Timelord X

User Manual

Timelord X is a framework designed to assist digital forensic investigators in detecting timestamp manipulation across multiple Windows artefacts. It focuses on **causal rule validation**, ensuring that the temporal order of artefacts (e.g., Prefetch, Amcache, MFT) aligns with expected behaviour and how Windows generates them. Any detected inconsistency via detection rule violations suggests a potential timestomp or anti-forensics attempt.

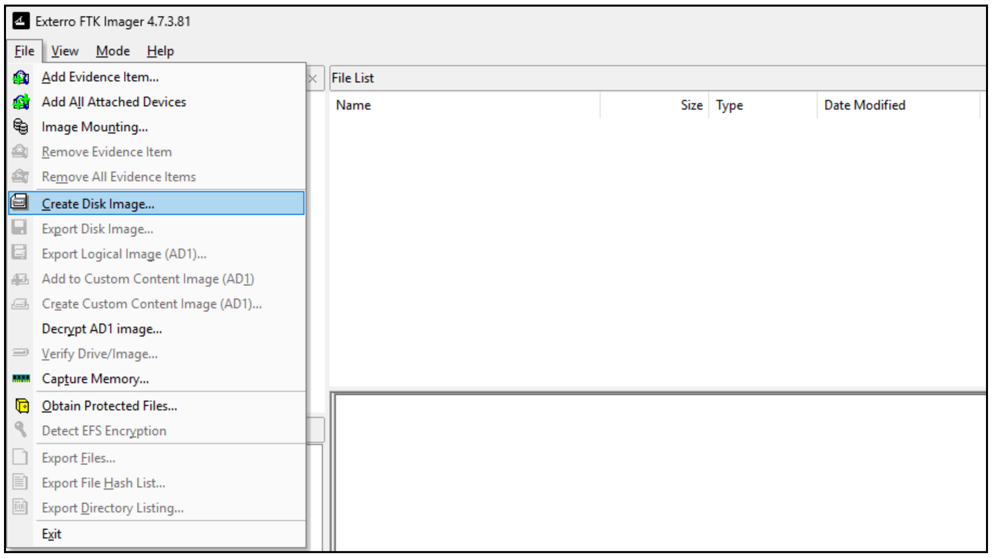
This manual shall provide detailed guidance for installation, usage, rule configuration, and interpretation of results.

Table of Contents

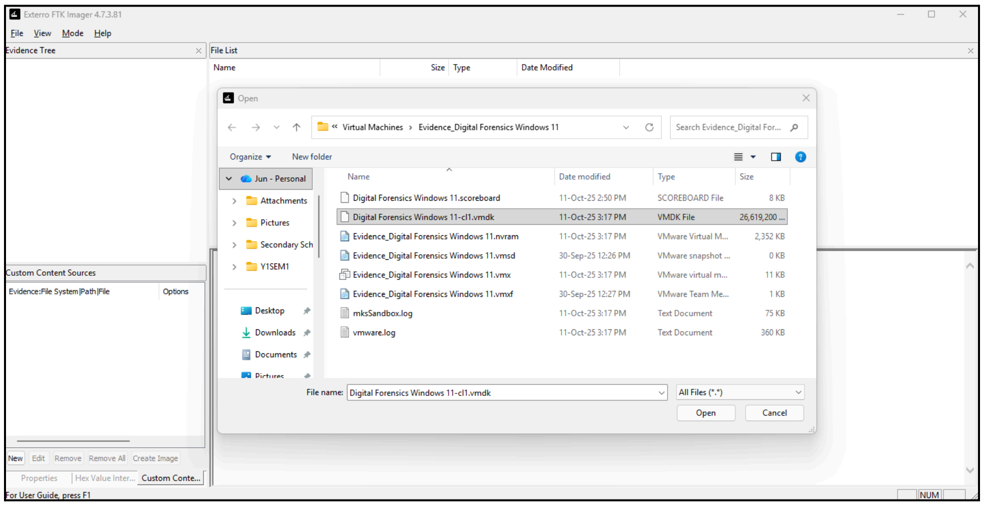
I. Data Attainment

An evidence disk image should be properly handled in a forensically sound manner. Upon ensuring that the disk image was properly collected and handled, an .E01 file should follow to begin performing digital forensics. In this guide, FTK Imager will be the primary tool used for imaging.

1) Obtain the .E01 file using tools such as FTK Imager, Encase Imager, or any suitable tool of your choice. In FTK Imager, select ‘File’ > ‘Create Disk Image’.



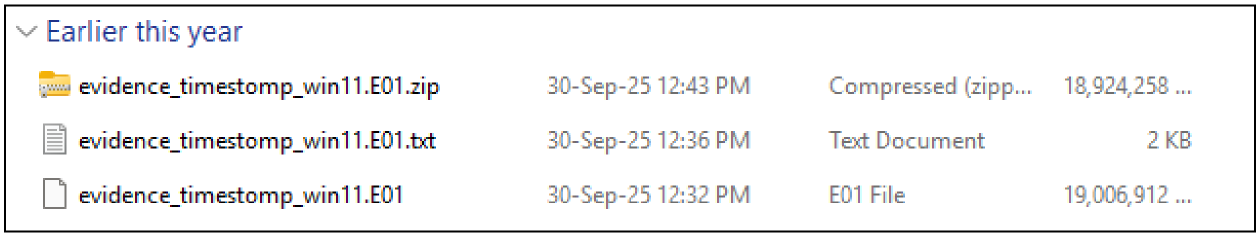
2) Select the target system to perform imaging on (e.g., .vmdk in this guide).



Name	Date modified	Type	Size
Digital Forensics Windows 11.scoreboard	11-Oct-25 2:50 PM	SCOREBOARD File	8 KB
Digital Forensics Windows 11-cl1.vmdk	11-Oct-25 3:17 PM	VMDK File	26,619,200 ...
Evidence_Digital Forensics Windows 11.nvram	11-Oct-25 3:17 PM	VMware Virtual M...	2,352 KB
Evidence_Digital Forensics Windows 11.vmsd	30-Sep-25 12:26 PM	VMware snapshot ...	0 KB
Evidence_Digital Forensics Windows 11.vmx	11-Oct-25 3:17 PM	VMware virtual m...	11 KB
Evidence_Digital Forensics Windows 11.vmxf	30-Sep-25 12:27 PM	VMware Team Me...	1 KB
mksSandbox.log	11-Oct-25 3:17 PM	Text Document	75 KB
vmware.log	11-Oct-25 3:17 PM	Text Document	360 KB

File name: Digital Forensics Windows 11-cl1.vmdk

3) Upon completion, an .E01 file will be generated.



Earlier this year

evidence_timestomp_win11.E01.zip	30-Sep-25 12:43 PM	Compressed (zipp...	18,924,258 ...
evidence_timestomp_win11.E01.txt	30-Sep-25 12:36 PM	Text Document	2 KB
evidence_timestomp_win11.E01	30-Sep-25 12:32 PM	E01 File	19,006,912 ...

