# Package 'mitre'

March 23, 2021

**Type** Package

**Title** Cybersecurity MITRE Standards Data and Digraphs

**Version** 0.5.2

**Maintainer** Humbert Costas <humbert.costas@gmail.com>

**Description** Extract, transform and load MITRE standards.
This package gives you an approach to cybersecurity data sets.
All data sets are build on runtime downloading raw data from MITRE public services.
MITRE <https://www.mitre.org/> is a government-funded research organization
based in Bedford and McLean. Current version includes most used standards as
data frames. It also provide a list of nodes and edges with all relationships.

**License** CC0

**URL** https://github.com/motherhack3r/mitre

**BugReports** https://github.com/motherhack3r/mitre/issues

**Encoding** UTF-8

**LazyData** true

**Imports** dplyr, plyr, tidyr, xml2, rvest, jsonlite, RJSONIO, stringr,
httr, yaml, rlang, igraph

**RoxygenNote** 7.1.1

**Suggests** visNetwork, knitr, rmarkdown, utils, shiny, curl, testthat
(>= 3.0.0)

**VignetteBuilder** knitr

**Depends** R (>= 2.10)

**Config/testthat/edition** 3

**NeedsCompilation** no

**Author** Humbert Costas [aut, cre]

**Repository** CRAN

**Date/Publication** 2021-03-22 23:00:08 UTC

# R topics documented:

as_igraph . . . 2
buildAttckTactics . . . 3
createATTCKedges . . . 3
downloadRawData . . . 4
getAttckData . . . 5
getCAPECData . . . 5
getCARData . . . 6
getCPEData . . . 6
getCVEData . . . 7
getCWEData . . . 7
getLatestDataSet . . . 8
getNodeNeighbors . . . 8
getShieldData . . . 9
MapCommonproperties . . . 9
MapGroups . . . 10
MapMitigation . . . 11
MapRelations . . . 11
MapSoftware . . . 12
MapTactics . . . 12
MapTechniques . . . 13
omitDeprecated . . . 14
parseAttck.Groups . . . 14
parseAttck.Mitigation . . . 15
parseAttck.Relationships . . . 15
parseAttck.Software . . . 16
parseAttck.Tactics . . . 17
parseAttck.Techniques . . . 17
parseAttckmodel.group . . . 18
parseAttckmodel.miti . . . 19
parseAttckmodel.rels . . . 19
parseAttckmodel.soft . . . 20
parseAttckmodel.tact . . . 21
parseAttckmodel.tech . . . 21
parseRawData . . . 22

**Index** **23**

---

`as_igraph` *Given a mitre network it returns the same as igraph*

---

## Description

Given a mitre network it returns the same as igraph

### Usage

```
as_igraph(mitrenet = getLatestDataSet()[["mitrenet"]], verbose = FALSE)
```

### Arguments

| | |
|---|---|
| mitrenet | MITRE network built with this package |
| verbose | default is FALSE |

### Value

igraph

---

## buildAttckTactics *Parse tactics*

---

### Description

Parse tactics

### Usage

```
buildAttckTactics(verbose = TRUE)
```

### Arguments

| | |
|---|---|
| verbose | Default set as FALSE |

### Value

data frame

---

## createATTCKedges *Create edges from ATTCK data frames*

---

### Description

Create edges from ATTCK data frames

### Usage

```
createATTCKedges(
  tactics,
  techniques,
  mitigations,
  groups,
  software,
  relations,
  verbose
)
```

### Arguments

| | |
|---|---|
| tactics | data.frame |
| techniques | data.frame |
| mitigations | data.frame |
| groups | data.frame |
| software | data.frame |
| relations | data.frame |
| verbose | Default set as FALSE |

### Value

data.frame

---

## downloadRawData *Download from official sources raw files saving them in [working_directory]/data-raw/*

---

### Description

Download from official sources raw files saving them in [working_directory]/data-raw/

### Usage

```
downloadRawData(verbose = FALSE)
```

### Arguments

| | |
|---|---|
| verbose | default is FALSE |

### Examples

```
## Not run:
mitre::downloadRawData(verbose = TRUE)

## End(Not run)
```

`getAttckData` *ETL process that download current attck definitions and return a list of data frames for each object. The list also contains a visNetwork object with ATT&CK objects as nodes and all relations as edges.*

## Description

ETL process that download current attck definitions and return a list of data frames for each object. The list also contains a visNetwork object with ATT&CK objects as nodes and all relations as edges.

