEMNXPTC.INF: ShellCode error"
            goto ShellCodeError
        endif
        CreateRegKey $(KeyOptions) {"3",$(NoTitle),GenericClass} "" $
(MAXIMUM_ALLOWED) "" Key3
        Shell $(UtilityInf) AddValueList, $(Key3),+
            {{"RegLocation", $(NoTitle), $(!REG_VT_SZ), "System\CurrentControlSet\
Services\?\Parameters\Tcpip\DhcpDefaultGateway"},+
            {"KeyType", $(NoTitle), $(!REG_VT_DWORD), $(!REG_VT_MULTI_SZ)}}
        CloseRegKey $(Key3)
```

```
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "OEMNXPTC.INF: ShellCode error"
            goto ShellCodeError
        endif
        CreateRegKey $(KeyOptions) {"6",$(NoTitle),GenericClass} "" $
(MAXIMUM_ALLOWED) "" Key6
        Shell $(UtilityInf) AddValueList, $(Key6),+
            {{"RegLocation", $(NoTitle), $(!REG_VT_SZ), "System\CurrentControlSet\
Services\Tcpip\Parameters\DhcpNameServer"},+
            {"KeyType", $(NoTitle), $(!REG_VT_DWORD), $(!REG_VT_SZ)}}
        CloseRegKey $(Key6)
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "OEMNXPTC.INF: ShellCode error"
            goto ShellCodeError
        endif
        CreateRegKey $(KeyOptions) {"15",$(NoTitle),GenericClass} "" $
(MAXIMUM_ALLOWED) "" Key15
        Shell $(UtilityInf) AddValueList, $(Key15),+
            {{"RegLocation", $(NoTitle), $(!REG_VT_SZ), "System\CurrentControlSet\
Services\Tcpip\Parameters\DhcpDomain"},+
            {"KeyType", $(NoTitle), $(!REG_VT_DWORD), $(!REG_VT_SZ)}}
        CloseRegKey $(Key15)
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "OEMNXPTC.INF: ShellCode error"
            goto ShellCodeError
        endif
        CreateRegKey $(KeyOptions) {"44",$(NoTitle),GenericClass} "" $
(MAXIMUM_ALLOWED) "" Key44
        Shell $(UtilityInf) AddValueList, $(Key44),+
            {{"RegLocation", $(NoTitle), $(!REG_VT_SZ), "System\CurrentControlSet\
Services\NetBT\Adapters\?\DhcpNameServer"},+
            {"KeyType", $(NoTitle), $(!REG_VT_DWORD), $(!REG_VT_SZ)}}
        CloseRegKey $(Key44)
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "OEMNXPTC.INF: ShellCode error"
            goto ShellCodeError
        endif
        CreateRegKey $(KeyOptions) {"46",$(NoTitle),GenericClass} "" $
(MAXIMUM_ALLOWED) "" Key46
        Shell $(UtilityInf) AddValueList, $(Key46),+
            {{"RegLocation", $(NoTitle), $(!REG_VT_SZ), "System\CurrentControlSet\
Services\NetBT\Parameters\DhcpNodeType"},+
            {"KeyType", $(NoTitle), $(!REG_VT_DWORD), $(!REG_VT_DWORD)}}
        CloseRegKey $(Key46)
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "OEMNXPTC.INF: ShellCode error"
            goto ShellCodeError
        endif
        CreateRegKey $(KeyOptions) {"47",$(NoTitle),GenericClass} "" $
(MAXIMUM_ALLOWED) "" Key47
        Shell $(UtilityInf) AddValueList, $(Key47),+
            {{"RegLocation", $(NoTitle), $(!REG_VT_SZ), "System\CurrentControlSet\
Services\NetBT\Parameters\DhcpScopeID"},+
            {"KeyType", $(NoTitle), $(!REG_VT_DWORD), $(!REG_VT_SZ)}}
        CloseRegKey $(Key47)
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "OEMNXPTC.INF: ShellCode error"
            goto ShellCodeError
        endif
```

```
        set RegistryErrorIndex = $($R0)
        CloseRegKey $(KeyOptions)
        CloseRegKey $(DhcpParameterKey)
        Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
            EndWait
            Debug-Output "OEMNXPTC.INF: Registry error: add value list."
            goto fatalregistry
        endif
    endif
    ifstr(i) $(DoTC) == TRUE
        set ThisOption = "TC"
        Set OEM_ABANDON_OPTIONS = >($(OEM_ABANDON_OPTIONS), $(ProductTCName))
        Shell $(UtilityInf), AddSoftwareComponent, $(Manufacturer),+
            $(Product$(ThisOption)Name),+
            $(Product$(ThisOption)Name),+
            $(Product$(ThisOption)DisplayName), $(STF_CONTEXTINFNAME),+
            $(Product$(ThisOption)ImagePath),+
            $(Product$(ThisOption)SvcType),+
            "PNP_TDI", {}, "", $(NetEventDLL)
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "OEMNXPTC.INF: ShellCode error, add software component"
            goto ShellCodeError
        endif
        set RegistryErrorIndex = $($R0)
        Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
            EndWait
            Debug-Output "OEMNXPTC.INF: Registry error: add software components"
            CloseRegKey $($R1)
            CloseRegKey $($R2)
            CloseRegKey $($R3)
            CloseRegKey $($R4)
            CloseRegKey $($R5)
            goto fatalregistry
        endif
        Set TcpVersKeyHandle  = $($R1)
        Set TcpRulesKeyHandle = $($R2)
        Set TcpSvcKeyHandle   = $($R3)
        Set TcpParmKeyHandle  = $($R4)
        Set TcpLinkageHandle  = $($R5)
        set NewValueList = {{SoftwareType,$(NoTitle),$(!REG_VT_SZ),$
(SoftwareType)},+
                            {MajorVersion,$(NoTitle),$(!REG_VT_DWORD),$
(ProductMajorVersion)},+
                            {MinorVersion,$(NoTitle),$(!REG_VT_DWORD),$
(ProductMinorVersion)},+
                            {Title,$(NoTitle),$(!REG_VT_SZ),$(Product$
(ThisOption)Title)},+
                            {Description,$(NoTitle),$(!REG_VT_SZ),$(Product$
(ThisOption)Description)},+
                            {ServiceName,$(NoTitle),$(!REG_VT_SZ),$(Product$
(ThisOption)Name)},+
                            {Review,$(NoTitle),$(!REG_VT_DWORD),1},+
                            {OperationsSupport,$(NoTitle),$(!REG_VT_DWORD),$
(ProductOpSupport)}, +
                            {InstallDate,$(NoTitle),$(!REG_VT_DWORD),*($(Now),1)}}
        Shell  $(UtilityInf), AddValueList, $(TcpVersKeyHandle), $(NewValueList)
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "OEMNXPTC.INF: ShellCode error, add value list"
            goto ShellCodeError
```

```
        endif
        set RegistryErrorIndex = $($R0)
        Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
            EndWait
            Debug-Output "OEMNXPTC.INF: Registry error: add value list."
            CloseRegKey $(TcpVersKeyHandle)
            CloseRegKey $(TcpRulesKeyHandle)
            CloseRegKey $(TcpSvcKeyHandle)
            CloseRegKey $(TcpParmKeyHandle)
            CloseRegkey $(TcpLinkageHandle)
            goto fatalregistry
        Endif
        CreateRegKey $(TcpParmKeyHandle) {"PersistentRoutes",$
(NoTitle),GenericClass} "" +
                $(MAXIMUM_ALLOWED) "" KeyPersistentRoutes
        CloseRegKey $(KeyPersistentRoutes)
        CreateRegKey $(TcpSvcKeyHandle) {"Performance",$(NoTitle),GenericClass} ""
+
                $(MAXIMUM_ALLOWED) "" KeyPerformance
        set NewValueList = {{Library,$(NoTitle),$(!REG_VT_SZ),"Perfctrs.dll"},+
                            {Open,$(NoTitle),$(!
REG_VT_SZ),"OpenTcpIpPerformanceData"},+
                            {Collect,$(NoTitle),$(!
REG_VT_SZ),"CollectTcpIpPerformanceData"},+
                            {Close,$(NoTitle),$(!
REG_VT_SZ),"CloseTcpIpPerformanceData"}}
        Shell  $(UtilityInf), AddValueList, $(KeyPerformance), $(NewValueList)
        set RegistryErrorIndex = $($R0)
        CloseRegKey $(KeyPerformance)
        Ifstr $(RegistryErrorIndex) != NO_ERROR
            EndWait
            Debug-Output "OEMNXPTC.INF: Registry error: add performance info."
            CloseRegKey $(TcpVersKeyHandle)
            CloseRegKey $(TcpRulesKeyHandle)
            CloseRegKey $(TcpSvcKeyHandle)
            CloseRegKey $(TcpParmKeyHandle)
            CloseRegkey $(TcpLinkageHandle)
            goto fatalregistry
        Endif
        CreateRegKey $(TcpSvcKeyHandle) {"ServiceProvider",$(NoTitle),GenericClass}
"" +
                $(MAXIMUM_ALLOWED) "" KeyNetworkProvider
        set NewValueList = {{Class,$(NoTitle),$(!REG_VT_DWORD),8},+
                            {DnsPriority,$(NoTitle),$(!REG_VT_DWORD),2000},+
                            {HostsPriority,$(NoTitle),$(!REG_VT_DWORD),500},+
                            {LocalPriority,$(NoTitle),$(!REG_VT_DWORD),499},+
                            {ProviderPath,$(NoTitle),$(!REG_VT_EXPAND_SZ),$
(Winsock11Provider)},+
                            {NetbtPriority,$(NoTitle),$(!REG_VT_DWORD),2001}}
        Shell  $(UtilityInf), AddValueList, $(KeyNetworkProvider), $(NewValueList)
        set RegistryErrorIndex = $($R0)
        CloseRegKey $(KeyNetworkProvider)
        Ifstr $(RegistryErrorIndex) != NO_ERROR
            EndWait
            Debug-Output "OEMNXPTC.INF: Registry error: add performance info."
            CloseRegKey $(TcpVersKeyHandle)
            CloseRegKey $(TcpRulesKeyHandle)
            CloseRegKey $(TcpSvcKeyHandle)
            CloseRegKey $(TcpParmKeyHandle)
```