4) Next, using log2timeline, generate a .plaso file using the following command:

```
log2timeline.py --storage-file <name of evidence>.plaso "<name of
image file"
```

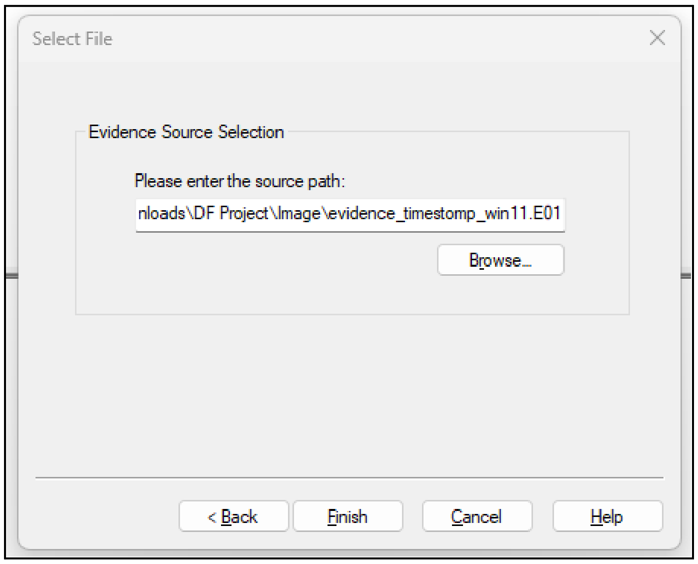
5) Subsequently, obtain a super timeline exported in the form of CSV. This is performed using the psort built into the Plaso framework.

```
psort.py -o l2tcsv -w timeline.csv <name of evidence>.plaso
```

II. Supporting Data

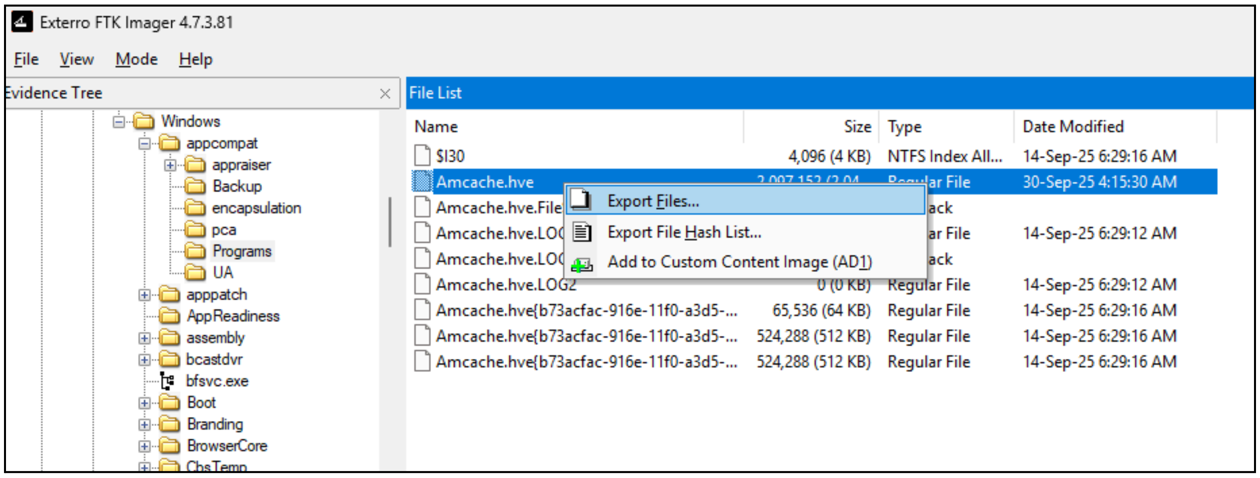
Additional artefacts such as the Amcache.hve and Prefetch files have to be obtained separately to assure the correctness of data prior to timestomp detection.

1) Launch FTK Imager. Under 'File', select 'Add Evidence Item' > 'Image File'. Select the previously exported .E01 file as the target.



Please enter the source path: nloads\DF Project\Image\evidence_timestomp_win11.E01

2) Navigate to the partition containing the operating system (OS). Thereafter, branch to Windows\AppCompat\Programs\Amcache.hve and export this file to your desired directory.


Exterro FTK Imager 4.7.3.81

File View Mode Help

Evidence Tree

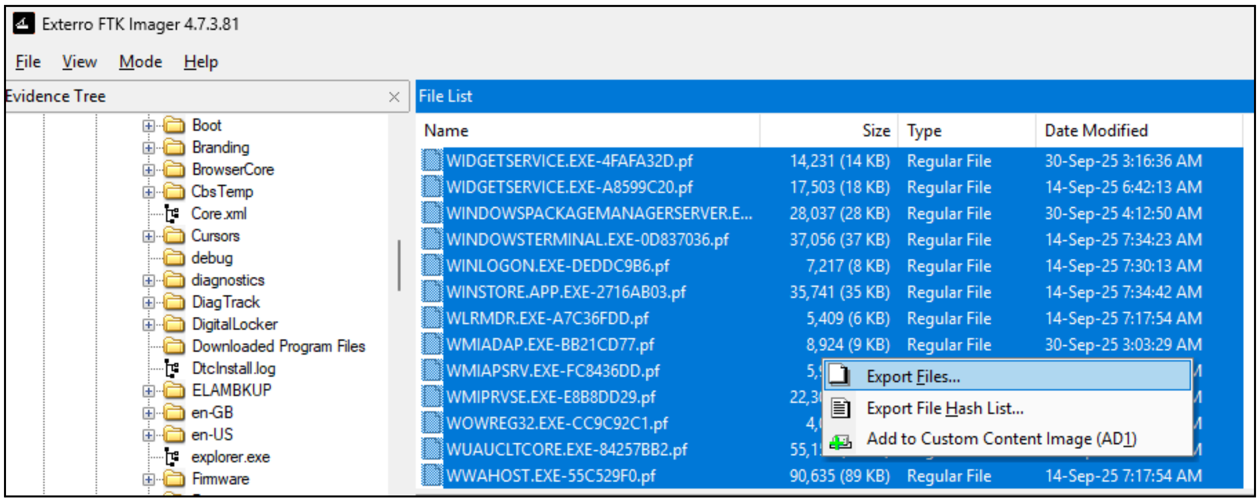
```
Windows
  appcompat
    appraiser
    Backup
    encapsulation
    pca
    Programs
    UA
  apppatch
  AppReadiness
  assembly
  bcastdvr
  bfsvc.exe
  Boot
  Branding
  BrowserCore
  CbsTemp
```

File List

Name	Size	Type	Date Modified
$I30	4,096 (4 KB)	NTFS Index All...	14-Sep-25 6:29:16 AM
Amcache.hve	2,097,152 (2,04	Regular File	30-Sep-25 4:15:30 AM
Amcache.hve.File		ack	
Amcache.hve.LO(		ar File	14-Sep-25 6:29:12 AM
Amcache.hve.LO(		ack	
Amcache.hve.LOG2	0 (0 KB)	Regular File	14-Sep-25 6:29:12 AM
Amcache.hve{b73acfac-916e-11f0-a3d5-...	65,536 (64 KB)	Regular File	14-Sep-25 6:29:16 AM
Amcache.hve{b73acfac-916e-11f0-a3d5-...	524,288 (512 KB)	Regular File	14-Sep-25 6:29:16 AM
Amcache.hve{b73acfac-916e-11f0-a3d5-...	524,288 (512 KB)	Regular File	14-Sep-25 6:29:16 AM

3) Next, create a directory or folder that will be used to store Prefetch (.pf) files. In FTK Imager, branch to Windows\Prefetch, then sort by 'Type' column. Select all of the .pf files containing a 'Regular File' type label and export to the created folder.

