TOWARDS PROCESS MIGRATION IN WINDOWS 10

Master Computer Science – Cyber Security

Wietze D. Mulder
Radboud University Nijmegen

July 12, 2023

Radboud Universiteit

Agenda

1) What is process migration?

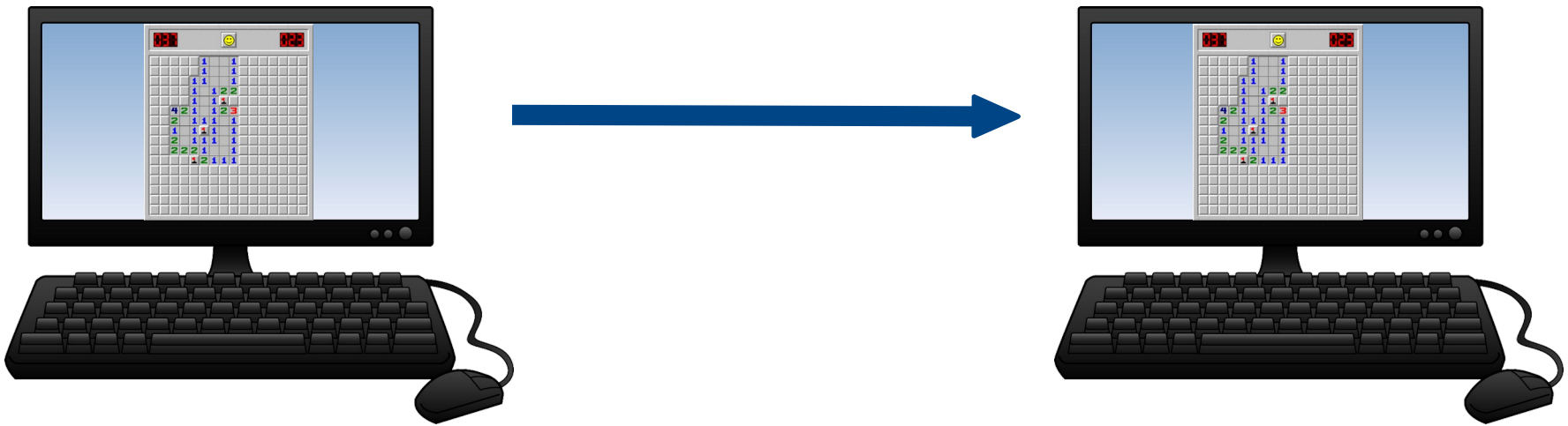
2) Research Question 1: Is process migration possible?

3) Research Question 2: Can we migrate malware?

4) Conclusions & Recommendations

5) Questions?

What is process migration?

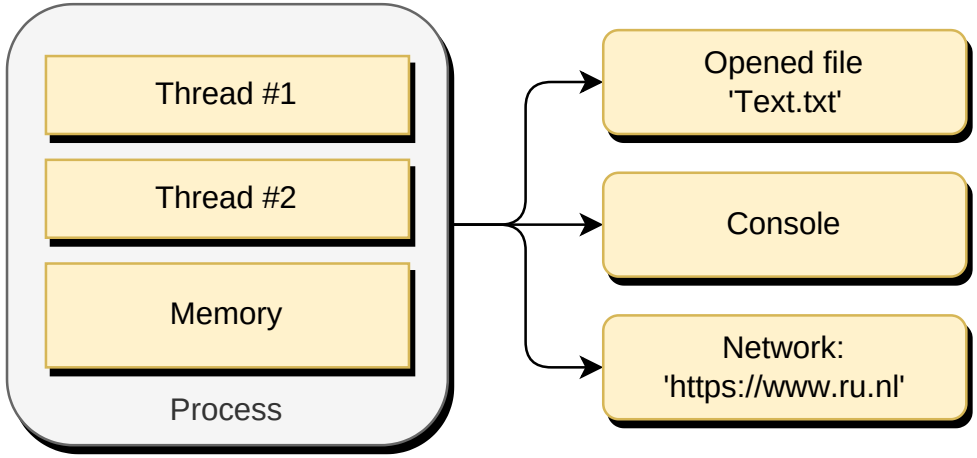


Process migration: background

What is a process?

What is a thread?

What is virtual memory?



Why process migration?

Why is process migration in Windows 10 relevant?

- Does not exist yet
- Already done in Linux

How can this be useful for malware analysis?