```
            CloseRegkey $(TcpLinkageHandle)
            goto fatalregistry
        Endif
        set NewValueList = {{type,$(NoTitle),$(!REG_VT_SZ),$(NetRule$
(ThisOption)Type)},+
                            {use,$(NoTitle),$(!REG_VT_SZ),$(NetRule$
(ThisOption)Use)},+
                            {bindform,$(NoTitle),$(!REG_VT_SZ),$(NetRule$
(ThisOption)BindForm)},+
                            {bindable,$(NoTitle),$(!REG_VT_MULTI_SZ),$(NetRule$
(ThisOption)Bindable)},+
                            {class,$(NoTitle),$(!REG_VT_MULTI_SZ),$(NetRule$
(ThisOption)Class)},+
                            {InfOption,$(NoTitle),$(!REG_VT_SZ),$(ThisOption)}}
        Shell  $(UtilityInf), AddValueList, $(TcpRulesKeyHandle), $(NewValueList)
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "OEMNXPTC.INF: ShellCode error."
            goto ShellCodeError
        endif
        set RegistryErrorIndex = $($R0)
        CloseRegKey $(TcpVersKeyHandle)
        CloseRegKey $(TcpRulesKeyHandle)
        LibraryProcedure ResultHostname $(!TCPCFG_HANDLE), ConvertHostname $(!
STF_COMPUTERNAME)
        set NewValueList = {{EnableDHCP,$(NoTitle),$(!REG_VT_DWORD),$
(EnableDHCPFlag)},+
                            {DataBasePath,$(NoTitle),$(!
REG_VT_EXPAND_SZ),"%SystemRoot%\System32\drivers\etc"},+
                            {Domain,$(NoTitle),$(!REG_VT_SZ),""},+
                            {Hostname,$(NoTitle),$(!REG_VT_SZ),$(ResultHostname)},+
                            {NameServer,$(NoTitle),$(!REG_VT_SZ),""},+
                            {ForwardBroadcasts,$(NoTitle),$(!REG_VT_DWORD),0},+
                            {IPEnableRouter,$(NoTitle),$(!REG_VT_DWORD),0},+
                            {SearchList,$(NoTitle),$(!REG_VT_SZ),""}}
        Shell  $(UtilityInf), AddValueList, $(TcpParmKeyHandle), $(NewValueList)
        CloseRegKey $(TcpParmKeyHandle)
        Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
            CloseRegKey $(TcpLinkageHandle)
            CloseRegKey $(TcpSvcKeyHandle)
            EndWait
            Debug-Output "OEMNXPTC.INF: Registry error: add value list."
            goto fatalregistry
        endif
        Shell $(UtilityInf) AddRpcProtocol "ncacn_ip_tcp" "RpcLtCcm.Dll"
"RpcLtScm.Dll"
        Ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "OEMNXPTC.INF: ShellCode error adding RPC procotol"
            goto ShellCodeError
        Endif
        Set RegistryErrorIndex = $($R0)
        Ifstr(i) $(RegistryErrorIndex) != "NO_ERROR"
            Debug-Output "OEMNXPTC.INF: ERROR adding RPC protocol data"
            EndWait
            goto fatalregistry
        Endif
        Shell $(UtilityInf) AddRpcProtocol "ncadg_ip_udp" "RpcLtCcm.Dll"
"RpcLtScm.Dll"
        Ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "OEMNXPTC.INF: ShellCode error adding RPC procotol"
```

```
            goto ShellCodeError
        Endif
        Set RegistryErrorIndex = $($R0)
        Ifstr(i) $(RegistryErrorIndex) != "NO_ERROR"
            Debug-Output "OEMNXPTC.INF: ERROR adding RPC protocol data"
            EndWait
            goto fatalregistry
        Endif
        Shell $(UtilityInf) AddMixRpcProtocol "Netbios" "tcpip" "ncacn_nb_tcp"
"rpcltccm.dll" "RpcLtScm.Dll"
        Ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "OEMNXPTC.INF: ShellCode error adding RPC procotol (2)"
            goto ShellCodeError
        endif
        Set RegistryErrorIndex = $($R0)
        Ifstr(i) $(RegistryErrorIndex) != "NO_ERROR"
            Debug-Output "OEMNXPTC.INF: ERROR adding RPC protocol data (2)"
            EndWait
            goto fatalregistry
        Endif
        Shell $(UtilityInf), AddWinsockInfo, +
             $(Product$(ThisOption)Name), +
             $(Product$(ThisOption)WshDllPath), +
             $(Sockmaxlength),$(Sockminlength)
        set RegistryErrorIndex = $($R0)
        Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
            Debug-Output "OEMNXPTC.INF: Registry error: Add Winsock Info."
        Endif
    endif
    Ifstr(i) $(DoNbt) == TRUE
        Set ThisOption = NETBT
        Set OEM_ABANDON_SOFTWARE = >($(OEM_ABANDON_SOFTWARE), $(Product$
(ThisOption)Name))
        Debug-Output "OEMNXPTC.INF: installing NETBT..."
        Shell $(UtilityInf), AddSoftwareComponent, $(Manufacturer),+
            $(Product$(ThisOption)Name),+
            $(Product$(ThisOption)Name),+
            $(Product$(ThisOption)DisplayName), $(STF_CONTEXTINFNAME),+
            $(Product$(ThisOption)ImagePath), $(Product$(ThisOption)SvcType),
"PNP_TDI", {"Tcpip"}, "", $(NetBTEventDLL)
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "OEMNXPTC.INF: ShellCode error"
            goto ShellCodeError
        endif
        set RegistryErrorIndex = $($R0)
        Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
            EndWait
            Debug-Output "OEMNXPTC.INF: Registry error: add software components"
            CloseRegKey $($R1)
            CloseRegKey $($R2)
            CloseRegKey $($R3)
            CloseRegKey $($R4)
            CloseRegKey $($R5)
            goto fatalregistry
        endif
        Set NBTProductKey       = $($R1)
        Set NBTNetRuleKey       = $($R2)
        Set NBTServiceKey       = $($R3)
        Set NBTParameterKey     = $($R4)
```

```
        Set NBTLinkageKey       = $($R5)
        set NewValueList = {{SoftwareType,$(NoTitle),$(!REG_VT_SZ),$
(SoftwareType)},+
                            {MajorVersion,$(NoTitle),$(!REG_VT_DWORD),$
(ProductMajorVersion)},+
                            {MinorVersion,$(NoTitle),$(!REG_VT_DWORD),$
(ProductMinorVersion)},+
                            {Hidden,0,$(!REG_VT_DWORD),1},+
                            {Title,$(NoTitle),$(!REG_VT_SZ),$(Product$
(ThisOption)Title)},+
                            {Description,$(NoTitle),$(!REG_VT_SZ),$(Product$
(ThisOption)Description)},+
                            {ServiceName,$(NoTitle),$(!REG_VT_SZ),$(Product$
(ThisOption)Name)},+
                            {InstallDate,$(NoTitle),$(!REG_VT_DWORD),*($(Now),1)}}
        Shell  $(UtilityInf), AddValueList, $(NBTProductKey), $(NewValueList)
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "OEMNXPTC.INF: ShellCode error."
            goto ShellCodeError
        endif
        set RegistryErrorIndex = $($R0)
        Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
            EndWait
            Debug-Output "OEMNXPTC.INF: Registry error: add value list."
            CloseRegKey $(NBTProductKey)
            CloseRegKey $(NBTNetRuleKey)
            CloseRegKey $(NBTServiceKey)
            CloseRegKey $(NBTParameterKey)
            CloseRegKey $(NBTLinkageKey)
            goto fatalregistry
        endif
        set NewValueList = {{OtherDependencies,$(NoTitle),$(!REG_VT_MULTI_SZ),
{"Tcpip"}}}
        Shell  $(UtilityInf), AddValueList, $(NBTLinkageKey), $(NewValueList)
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "OEMNXPTC.INF: ShellCode error."
            goto ShellCodeError
        endif
        set RegistryErrorIndex = $($R0)
        Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
            EndWait
            Debug-Output "OEMNXPTC.INF: Registry error: add value list."
            CloseRegKey $(NBTProductKey)
            CloseRegKey $(NBTNetRuleKey)
            CloseRegKey $(NBTServiceKey)
            CloseRegKey $(NBTParameterKey)
            CloseRegKey $(NBTLinkageKey)
            goto fatalregistry
        endif
        set NewValueList = {{type,$(NoTitle),$(!REG_VT_SZ),$(NetRule$
(ThisOption)Type)},+
                             {use,$(NoTitle),$(!REG_VT_SZ),$(NetRule$
(ThisOption)Use)},+
                             {bindform,$(NoTitle),$(!REG_VT_SZ),$(NetRule$
(ThisOption)BindForm)},+
                             {class,$(NoTitle),$(!REG_VT_MULTI_SZ),$(NetRule$
(ThisOption)Class)},+
                             {InfOption,$(NoTitle),$(!REG_VT_SZ),$(ThisOption)}}
        Shell  $(UtilityInf), AddValueList, $(NBTNetRuleKey), $(NewValueList)
```

