PILAR Batch Mode

May 2021

1 Introduction

PILAR may be run in batch mode; that is, without graphical interface. This mode is useful for:

- unattended evaluation of risks (e.g. overnight)
- reactive risk analysis (e.g. upon reporting of vulnerabilities)

PILAR reads

1. a valid license (a ".lic" file)
2. a configuration file (a ".car" file)
3. a system project (either from a ".mgr" file or from tables in a database)

Then you may apply changes to the project:

- load and apply one or more threat profile(s) (".tsv" files)
- import and set some values for assets, threats, ...
- generate reports in XML format, or in database tables.

The working scenario is stored in plan files (XML).

See an example:

```
<?xml version="1.0" encoding="iso-8859-1" ?>
<pilar_batch
  working_dir="C:\Users\jam\pilar"
  lib_dir="/Program Files/PILAR_4.1/bib_en"
  quantitative="false"
>
  <lic>/Program Files/PILAR_4.1/lics/L30000.lic</lic>
  <car>/Program Files/PILAR_4.1/STIC_en.car</car>
  <log>daily.log</log>

  <mgr>dmz.mgr</mgr>

  <apply-tsv>threats.tsv</apply-tsv>

  <report what="risk_down">
    risk_down.xml
  </report>
  <report what="risk_up" phase="target">
    risk_up_target.xml
  </report>
</pilar_batch>
```

2 Set up

This section covers the preliminary steps:

- .plan file header attributed

```
<?xml version="1.0" encoding="iso-8859-1" ?>
<pilar_batch attributes>
```

- <car> to set configuration file
- <lic> to set license file
- <log> to save an activity report
- <mgr> or <db> to load a risk project

2.1 Header

The header tag is

pilar_batch

The following attributes are available:

working_dir="..."

mandatory

working directory

lib_dir="..."

mandatory

library directory, to load .bgr, .lle, and .kb

quantitative="true | false"

optional; default: false

if TRUE, pilar uses a quantitative mode;
else, a qualitative mode

2.2 <car>: Configuration file

Mandatory. Only one.

Specifies the configuration file (the .car file) to drive the risk analysis.

Format:

```
<car> absolute path name </car>
```

2.3 <lic>: License

Mandatory. Only one.

Provides the license (the .lic file).

Format

```
<lic> absolute path name </lic>
```

2.4 <log>: Activity log

Optional. Default is system console.

A file to record the batch activity.

Format:

```
<log> path relative to working_dir </log>
```

2.5 <bgr> or <pl5>: Library

Mandatory.

Specifies the library (the .bgr file).

Format:

```
<bgr> path relative to lib_dir </bgr>
```

Since PILAR version 5, the BGR files have the extension PL5. The following tag is as synonymous:

```
<pl5> path relative to lib_dir </pl5>
```

2.6 <ext>: Library extensions

Optional: 0 or more.

Loads library extensions (.lle files) and specific protections (.kb files).

Format:

```
<ext> path relative to lib_dir </ext>
```

2.7 <ext_db>: Library extensions

Optional: 0 or more. You need a license that enables database access.

Loads library extensions from a database.

Format:

```
<ext_db user="..." password="..."> jdbc url </ext_db>
```

In order to read from and write to a database, PILAR uses a JDBC connector that must be provided:

```
<jar> absolute path to .jar connector </jar>
<class> Driver class in jar </class>
```

Example:

```
<jar> /java/mysql-connector-java-5.1.8-bin.jar </jar>
<class> com.mysql.jdbc.Driver </class>
```

2.8 <evl>: Security profiles

Optional: 0 or more.

Loads the security profile (.evl files).

Format:

```
<evl> path relative to lib_dir </ext>
```

2.9 Risk project

It is mandatory to load a project.

Formats:

```
<mgr> path relative to working_dir </mgr>

<db user="..." password="..."> jdbc url </db>
```

Examples:

```
<mgr>risk-model_2000-12-28.mgr</mgr>

<db user="john" password= "password">
  jdbc:mysql://localhost/risk_model
</db>
```

In order to read from and write to a database, PILAR uses a JDBC connector that must be provided:

```
<jar> absolute path to .jar connector </jar>
<class> Driver class in jar </class>
```

Example:

```
<jar> /java/mysql-connector-java-5.1.8-bin.jar </jar>
<class> com.mysql.jdbc.Driver </class>
```

3 Definitions

The user may define strings and reuse them later replacing. this is useful for repetitive strings.

