Windows SECURE RING0 DEVELOPMENT

Agenda

Introduction

Introduction

Introduction

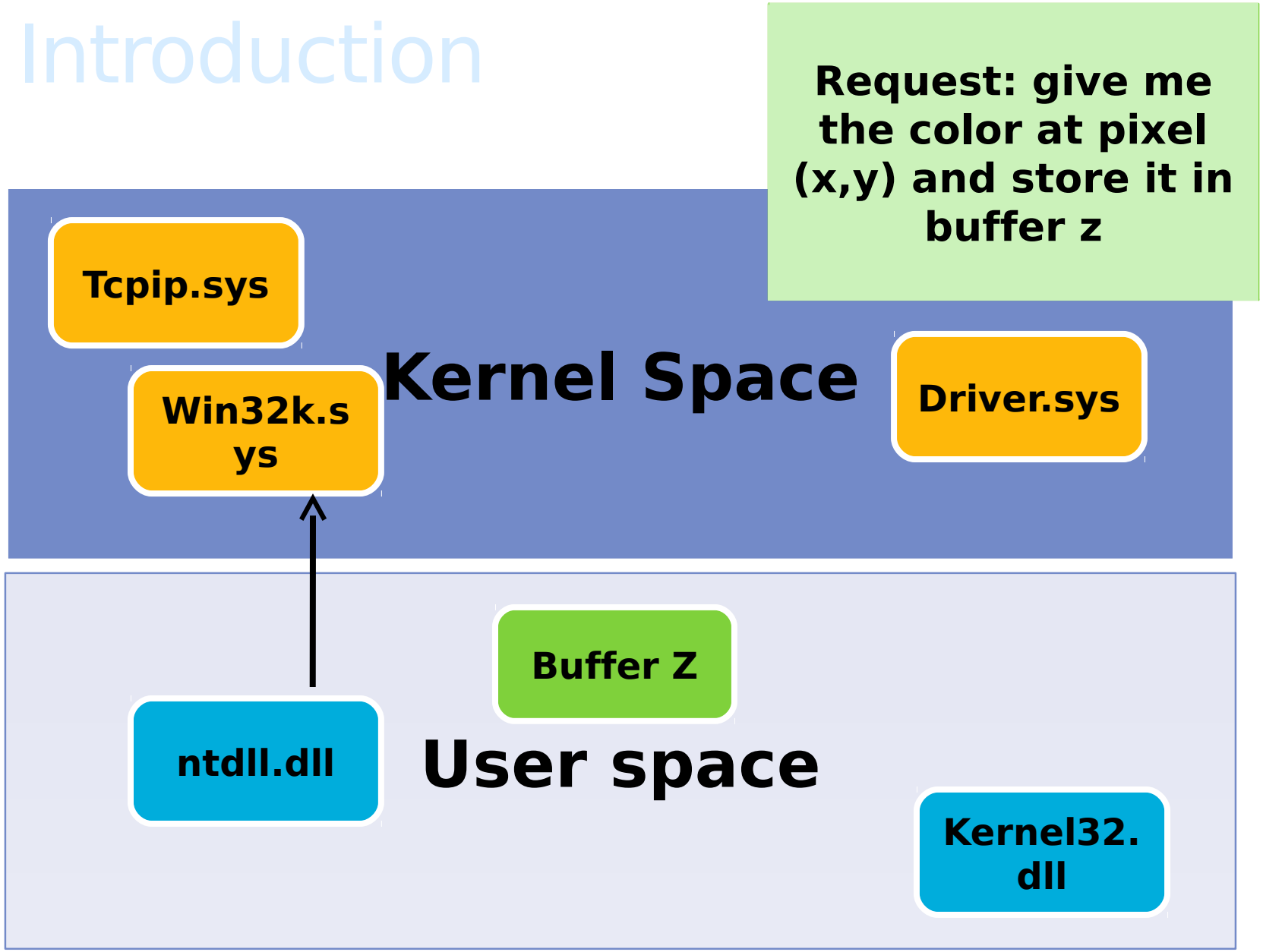
Introduction

	User Space	Kernel Space
User priv (ring3)		
Kernel priv (ring0)		