```
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "OEMNXPTC.INF: ShellCode error."
            goto ShellCodeError
        endif
        set RegistryErrorIndex = $($R0)
        Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
            EndWait
            Debug-Output "OEMNXPTC.INF: Registry error: add value list."
            CloseRegKey $(NBTProductKey)
            CloseRegKey $(NBTNetRuleKey)
            CloseRegKey $(NBTServiceKey)
            CloseRegKey $(NBTParameterKey)
            CloseRegKey $(NBTLinkageKey)
            goto fatalregistry
        endif
        Set NewValueList = {{NbProvider,$(NoTitle),$(!REG_VT_SZ),"_tcp"},+
                {NameServerPort,$(NoTitle),$(!REG_VT_DWORD),137},+
                {EnableLMHOSTS,$(NoTitle),$(!REG_VT_DWORD),1},+
                {EnableProxy,$(NoTitle),$(!REG_VT_DWORD),0},+
                {EnableDNS,$(NoTitle),$(!REG_VT_DWORD),0},+
                {CacheTimeout,$(NoTitle),$(!REG_VT_DWORD),600000},+
                {BcastNameQueryCount,$(NoTitle),$(!REG_VT_DWORD),3},+
                {BcastQueryTimeout,$(NoTitle),$(!REG_VT_DWORD),750},+
                {NameSrvQueryCount,$(NoTitle),$(!REG_VT_DWORD),3},+
                {NameSrvQueryTimeout,$(NoTitle),$(!REG_VT_DWORD),1500},+
                {Size/Small/Medium/Large,$(NoTitle),$(!REG_VT_DWORD),1},+
                {SessionKeepAlive,$(NoTitle),$(!REG_VT_DWORD),3600000},+
                {ScopeID,$(NoTitle),$(!REG_VT_SZ),""},+
                {TransportBindName,$(NoTitle),$(!REG_VT_SZ),"\Device\"},+
                }
        Shell  $(UtilityInf), AddValueList, $(NBTParameterKey), $(NewValueList)
        Ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "OEMNXPTC.INF: ShellCode error."
            goto ShellCodeError
        Endif
        set RegistryErrorIndex = $($R0)
        Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
            EndWait
            Debug-Output "OEMNXPTC.INF: Registry error: add value list."
            CloseRegKey $(NBTProductKey)
            CloseRegKey $(NBTNetRuleKey)
            CloseRegKey $(NBTServiceKey)
            CloseRegKey $(NBTParameterKey)
            CloseRegKey $(NBTLinkageKey)
            goto fatalregistry
        endif
        CreateRegKey $(NBTServiceKey) {"Adapters",$(NoTitle),GenericClass} "" $
(MAXIMUM_ALLOWED) "" AdaptersKey
        CloseRegKey $(AdaptersKey)
        set RegistryErrorIndex = $($R0)
        CloseRegKey $(NBTProductKey)
        CloseRegKey $(NBTNetRuleKey)
        CloseRegKey $(NBTServiceKey)
        CloseRegKey $(NBTParameterKey)
        CloseRegKey $(NBTLinkageKey)
        Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
            EndWait
            Debug-Output "OEMNXPTC.INF: Registry error: add value list."
            goto fatalregistry
```

```
        endif
    Endif
    Ifstr(i) $(DoAFD) == TRUE
        Shell $(UtilityInf) AddAFD
        OpenRegKey $(!REG_H_LOCAL) "" $(!NTN_ServiceBase)"\Afd" $(MAXIMUM_ALLOWED)
AfdKey
        ifstr(i) $(AfdKey) != ""
            GetRegValue $(AfdKey) "DependOnGroup" GroupList
            debug-output "DependOnGroup:"$(GroupList)
            set GroupValues = *($(GroupList),4)
            debug-output "DependOnGroup:"$(GroupValues)
            set NewGroupValues = {}
            ForListDo $(GroupValues)
                ifstr(i) $($) != "TDI"
                    ifstr(i) $(NewGroupValues) == {}
                        Set NewGroupValues = {$($)}
                    else
                        Set NewGroupValues = >($(NewGroupValues), $($))
                    endif
                endif
            EndForListDo
            SetRegValue $(AfdKey) {"DependOnGroup",$(NoTitle),$(!REG_VT_MULTI_SZ),$
(NewGroupValues)}
            CloseRegKey $(AfdKey)
        endif
    Endif
    Shell $(UtilityInf), AddNameSpaceProvider, +
            $(ProviderDisplayName), +
            $(Winsock20Provider), +
            $(ProductProviderNameSpaces), +
            FALSE, +
            $(ProductProviderGUID)
    set RegistryErrorIndex = $($R0)
    Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
        EndWait
        Debug-Output "OEMNXPTC.INF: Registry error: add name space provider"
        goto fatalregistry
    endif
    Shell $(UtilityInf), AddServiceProvider, $(ProviderName), +
        $(Winsock11Provider), $(ProviderDisplayName), 8
    set RegistryErrorIndex = $($R0)
    Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
        EndWait
        Debug-Output "OEMNXPTC.INF: Registry error: add software components"
        goto fatalregistry
    endif
    ifstr(i) $(!STF_GUI_UNATTENDED) != YES
        ifstr(i) $(InstallFromRas) != "YES"
            OpenRegKey $(!REG_H_LOCAL) "" $(!NTN_ServiceBase)"\RASMan" $
(MAXIMUM_ALLOWED) RASKey
            ifstr(i) $(RASKey) != ""
                Shell $(subroutineinf) SetupMessage, $(!STF_LANGUAGE), "WARNING", $
(RASRebindError)
                ifint $($ShellCode) != $(!SHELL_CODE_OK)
                    goto ShellCodeError
                endif
                CloseRegKey $(RASKey)
                ifstr(i) $($R1) == "OK"
                    set SaveNTN_InstallMode = $(!NTN_InstallMode)
```

```
                    set !NTN_InstallMode = configure
                    Shell "oemnsvra.inf" InstallOption $(!STF_LANGUAGE) "RAS" $
(SrcDir) $(AddCopy) $(DoCopy) $(DoConfig)
                    set !NTN_InstallMode = $(SaveNTN_InstallMode)
                endif
            endif
        endif
    endif
    Ifint $(BillboardVisible) != 0
        Shell "subroutn.inf" PopBillboard
        Set BillboardVisible = 0
    Endif
    EndWait
    goto successful
configureadapter = +
    ifstr(i) $(Option) == "TC"
        OpenRegKey $(!REG_H_LOCAL) "" $(ParametersKeyName) $(MAXIMUM_ALLOWED)
ParametersKey
        GetRegValue $(ParametersKey) "EnableDHCP" EnableDHCPInfo
        Set FirstTimeBinding = 0
        set EnableDHCPFlag = 0
        ifint $(RegLastError) == $(!REG_ERROR_SUCCESS)
            Set FirstTimeBinding = 1
            set EnableDHCPFlag = *($(EnableDHCPInfo),4)
            DeleteRegValue $(ParametersKey) "EnableDHCP"
            ifint $(FirstTimeBinding) == 1
                ifint $(EnableDHCPFlag) == 0
                    Shell $(IPINFOINF), GetIPInfo
                    Ifint $($ShellCode) != $(!SHELL_CODE_OK)
                    Else
                        set DefaultGateway = $($R0)
                        set IPAddressList  = $($R1)
                        set SubnetMaskList = $($R2)
                    Endif
                endif
            endif
        EndIf
        CloseRegKey $(ParametersKey)
        OpenRegKey $(!REG_H_LOCAL) "" $(LinkageKeyName) $(MAXIMUM_ALLOWED)
LinkageKey
        GetRegValue $(LinkageKey) "Bind" BindList
        ifstr(i) $(BindList) == {}
           Set BindList = ""
        endif
        ifstr(i) $(BindList) == ""
            LibraryProcedure Result, $(!LIBHANDLE), SetupChangeServiceStart "DHCP",
4
            debug-output "OEMNXPTC.INF: no binding info; warning user"
            read-syms InfoDlgCantConfigure
            goto infomsg
        endif
        set OldVersionExisted = $(TRUE )
        set CardList = ""
        set CardCount = 0
        ForListDo *($(BindList),4)
            Split-String $($), "\", BindInfo
            QueryListSize BindListSize $(BindInfo)
            set CardName = *($(BindInfo),$(BindListSize))
            OpenRegKey $(!REG_H_LOCAL) "" $(!NTN_ServiceBase)"\"*($(BindInfo),$
```