Example

```
<define key="here">C:\tmpp\directory</define>

<copy file="$here$\baseline.mgr">
```

3.1 define

Format:

```
<define key=”ID” /> STRING </define>
```

3.2 replace

In attributes and text nodes.

```
$ID$
```

4 Execution

Once the working framework has been established with the header commands, you may specify a number of actions to execute. These are executed in the order they appear in the plan file. You may refer to it as “scripting”.

4.1 Set of asset values

Format:

```
<set asset=”id” dimension=”id” value=”v” />
```

Examples:

```
<set A="req" D="en:A" V="A" />
<set A="req" D="en:I" V="A" />
<set A="req" D="en:C" V="A" />
```

4.2 Valuation of domains

Format:

```
<import> filename in working_dir </import>
```

See document “XML Import”.

4.3 Valuation of assets

Format:

```
<import> filename in working_dir </import>
```

See document “XML Import”.

4.4 Valuation of threats

Format:

```
<import> filename in working_dir </import>
```

See document “XML Import”.

4.5 Valuation of safeguards

Format:

```
<import> filename in working_dir </import>
```

See document “XML Import”.

4.6 Set threat values

Pilar recognizes the following formats:

```
<set
    A=”asset code”
    Z=”threat code”
```

```
        freq="frequency value"
    />
```

```
    <set
        A="asset code"
        Z="threat code"
        deg="degradation value"
        D="dimension code"
    />
    <set
        A="asset code"
        Z="threat code"
        freq="frequency value"
        deg="degradation value"
        D="dimension code"
    />
```

```
    <set
        A="asset code"
        Z="threat code"
        step="seconds"
    />
    <set
        A="asset code"
        Z="threat code"
        freq="frequency value"
        step="seconds"
    />
```

Setting the frequency and/or the degradation of a given threat on a given asset.

Or the interruption step.

Frequency:

e.g. "1.5" (floating point, English notation)

Dimension code:

e.g. "C" for confidentiality

Degradation:

An integer between 0 and 100.

Step:

examples			
1 hour	"3600" "3600s" "60m" "1h"	1,5 hours	"5400" "5400s" "90m" "1h30m"

4.7 Apply tsv (threat standard values)

Format:

```
<tsv> filename in working_dir </tsv>
```

or

```
<apply-tsv> filename in working_dir </apply-tsv>
```

Loads the file, and applies the matching values. The content of the file is a standard TSV extension file.

4.8 Search for vulnerabilities

Format:

```
<search-vulnerabilities>
  filename in working_dir
</search-vulnerabilities>
```

See documentation on how PILAR searches for vulnerabilities using and NVD formats, and matching according to CPE names.

Matching vulnerabilities are assigned to assets.

4.9 Update vulnerabilities

Format:

```
<update-vulnerabilities>
  filename in working_dir
</update-vulnerabilities>
```

See documentation on how PILAR specifies CVSS refinements, and applies to vulnerabilities identified by the CVE-ID.

Matching vulnerabilities are updated.

4.10 Copy from another project

You may import the valuation of safeguards and security profiles from another project.

```
copy ::=
  <copy
    from="project.mgr"
  >
    [ safeguards ]
    [ evl ]
    [ domain ]
    [ phase ]
  </copy>

safeguards ::=
  <safeguards ... />

evl ::=
  <evl
    code="code" ...
  />

domain ::=
  <domain
    code="code"
```

```
    [ target=”code” ]
  />

phase ::=
  <phase
    code=”code”
    [target=”code” ]
  </phase>
```

Safeguards are copied, if the tag “safeguards” is present. See below.

Evaluation profiles are copied, if mentioned. Only those mentioned. See below.

Only values in the mentioned domains and phases are copied. If a target domain is specified, PILAR copies values from the domain in the from-project into the target domain in the working project. If a target phase is specified, PILAR copies from the phase in the from-project into the target phase in the working project. If the target domain or the target phase does not exist, they are created and added to the end of the current domains and phases, respectively.

4.10.1 Import safeguards

Safeguards are copied if the tag “safeguards” is present.