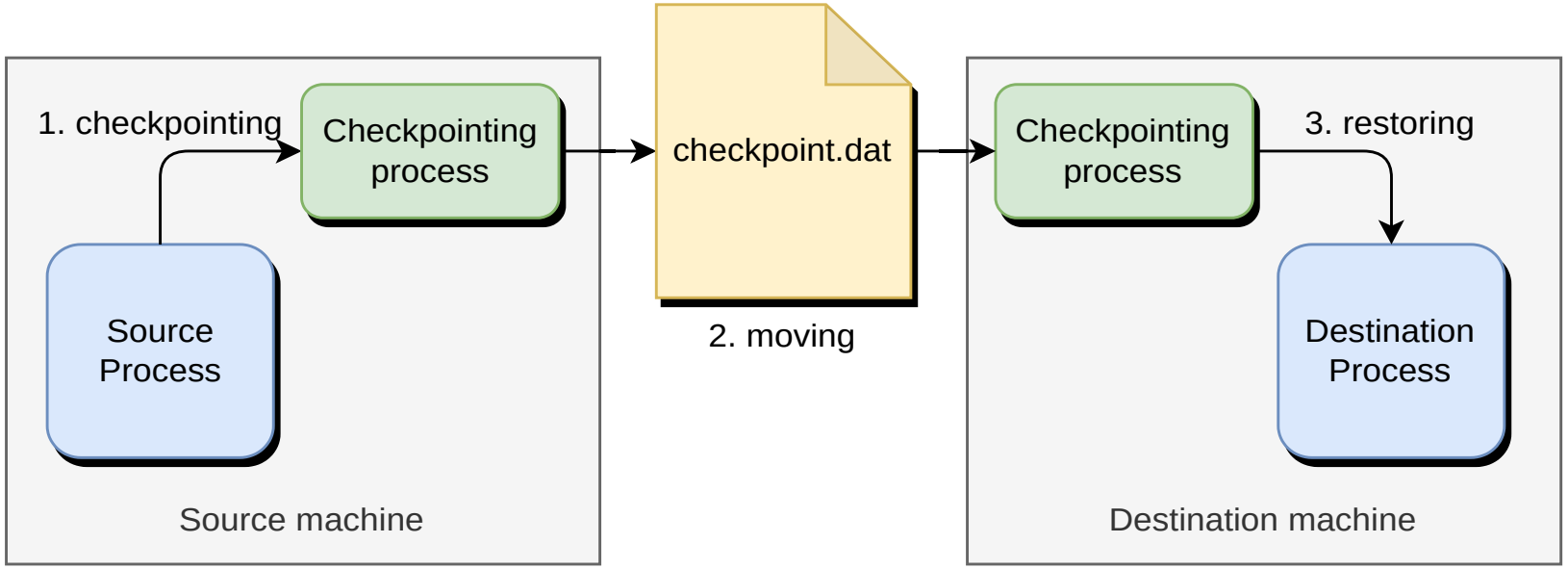
- 83% of malware in Windows

Process migration: Feasibility

Research Question 1:

Is it possible to migrate a running process from one Windows 10 machine to another machine?

Process migration:
Overview of the main approach used

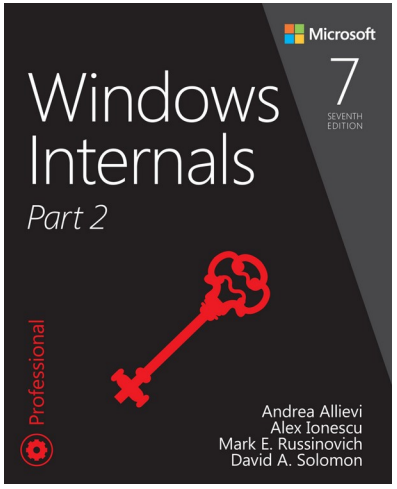


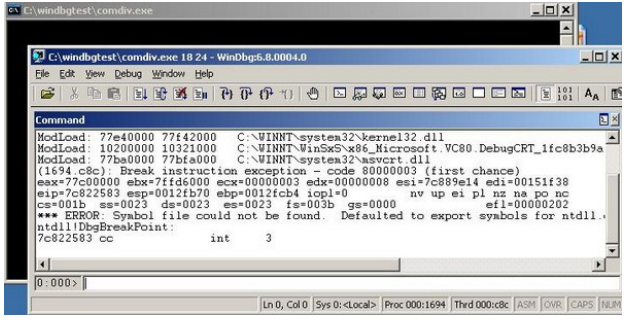
Process migration: Scope

- Windows 10 only (version 22H2).
- x86-64 bit architecture.
- No GUI applications.
- Only in user-space; we cannot modify kernel-space