```
(BindListSize)) $(MAXIMUM_ALLOWED) CardServiceKey
            ifstr(i) $(CardServiceKey) != ""
                CloseRegKey $(CardServiceKey)
                set-add CardCount = $(CardCount),1
                ifstr $(CardList) == ""
                    set CardList = {$(CardName)}
                else
                    set CardList = >($(CardList),$(CardName))
                endif
            endif
        EndForListDo
        debug-output $(CardList)
        set OldCardList = $(CardList)
        set CardList = ""
        set CardServiceKey = ""
        set CardCount = 1
        ForListDo $(OldCardList)
            ifstr(i) $(CardServiceKey) == ""
                ifstr $(CardList) == ""
                    set CardList = $($)
                else
                    set CardList = $(CardList)"@"$($)
                endif
                debug-output "Doing:"$($)
                debug-output $(CardList)
                set DefaultIPAddress = *($(IPAddressList),$(CardCount))
                set DefaultSubnetMask = *($(SubnetMaskList),$(CardCount))
                OpenRegKey $(!REG_H_LOCAL) "" "System\CurrentControlSet\Services\"$
($)"\Parameters" $(MAXIMUM_ALLOWED) ServiceParamKey
                ifstr(i) $(ServiceParamKey) != ""
                    OpenRegKey $(ServiceParamKey) "" "Tcpip" $(MAXIMUM_ALLOWED)
TCPIPKey
                    GetRegValue $(ParametersKey) "AutoIPAddress" AutoIPInfo
                    set AutoIPFlag = *($(AutoIPInfo),4)
                    set EnableDHCPForThisAdapter = $(EnableDHCPFlag)
                    ifint $(AutoIPFlag) == 1
                        set EnableDHCPForThisAdapter = 0
                    endif
                    ifstr(i) $(TCPIPKey) == ""
                        CreateRegKey $(ServiceParamKey) {"Tcpip",$
(NoTitle),GenericClass} "" +
                        $(MAXIMUM_ALLOWED) "" TCPIPKey
                        debug-output "set Parameters"
                        debug-output "EnableDHCPFlag:"$(EnableDHCPForThisAdapter)
                        set NewValueList = {{EnableDHCP,$(NoTitle),$(!
REG_VT_DWORD),$(EnableDHCPForThisAdapter)},+
                                            {UseZeroBroadcast,$(NoTitle),$(!
REG_VT_DWORD),0},+
                                            {LLInterface,$(NoTitle),$(!REG_VT_SZ),""}}
                        Shell  $(UtilityInf), AddValueList, $(TCPIPKey), $
(NewValueList)
                        ifstr(i) $(TCPIPKey) != ""
                            debug-output $(RegistryErrorIndex)
                            ifint $(EnableDHCPForThisAdapter) == 1
                                set NewValueList = {{IPAddress,$(NoTitle),$(!
REG_VT_MULTI_SZ),{"0.0.0.0"}},+
                                                    {DefaultGateway,$(NoTitle),$(!
REG_VT_MULTI_SZ),{}},+
                                                    {SubnetMask,$(NoTitle),$(!
```

```
REG_VT_MULTI_SZ),{"0.0.0.0"}}}
                                Shell  $(UtilityInf), AddValueList, $(TCPIPKey), $
(NewValueList)
                                ifint $($ShellCode) != $(!SHELL_CODE_OK)
                                    Debug-Output "OEMNXPTC.INF: ShellCode error:
cannot write default subnet mask error."
                                    goto ShellCodeError
                                endif
                            else
                                ifstr(i) $(DefaultIPAddress) != ""
                                    set IPAddressInfo = {}
                                    GetRegValue $(TCPIPKey) "IPAddress"
IPAddressInfo
                                    set IPAddress = *($(IPAddressInfo), 4)
                                    ifstr(i) $(IPAddress) == ""
                                        set NewValueList = {{IPAddress,$(NoTitle),$
(!REG_VT_MULTI_SZ),{$(DefaultIPAddress)}}}
                                        Shell  $(UtilityInf), AddValueList, $
(TCPIPKey), $(NewValueList)
                                        ifint $($ShellCode) != $(!SHELL_CODE_OK)
                                            Debug-Output "OEMNXPTC.INF: ShellCode
error: cannot write default subnet mask error."
                                            goto ShellCodeError
                                        endif
                                    endif
                                endif
                                ifstr(i) $(DefaultSubnetMask) != ""
                                    set SubnetMasInfo = {}
                                    GetRegValue $(TCPIPKey) "SubnetMask"
SubnetMaskInfo
                                    set SubnetMask = *($(SubnetMaskInfo), 4)
                                    ifstr(i) $(SubnetMask) == ""
                                        set NewValueList = {{SubnetMask,$(NoTitle),
$(!REG_VT_MULTI_SZ),{$(DefaultSubnetMask)}}}
                                        Shell  $(UtilityInf), AddValueList, $
(TCPIPKey), $(NewValueList)
                                        ifint $($ShellCode) != $(!SHELL_CODE_OK)
                                            Debug-Output "OEMNXPTC.INF: ShellCode
error: cannot write default subnet mask error."
                                            goto ShellCodeError
                                        endif
                                    endif
                                endif
                                ifstr(i) $(DefaultGateway) != ""
                                    GetRegValue $(TCPIPKey) "DefaultGateway"
GatewayInfo
                                    set Gateway = *($(GatewayInfo), 4)
                                    ifstr(i) $(Gateway) == ""
                                        set NewValueList = {{DefaultGateway,$
(NoTitle),$(!REG_VT_MULTI_SZ),{$(DefaultGateway)}}}
                                        Shell  $(UtilityInf), AddValueList, $
(TCPIPKey), $(NewValueList)
                                        ifint $($ShellCode) != $(!SHELL_CODE_OK)
                                            Debug-Output "OEMNXPTC.INF: ShellCode
error: cannot write default subnet mask error."
                                            goto ShellCodeError
                                        endif
                                    endif
                                endif
```

```
                        endif
                    endif
                endif
                CloseRegKey $(TCPIPKey)
            endif
            CloseRegKey $(ServiceParamKey)
            set-add CardCount = $(CardCount),1
        Endif
    EndForListDo
    ifstr(i) $(CardList) != ""
        ifstr(i) $(!NTN_InstallMode) == bind
            set FLibraryErrCtl = 1
            LibraryProcedure ResultList, $(!TCPCFG_HANDLE), TcpCfgCheck
            set FLibraryErrCtl = 0
            Set Result = *($(ResultList),1)
            ifint $(Result) == 0
                Debug-Output "OEMNXPTC.INF:  reconfiguration not required"
                set CommonStatus = STATUS_USERCANCEL
                goto checkBootp
            else
                goto configureTcp
            endif
checkBootp =+
            OpenRegKey $(!REG_H_LOCAL) "" "System\Setup\DHCPRelay" $
(MAXIMUM_ALLOWED) KeyRelay
            set DHCP_UNATTENDED = ""
            set DHCP_Temp = "NO"
            ifstr(i) $(KeyRelay) != ""
                GetRegValue $(KeyTcpip) "GuiUnattended" DHCP_Temp
                set DHCP_UNATTENDED = *($(DHCP_Temp),4)
                CloseRegKey $(KeyRelay)
                OpenRegKey $(!REG_H_LOCAL) "" "System\Setup" $(MAXIMUM_ALLOWED)
KeySetup
                DeleteRegTree $(KeySetup) "DHCPRelay"
                CloseRegKey $(KeySetup)
            endif
            Debug-Output "DHCP_UNATTENDED="$(DHCP_UNATTENDED)
            Ifstr(i) $(DHCP_UNATTENDED) != YES
                set FLibraryErrCtl = 1
                LibraryProcedure ResultList, $(!TCPCFG_HANDLE),
TcpBootRelayCfgCheck
                set FLibraryErrCtl = 0
                Set Result = *($(ResultList),1)
                ifint $(Result) == 0
                    Debug-Output "OEMNXPTC.INF:  bootp configuration not
required"
                    set CommonStatus = STATUS_USERCANCEL
                    goto end
                endif
            endif
        endif
configureTcp =+
        set FLibraryErrCtl = 1
        OpenRegKey $(!REG_H_LOCAL) "" "System\Setup\Tcpip" $(MAXIMUM_ALLOWED)
KeyTcpip
        set Tcpip_GUI_UNATTENDED = "NO"
        set Tcpip_UNATTENDED = ""
        set Tcpip_UNATTENDED_SECTION = ""
        set Tcpip_Temp = ""
```

```
            ifstr(i) $(KeyTcpip) != ""
                GetRegValue $(KeyTcpip) "GuiUnattended" Tcpip_Temp
                set Tcpip_GUI_UNATTENDED = *($(Tcpip_Temp),4)
                GetRegValue $(KeyTcpip) "Unattended" Tcpip_Temp
                set Tcpip_UNATTENDED = *($(Tcpip_Temp),4)
                GetRegValue $(KeyTcpip) "UnattendedSection" Tcpip_Temp
                set Tcpip_UNATTENDED_SECTION = *($(Tcpip_Temp), 4)
                CloseRegKey $(KeyTcpip)
                OpenRegKey $(!REG_H_LOCAL) "" "System\Setup" $(MAXIMUM_ALLOWED)
KeySetup
                DeleteRegTree $(KeySetup) "Tcpip"
                CloseRegKey $(KeySetup)
            endif
            Debug-Output "Tcpip_GUI_UNATTENDED="$
(Tcpip_GUI_UNATTENDED)"Tcpip_UNATTENDED="$
(Tcpip_UNATTENDED)"Tcpip_UNATTENDED_SECTION="$(Tcpip_UNATTENDED_SECTION)
            LibraryProcedure ResultList, $(!TCPCFG_HANDLE), CPlTcpip, $(!STF_HWND),
$(CardList), $(!STF_COMPUTERNAME), "", $(!STF_PRODUCT), $(Tcpip_GUI_UNATTENDED), $
(Tcpip_UNATTENDED), $(Tcpip_UNATTENDED_SECTION)
            set FLibraryErrCtl = 0
            Set Result = *($(ResultList),1)
            ifint $(Result) == 2
                set CommonStatus = STATUS_REBOOT
            else
                set CommonStatus = STATUS_USERCANCEL
            endif
        else
            set CommonStatus = STATUS_USERCANCEL
        endif
    else
        Shell $(UtilityInf),RegistryErrorString,CANNOT_CONFIGURE_SOFTWARE
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "OEMNXPTC.INF: ShellCode error: cannot get an error
string."
            goto ShellCodeError
        endif
        set Error = $($R0)
        set from = end
        set to = end
        goto nonfatalinfo
    endif
        goto end
bindingadapter =+
    ifstr(i) $(Option) == "TC"
        ForListDo {"Browser","NetLogon"}
            OpenRegKey $(!REG_H_LOCAL) "" "System\CurrentControlSet\Services\"$($)
$(MAXIMUM_ALLOWED) ServicesKey
            ifstr(i) $(ServicesKey) != ""
                GetRegValue $(ServicesKey) "DependOnService" ServicesList
                debug-output "DependOnService:"$(ServicesList)
                set ServiceValues = *($(ServicesList),4)
                debug-output "DependOnService:"$(ServiceValues)
                ifstr(i) $(ServiceValues) == {}
                    Set ServiceValues = {"LmHosts"}
                    LibraryProcedure Result, $(!LIBHANDLE),
SetupChangeServiceConfig, $($), $(SERVICE_NO_CHANGE), $(SERVICE_NO_CHANGE), $
(SERVICE_NO_CHANGE), "", "", $(ServiceValues), "", "", ""
                else-ifstr(i) $(ServiceValues) == ""
                    Set ServiceValues = {"LmHosts"}
```