Note: It is important that the Prefetch files are only exported and have no other filesystem activities (e.g., moving, copying) performed on them.


Exterro FTK Imager 4.7.3.81

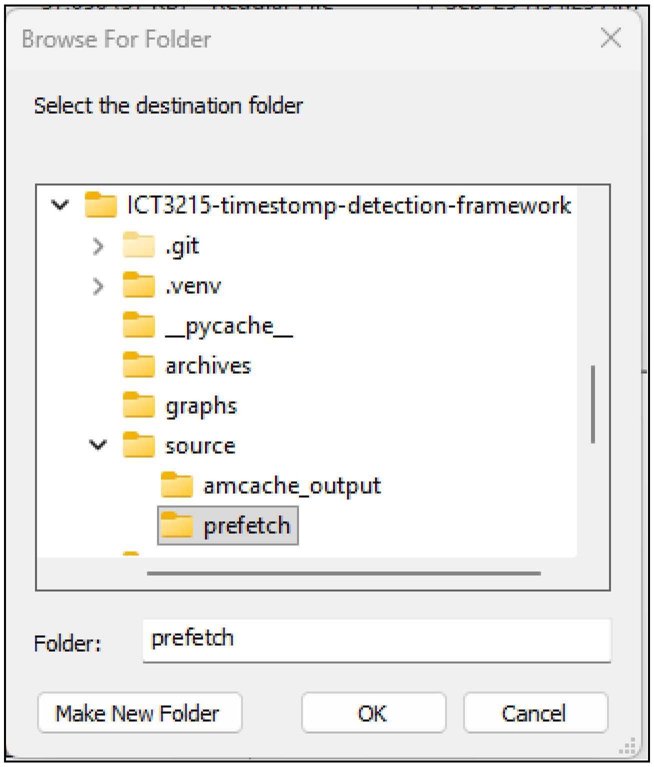
File View Mode Help

Evidence Tree

```
Boot
Branding
BrowserCore
CbsTemp
Core.xml
Cursors
debug
diagnostics
DiagTrack
DigitalLocker
Downloaded Program Files
DtcInstall.log
ELAMBKUP
en-GB
en-US
explorer.exe
Firmware
```

File List

Name	Size	Type	Date Modified
WIDGETSERVICE.EXE-4FAFA32D.pf	14,231 (14 KB)	Regular File	30-Sep-25 3:16:36 AM
WIDGETSERVICE.EXE-A8599C20.pf	17,503 (18 KB)	Regular File	14-Sep-25 6:42:13 AM
WINDOWSPACKAGEMANAGERSERVER.E...	28,037 (28 KB)	Regular File	30-Sep-25 4:12:50 AM
WINDOWSTERMINAL.EXE-0D837036.pf	37,056 (37 KB)	Regular File	14-Sep-25 7:34:23 AM
WINLOGON.EXE-DEDDC9B6.pf	7,217 (8 KB)	Regular File	14-Sep-25 7:30:13 AM
WINSTORE.APP.EXE-2716AB03.pf	35,741 (35 KB)	Regular File	14-Sep-25 7:34:42 AM
WLRMDR.EXE-A7C36FDD.pf	5,409 (6 KB)	Regular File	14-Sep-25 7:17:54 AM
WMIADAP.EXE-BB21CD77.pf	8,924 (9 KB)	Regular File	30-Sep-25 3:03:29 AM
WMIAPSRV.EXE-FC8436DD.pf	5,		M
WMIPRVSE.EXE-E8B8DD29.pf	22,3		M
WOWREG32.EXE-CC9C92C1.pf	4,		M
WUAUCLTCORE.EXE-84257BB2.pf	55,1		M
WWAHOST.EXE-55C529F0.pf	90,635 (89 KB)	Regular File	14-Sep-25 7:17:54 AM



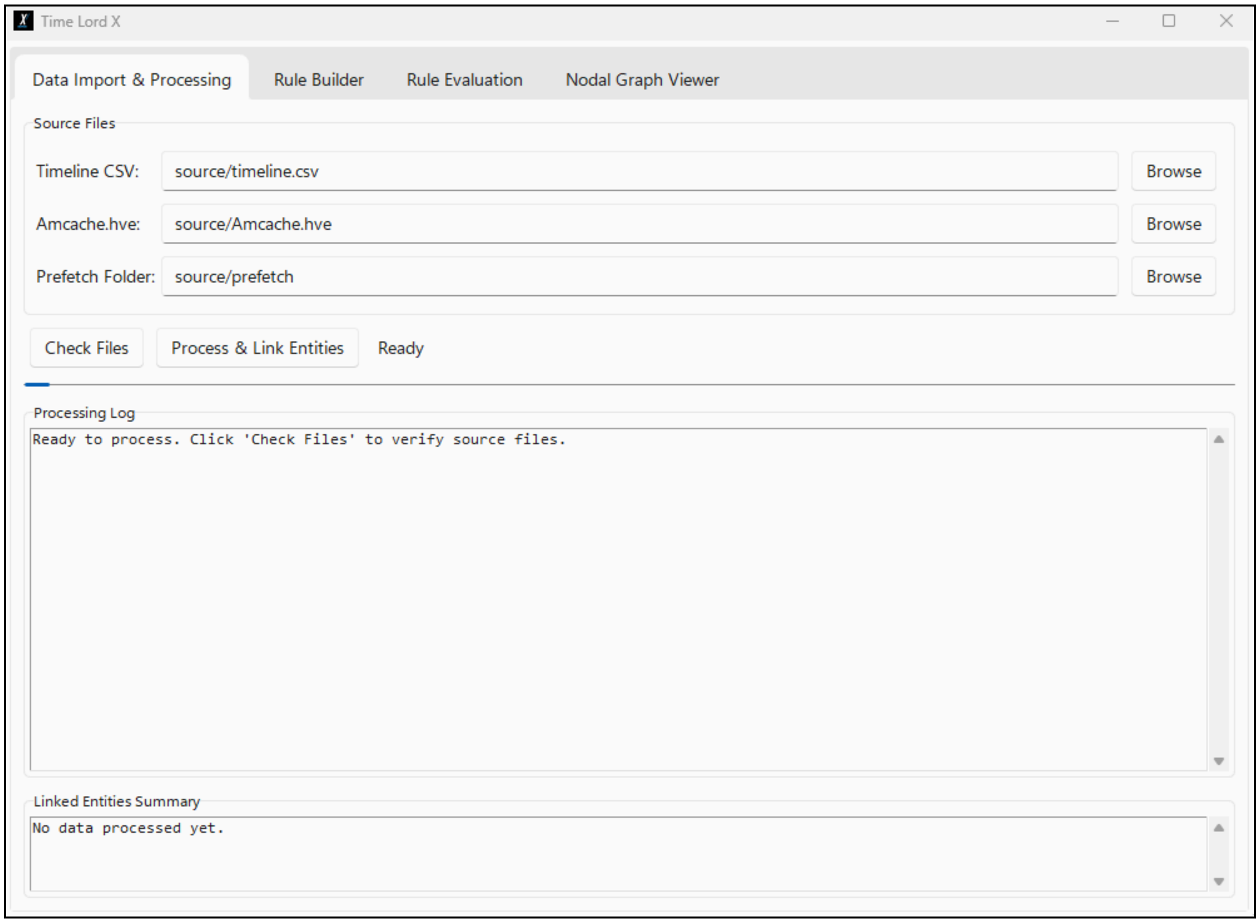
III. Environment Setup & Initial Execution