Process migration:
A lot of Windows API calls

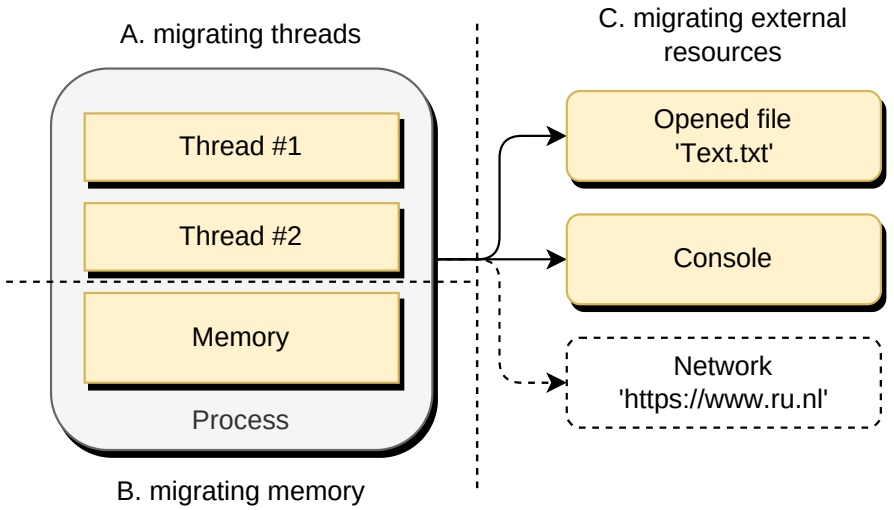
- Windows API calls are mostly documented
- Not everything however





Process migration: Chosen approach

A) Migrating threads
B) Migrating Memory
C) Migrating external resources



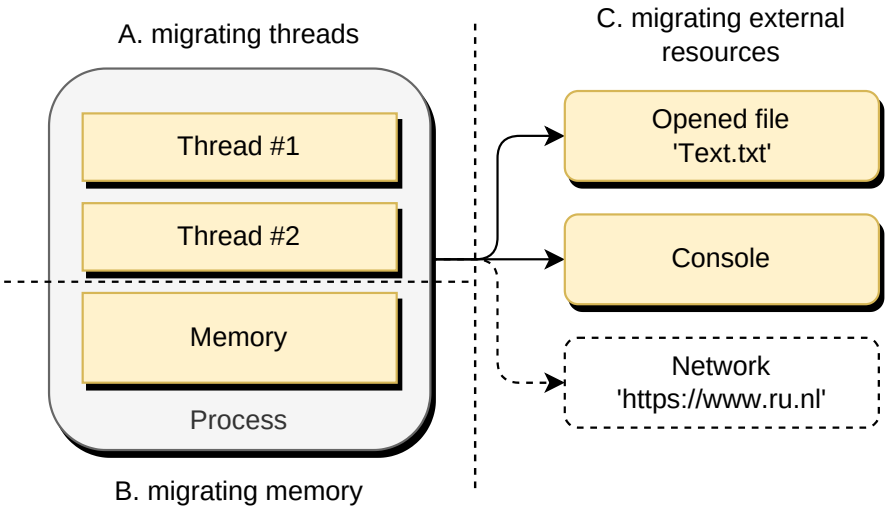
Process migration:
A) Migrating threads

Collecting Threads

- Get list of thread of process
- Get thread registers (GetThreadContext)

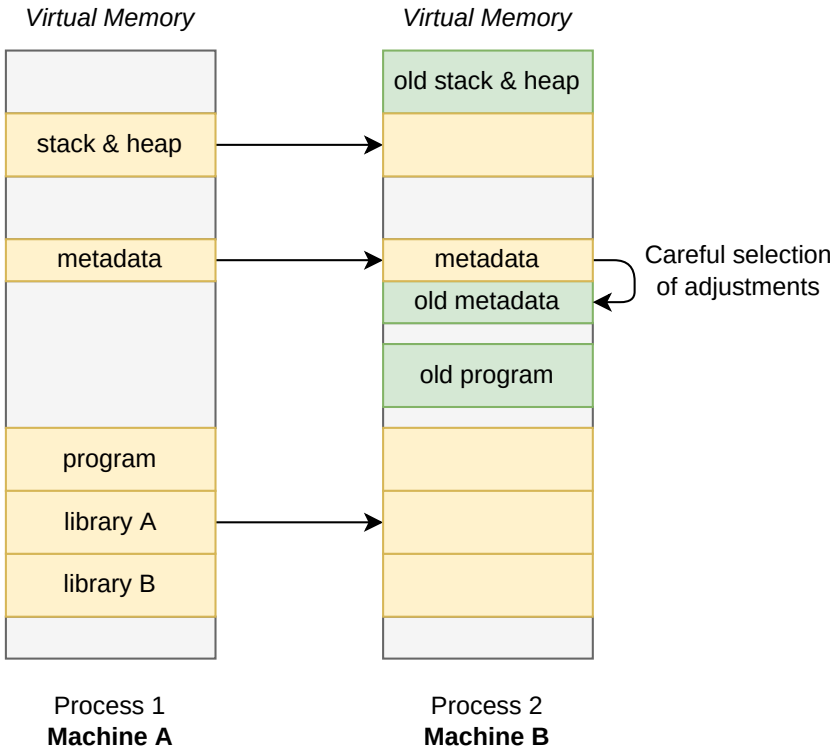
Restoring Threads

- Create thread (CreateRemoteThread)
- Set thread registers (SetThreadContext)