```
                    LibraryProcedure Result, $(!LIBHANDLE),
SetupChangeServiceConfig, $($), $(SERVICE_NO_CHANGE), $(SERVICE_NO_CHANGE), $
(SERVICE_NO_CHANGE), "", "", $(ServiceValues), "", "", ""
                else-ifcontains(i) "LMHOSTS" in $(ServiceValues)
                else
                    Set ServiceValues = >($(ServiceValues), "LmHosts")
                    LibraryProcedure Result, $(!LIBHANDLE),
SetupChangeServiceConfig, $($), $(SERVICE_NO_CHANGE), $(SERVICE_NO_CHANGE), $
(SERVICE_NO_CHANGE), "", "", $(ServiceValues), "", "", ""
                endif
                CloseRegKey $(ServicesKey)
            endif
        EndForListDo
        debug-output "binding for NetBT"
        OpenRegKey $(!REG_H_LOCAL) "" $(NetBTLinkageKeyName) $(MAXIMUM_ALLOWED)
LinkageKey
        GetRegValue $(LinkageKey) "Bind" BindList
        set CombineBindInfo = *($(BindList),4)
        CloseRegKey $(LinkageKey)
        Shell "nbinfo.inf" CheckMixRpcProtocol "NetBT" 5 "tcpip" "ncacn_nb_tcp"
"rpcltccm.dll" "RpcLtScm.Dll"
        OpenRegKey $(!REG_H_LOCAL) "" $(NetBTLinkageKeyName)"\Disabled" $
(MAXIMUM_ALLOWED) LinkageKey
        GetRegValue $(LinkageKey) "Bind" BindList
        ForListDo *($(BindList),4)
            set CombineBindInfo = >($(CombineBindInfo), $($))
        EndForListDo
        CloseRegKey $(LinkageKey)
        OpenRegKey $(!REG_H_LOCAL) "" $(AdaptersKeyName) $(MAXIMUM_ALLOWED)
AdaptersKey
        EnumRegKey $(AdaptersKey) AdaptersListInfo
        set AdaptersList = {}
        ForListDo $(AdaptersListInfo)
            ifstr(i) $(AdaptersList) == {}
                set AdaptersList = {*($($),1)}
            else
                set AdaptersList = >($(AdaptersList),*($($),1))
            endif
        EndForListDo
        set CardList = {}
        set CreateCardList = {}
        ForListDo $(CombineBindInfo)
            Split-String $($), "\", BindInfo
            QueryListSize BindListSize $(BindInfo)
            set CardName = *($(BindInfo),$(BindListSize))
            ifstr(i) $(CardList) == {}
                set CardList = {$(CardName)}
            else
                set CardList = >($(CardList),$(CardName))
            endif
            ifContains(i) $(CardName) not-in $(AdaptersList)
                ifstr(i) $(CreateCardList) == {}
                    set CreateCardList = {$(CardName)}
                else
                    set CreateCardList = >($(CreateCardList),$(CardName))
                endif
            endif
        EndForListDo
        set DeleteCardList = {}
```

```
        ForListDo $(AdaptersList)
            ifcontains(i) $($) not-in $(CardList)
                ifstr(i) $(DeleteCardList) == {}
                    set DeleteCardList = {$($)}
                else
                    set DeleteCardList = >($(DeleteCardList),$($))
                endif
            endif
        EndForListDo
        ForListDo $(CreateCardList)
            CreateRegKey $(AdaptersKey) {$($),$(NoTitle),GenericClass} "" $
(MAXIMUM_ALLOWED) "" CardKey
            CloseRegKey $(CardKey)
        EndForListDo
        ForListDo $(DeleteCardList)
            DeleteRegKey $(AdaptersKey) $($)
        EndForListDo
        CloseRegKey $(AdaptersKey)
        debug-output "binding for TC"
        goto configureadapter
    endif
    goto successful
removeadapter = +
    Shell $(UtilityInf) RemoveDependentComponents Microsoft TCPIP
    Shell $(UtilityInf) RemoveRpcProtocol "ncadg_ip_udp"
    Shell $(UtilityInf) RemoveRpcProtocol "ncacn_ip_tcp"
    Shell $(UtilityInf) RemoveRpcProtocol "ncacn_nb_tcp"
    Shell $(UtilityInf) RemoveServiceProvider $(ProviderName)
    Shell $(UtilityInf), RemoveNameSpaceProvider, $(ProductProviderGUID)
    ifint $($ShellCode) != $(!SHELL_CODE_OK)
        Debug-Output "OEMNXPTC.INF: ShellCode error"
        goto ShellCodeError
    endif
    set RegistryErrorIndex = $($R0)
    Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
        goto fatalregistry
    endif
    ForListDo  $(OEM_ABANDON_SOFTWARE)
        ifstr(i) $($) == "Tcpip"
            OpenRegKey $(!REG_H_LOCAL) "" $(LinkageKeyName) $(MAXIMUM_ALLOWED)
LinkageKey
            GetRegValue $(LinkageKey) "Bind" BindList
            ForListDo *($(BindList),4)
                Split-String $($), "\", BindInfo
                QueryListSize BindListSize $(BindInfo)
                set CardName = *($(BindInfo),$(BindListSize))
                OpenRegKey $(!REG_H_LOCAL) "" $(!NTN_ServiceBase)"\"*($(BindInfo),$
(BindListSize))"\Parameters" $(MAXIMUM_ALLOWED) CardParamKey
                ifstr(i) $(CardParamKey) != ""
                    DeleteRegTree $(CardParamKey) "Tcpip"
                    CloseRegKey $(CardParamKey)
                endif
            EndForListDo
            Shell $(UtilityInf), RemoveWinsockInfo, $(ProductTCName)
            set RegistryErrorIndex = $($R0)
            Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
                Debug-Output "OEMNXPTC.INF: Registry error: remove Winsock Info."
            Endif
        endif
```

```
        debug-output "Remove component: "$($)
        ifstr(i) $($) == "Streams"
            Shell $(UtilityInf), RemoveStreams
        else
            Shell $(UtilityInf), RemoveSoftwareComponent, $(Manufacturer), $($)
        endif
    EndForListDo
    ForListDo  $(OEM_ABANDON_OPTIONS)
        debug-output "Remove component: "$($)
        Ifstr(i) $($) == $(ProductLOOPName)
            Set UseSvcctrl = "NO"
        Else
            Set UseSvcctrl = "YES"
        Endif
        ifstr(i) $($) == $(ProductLMHOSTSName)
            debug-output "Remove "$($)"'s LMHOSTS dependency"
            ForListDo {"Browser","NetLogon"}
                OpenRegKey $(!REG_H_LOCAL) "" "System\CurrentControlSet\Services\"$
($) $(MAXIMUM_ALLOWED) ServicesKey
                ifstr(i) $(ServicesKey) != ""
                    GetRegValue $(ServicesKey) "DependOnService" ServicesList
                    debug-output "DependOnService List:"$(ServicesList)
                    set ServiceValues = *($(ServicesList),4)
                    debug-output "ServiceValues: "$(ServiceValues)
                    ifcontains(i) "LMHOSTS" in $(ServiceValues)
                        set NewServiceValues = {}
                        ForListDo $(ServiceValues)
                            ifstr(i) $($) != "LmHosts"
                                set NewServiceValues = >($(NewServiceValues), $($))
                            endif
                        EndForListDo
                        debug-output "Final dependency list for "$($)": "$
(NewServiceValues)
                        LibraryProcedure Result, $(!LIBHANDLE),
SetupChangeServiceConfig, $($), $(SERVICE_NO_CHANGE), $(SERVICE_NO_CHANGE), $
(SERVICE_NO_CHANGE), "", "", $(NewServiceValues), "", "", ""
                    else
                    endif
                    CloseRegKey $(ServicesKey)
                endif
            EndForListDo
        endif
        Shell $(UtilityInf), RemoveService, $($), $(UseSvcctrl)
    EndForListDo
    goto end
UpgradeSoftware = +
    OpenRegKey $(!REG_H_LOCAL) "" "SYSTEM\CurrentControlSet\Services\NetBT" $
(MAXIMUM_ALLOWED) NBTParent
    ifstr(i) $(NBTParent) != ""
        DeleteRegKey $(NBTParent) "Performance"
        CloseRegKey $(NBTParent)
    endif
    OpenRegKey $(!REG_H_LOCAL) "" $(OldProductKeyName) $(MAXIMUM_ALLOWED)
KeyProduct
    Ifstr(i) $(KeyProduct) == $(KeyNull)
        goto end
    endif
    GetRegValue $(KeyProduct) "DefaultGateway" DefaultGatewayList
    set DefaultGatewayValue = *($(DefaultGatewayList),4)
```