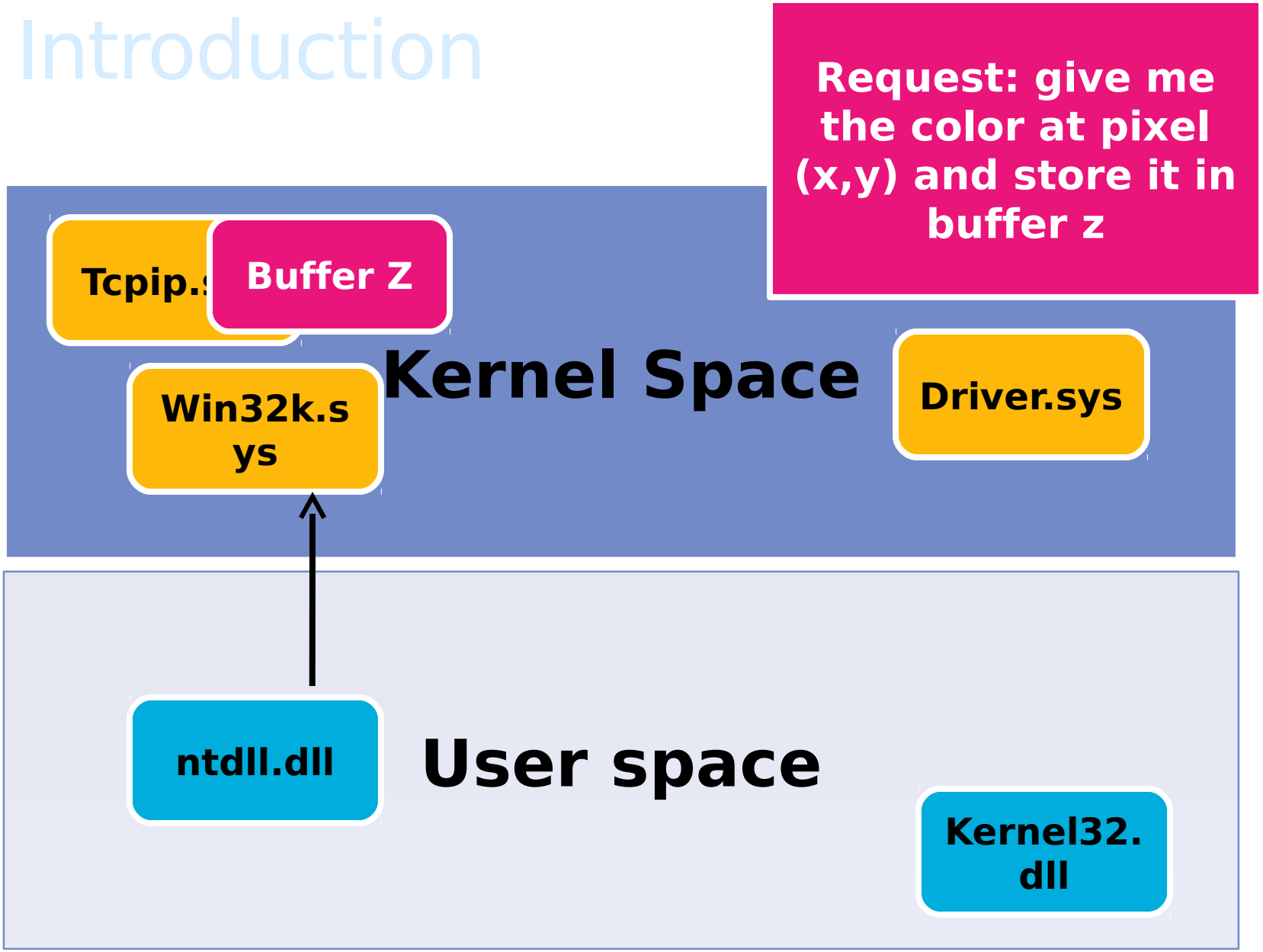
Introduction

Introduction

Introduction



Introduction



Introduction

Introduction

Common Mistakes

```
void foo(struct bar *x, USHORT len) {

USHORT blah;

   blah=get_actual_counter();
   blah+=len;

   new_entries =
ExAllocatePoolWithTag(   NonPagedPool,blah*sizeof(
),TAG_NAME);

   for (USHORT counter=0; counter<get_actual_count
counter++) {
      // Copy old entries
   }
   for (USHORT counter=0; counter<len;counter++)  {
      // Copy new entries
   }
```