The “safeguards” node may specify some options to drive import:

```
<safeguards comments=”true” doubts=”true” na=”true”
sources=”true” />
```

If not specified, each attribute is TRUE.

4.10.2 Import security profiles (evl)

Evaluation profiles are copied, if mentioned. Only those mentioned.

The “evl” node may specify some options to drive import:

```
<evl code=”...” comments=”true” doubts=”true” na=”true”
sources=”true” propagate=”true” />
```

If not specified, each attribute is TRUE.

4.11 Remove phases

You may remove one or more phases:

```
remove ::=
  <remove>
    { phase }0+
  </remove>

phase ::=
  <phase
    code=”code”
```

```
/>
```

The phases are removed, along with the associated valuations.

4.12 Output: save the project

To save the working project with the modifications applied while scripting:

```
<output file="mgr file" />
```

The file is relative to the working directory. If no file is specified, the project is saved as "output.mgr".

In order to save on a database, use the following format:

```
<output db="url" user="..." password="..." />
```

as in the following example

```
<output db="jdbc:mysql://localhost/risk_model"
        user="john"
        password= "password
/>
```

4.13 Reports (RTF)

You may generate a report using a template. See specification of templates.

```
<report template="file" output="file" />
```

where the template and the output files are relative to the working directory.

5 XML Reporting

The XML syntax is presented using a variant of BNF notation, namely:

notation	meaning
`x | y`	choice “x” or “y”
`[ x ]`	zero or one occurrence of “x”; that is, “x” is optional
`{ x }0+`	zero or more occurrences of “x”
`{ x }1+`	one or more occurrences of “x”

5.1 Assets

Format:

```
<report what=”assets”>
  filename in working_dir
</report>
```

assets

```
file ::=
  <assets>
    { asset }0+
  </assets>

asset ::=
  <asset code="...">
    name
  </asset>
```

5.2 Threats

Format:

```
<report what=”threats”>
  filename in working_dir
</report>
```

threats

```
file ::=
  <threats>
    { threat }0+
  </threats>

threat ::=
  <threat code="...">
```

```
    name
  </threat>
```

5.3 Valuation of domains

Format:

```
<report what="valuation_domains">
  filename in working_dir
</report>
```

standard risk	continuity

file ::=
  <pilar_domain_values>
  { value }0+
  </pilar_domain_values>
 	
value ::=
  <set
  domain="code"
  dimension="code"
  [vl="level"]
  [vn="value"]
  >
  { why }0+
  [comment]
  </set>
 	
why ::=
  <why>
  criterion
  </why>
 	
comment ::=
  <comment>
  comment
  </comment>
	file ::=
	  <pilar_bcm_domain_values>
	  { value }0+
	  </pilar_bcm_domain_values>
	
	value ::=
	  <set
	  domain="code"
	  step="seconds"
	  [vl="level"]
	  [vn="value"]
	  >
	  { why }0+
	  [comment]
	  </set>
	
	why ::=
	  <why>
	  criterion
	  </why>
	
	comment ::=
	  <comment>
	  comment
	  </comment>

5.4 Valuation of assets

Format:

```
<report what="valuation_assets">
  filename in working_dir
</report>
```

standard risk	continuity

file ::=
  <**pilar_asset_values**>
  { value }0+
	file ::=
	  <**pilar_bcm_asset_values**>
	  { value }0+
	  </pilar_bcm_asset_values>

```
  </pilar_asset_values>

value ::=
  <set
      asset="code"
      dimension="code"
      [ vl="level" ]
      [ vn="value" ]
  >
    { why }0+
    [ comment ]
  </set>

why ::=
  <why>
    criterion
  </why>

comment ::=
  <comment>
    comment
  </comment>
```

```
value ::=
  <set
      asset="code"
      step="seconds"
      [ vl="level" ]
      [ vn="value" ]
  >
    { why }0+
    [ comment ]
  </set>

why ::=
  <why>
    criterion
  </why>

comment ::=
  <comment>
    comment
  </comment>
```