Process migration:
B) Migrating memory

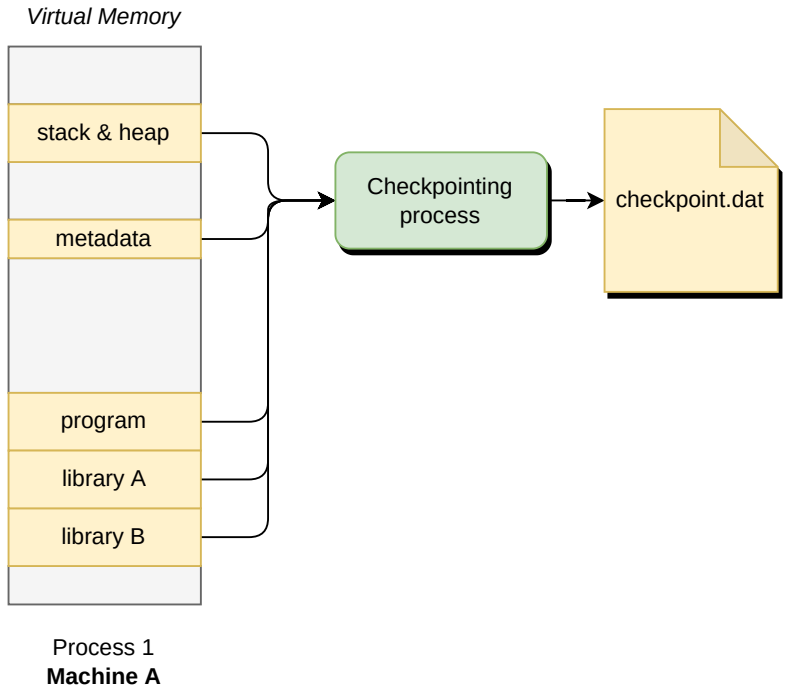
1) Collect info & content per memory region
2) Start an empty process + process initialization
3) Allocate content per memory region
4) Adjust metadata of new process



Process migration:
B) Collecting memory

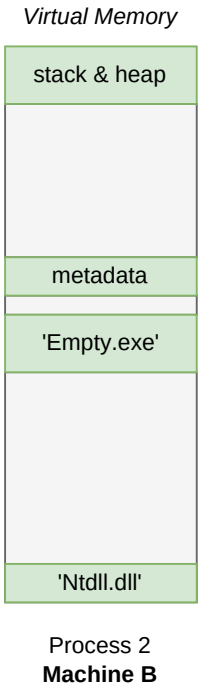
Iterate over every memory region in the virtual memory:

- Collect information about memory region (VirtualQueryEx)
- Read Content (ReadProcessMemory)