```
    Ifstr(i) $(DefaultGatewayValue) != $(KeyNull)
        install "Install-Update"
        ifstr(i) $(STF_INSTALL_OUTCOME) != STF_SUCCESS
            goto fatal
        endif
        CloseRegKey $(KeyProduct)
        set FLibraryErrCtl = 1
        ifstr(i) $(!UpgradeEnableDhcp) == ""
            LibraryProcedure ResultList $(!TCPCFG_HANDLE), UpgradeTcpip, $(!
STF_HWND) "NO" "YES"
        else
            LibraryProcedure ResultList $(!TCPCFG_HANDLE), UpgradeTcpip, $(!
STF_HWND) "YES" $(!UpgradeEnableDhcp)
        endif
        set FLibraryErrCtl = 0
        set UserWantsDHCP = *($(ResultList),2)
        OpenRegKey $(!REG_H_LOCAL) "" "System\CurrentControlSet\Services\Tcpip\
Parameters" $(MAXIMUM_ALLOWED) TcpipParamKey
        ifstr(i) $(TcpipParamKey) != ""
            GetRegValue $(TcpipParamKey) "DNSLookupOrder" LookupInfo
            debug-output "LookupOrder:"$(LookupInfo)
            GetRegValue $(TcpipParamKey) "Domain" DomainInfo
            debug-output "Domain:"$(DomainInfo)
            GetRegValue $(TcpipParamKey) "Hostname" HostnameInfo
            debug-output "Hostname:"$(HostnameInfo)
            GetRegValue $(TcpipParamKey) "NameServer" NameServerInfo
            debug-output "NameServer:"$(NameServerInfo)
            GetRegValue $(TcpipParamKey) "SearchList" SearchListInfo
            debug-output "SearchListInfo:"$(SearchListInfo)
            CloseRegKey $(TcpipParamKey)
        endif
        OpenRegKey $(!REG_H_LOCAL) "" "System\CurrentControlSet\Services\Nbt\
Parameters" $(MAXIMUM_ALLOWED) NbtParamKey
        ifstr(i) $(NbtParamKey) != ""
            GetRegValue $(NbtParamKey) "ScopeID" ScopeIDInfo
            debug-output "ScopeID:"$(ScopeID)
            CloseRegKey $(NbtParamKey)
        endif
        set OEM_ABANDON_OPTIONS = {+
            "TcpipSys",+
            "NbtSys",+
            "Telnet",+
            "LMHOSTS",+
            "Telnetsys",+
            "Loop"}
        set OEM_ABANDON_SOFTWARE = {+
            "Tcpip",+
            "Nbt",+
            "Streams"}
        Shell $(UtilityInf) RemoveRpcProtocol "ncacn_ip_tcp"
        Ifstr(i) $($0) != NO_ERROR
            Debug-Output "OEMNXPTC.INF: ERROR deleting RPC protocol data"
        Endif
        Shell $(UtilityInf) RemoveRpcProtocol "ncacn_nb_tcp"
        Ifstr(i) $($0) != NO_ERROR
            Debug-Output "OEMNXPTC.INF: ERROR deleting RPC protocol data (2)"
        Endif
        ForListDo  $(OEM_ABANDON_SOFTWARE)
            debug-output "Remove component: "$($)
```

```
            ifstr(i) $($) == "Streams"
                Shell $(UtilityInf), RemoveStreams
            else
                Shell $(UtilityInf), RemoveSoftwareComponent, $(Manufacturer), $($)
            endif
            ifstr(i) $($) == "Tcpip"
                Shell $(UtilityInf), RemoveWinsockInfo, $($)
                set RegistryErrorIndex = $($R0)
                Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
                    Debug-Output "OEMNXPTC.INF: Registry error: remove Winsock
Info."
                Endif
            endif
        EndForListDo
        ForListDo  $(OEM_ABANDON_OPTIONS)
            debug-output "Remove component: "$($)
            Ifstr(i) $($) == "loop"
                Set UseSvcctrl = "NO"
            Else
                Set UseSvcctrl = "YES"
            Endif
            ifstr(i) $($) == "lmhosts"
                debug-output "Remove "$($)"'s LMHOSTS dependency"
                ForListDo {"Browser","NetLogon"}
                    OpenRegKey $(!REG_H_LOCAL) "" "System\CurrentControlSet\
Services\"$($) $(MAXIMUM_ALLOWED) ServicesKey
                    ifstr(i) $(ServicesKey) != ""
                        GetRegValue $(ServicesKey) "DependOnService" ServicesList
                        debug-output "DependOnService List:"$(ServicesList)
                        set ServiceValues = *($(ServicesList),4)
                        debug-output "ServiceValues: "$(ServiceValues)
                        ifcontains(i) "LMHOSTS" in $(ServiceValues)
                            set NewServiceValues = {}
                            ForListDo $(ServiceValues)
                                ifstr(i) $($) != "LmHosts"
                                    set NewServiceValues = >($(NewServiceValues), $
($))
                                endif
                            EndForListDo
                            debug-output "Final dependency list for "$($)": "$
(NewServiceValues)
                            LibraryProcedure Result, $(!LIBHANDLE),
SetupChangeServiceConfig, $($), $(SERVICE_NO_CHANGE), $(SERVICE_NO_CHANGE), $
(SERVICE_NO_CHANGE), "", "", $(NewServiceValues), "", "", ""
                        else
                        endif
                        CloseRegKey $(ServicesKey)
                    endif
                EndForListDo
            endif
            Shell $(UtilityInf), RemoveService, $($), $(UseSvcctrl)
        EndForListDo
        set OldSTFInstallMode = $(!STF_INSTALL_MODE)
        set !STF_INSTALL_MODE = EXPRESS
        set OldInstallMode = $(!NTN_InstallMode)
        set !NTN_InstallMode = install
        Shell "" "InstallOption" $(!STF_LANGUAGE) "TC" $(!STF_SRCDIR) "NO" "NO"
"NO"
        set !STF_INSTALL_MODE = $(OldSTFInstallMode)
```

```
        set !NTN_InstallMode = $(OldInstallMode)
        OpenRegKey $(!REG_H_LOCAL) "" $(ParametersKeyName) $(MAXIMUM_ALLOWED)
TcpipParamKey
        ifstr(i) $(TcpipParamKey) != ""
            set NewValueList = {$(LookupInfo),$(DomainInfo),$(HostnameInfo),$
(NameServerInfo),$(SearchListInfo)}
            Shell  $(UtilityInf), AddValueList, $(TcpipParamKey), $(NewValueList)
            SetRegValue $(TcpipParamKey) {"EnableDHCP",$(NoTitle),$(!REG_VT_DWORD),
$(UserWantsDHCP)}
            CloseRegKey $(TcpipParamKey)
        endif
        OpenRegKey $(!REG_H_LOCAL) "" $(NetBTParamKeyName) $(MAXIMUM_ALLOWED)
NetbtParamKey
        ifstr(i) $(NetbtParamKey) != ""
            Shell  $(UtilityInf), AddValueList, $(NetbtParamKey), {$(ScopeIDInfo)}
            CloseRegKey $(NetbtParamKey)
        endif
    else
        install "Install-Update"
        ifstr(i) $(STF_INSTALL_OUTCOME) != STF_SUCCESS
            goto fatal
        endif
        CloseRegKey $(KeyProduct)
        Shell $(UtilityInf) UpdateWinsockMappings $(ProductTCName) $
(ProductTCWshDllPath)
        ifstr(i) $($R0) != NO_ERROR
            goto fatal
        endif
        OpenRegKey $(!REG_H_LOCAL) "" "Software\Microsoft\NetBT\CurrentControlSet\
NetRules" $(MAXIMUM_ALLOWED) NetRulesKey
        ifstr(i) $(NetRulesKey) != ""
            SetRegValue $(NetRulesKey) {"use",$(NoTitle),$(!REG_VT_SZ),"transport
yes yes"}
            CloseRegKey $(NetRulesKey)
        endif
        Shell $(UtilityInf) RemoveNetworkProvider $(ProviderName)
        OpenRegKey $(!REG_H_LOCAL) "" "System\CurrentControlSet\Services\DHCP\
Parameters\Options" $(MAXIMUM_ALLOWED) KeyOptions
        ifstr(i) $(KeyOptions) != ""
            OpenRegKey $(KeyOptions) "" "1" $(MAXIMUM_ALLOWED) Key1
            ifstr(i) $(Key1) == ""
                CreateRegKey $(KeyOptions) {"1",$(NoTitle),GenericClass} "" $
(MAXIMUM_ALLOWED) "" Key1
                Shell $(UtilityInf) AddValueList, $(Key1),+
                    {{"RegLocation", $(NoTitle), $(!REG_VT_SZ), "System\
CurrentControlSet\Services\?\Parameters\Tcpip\DhcpSubnetMaskOpt"},+
                    {"KeyType", $(NoTitle), $(!REG_VT_DWORD), $(!REG_VT_MULTI_SZ)}}
            endif
            CloseRegKey $(Key1)
            OpenRegKey $(KeyOptions) "" "15" $(MAXIMUM_ALLOWED) Key15
            ifstr(i) $(Key15) == ""
                CreateRegKey $(KeyOptions) {"15",$(NoTitle),GenericClass} "" $
(MAXIMUM_ALLOWED) "" Key15
                Shell $(UtilityInf) AddValueList, $(Key15),+
                    {{"RegLocation", $(NoTitle), $(!REG_VT_SZ), "System\
CurrentControlSet\Services\Tcpip\Parameters\DhcpDomain"},+
                    {"KeyType", $(NoTitle), $(!REG_VT_DWORD), $(!REG_VT_SZ)}}
            endif
            CloseRegKey $(Key15)
```