5.5 Valuation of threats

Format:

```
<report what="valuation_threats">
  filename in working_dir
</report>
```

standard risk

```
file ::=
  <pilar_threats>
    { value }0+
  </pilar_threats>

value ::=
  <set
      asset="code"
      threat="code"
      frequency="value"
  >
    { degradation }0+
  </set>

degradation ::=
  <degradation
      dimension="code"
      degradation="percent"
  </degradation>
```

continuity

```
file ::=
  <pilar_bcm_threats>
    { value }0+
  </pilar_bcm_threats>

value ::=
  <set
      asset="code"
      threat="code"
      frequency="value"
      step="seconds"
  >
  </set>
```

5.6 Valuation of safeguards

Format:

```
<report
    what="valuation_safeguards"
    [ domain="comma-separated list of domain codes" ] >
    [ phase="comma-separated list of phase codes" ] >
  filename in working_dir
</report>
```

If no "domain" is specified, all the domains are used.

If no "phase" is specified, all the phases are used.

safeguards

```
file ::=
  <pilar_safeguards>
    { per_domain | per_asset }0+
  </pilar_safeguards>

per_domain ::=
  <domain
      code="code"
  >
    { per_phase }1+
  </domain>

per_asset ::=
  <asset
      code="code"
  >
    { per_phase }1+
  </asset>

per_phase ::=
  <phase
      code="code"
  >
    { value }1+
  </phase>

value ::=
  <safeguard
      code="code"
      value="maturity"
  >
    [ comment ]
  </safeguard>
```

maturity

```
"L0" | "L1" | "L2" | "L3" | "L4" | "L5" | "" | "na" | "?"
```

5.7 Accumulated impact and risk

Format:

```
<report what="risk_down"
    [ domain="comma-separated list of domain codes" ] >
    [ phase="comma-separated list of phase codes" ] >
   filename in working_dir
</report>
```

If no "domain" is specified, all the domains are used.

If no "phase" is specified, all the phases are used. The potential values are reported under phase "null".

standard risk	**continuity**

`file ::=`
`  <risks_down>`
`  { per_phase }0+`
`  </risks_down>`
 	
`per_phase ::=`
`  <phase code="code">`
`  { item }0+`
`  </phase>`
 	
`item ::=`
`  <item`
`  asset="code"`
`  [ dcode="code" ]`
`  dimension="code"`
`  value="a_value"`
`  accumulated="a_value"`
`  threat="code"`
`  degradation="percent"`
`  frequency="value"`
`  impact="a_value"`
`  risk="b_value"`
`  >`
	`file ::=`
	`  <risks_down bcm="true">`
	`  { per_phase }0+`
	`  </risks_down>`
	
	`per_phase ::=`
	`  <phase code="code">`
	`  { item }0+`
	`  </phase>`
	
	`item ::=`
	`  <item`
	`  asset="code"`
	`  threat="code"`
	`  frequency="value"`
	`  step="seconds"`
	`  impact="a_value"`
	`  risk="b_value"`
	`  >`

dcode

Optional: the internal code used by Pilar to identify the dimension in any language. Please, note that the 'code' in attribute 'dimension' depends on the language.

percent

An integer between 0 and 100.

a_value

The value of the assets, and of the impact may be numerical (in quantitative analysis) or a value level (in qualitative analysis; e.g. “[7]”).

b_value

The risk value may be numerical (in quantitative analysis) or a criticality level (in qualitative analysis; e.g. "{4.8}").

5.8 Deflected impact and risk

Format:

```
<report what=”risk_up”
    [ domain=”comma-separated list of domain codes” ] >
    [ phase=”comma-separated list of phase codes” ] >
  filename in working_dir
</report>
```

If no “domain” is specified, all the domains are used.

If no “phase” is specified, all the phases are used. The potential values are reported under phase “null”.

standard risk	**continuity**

`file ::=`
`  <risks_up>`
`  { per_phase }0+`
`  </risks_up>`
 	
`per_phase ::=`
`  <phase code="code">`
`  { item }0+`
`  </phase>`
 	
`item ::=`
`  <item`
`  above="code"`
`  below="code"`
`  [ dcode=”code” ]`
`  dimension="code"`
`  value="a_value"`
`  threat="code"`
`  degradation="percent"`
`  frequency="value"`
`  impact="a_value"`
`  risk="b_value"`
`  >`
	`file ::=`
	`  <risks_up bcm="true">`
	`  { per_phase }0+`
	`  </risks_up>`
	