1) Launch the command prompt as administrator and navigate to the project directory. Within it, create a virtual environment and activate it using the following:

```
py -m venv .venv
.venv\Scripts\activate
```

2) Execute rule_builder_gui.py with the command:

```
py rule_builder_gui.py
```



Timeline CSV: source/timeline.csv

Amcache.hve: source/Amcache.hve

Prefetch Folder: source/prefetch

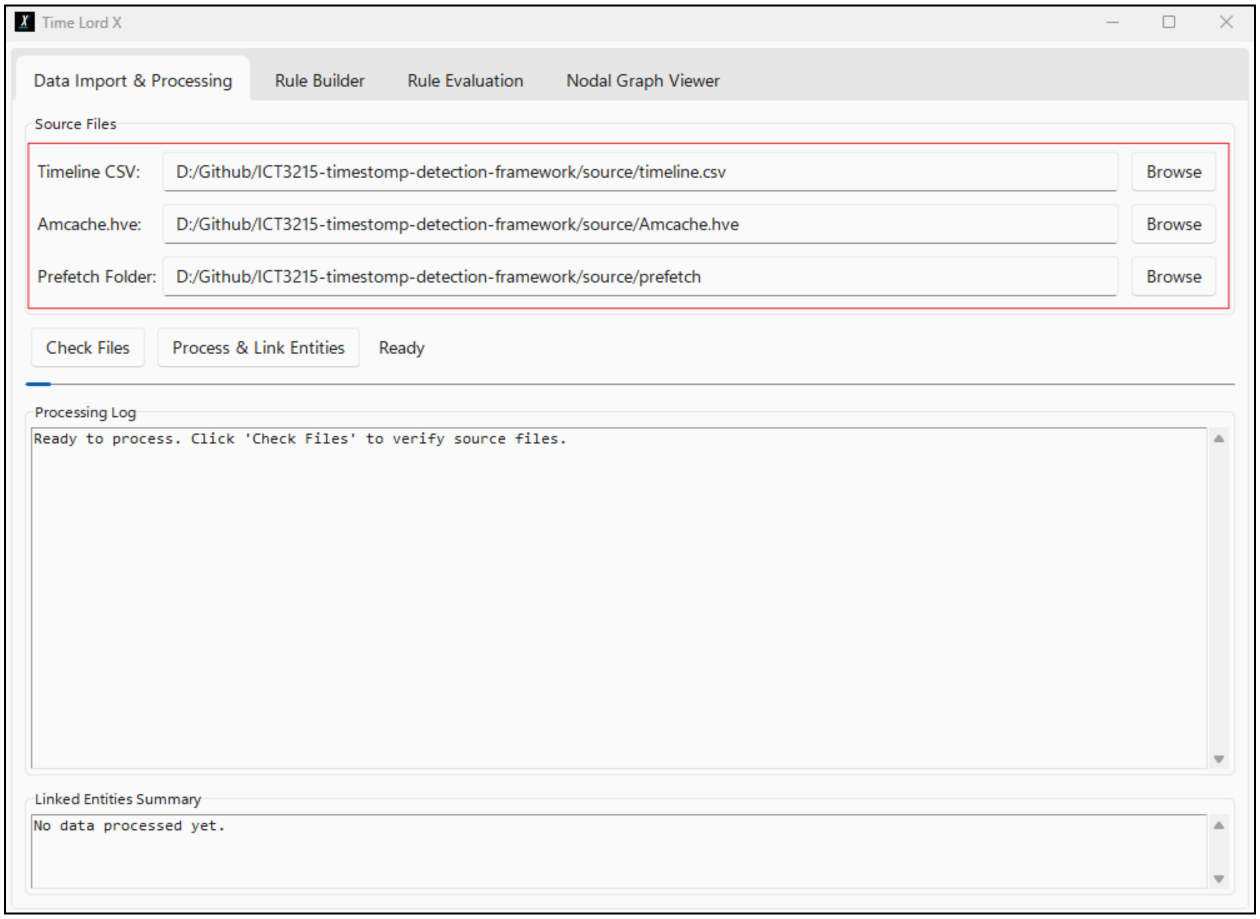
```
Ready to process. Click 'Check Files' to verify source files.
```

```
No data processed yet.
```

IV. Data Import & Processing

The timeline.csv generated using psort, Amcache.hve, and Prefetch log details have to be processed and have their entity linkage performed. Entity linkage groups artefacts referring to the same executable or file path using correlating logic from the various information within each log source type.

1) Browse for timeline.csv, Amcache.hve, and the folder containing all of the Prefetch (.pf) files.