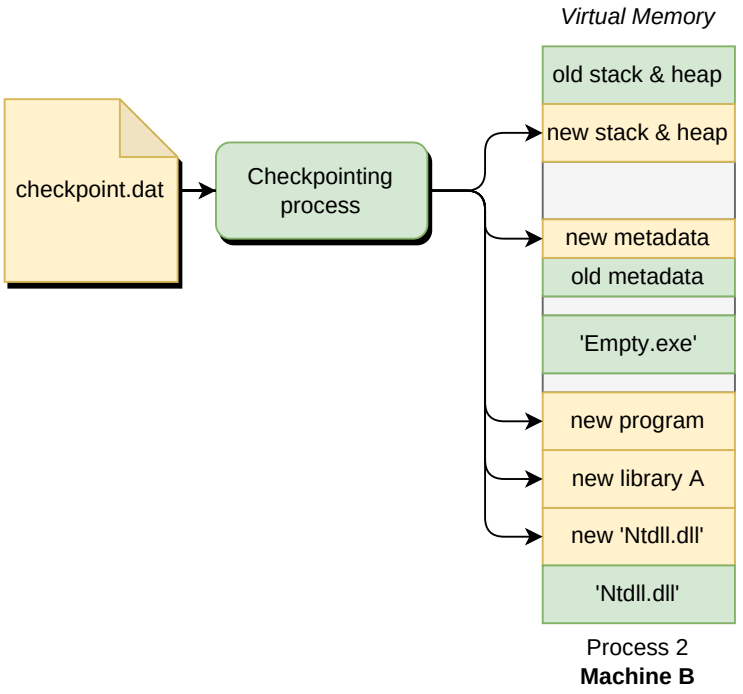
Process migration:
B) Restoring memory

1) **Start ‘Empty.exe’**
   - For correct initialization



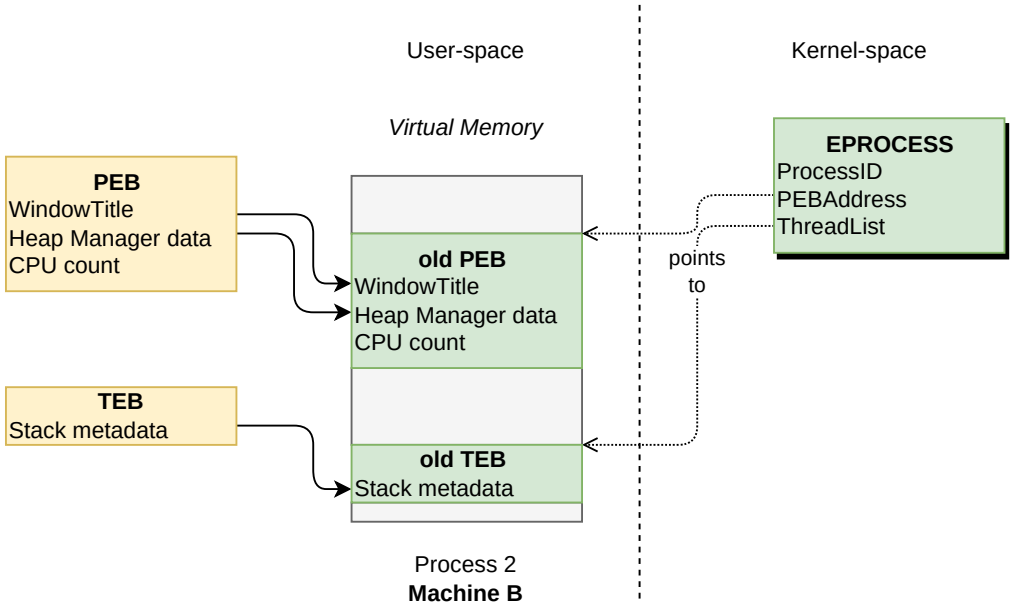
Process migration:
B) Restoring memory

1) Start ‘Empty.exe’
2) **For every memory region:**
    1) Allocate if region is free (VirtualAllocEx)
    2) Write content to region (WriteProcessMemory)
    3) Adjust read/write protection (VirtualProtectEx)



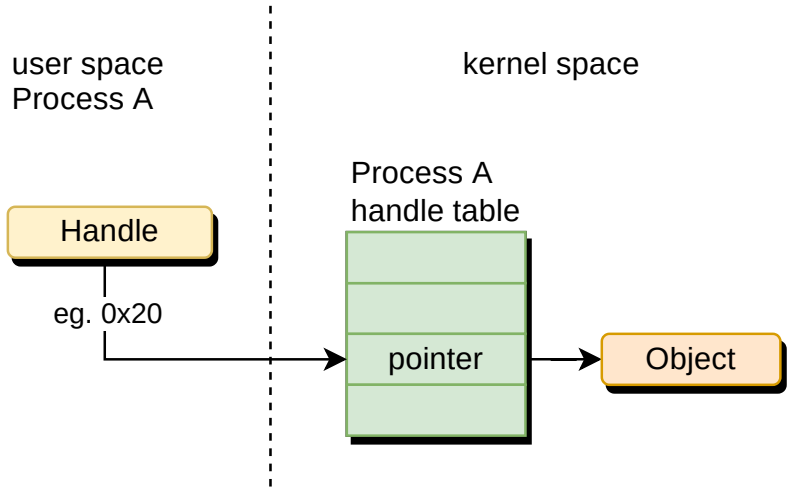
Process migration:
B) Migrating Memory: adjusting the metadata

- **What is the metadata?**
  - EPROCESS structure
  - Process Environment Block (TEB)
  - Thread Environment Block (TEB)
- **How to restore the metadata?**



Process migration:
C) Migrating external resources: what is a handle?