blah can overflow

size can overflow

NULL deref

Memory corruption

Common Mistakes

```
void NtUser_giveme_foo(struct bar *x, USHORT len)


   for (USHORT counter=0; counter<len;counter++)
      // Copy entries to x
   }

}
```

No NULL checking on X

No ProbeForWrite on X

No length check

Common Mistakes

```
case IOCTL_FOO_STATS:  // METHOD_BUFFERED

    Stats = Irp->AssociatedIrp.SystemBuffer;
    Irp->IoStatus.Information = sizeof(*Stats);


    if (Stats->Version != FOO_STATS_VERSION) {
        Irp->IoStatus.Status = STATUS_INVALID_PARAMETER;
        break;
    }

    *Stats->data = *FooSt;
```

Usermode Data

Deref it without NULL checking

If attacker controls NULL page…

Common Mistakes

```
// lParam and wParam are untrusted DWORDs
LARGE_STRING str;
try {
    str.bAnsi = bAnsi;
    str.MaximumLength = wParam;

    if (!bAnsi) {
         str.MaximumLength *= sizeof(WCHAR);
    }
    str.Length = 0;
    str.Buffer = (LPBYTE)lParam;

    ProbeForWrite((PVOID)str.Buffer, str.MaximumLength,1);
} except (StubExceptionHandler(FALSE)) {
   MSGERROR(0);
}
[... later write into str.Buffer pointer based on wParam ...]
```

This can overflow and become zero

PFW will say Ok for a zero length

Later on we write wParam Tchars

Common Mistakes

```
// Attacker controls OutputBuffer and OutputBufferLength

 void IOCTL_foo_handler(...)
{

[...]

try {

  ProbeForWrite(OutputBuffer,OutputBufferLength,siz
  RtlCopyMemory(OutputBuffer,context->offset,contex

} except( FOO_EXCEPTION_FILTER(&status) ) {
}

[...]

}
```

Outputbufferlen gthcan be zero

OutputBuffer can point to kmode

PFW will say it is OK

We copy based on another size

Common Mistakes

```
// lParam and wParam are untrusted DWORDs

LARGE_STRING str;

try {
    str.bAnsi = bAnsi;
    str.MaximumLength = WORD_TRUNCATION(wParam);

    str.Length = 0;
    str.Buffer = (LPBYTE)lParam;
    ProbeForWrite((PVOID)str.Buffer,str.MaximumLength,1);

} except (StubExceptionHandler(FALSE)) {
    MSGERROR(0);
}

[... later write into str.Buffer pointer based on wParam ...]
```

This can be become zero

PFW will say Ok for a zero length

Later on we write wParam Tchars

Common Mistakes

```
// lParam and wParam are untrusted DWORDs

LARGE_STRING str;

    str.bAnsi = bAnsi;
    str.MaximumLength = (ULONG)wParam;

    str.Length = 0;
    str.Buffer = (LPBYTE)lParam;

    ProbeForWrite((PVOID)str.Buffer, str.MaximumLer

    *str.Buffer='\0';
```

We initialize LARG_STRING

PFW can throw an exception

Writting to user mode too!