Timeline CSV: D:/Github/ICT3215-timestomp-detection-framework/source/timeline.csv

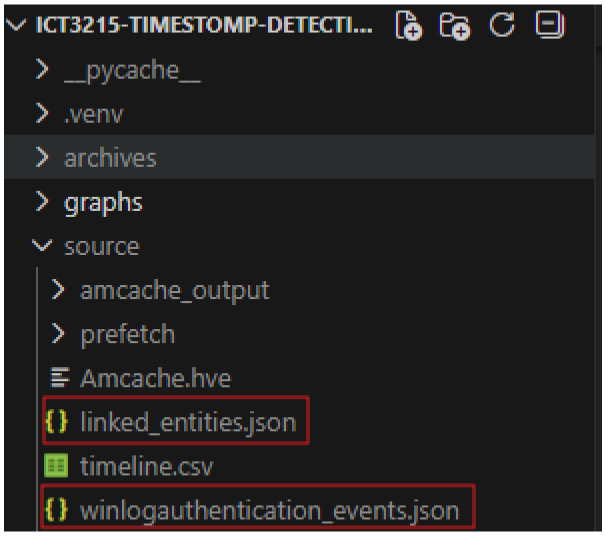
Amcache.hve: D:/Github/ICT3215-timestomp-detection-framework/source/Amcache.hve

Prefetch Folder: D:/Github/ICT3215-timestomp-detection-framework/source/prefetch

```
Ready to process. Click 'Check Files' to verify source files.
```

```
No data processed yet.
```

2) Once selected, proceed to click on ‘Process & Link Entities’. This will perform log correlation and link all of the events tied to each executable or file found within the evidence. Its corresponding output will produce linked_entities.json and winlogauthentication_events.json files under the source folder of the project root directory.



```
ICT3215-TIMESTOMP-DETECTI...
  > __pycache__
  > .venv
  > archives
  > graphs
  v source
    > amcache_output
    > prefetch
      Amcache.hve
      linked_entities.json
      timeline.csv
      winlogauthentication_events.json
```

V. Rule Builder

General Information

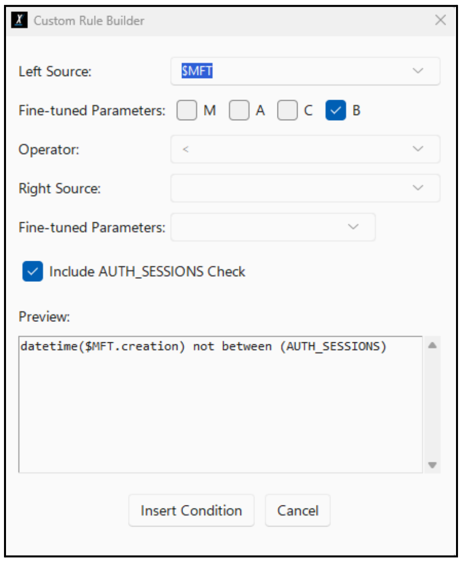
Field	**Description**
Rule ID	Unique Rule Reference
Rule Name	Name for the Rule
Severity	Potential seriousness of the detected anomaly
Explanation	Rule rationale
Logic Type	Tell the framework how to evaluate multiple conditions inside a single rule 'any_of': The rule triggers a violation if any one of the listed conditions is true 'all_of': The rule triggers a violation only when all listed conditions are true at the same time
Conditions	Each condition is a Boolean comparison that evaluates two timestamps or state values drawn from artefacts such as $MFT, PREFETCH, AMCACHE, $USN_JOURNAL, etc.

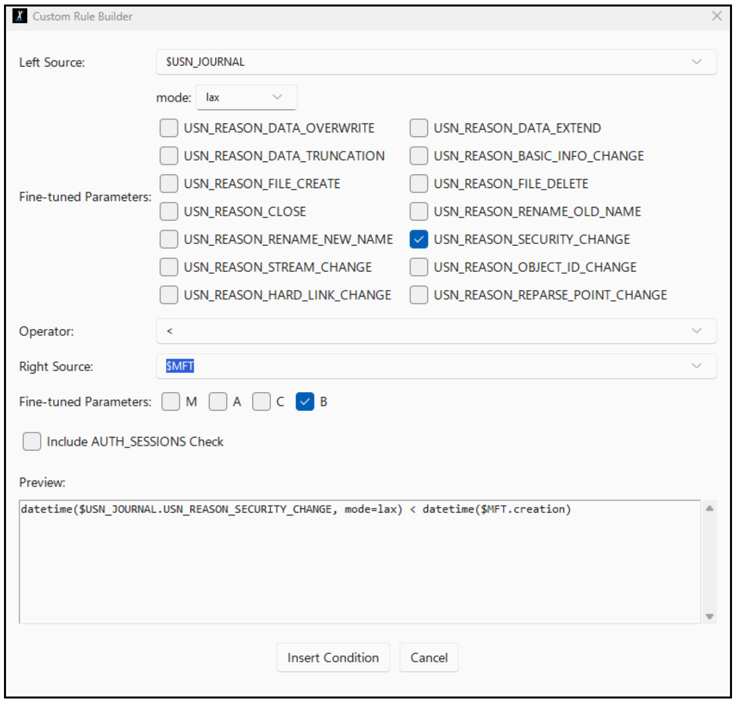
Source Types

Source Type	**Description**
$MFT	Uses MACB attribute filtering to return only the specific timestamp type requested (e.g., creation, modified).
Prefetch	Supports firstrun (created time) and lastrun (latest run). Defaults to up to 8 recorded run timestamps.
$UsnJournal	Allows strict or lax matching of USN_REASON flags to extract timestamps tied to specific file-system changes. **Note**: ● Lax Mode: a record is considered relevant if it contains any reason that belongs to the investigator's selected subset. ● Strict Mode: requires that the record's assigned reasons match exactly and exclusively the chosen set.
Others	Other source types are evaluated via a normal basis. For all of the found timestamps, they are compared iteratively.

AUTH_SESSIONS	Defaults to flagging sources that are not within the timeframe of authenticated sessions.

Some available examples of rule design syntax are as follows to familiarize with how the rule building works. It is worth noting that the crafting of the ruleset is heavily dependent on semantic and forensic correctness to obtain accurate information:

Purpose	**Description**
Identify files with $MFT birth (...b) during unauthenticated session	 Custom Rule Builder Left Source: $MFT Fine-tuned Parameters: M A C [✓ B] Operator: < Right Source: Fine-tuned Parameters: [✓ Include AUTH_SESSIONS Check] Preview: `datetime($MFT.creation) not between (AUTH_SESSIONS)` Insert Condition Cancel

Identify $UsnJournal Security Change events before $MFT birth time	

Custom Rule Builder

Left Source: $USN_JOURNAL

mode: lax

Fine-tuned Parameters:

- USN_REASON_DATA_OVERWRITE
- USN_REASON_DATA_EXTEND
- USN_REASON_DATA_TRUNCATION
- USN_REASON_BASIC_INFO_CHANGE
- USN_REASON_FILE_CREATE
- USN_REASON_FILE_DELETE
- USN_REASON_CLOSE
- USN_REASON_RENAME_OLD_NAME
- USN_REASON_RENAME_NEW_NAME
- [✓ USN_REASON_SECURITY_CHANGE]
- USN_REASON_STREAM_CHANGE
- USN_REASON_OBJECT_ID_CHANGE
- USN_REASON_HARD_LINK_CHANGE
- USN_REASON_REPARSE_POINT_CHANGE

Operator: <

Right Source: $MFT

Fine-tuned Parameters: M A C [✓ B]

Include AUTH_SESSIONS Check

Preview:

```
datetime($USN_JOURNAL.USN_REASON_SECURITY_CHANGE, mode=lax) < datetime($MFT.creation)
```