## Usage

```
getAttckData(verbose = FALSE)
```

## Arguments

| | |
|---|---|
| `verbose` | Default set as FALSE |

## Value

list of data frames

`getCAPECData` *ETL process that download current CAPEC definitions and return a list with a data frame for CAPEC objects. The list also contains a visNetwork object with CAPEC objects as nodes and all relations as edges.*

## Description

ETL process that download current CAPEC definitions and return a list with a data frame for CAPEC objects. The list also contains a visNetwork object with CAPEC objects as nodes and all relations as edges.

## Usage

```
getCAPECData(verbose = FALSE)
```

## Arguments

| | |
|---|---|
| `verbose` | Default set as FALSE |

## Value

list of data frames

---

`getCARData` *ETL process that download current CAR definitions and return a list with a data frame for CAR objects. The list also contains a visNetwork object with CAR objects as nodes and all relations as edges.*

---

## Description

ETL process that download current CAR definitions and return a list with a data frame for CAR objects. The list also contains a visNetwork object with CAR objects as nodes and all relations as edges.

## Usage

```
getCARData(verbose = FALSE)
```

## Arguments

| | |
|---|---|
| `verbose` | Default set as FALSE |

## Value

list of data frames

---

`getCPEData` *ETL process that download current CPE definitions and return a list with a data frame for CPE objects. The list also contains a visNetwork object with CPE objects as nodes and all relations as edges.*

---

## Description

ETL process that download current CPE definitions and return a list with a data frame for CPE objects. The list also contains a visNetwork object with CPE objects as nodes and all relations as edges.

## Usage

```
getCPEData(verbose = FALSE)
```

## Arguments

| | |
|---|---|
| `verbose` | Default set as FALSE |

## Value

list of data frames

---

`getCVEData` *ETL process that download current CVE definitions and return a list with a data frame for CVE objects. The list also contains a visNetwork object with CVE objects as nodes and all relations as edges.*

---

## Description

ETL process that download current CVE definitions and return a list with a data frame for CVE objects. The list also contains a visNetwork object with CVE objects as nodes and all relations as edges.

## Usage

```
getCVEData(verbose = FALSE)
```

## Arguments

| | |
|---|---|
| `verbose` | Default set as FALSE |

## Value

list of data frames

---

`getCWEData` *ETL process that download current CVE definitions and return a list with a data frame for CVE objects. The list also contains a visNetwork object with CVE objects as nodes and all relations as edges.*

---

## Description

ETL process that download current CVE definitions and return a list with a data frame for CVE objects. The list also contains a visNetwork object with CVE objects as nodes and all relations as edges.

## Usage

```
getCWEData(verbose = FALSE)
```

## Arguments

| | |
|---|---|
| `verbose` | Default set as FALSE |

## Value

data frame

`getLatestDataSet` *Download latest R data sets from Github previously parsed with this package.*

## Description

Download latest R data sets from Github previously parsed with this package.

## Usage

```
getLatestDataSet(verbose = FALSE)
```

## Arguments

| | |
|---|---|
| `verbose` | default is FALSE |

## Value

list of standards and network

## Examples

```
## Not run:
mitredata <- mitre::getLatestDataSet(TRUE)

## End(Not run)
```

`getNodeNeighbors` *Returns a list of nodes and edges (neighbors) based on input node.*

## Description

Returns a list of nodes and edges (neighbors) based on input node.

## Usage

```
getNodeNeighbors(
  nodes = c("T1104"),
  direction = "both",
  mitrenet = getLatestDataSet()[["mitrenet"]],
  verbose = FALSE
)
```

### Arguments

| | |
|---|---|
| nodes | MITRE Standard Id as character vector |
| direction | value should be: "from", "to" or "both" |
| mitrenet | MITRE network built with this package |
| verbose | default is FALSE |

### Value

list of nodes and edges

---

## getShieldData *ETL process that download current shield definitions and return a list of data frames for each object. The list also contains a visNetwork object with SHIELD objects as nodes and all relations as edges.*

---

### Description

ETL process that download current shield definitions and return a list of data frames for each object. The list also contains a visNetwork object with SHIELD objects as nodes and all relations as edges.