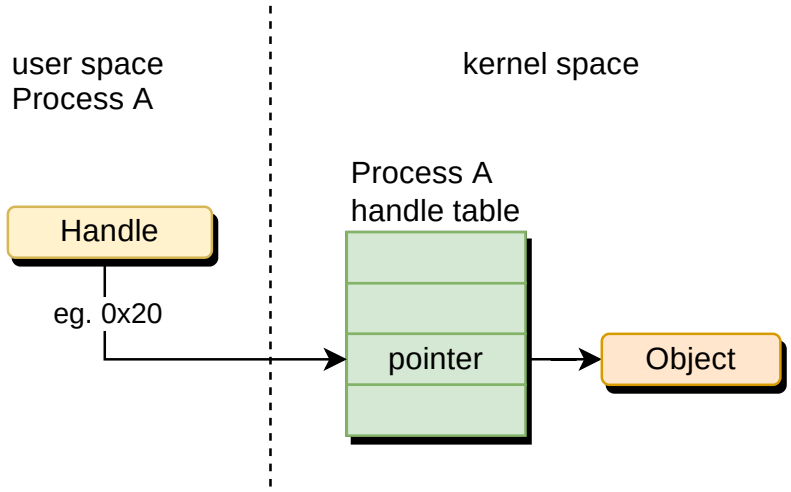
- A handle table for every process
- For communication with external resources
- Around 70 different types of handles



Process migration:
C) Migrating external resources: chosen approach

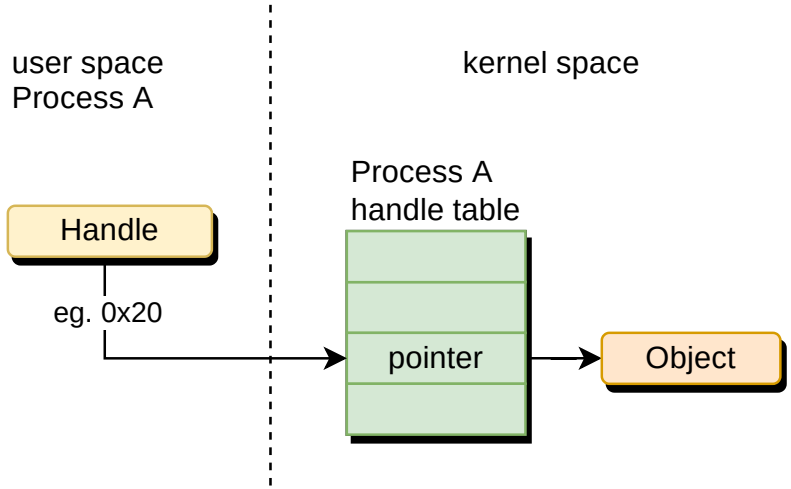
External sources we can migrate:

- Files
- Console handles (standard in/out)
- Events, Mutexes, Semaphores



Process migration:
C) Collecting handles

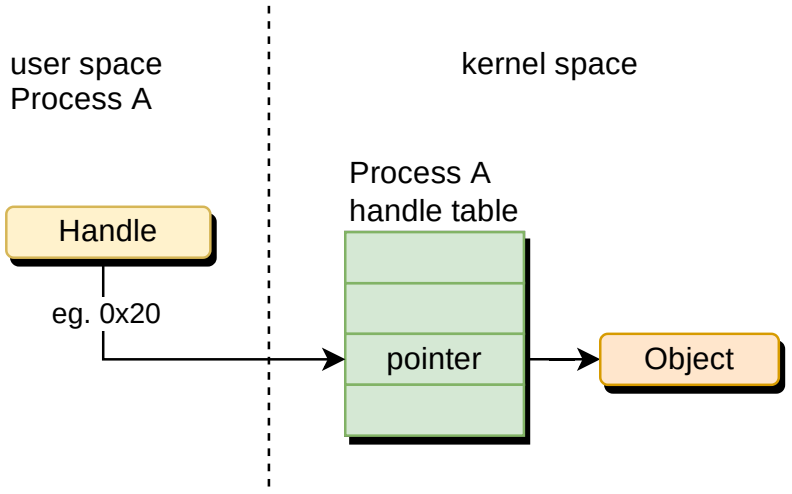
1) **Retrieve all handles from the system**
2) **For every handle with a matching process ID:**
   - Copy handle to checkpoint process (DuplicateHandle)
   - Read index, name, type, and access granted (NtQueryObject)



Source machine

Process migration:
C) Restoring handles

1) **Iterate over every handle from the checkpoint file**
2) **For every handle:**
   - Create the handle (type specific)
   - Copy handle over to restored process



Destination machine

Process migration:
C) Migrating files

Information to collect:

- Filename (NtQueryObject)
- Content of file (ReadFile)
- Current position in file (SetFilePointerEx)

Restoring a file:

- Create the file (CreateFile)
- Write the content (WriteFile)
- Set current position in file (SetFilePointerEx)

Process migration:
C) Migrating console handles

How to collect console handles:

- Find console handle type in handle table
- Locate standard in/out/err from metadata

How to restore console handles

- Inherited from checkpointing process
- Placed at the start of the handle table
- Also duplicated at position of standard in/out

Process migration:
C) Events, mutexes, semaphores

How to Collect:

- Events
- Mutex (WaitForSingleObject, ReleaseMutex)
- Semaphore (NtQuerySemaphore)