Target Selection

The target selection feature accepts wildcards, substring matching, precise binary paths, and regular expressions. Below are some possible examples for each category:

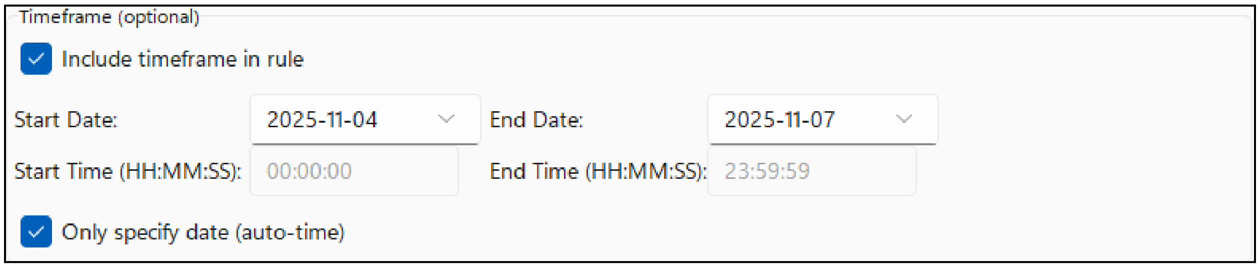
- Wildcard: “*”
- Substring Search: “creation_future”
- Precise Binary Path: “users/timel/desktop/cases/creation_future.exe”
- Regular Expression: “r/<regex_expression>”

Note: regular expression must be pre-fixed with ‘r/’ prior to the regex expression. The prefix is not taken into account in the regular expression syntax.

Timeframe Selection

Timeframe selection is an optional field allowing investigators to specify the period that he/she may want to conduct the investigation on. Events outside of the specified timeframe will not be included in the checking against the detection logic. If necessary, the timeframe can be selected by enabling ‘include timeframe in rule’.

When ‘only specify date (auto-time)’ is enabled, the framework accounts for the period of the start date and the end date, inclusive of the day itself.


Timeframe (optional)

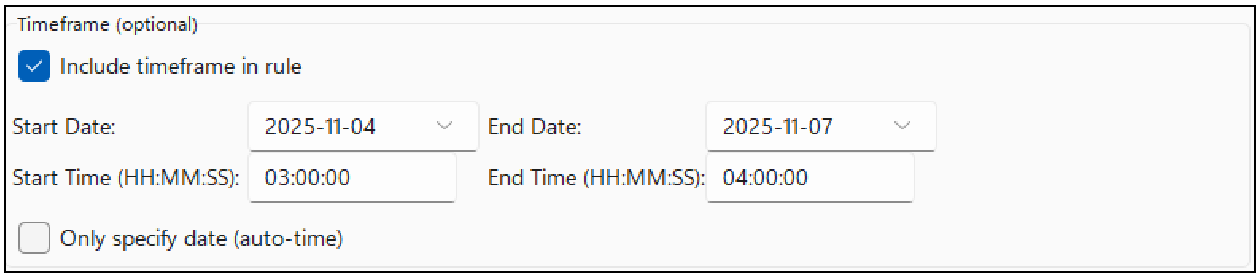
☑ Include timeframe in rule

Start Date: 2025-11-04 End Date: 2025-11-07

Start Time (HH:MM:SS): 00:00:00 End Time (HH:MM:SS): 23:59:59

☑ Only specify date (auto-time)

Otherwise, investigators are also able to define the start and end times of the start and end dates, respectively, as shown below.


Timeframe (optional)

☑ Include timeframe in rule

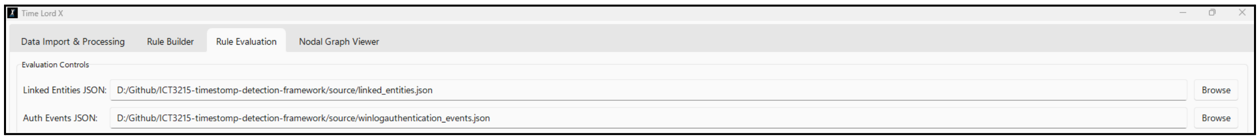
Start Date: 2025-11-04 End Date: 2025-11-07

Start Time (HH:MM:SS): 03:00:00 End Time (HH:MM:SS): 04:00:00

☐ Only specify date (auto-time)

VI. Rule Evaluation

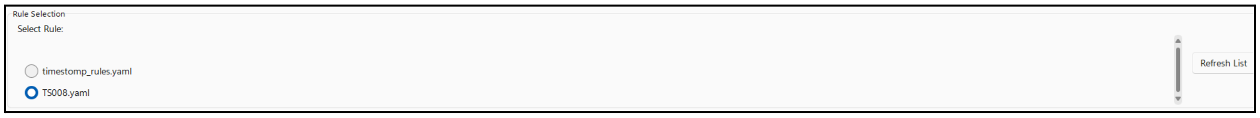
1) Link and target the linked_entities.json and winlogauthentication_events.json.



Linked Entities JSON: D:/Github/ICT3215-timestomp-detection-framework/source/linked_entities.json

Auth Events JSON: D:/Github/ICT3215-timestomp-detection-framework/source/winlogauthentication_events.json

2) Select the ruleset that you have created to be used for timestomp detection. You may be required to refresh the list whenever necessary.

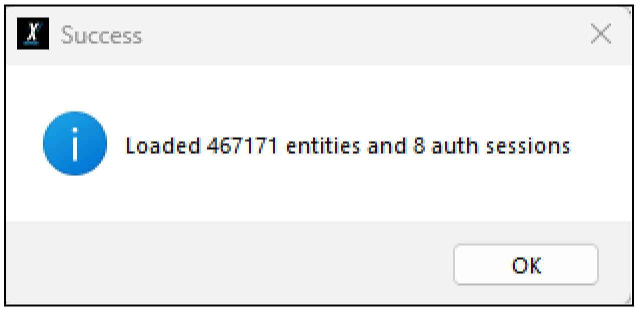


Select Rule:

○ timestomp_rules.yaml

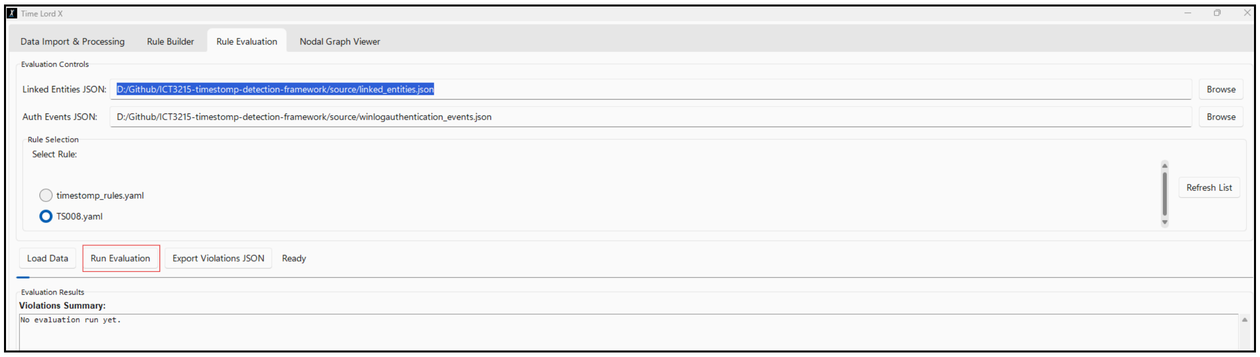
● TS008.yaml

3) Click on ‘Load Data’ to process the information. Upon successful loading, a pop-up window will appear suggesting completion.