### Usage

```
getShieldData(verbose = FALSE)
```

### Arguments

| | |
|---|---|
| verbose | Default set as FALSE |

### Value

list of data frames

---

## MapCommonproperties *Extract common properties from attack pattern object (parsed with RJSONIO::fromJSON)*

---

### Description

Extract common properties from attack pattern object (parsed with RJSONIO::fromJSON)

### Usage

```
MapCommonproperties(attack.obj = NA, domain = NA)
```

### Arguments

| | |
|---|---|
| attack.obj | list |
| domain | must be "pre-attack", "enterprise-attack" or "mobile-attack" |

### Value

data.frame compliant with CTI USAGE document

### Examples

```
## Not run:
sf <- "https://github.com/mitre/cti/raw/master/<domain>/<object>/<file>.json"
attack.pattern <- RJSONIO::fromJSON(sf)
df.common <- MapCommonproperties(attack.pattern)

## End(Not run)
```

---

## MapGroups *Extract Group properties from intrusion set object (parsed with RJSONIO::fromJSON)*

---

### Description

Extract Group properties from intrusion set object (parsed with RJSONIO::fromJSON)

### Usage

```
MapGroups(intrusion.set = NA, domain = NA)
```

### Arguments

| | |
|---|---|
| intrusion.set | list based on STIX |
| domain | must be "pre-attack", "enterprise-attack" or "mobile-attack" |

### Value

data.frame

`MapMitigation` *Extract Mitigation properties from course.action object (parsed with RJSONIO::fromJSON)*

## Description

Extract Mitigation properties from course.action object (parsed with RJSONIO::fromJSON)

## Usage

```
MapMitigation(course.action = NA, domain = domain)
```

## Arguments

| | |
|---|---|
| `course.action` | list based on STIX |
| `domain` | must be "pre-attack", "enterprise-attack" or "mobile-attack" |

## Value

data.frame

`MapRelations` *Extract object relationships from relationship object (parsed with RJSONIO::fromJSON)*

## Description

Extract object relationships from relationship object (parsed with RJSONIO::fromJSON)

## Usage

```
MapRelations(relationship = NA, domain = NA)
```

## Arguments

| | |
|---|---|
| `relationship` | list based on STIX |
| `domain` | must be "pre-attack", "enterprise-attack" or "mobile-attack" |

## Value

data.frame

`MapSoftware` *Extract Software properties from malware and tool object (parsed with RJSONIO::fromJSON)*

## Description

Extract Software properties from malware and tool object (parsed with RJSONIO::fromJSON)

## Usage

```
MapSoftware(software.obj = NA, domain = domain)
```

## Arguments

| | |
|---|---|
| `software.obj` | list based on STIX |
| `domain` | must be "pre-attack", "enterprise-attack" or "mobile-attack" |

## Value

data.frame

`MapTactics` *Extract Tactic properties from x-mitre-tactic object (parsed with RJSONIO::fromJSON)*

## Description

Extract Tactic properties from x-mitre-tactic object (parsed with RJSONIO::fromJSON)

## Usage

```
MapTactics(x.mitre.tactic = NA, domain = NA)
```

## Arguments

| | |
|---|---|
| `x.mitre.tactic` | list based on STIX |
| `domain` | must be "pre-attack", "enterprise-attack" or "mobile-attack" |

## Value

data.frame compliant with CTI USAGE document

## Examples

```
## Not run:
sf <- "https://github.com/mitre/cti/raw/master/<domain>/<object>/<file>.json"
x.mitre.tactic <- RJSONIO::fromJSON(sf)
df.ent.tact <- MapTactics(x.mitre.tactic, "enerprise-attack")

## End(Not run)
```

---

`MapTechniques` *Extract Technique properties from attack pattern object (parsed with RJSONIO::fromJSON)*

---

## Description

Extract Technique properties from attack pattern object (parsed with RJSONIO::fromJSON)

## Usage

```
MapTechniques(attack.pattern = NA, domain = NA)
```

## Arguments

| | |
|---|---|
| `attack.pattern` | list based on STIX |
| `domain` | must be "pre-attack", "enterprise-attack" or "mobile-attack" |

## Value

data.frame compliant with CTI USAGE document

## Examples

```
## Not run:
sf <- "https://github.com/mitre/cti/raw/master/<domain>/<object>/<file>.json"
attack.pattern <- RJSONIO::fromJSON(sf)
df.ent.tech <- MapTechniques(attack.pattern, "enerprise-attack")

## End(Not run)
```

`omitDeprecated` *Given a mitre network it returns the same without deprecated nodes*

## Description

Given a mitre network it returns the same without deprecated nodes

## Usage

```
omitDeprecated(mitrenet = getLatestDataSet()[["mitrenet"]], verbose = FALSE)
```

## Arguments

| | |
|---|---|
| `mitrenet` | MITRE network built with this package |
| `verbose` | default is FALSE |

## Value

list of nodes and edges

`parseAttck.Groups` *Read MITRE CTI Repository browsing domain directories to extract data from intrusion-set files, map variables from STIX to ATT&CK model and return tidy data.frame with Group variables.*

## Description

Read MITRE CTI Repository browsing domain directories to extract data from intrusion-set files, map variables from STIX to ATT&CK model and return tidy data.frame with Group variables.