```
        endif
        Shell $(UtilityInf), RemoveNameSpaceProvider, $(ProductProviderGUID)
        Shell $(UtilityInf), AddNameSpaceProvider, +
                $(ProviderDisplayName), +
                $(Winsock20Provider), +
                $(ProductProviderNameSpaces), +
                FALSE, +
                $(ProductProviderGUID)
        OpenRegKey $(!REG_H_LOCAL) "" "System\CurrentControlSet\Services\Tcpip\
ServiceProvider" $(MAXIMUM_ALLOWED) ProviderKey
        ifstr(i) $(ProviderKey) == ""
            Shell $(UtilityInf), AddServiceProvider, $(ProviderName), +
                $(Winsock11Provider), 8
            set RegistryErrorIndex = $($R0)
            Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
                EndWait
                Debug-Output "OEMNXPTC.INF: Registry error: add software
components"
                goto fatalregistry
            endif
            OpenRegKey $(!REG_H_LOCAL) "" "System\CurrentControlSet\Services\Tcpip"
$(MAXIMUM_ALLOWED) TcpSvcKeyHandle
            ifstr(i) $(TcpSvcKeyHandle) != ""
                OpenRegKey $(!REG_H_LOCAL) "" "System\CurrentControlSet\Services\
Tcpip\ServiceProvider" $(MAXIMUM_ALLOWED) KeyNetworkProvider
                ifstr(i) $(KeyNetworkProvider) == ""
                   CreateRegKey $(TcpSvcKeyHandle) {"ServiceProvider",$
(NoTitle),GenericClass} "" +
                         $(MAXIMUM_ALLOWED) "" KeyNetworkProvider
                endif
                set NewValueList = {{DnsPriority,$(NoTitle),$(!REG_VT_DWORD),2000},
+
                           {Name,$(NoTitle),$(!REG_SZ),$(ProviderDisplayName)},+
                           {HostsPriority,$(NoTitle),$(!REG_VT_DWORD),500},+
                           {LocalPriority,$(NoTitle),$(!REG_VT_DWORD),499},+
                           {ProviderPath,$(NoTitle),$(!REG_VT_EXPAND_SZ),$
(Winsock11Provider)},+
                           {NetbtPriority,$(NoTitle),$(!REG_VT_DWORD),2001}}
                Shell  $(UtilityInf), AddValueList, $(KeyNetworkProvider), $
(NewValueList)
                CloseRegKey $(KeyNetworkProvider)
                CloseRegKey $(TcpSvcKeyHandle)
            endif
        else
            CloseRegKey $(ProviderKey)
        endif
        OpenRegKey $(!REG_H_LOCAL) "" "System\CurrentControlSet\Services\Tcpip\
Parameters" $(MAXIMUM_ALLOWED) TcpParmKeyHandle
        ifstr(i) $(TcpParmKeyHandle) != ""
            CreateRegKey $(TcpParmKeyHandle) {"PersistentRoutes",$
(NoTitle),GenericClass} "" +
                  $(MAXIMUM_ALLOWED) "" KeyPersistentRoutes
            CloseRegKey $(KeyPersistentRoutes)
            CloseRegKey $(TcpParmKeyHandle)
        endif
    endif
    OpenRegKey $(!REG_H_LOCAL) "" "System\CurrentControlSet\Services\Tcpip" $
(MAXIMUM_ALLOWED) TcpSvcKeyHandle
    ifstr(i) $(TcpSvcKeyHandle) != $(KeyNull)
```

```
        SetRegValue $(TcpSvcKeyHandle) {Group,$(NoTitle),$(!REG_VT_SZ),"PNP_TDI"}
        CloseRegKey $(TcpSvcKeyHandle)
    endif
    OpenRegKey $(!REG_H_LOCAL) "" "System\CurrentControlSet\Services\NetBT" $
(MAXIMUM_ALLOWED) NetBTSvcKeyHandle
    ifstr(i) $(NetBTSvcKeyHandle) != $(KeyNull)
        SetRegValue $(NetBTSvcKeyHandle) {Group,$(NoTitle),$(!REG_VT_SZ),"PNP_TDI"}
        CloseRegKey $(NetBTSvcKeyHandle)
    endif
    OpenRegKey $(!REG_H_LOCAL) "" $(TCNetRuleKeyName) $(MAXIMUM_ALLOWED)
TCKeyNetRules
    Ifstr $(TCKeyNetRules) != $(KeyNull)
        SetRegValue $(TCKeyNetRules) {use,$(NoTitle),$(!REG_VT_SZ),$(NetRuleTCUse)}
        CloseRegKey $(TCKeyNetRules)
    endif
    OpenRegKey $(!REG_H_LOCAL) "" $(NetBTNetRuleKeyName) $(MAXIMUM_ALLOWED)
NetBTKeyNetRules
    Ifstr $(NetBTKeyNetRules) != $(KeyNull)
        SetRegValue $(NetBTKeyNetRules) {use,$(NoTitle),$(!REG_VT_SZ),$
(NetRuleNETBTUse)}
        CloseRegKey $(NetBTKeyNetRules)
    endif
    OpenRegKey $(!REG_H_LOCAL) "" $(ProductTCKeyName) $(MAXIMUM_ALLOWED)
TCKeyCurrentVersion
    Ifstr $(TCKeyCurrentVersion) != $(KeyNull)
        SetRegValue $(TCKeyCurrentVersion) {Description,$(NoTitle),$(!REG_VT_SZ),$
(ProductTCDescription)}
        SetRegValue $(TCKeyCurrentVersion) {OperationsSupport,$(NoTitle),$(!
REG_VT_DWORD),$(ProductOpSupport)}
        CloseRegKey $(TCKeyCurrentVersion)
    endif
    goto end
successful = +
    goto end
infomsg =+
   read-syms InfoDlg
   ui start "Warning"
   set CommonStatus = STATUS_USERCANCEL
   goto end
warning = +
    Shell $(subroutineinf) SetupMessage, $(!STF_LANGUAGE), "WARNING", $(Error)
    ifint $($ShellCode) != $(!SHELL_CODE_OK)
        goto ShellCodeError
    endif
    ifstr(i) $($R1) == "OK"
        goto $(to)
    else-ifstr(i) $($R1) == "CANCEL"
        goto $(from)
    else
        goto "end"
    endif
nonfatalinfo = +
    Set CommonStatus = STATUS_USERCANCEL
    Set Severity = STATUS
    goto nonfatalmsg
nonfatal = +
    Set Severity = NONFATAL
    goto nonfatalmsg
nonfatalmsg = +
```

```
    ifstr(i) $(Error) == ""
        Set Severity = NONFATAL
        Shell $(UtilityInf) RegistryErrorString "SETUP_FAIL"
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            goto ShellCodeError
        endif
        set Error = $($R0)
    endif
    Shell $(subroutineinf) SetupMessage, $(!STF_LANGUAGE), $(Severity), $(Error)
    ifint $($ShellCode) != $(!SHELL_CODE_OK)
        goto ShellCodeError
    endif
    ifstr(i) $($R1) == "OK"
        goto $(from)
    else
        goto "end"
    endif
fatalregistry = +
    Shell $(UtilityInf) RegistryErrorString $(RegistryErrorIndex)
    ifint $($ShellCode) != $(!SHELL_CODE_OK)
        goto ShellCodeError
    endif
    set Error = $($R0)
    goto fatal
fatal = +
    ifstr(i) $(Error) == ""
        Shell $(UtilityInf) RegistryErrorString "SETUP_FAIL"
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            goto ShellCodeError
        endif
        set Error = $($R0)
    endif
    Ifint $(BillboardVisible) != 0
        Shell "subroutn.inf" PopBillboard
        Set BillboardVisible = 0
    Endif
    Shell $(subroutineinf) SetupMessage, $(!STF_LANGUAGE), "FATAL", $(Error)
    ifint $($ShellCode) != $(!SHELL_CODE_OK)
        goto ShellCodeError
    endif
   goto setfailed
ShellCodeError = +
    set DlgType      = "MessageBox"
    set STF_MB_TITLE = $(ShellCodeErrorTitle)
    set STF_MB_TEXT  = $(ShellCodeErrorText)
    set STF_MB_TYPE  = 1
    set STF_MB_ICON  = 3
    set STF_MB_DEF   = 1
    ui start "Error Message"
    goto setfailed
setfailed = +
    set CommonStatus = STATUS_FAILED
    ifstr(i) $(OEM_ABANDON_ON) == TRUE
        set OEM_ABANDON_ON = FALSE
        goto removeadapter
    endif
    goto end
end = +
    freeLibrary $(!TCPCFG_HANDLE)
```