Success

Loaded 467171 entities and 8 auth sessions

4) To evaluate evidence of artefacts against the ruleset to identify violations, click ‘Run Evaluation’. Its status will transition to “Evaluating…”.



Linked Entities JSON: D:/Github/ICT3215-timestomp-detection-framework/source/linked_entities.json

Auth Events JSON: D:/Github/ICT3215-timestomp-detection-framework/source/winlogauthentication_events.json

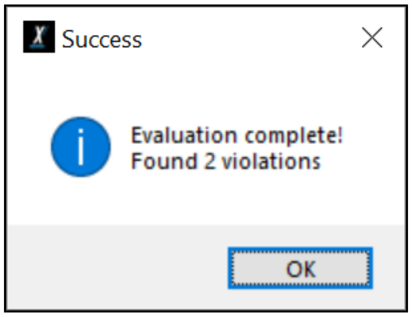
Select Rule: timestomp_rules.yaml [✓ TS008.yaml]

Ready

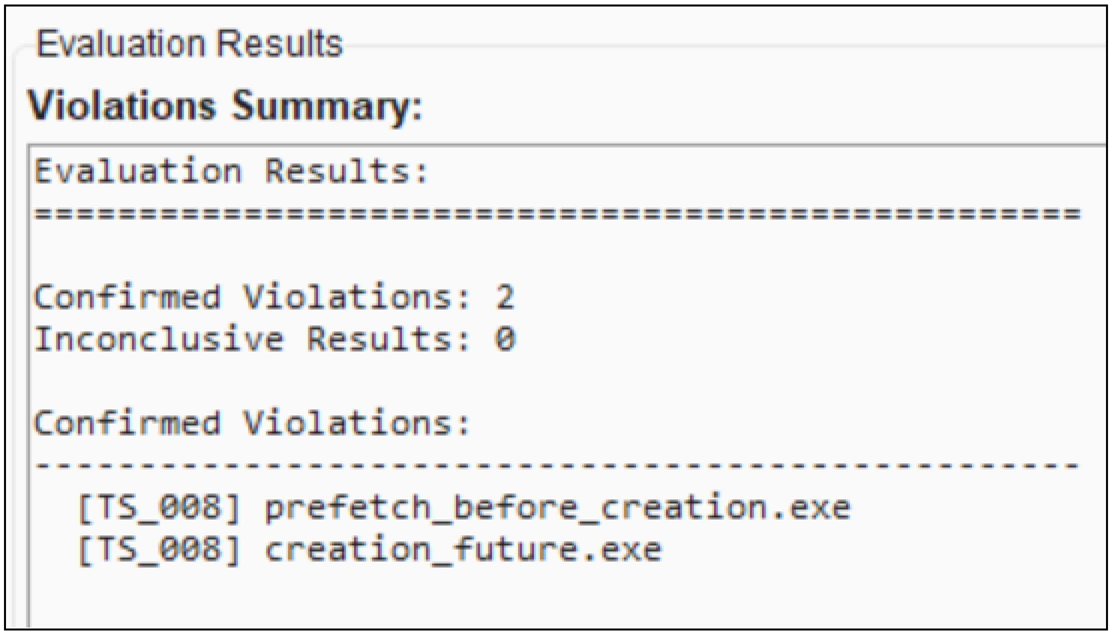
Violations Summary:

```
No evaluation run yet.
```

5) The count of violations detected will be displayed in a pop-up window. Additional details can be seen in the ‘Evaluation Results’ panel.


Success

Evaluation complete!
Found 2 violations

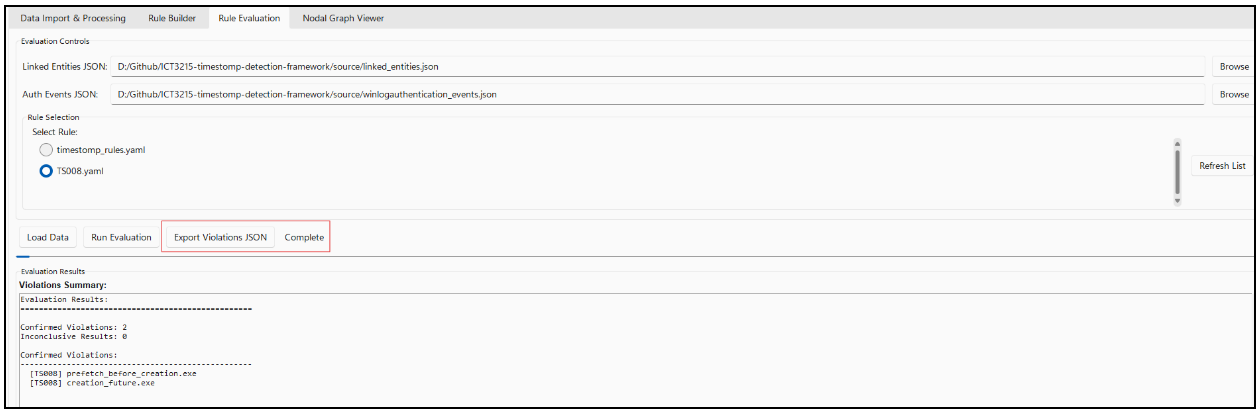


```
Evaluation Results:
==================================================

Confirmed Violations: 2
Inconclusive Results: 0

Confirmed Violations:
--------------------------------------------------
  [TS_008] prefetch_before_creation.exe
  [TS_008] creation_future.exe
```

6) The violations can be exported in a JSON format using the ‘Export Violations JSON’ button. This JSON can be used for nodal graph representation for visualization.