Exception handling? bugcheck

Common Mistakes

```
// Attacker controls lParam
void win32k_foo_entry_point(...) {

 // lParam has already passed successfully the Prob
 my_struct = (PMY_STRUCT)lParam;
 if (my_struct ->lpData) {
    cbCapture = sizeof(MY_STRUCT) + my_struct->cbDa

    […]

  if (my_alloc=UserAllocPoolWithQuota(cbCapture,TAG

   {
   RtlCopyMemory(my_alloc,my_struct->lpData,my_struct->cbData);
   }
 }

}
```

my_struct is already validated

But not captured

First user mode fetch

Second one…

MSRC Cases

```
IGMPv3GroupRecord * GetGSIsInRecord(IN IGMPAddr *AddrPtr, IN
uint  *RecSize) {
    IGMPSrcAddr       *Src, *PrevSrc;
    IGMPv3GroupRecord *Rec;
    ushort             Count = 0;

for (Src=AddrPtr->iga_srclist; Src; Src=Src->isa_ne
        if (!IS_SOURCE_ALLOWED(AddrPtr, Src))
            continue;
        if (!Src->isa_csmarked)
            continue;
      Count++;
}

    // Allocate record
    Rec = CTEAllocMemN(RECORD_SIZE(Count,0), 'qICT');
    if (Rec == NULL) {
        *RecSize = 0;
        return NULL;
    }
```

Count initializaed

This can overflow 65535

Small allocation

Later on pool overflow

MSRC Cases

```
MESSAGECALL(OUTSTRING)
{
    LARGE_STRING str;

    BEGINRECV_MESSAGECALL(0);
    try {
        str.bAnsi = bAnsi;
        str.MaximumLength = (ULONG)wParam;
        if (!bAnsi) {
            str.MaximumLength *= sizeof(WCHAR);
        }
        str.Length = 0;
        str.Buffer = (PVOID)lParam;
        ProbeForWrite((PVOID)str.Buffer, str.MaximumLength,
sizeof(BYTE));
    } except (StubExceptionHandler(FALSE)) {
        MSGERROR(0);
    }
```

Truncation to 31 bits

Also an integer overflow

Length can be zero and PFW Bypass

Later on we had a RtlCopyMemory

MSRC Cases

```
try {
      if (RequestorMode != KernelMode ) {
      ProbeForWrite (OutputBuffer, OutputBufferLeng
(UCHAR));
      RtlCopyMemory(
      OutputBuffer,
      (PUCHAR)context+endpoint-
>Common.VcConnecting.RemoteSocketAddressOffset,
      endpoint->Common.VcConnecting.RemoteSocketAdd
      );

      *Information = endpoint->ContextLength;

  } except( AFD_EXCEPTION_FILTER (status) ) {
      ASSERT (NT_ERROR (status));
  }
```

Validate with supplied length

Actual usage with different one

If kmode addr and length zero

Kernel memory corruption

MSRC Cases

```
AlpcpIsReplyAllowed( __in PALPC_PORT PortObject,__in
PKALPC_MESSAGE Message )
{
    BOOLEAN Allowed;
    PALPC_COMMUNICATION_INFO CommunicationInfo;
    PALPC_PORT OwnerPort;

    ALPCASSERT_PTR(AlpcpIsLocked(Message) != FALSE,
    ALPCASSERT_PTR(AlpcpIsCanceled(Message) == FALS

    if (Message->PortQueue != PortObject) {
        if (Message->PortQueue == NULL) {
            OwnerPort = AlpcpGetOwnerPortMessage(Message);

            ALPCASSERT_PTR(OwnerPort != NULL, Message);

            CommunicationInfo = OwnerPort->CommunicationInfo;
```

This function can return NULL

ASSERT only,not present in free build

ReadAV from NULL page

MSRC Cases

```
      pcds = (PCOPYDATASTRUCT)lParam;
      if (pcds->lpData) {
          cbCapture = sizeof(COPYDATASTRUCT) + pcds-



      } else {
          cbCapture = sizeof(COPYDATASTRUCT);
      }

      if (cbCapture && (psms->pvCapture =
UserAllocPoolWithQuota (cbCapture, TAG_SMS_CAPTURE))

          if (pcds->lpData) {
             pcdsNew->lpData=(PVOID)((PBYTE)pcdsNew
sizeof(COPYDATASTRUCT));
            RtlCopyMemory(pcdsNew->lpData, pcds->lpData, pcds-
>cbData);
          }
```

First fetch from usermode

Allocation based on it

Second fetch from usermode

Copy based on it

Detect/Protect

Detect/Protect

Detect/Protect

Detect/Protect

Door-Points

Greetings

Questions

-
-
-