## Usage

```
parseAttck.Groups(verbose = TRUE)
```

## Arguments

| | |
|---|---|
| `verbose` | default is FALSE |

## Value

data.frame

## Examples

```
## Not run:
df.groups <- parseAttck.Groups()

## End(Not run)
```

`parseAttck.Mitigation` *Read MITRE CTI Repository browsing domain directories to extract data from course-of-action files, build model and return tidy data.frame with Mitigation variables.*

## Description

Read MITRE CTI Repository browsing domain directories to extract data from course-of-action files, build model and return tidy data.frame with Mitigation variables.

## Usage

```
parseAttck.Mitigation(verbose = TRUE)
```

## Arguments

| | |
|---|---|
| `verbose` | default is FALSE |

## Value

data.frame

## Examples

```
## Not run:
df.mitigations <- parseAttck.Mitigation()

## End(Not run)
```

`parseAttck.Relationships` *Read MITRE CTI Repository browsing domain directories to extract data from relationship files, build model and return tidy data.frame with relationship variables.*

## Description

Read MITRE CTI Repository browsing domain directories to extract data from relationship files, build model and return tidy data.frame with relationship variables.

## Usage

```
parseAttck.Relationships(verbose = TRUE)
```

## Arguments

| | |
|---|---|
| `verbose` | default is FALSE |

## Value

data.frame

## Examples

```
## Not run:
df.relationships <- parseAttck.Relationships()

## End(Not run)
```

---

`parseAttck.Software` *Read MITRE CTI Repository browsing domain directories to extract data from malware and tool files, build model and return tidy data.frame with Software variables.*

---

## Description

Read MITRE CTI Repository browsing domain directories to extract data from malware and tool files, build model and return tidy data.frame with Software variables.

## Usage

```
parseAttck.Software(verbose = TRUE)
```

## Arguments

| | |
|---|---|
| `verbose` | default is FALSE |

## Value

data.frame

## Examples

```
## Not run:
df.software <- parseAttck.Software()

## End(Not run)
```

`parseAttck.Tactics` *Read MITRE CTI Repository browsing domain directories to extract data from x-mitre-tactic files, map variables from STIX to ATT&CK model and return tidy data.frame with Tactic variables.*

## Description

Read MITRE CTI Repository browsing domain directories to extract data from x-mitre-tactic files, map variables from STIX to ATT&CK model and return tidy data.frame with Tactic variables.

## Usage

```
parseAttck.Tactics(verbose = TRUE)
```

## Arguments

| | |
|---|---|
| `verbose` | default is FALSE |

## Value

data.frame

## Examples

```
## Not run:
df.tactics <- parseAttck.Tactics()

## End(Not run)
```

`parseAttck.Techniques` *Read MITRE CTI Repository browsing domain directories to extract data from attack-pattern files, map variables from STIX to ATT&CK model and return tidy data.frame with Technique variables.*

## Description

Read MITRE CTI Repository browsing domain directories to extract data from attack-pattern files, map variables from STIX to ATT&CK model and return tidy data.frame with Technique variables.

## Usage

```
parseAttck.Techniques(verbose = TRUE)
```

## Arguments

| | |
|---|---|
| `verbose` | default is FALSE |

## Value

data.frame

## Examples

```
## Not run:
df.techniques <- parseAttck.Techniques()

## End(Not run)
```

---

`parseAttckmodel.group` *Read MITRE CTI Repository files related to intrusion-set, extract data, map variables from STIX to ATT&CK model and return tidy data.frame.*

---

## Description

Read MITRE CTI Repository files related to intrusion-set, extract data, map variables from STIX to ATT&CK model and return tidy data.frame.

## Usage

```
parseAttckmodel.group(
 domain = sample(c("pre-attack", "ics-attack", "enterprise-attack", "mobile-attack"),
    1),
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| `domain` | must be "pre-attack", "enterprise-attack" or "mobile-attack" |
| `verbose` | default is FALSE |

## Value

data.frame

`parseAttckmodel.miti` *Read MITRE CTI Repository files related to course.action, extract data, map variables from STIX to ATT&CK model and return tidy data.frame.*

## Description

Read MITRE CTI Repository files related to course.action, extract data, map variables from STIX to ATT&CK model and return tidy data.frame.

## Usage

```
parseAttckmodel.miti(
 domain = sample(c("pre-attack", "ics-attack", "enterprise-attack", "mobile-attack"),
    1),
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| `domain` | must be "pre-attack", "enterprise-attack" or "mobile-attack" |
| `verbose` | default is FALSE |

## Value

data.frame

`parseAttckmodel.rels` *Read MITRE CTI Repository files related to relationship, extract data, map variables from STIX to ATT&CK model and return tidy data.frame.*

## Description

Read MITRE CTI Repository files related to relationship, extract data, map variables from STIX to ATT&CK model and return tidy data.frame.