Linked Entities JSON: D:/Github/ICT3215-timestomp-detection-framework/source/linked_entities.json

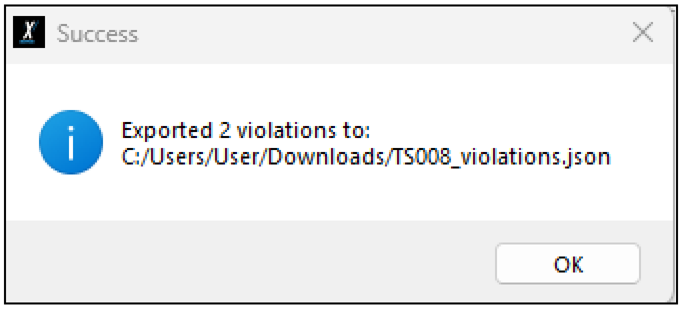
Auth Events JSON: D:/Github/ICT3215-timestomp-detection-framework/source/winlogauthentication_events.json

Select Rule:
timestomp_rules.yaml [✓ TS008.yaml]

```
Evaluation Results:
==================================================

Confirmed Violations: 2
Inconclusive Results: 0

Confirmed Violations:
--------------------------------------------------
  [TS008] prefetch_before_creation.exe
  [TS008] creation_future.exe
```



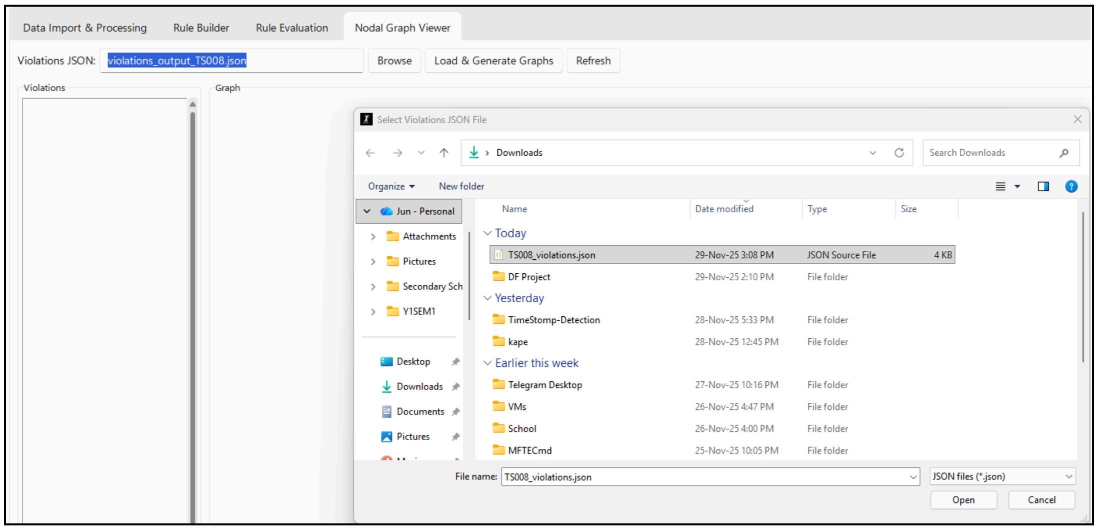
Exported 2 violations to:
C:/Users/User/Downloads/TS008_violations.json

VII. Nodal Graph Representation

The nodal graphs can be viewed under the ‘Nodal Graph Viewer’ tab. For each violation, it will be prefixed with the rule ID followed by the violation binary name. A nodal graph will be produced to illustrate the time delta difference behind the violation.

1) Under the ‘Nodal Graph Viewer’ tab, select the violation JSON by pressing on ‘Browse’.

2) Select the exported violation JSON, then click ‘Load & Generate Graphs’.

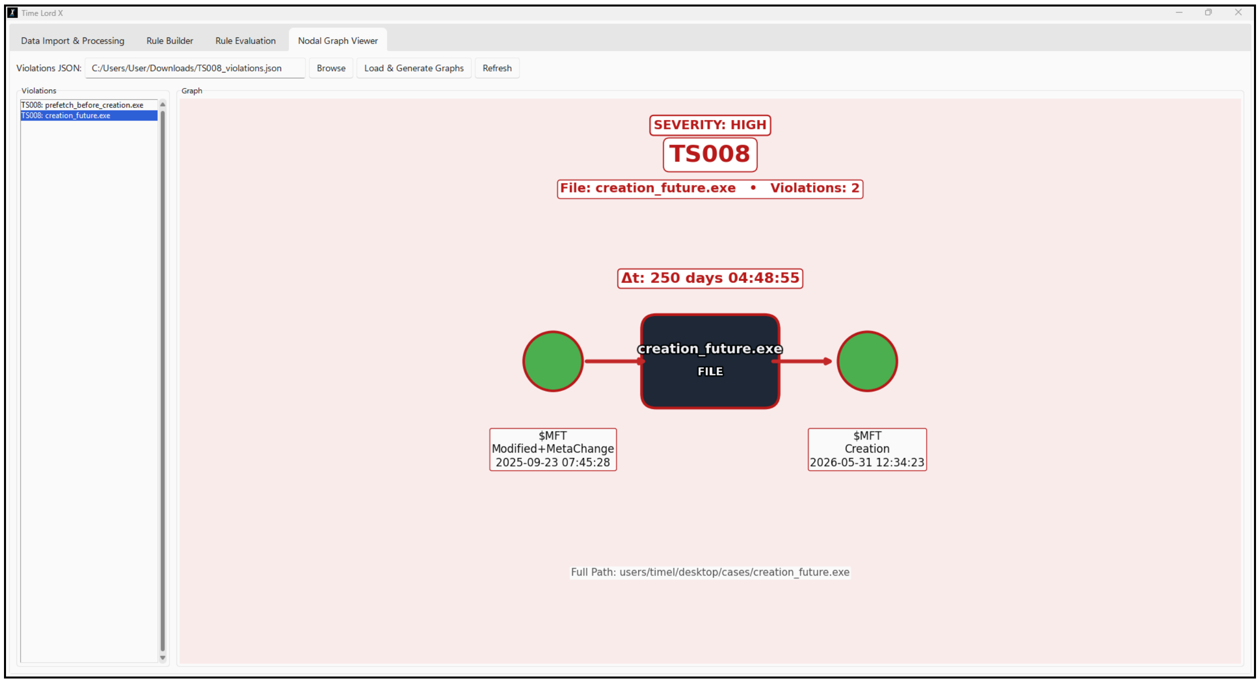


Violations JSON: violations_output_TS008.json

Name	Date modified	Type	Size
Today			
TS008_violations.json	29-Nov-25 3:08 PM	JSON Source File	4 KB
DF Project	29-Nov-25 2:10 PM	File folder	
Yesterday			
TimeStomp-Detection	28-Nov-25 5:33 PM	File folder	
kape	28-Nov-25 12:45 PM	File folder	
Earlier this week			
Telegram Desktop	27-Nov-25 10:16 PM	File folder	
VMs	26-Nov-25 4:47 PM	File folder	
School	26-Nov-25 4:00 PM	File folder	
MFTECmd	25-Nov-25 10:05 PM	File folder	

File name: TS008_violations.json
JSON files (*.json)

3) View the nodal graph representation for each violation by clicking on the desired entry.



Violations JSON: C:/Users/User/Downloads/TS008_violations.json

TS008: prefetch_before_creation.exe
TS008: creation_future.exe

Full Path: users/timel/desktop/cases/creation_future.exe