How to Restore:

- Events: (CreateEvent)
- Mutex: (CreateMutex)
- Semaphore: (CreateSemaphore)

Process migration: Verification

Demonstration

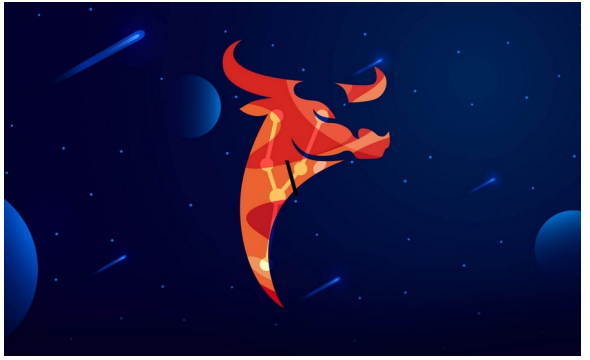
Process migration: Viability

Research Question 2:

To what extent is process migration a useful method for the analysis of malware?

Real-world malware

- Motivation
- Possible challenges
- We have tried to migrate two malware samples
  - Shadowpad malware
  - Turian malware



Real-world malware

Turian malware (sample of malware)

Behavior:

1) Creates a service and starts the service
2) The service connects to C2 server

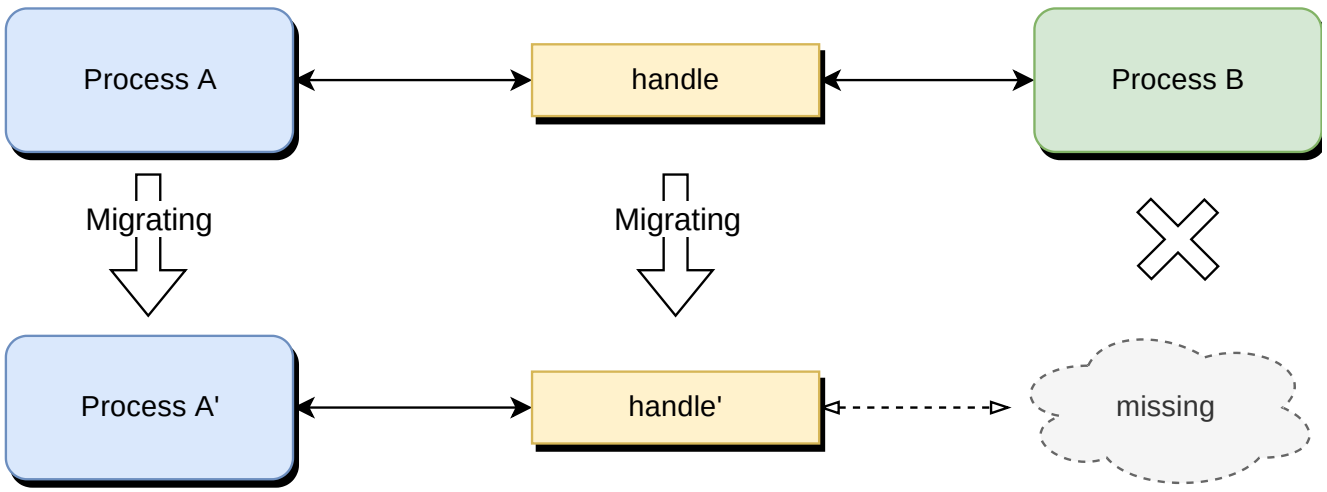
Goal: migrate the service

Real-world malware – Crash report

1) Migrate the process
2) Attach debugger before we resume the process

Observation: Crash due to missing ALPC Port handle (Inter-process communication)

The issue with handles



Final notes

Conclusions

Conclusions

Research Question 1:

Process migration is possible in Windows 10

- Threads & memory can be migrated
- A selection of handles can be migrated

Research Question 2:

A great start for migrating malware

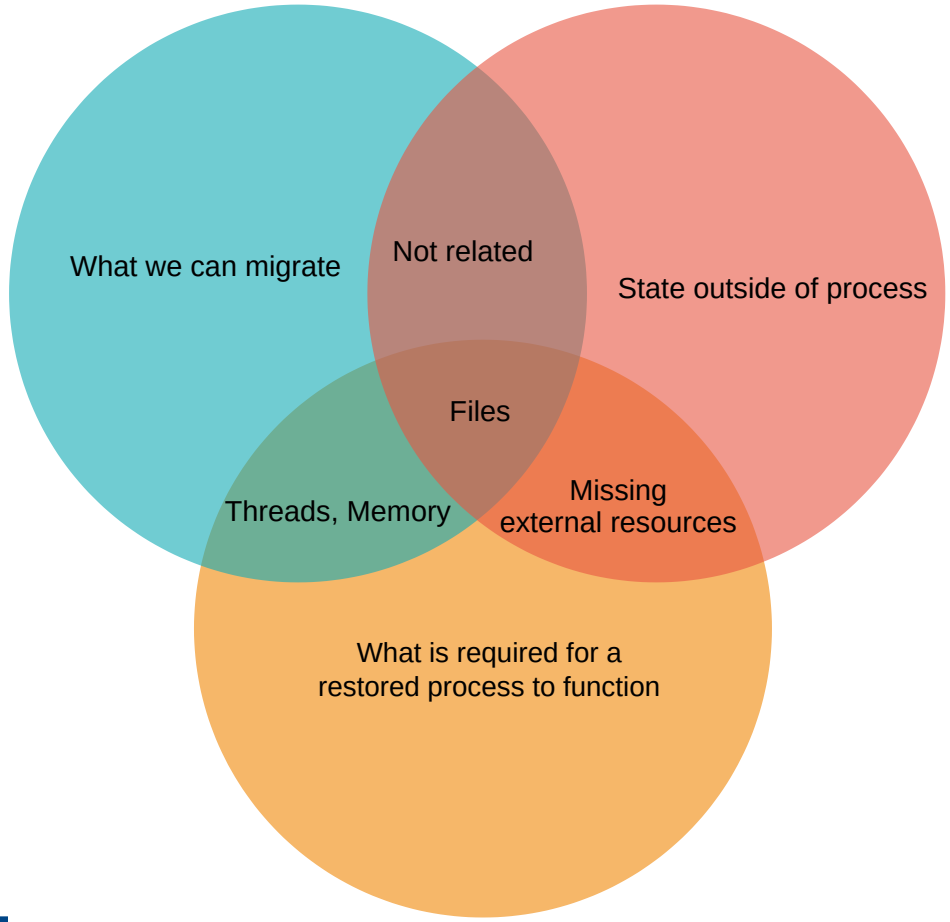
- Further research to migrate more handle types

Final notes

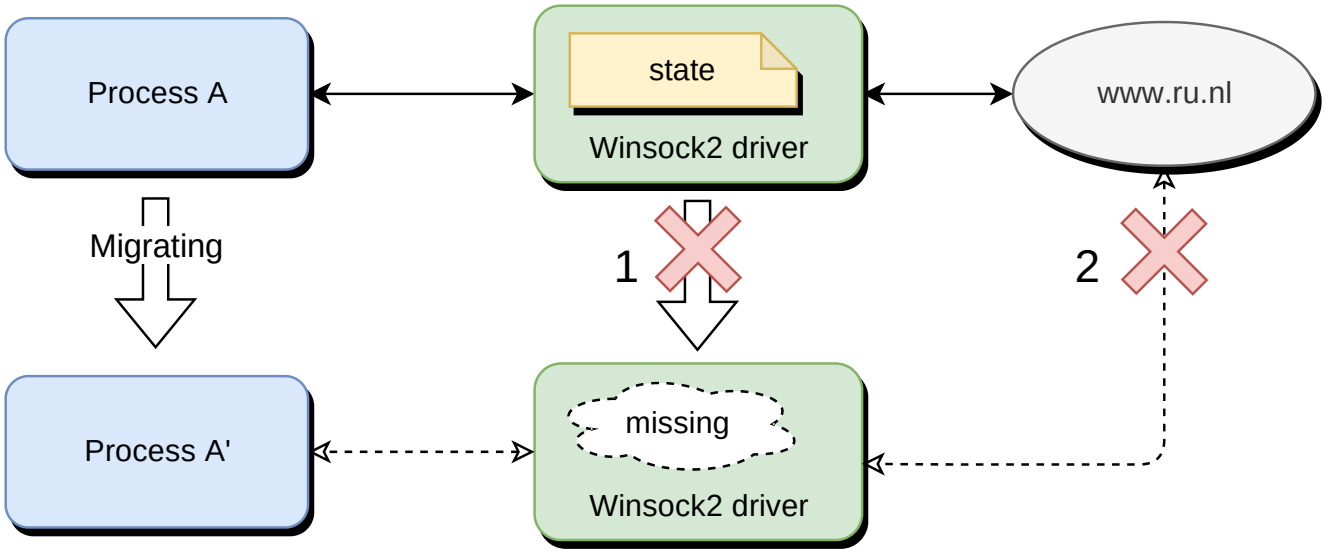
Find the code on Github:

https://github.com/keukentrap/process-copy

What makes process migration challenging?



Extra: Networking



Agenda

1) What is process migration?

2) Research Question 1: Is process migration possible?

3) Research Question 2: Can we migrate malware?

4) Conclusions & Recommendations

5) Questions?