## Usage

```
parseAttckmodel.rels(
 domain = sample(c("pre-attack", "ics-attack", "enterprise-attack", "mobile-attack"),
    1),
  verbose = TRUE
)
```

### Arguments

| | |
|---|---|
| domain | must be "pre-attack", "enterprise-attack" or "mobile-attack" |
| verbose | default is FALSE |

### Value

data.frame

---

## parseAttckmodel.soft *Read MITRE CTI Repository files related to malware and tool, extract data, map variables from STIX to ATT&CK model and return tidy data.frame.*

---

### Description

Read MITRE CTI Repository files related to malware and tool, extract data, map variables from STIX to ATT&CK model and return tidy data.frame.

### Usage

```
parseAttckmodel.soft(
 domain = sample(c("pre-attack", "ics-attack", "enterprise-attack", "mobile-attack"),
    1),
  verbose = TRUE
)
```

### Arguments

| | |
|---|---|
| domain | must be "pre-attack", "enterprise-attack" or "mobile-attack" |
| verbose | default is FALSE |

### Value

data.frame

`parseAttckmodel.tact` *Read MITRE CTI Repository files related to x-mitre-tactic, extract data, map variables from STIX to ATT&CK model and return tidy data.frame.*

## Description

Read MITRE CTI Repository files related to x-mitre-tactic, extract data, map variables from STIX to ATT&CK model and return tidy data.frame.

## Usage

```
parseAttckmodel.tact(
 domain = sample(c("pre-attack", "ics-attack", "enterprise-attack", "mobile-attack"),
    1),
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| `domain` | must be "pre-attack", "enterprise-attack" or "mobile-attack" |
| `verbose` | default is FALSE |

## Value

data.frame

`parseAttckmodel.tech` *Read MITRE CTI Repository files related to attack-pattern, extract data, map variables from STIX to ATT&CK model and return tidy data.frame.*

## Description

Read MITRE CTI Repository files related to attack-pattern, extract data, map variables from STIX to ATT&CK model and return tidy data.frame.

## Usage

```
parseAttckmodel.tech(
 domain = sample(c("pre-attack", "ics-attack", "enterprise-attack", "mobile-attack"),
    1),
  verbose = TRUE
)
```

### Arguments

| | |
|---|---|
| domain | must be "pre-attack", "enterprise-attack" or "mobile-attack" |
| verbose | default is FALSE |

### Value

data.frame

---

`parseRawData` *ETL process for all standards, it also create a list of nodes and edges representing the relationships between standard objects. It needs raw files downloaded from official MITRE repositories stored in a folder named "data-raw"; set downloadLatest parameter to TRUE and the function will create it for you.*

---

### Description

ETL process for all standards, it also create a list of nodes and edges representing the relationships between standard objects. It needs raw files downloaded from official MITRE repositories stored in a folder named "data-raw"; set downloadLatest parameter to TRUE and the function will create it for you.

### Usage

```
parseRawData(verbose = FALSE, downloadLatest = TRUE)
```

### Arguments

| | |
|---|---|
| verbose | default is TRUE |
| downloadLatest | default as FALSE, set to TRUE to download latest raw source files from official MITRE repositories |

### Value

list of two data frames: nodes and edges

### Examples

```
## Not run:
mitredata <- mitre::parseRawData(TRUE)

## End(Not run)
```

# Index

as_igraph, 2

buildAttckTactics, 3

createATTCKedges, 3

downloadRawData, 4

getAttckData, 5
getCAPECData, 5
getCARData, 6
getCPEData, 6
getCVEData, 7
getCWEData, 7
getLatestDataSet, 8
getNodeNeighbors, 8
getShieldData, 9

MapCommonproperties, 9
MapGroups, 10
MapMitigation, 11
MapRelations, 11
MapSoftware, 12
MapTactics, 12
MapTechniques, 13

omitDeprecated, 14

parseAttck.Groups, 14
parseAttck.Mitigation, 15
parseAttck.Relationships, 15
parseAttck.Software, 16
parseAttck.Tactics, 17
parseAttck.Techniques, 17
parseAttckmodel.group, 18
parseAttckmodel.miti, 19
parseAttckmodel.rels, 19
parseAttckmodel.soft, 20
parseAttckmodel.tact, 21
parseAttckmodel.tech, 21
parseRawData, 22