```
    goto term
term = +
    Return $(CommonStatus)
[CreateEventLog]
    read-syms GeneralConstants
    set NameOfService = $($0)
    set EventFile     = $($1)
    set TypeSupported = $($2)
    OpenRegKey $(!REG_H_LOCAL) "" "SYSTEM\CurrentControlSet\Services\EventLog\
System" $(MAXIMUM_ALLOWED) KeyEventLog
    Ifstr $(KeyEventLog) != ""
        OpenRegKey $(KeyEventLog) "" $(NameOfService) $(MAXIMUM_ALLOWED) KeyService
           ifstr(i) $(KeyService) == ""
               CreateRegKey $(KeyEventLog) {$(NameOfService),$
(NoTitle),GenericClass} "" $(MAXIMUM_ALLOWED) "" KeyService
           endif
           Ifstr $(KeyService) != ""
               SetRegValue $(KeyService) {EventMessageFile,$(NoTitle),$(!
REG_VT_EXPAND_SZ),$(EventFile)}
               SetRegValue $(KeyService) {TypesSupported,$(NoTitle),$(!
REG_VT_DWORD),$(TypeSupported)}
           endif
        Endif
    endif
CreateEventLogEnd = +
    return NO_ERROR
[GetFilesSize]
    set FileSizeList = >(>(>(^(Files-TC,3),^(Files-TCPIPEXE,3)),^(Files-
ETC,3)),^(Files-LMHOST,3))
    set TotalSize = 0
    ForListDo $(FileSizeList)
        ForListDo $($)
            Split-String $($) "=" SplitString
            set Size = *($(SplitString),3)
            set-add TotalSize = $(TotalSize) $(Size)
        EndForListDo
    EndForListDo
    set-div SizeInK = $(TotalSize) 1024
    return $(SizeInK)
[Install-Option]
    set STF_VITAL = ""
    ifstr(i) $(AddCopy) == "YES"
        AddSectionFilesToCopyList Files-TC $(SrcDir) $(!STF_WINDOWSSYSPATH)\drivers
        AddSectionFilesToCopyList Files-TCPIPEXE $(SrcDir) $(!STF_WINDOWSSYSPATH)
        AddSectionFilesToCopyList Files-ETC $(SrcDir) $(!STF_WINDOWSSYSPATH)\
drivers\etc
        AddSectionFilesToCopyList Files-LMHOST $(SrcDir) $(!STF_WINDOWSSYSPATH)\
drivers\etc
    endif
    ifstr(i) $(DoCopy) == "YES"
       set !STF_NCPA_FLUSH_COPYLIST = TRUE
       Debug-Output "OEMNXPTC.INF: *** CopyFilesInCopyList [2]"
       CopyFilesInCopyList
    endif
    ifstr(i) $(DoConfig) == "YES"
    endif
    Exit
[Install-Update]
   set STF_VITAL        = ""
```

```
   set STF_OVERWRITE    = "VERIFYSOURCEOLDER"
   AddSectionFilesToCopyList Files-TC $(SrcDir) $(!STF_WINDOWSSYSPATH)\drivers
   AddSectionFilesToCopyList Files-TCPIPEXE $(SrcDir) $(!STF_WINDOWSSYSPATH)
   AddSectionFilesToCopyList Files-LMHOST $(SrcDir) $(!STF_WINDOWSSYSPATH)\drivers\
etc
   AddSectionFilesToCopyList Files-ETC-Upgrade $(SrcDir) $(!STF_WINDOWSSYSPATH)\
drivers\etc
   Exit
[Source Media Descriptions]
    1 = "Windows NT Workstation CD-ROM" , TAGFILE = cdrom_w.40
[Signature]
    FileType = MICROSOFT_FILE
[GetSignature]
    read-syms Signature
    return $(FileType)
[ProductType]
STF_PRODUCT  = Winnt
STF_PLATFORM = I386
[Files-Inf]
2,    oemsetup.inf,      SIZE=1000, RENAME=$(!UG_Filename)
[Files-ETC-Upgrade]
1,HOSTS , SIZE=999, RENAME=HOSTS.SAM
1,SERVICES , SIZE=999, RENAME=SERVICES.SAM
[Files-ETC]
1,HOSTS , SIZE=999
1,NETWORKS , SIZE=999
1,PROTOCOL , SIZE=999
1,SERVICES , SIZE=999
[Files-LMHOST]
1,LMHOSTS.SAM , SIZE=999
[Files-LMHOSTOTHER]
99,LMHOSTS.SAM , SIZE=999
[Files-TC]
1,NETBT.SYS , SIZE=999
1,TCPIP.SYS , SIZE=999, OVERWRITE=OLDER
[Files-TCPIPEXE]
1, DHCPCSVC.DLL, SIZE=999
1, DHCPSAPI.DLL, SIZE=999
1, ICMP.DLL, SIZE=999
1, IPCONFIG.EXE, SIZE=999
1, NBTSTAT.EXE, SIZE=999
1, SNMPAPI.DLL, SIZE=999
1, TCPSVCS.EXE, SIZE=999
1, TRACERT.EXE, SIZE=999
1,ARP.EXE , SIZE=999
1,HOSTNAME.EXE , SIZE=999
1,INETMIB1.DLL , SIZE=999
1,IPINFO.INF , SIZE=999
1,LMHSVC.DLL , SIZE=999
1,NETSTAT.EXE , SIZE=999
1,PING.EXE , SIZE=999
1,ROUTE.EXE , SIZE=999
[LanguagesSupported]
    ENG
[OptionsTextENG]
    TC     = "TCP/IP"
[FileConstantsENG]
ProCaption   = "Instalator Windows NT"
ProCancel    = "Anuluj"
```

```
ProCancelMsg = "Sie Windows NT nie jest zainstalowana poprawnie. "+
               "Czy na pewno chcesz anulowa kopiowanie plik w?"
ProCancelCap = "Komunikat Instalatora sieci"
ProText1     = "Kopiowanie:"
ProText2     = "Do:"
NoNewComponents = "Brak nowych sk adnik w do instalacji."
FunctionTitle   = "Instalacja TCP/IP"
ProductNETBTDescription           = "Sterownik systemowy NetBT"
ProductNETBTDisplayName   = "Klient WINS (TCP/IP)"
ProductNETBTTitle         = "Klient WINS (TCP/IP)"
ProductTELNETDisplayName = "Telnet"
ProductTELNETTitle      = "Telnet Networking Support Environment"
ProductTELNETSYSDisplayName = "Telnetsys"
ProductTELNETSYSTitle     = "Sterownik Telnet"
ProductLMHOSTSDisplayName = "Pomocnik TCP/IP NetBIOS"
ProductLMHOSTSTitle     = "Pomocnik TCP/IP NetBIOS"
ProductDHCPDisplayName = "Klient DHCP"
ProductDHCPTitle     = "Klient DHCP"
ProductTCDisplayName = "TCP/IP"
ProductTCTitle     = "Protok  TCP/IP"
ProductTCDescription         = "Protok  TCP/IP jest domy lnym protoko em sieci
rozleg ej pozwalaj cym na komunikacj  z wieloma pod czonymi ze sob  sieciami."
ProviderDisplayName     = "TCP/IP"
LDAPProviderDisplayName  = "NTDS"
ShellCodeErrorTitle     = "B d: "$(FunctionTitle)
ShellCodeErrorText      = "B d kodu pow oki."
RASRebindError = "Instalator wykry  zainstalowan  Us ug  zdalnego dost pu (RAS).
Czy chcesz skonfigurowa  RAS do obs ugi TCP/IP?"
[DialogConstantsENG]
Help        = "Pomo&c"
Exit        = "Anuluj"
OK          = "OK"
HelpContext = ""
Continue    = "Kontynuuj"
Cancel      = "Anuluj"
[FileDependentDlgENG]
[InfoDlg]
STF_MB_TITLE = "Informacja"
DlgType = "MessageBox"
STF_MB_TEXT  = $(InfoMsgText)
STF_MB_TYPE  = 1
STF_MB_ICON  = 5
STF_MB_DEF   = 1
[DHCPDialog]
DHCPText = "Je li w twojej sieci znajduje si  serwer DHCP, protok  TCP/IP mo e
zosta  skonfigurowany do dynamicznego dostarczenia"+
                " adresu IP. Je li nie masz pewno ci, zapytaj administratora
sieci. Czy chcesz u ywa  DHCP?"
STF_MB_TITLE = "Instalacja TCP/IP"
DlgType = "MessageBox"
STF_MB_TEXT  = $(DHCPText)
STF_MB_TYPE  = 3
STF_MB_ICON  = 5
STF_MB_DEF   = 2
[InfoDlgCantConfigure]
InfoMsgText = "TCP/IP nie jest skonfigurowany z adn  kart  sieciow ."+
    " Je li w a nie zainstalowa e  j , powr  do okna dialogowego Sieci i kliknij"+
    "przycisk OK lub Zamknij. Wtedy b dziesz m g  skonfigurowa  karty sieciowe."
[WarningDlgENG]
```

```
STF_MB_TITLE = "Ostrze?enie Instalatora TCP/IP"
DlgType = "MessageBox"
STF_MB_TEXT  = $(Error)
STF_MB_TYPE  = 1
STF_MB_ICON  = 5
STF_MB_DEF   = 1
[Billboard1ENG]
Status = "Instalowanie TCP/IP i zwi?zanych z nim us?ug..."
```