	`per_phase ::=`
	`  <phase code="code">`
	`  { item }0+`
	`  </phase>`
	
	`item ::=`
	`  <item`
	`  above="code"`
	`  below="code"`
	`  threat="code"`
	`  frequency="value"`
	`  step="seconds"`
	`  impact="a_value"`
	`  risk="b_value"`
	`  >`

dcode

Optional: the internal code used by Pilar to identify the dimension in any language. Please, note that the ‘code’ in attribute ‘dimension’ depends on the language.

percent

An integer between 0 and 100.

a_value

The value of the assets, and of the impact may be numerical (in quantitative analysis) or a value level (in qualitative analysis; e.g. “[7]”).

b_value

The risk value may be numerical (in quantitative analysis) or a criticality level (in qualitative analysis; e.g. "{4.8}").

5.9 Security profile (.evl)

Format:

```
<report evl=”evl filename in working_dir”
    [ domain=”comma-separated list of domain codes” ] >
    [ phase=”comma-separated list of phase codes” ] >
  filename in working_dir
</report>
```

If no “domain” is specified, all the domains are used.

If no “phase” is specified, all the phases are used.

security profile

```
file ::=
  <controls
      code=”code”
  >
    { per_domain }0+
  </controls>

per_domain ::=
  <domain
      code="code"
  >
    { per_phase }1+
  </domain>

per_phase ::=
  <phase code="code">
    { value }0+
  </phase>

value ::=
  <control
      code="code"
      value="percent"
  >
    [ comment ]
  </control>
```

percent

An integer between 0 and 100.

5.10 Delta reports

You may evaluate risks at some point, change some values, recalculate risks, and report only on differences. So you can analyse the consequences of some change of valuation of assets, threats o safeguards.

Pilar takes a snapshot of risks

```
<mark label=”name” />
```

and when a report is produced, it may be instructed to compare with a previous mark

```
<report diff="name" what="risk_down">
  risk_down.xml
</report>

<report diff="name" what="risk_up">
  risk_up.xml
</report>
```

The format is the standard one, but only changes are reported, showing both old and new values. For instance

```
<item asset="LAN" dcode="D" dimension="A"
  value=" " accumulated="[5]"
  threat="E.9" degradation="100"
  frequency="10.0"
  old_impact="" impact="[5]"
  old_risk="{1.2}" risk="{4.8}"
/>
```

“old_impact” is shown when the impact changes. “old_risk” is shown when risk changes. The “item” is shown when either impact or risk change.

There may be several marks; PILAR stores the risks at the labelled mark, and later on compares the current situation with the situation when the mark was established.

6 Database reporting

Results may be exported to a database. In order to write into a database, the script must specify a JDBC connector and a driver class (see section 2.9)

Database tables are reported in a separate document

DB_structures

The following tables may be generated from a batch plan:

Risk analysis: impact and risk

```
riskdown1
riskdown2
riskup1
riskup2
```

Business continuity: impact and risk

```
bcmdown1
bcmdown2
bcmup1
bcmup2
```

Security profiles

```
EVL_apps
EVL.value
```

6.1 Accumulated impact and risk

Format:

```
<report what=”risk_down”
    format=”sql” user=”...” password=”...”
    [ domain=”comma-separated list of domain codes” ] >
    [ phase=”comma-separated list of phase codes” ] >
  jdbc url
</report>
```

If no “domain” is specified, all the domains are used.

If no “phase” is specified, all the phases are used. The potential values are reported under phase “null”.

6.2 Deflected impact and risk

Format:

```
<report what=”risk_up”
    format=”sql” user=”...” password=”...”
    [ domain=”comma-separated list of domain codes” ] >
    [ phase=”comma-separated list of phase codes” ] >
  jdbc url
</report>
```

If no “domain” is specified, all the domains are used.

If no “phase” is specified, all the phases are used. The potential values are reported under phase “null”.

6.3 Security profile (.evl)

Format:

```
<report evl=”code”
    format=”sql” user=”...” password=”...”
    [ domain=”comma-separated list of domain codes” ] >
    [ phase=”comma-separated list of phase codes” ] >
 jdbc url
</report>
```

If no “domain” is specified, all the domains are used.

If no “phase” is specified, all the phases are used.