Management of TM1 Security: an introduction

Prepared:
January 2014
Last Updated:
April 2020

Created By:
Andreas Kugelmeier
Executive Consultant, FOPM
Planning Analytics Architect
IBM Data and AI Expert Labs
Mobile Phone: +1-215-384-7302
Email: kugelmeier@us.ibm.com

Notices & Disclaimers

U.S. Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM.
Information in these presentations and papers (including information relating to products that have not yet been announced by IBM) has been reviewed for accuracy as of the date of initial publication and could include unintentional technical or typographical errors. IBM shall have no responsibility to update this information. THIS document is distributed "AS IS" without any warranty, either express or implied. In no event shall IBM be liable for any damage arising from the use of this information, including but not limited to, loss of data, business interruption, loss of profit or loss of opportunity. IBM products and services are warranted according to the terms and conditions of the agreements under which they are provided.
Any statements regarding IBM's future direction, intent or product plans are subject to change or withdrawal without notice.
Performance data contained herein was generally obtained in a controlled, isolated environments. Customer examples are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual performance, cost, savings or other results in other operating environments may vary.
References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business.
Workshops, sessions and associated materials may have been prepared by independent session speakers, and do not necessarily reflect the views of IBM. All materials and discussions are provided for informational purposes only, and are neither intended to, nor shall constitute legal or other guidance or advice to any individual participant or their specific situation.
It is the customer's responsibility to insure its own compliance with legal requirements and to obtain advice of competent legal counsel as to the identification and interpretation of any relevant laws and regulatory requirements that may affect the customer's business and any actions the customer may need to take to comply with such laws. IBM does not provide legal advice or represent or warrant that its services or products will ensure that the customer is in compliance with any law.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products. IBM does not warrant the quality of any third-party products, or the ability of any such third-party products to interoperate with IBM's products. IBM expressly disclaims all warranties, expressed or implied, including but not limited to, the implied warranties of merchantability and fitness for a particular purpose.
The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents, copyrights, trademarks or other intellectual property right.
IBM, the IBM logo, ibm.com, Aspera®, Bluemix, Blueworks Live, CICS, Clearcase, Cognos®, DOORS®, Emptoris®, Enterprise Document Management System™, FASP®, FileNet®, Global Business Services ®, Global Technology Services ®, IBM ExperienceOne™, IBM SmartCloud®, IBM Social Business®, Information on Demand, ILOG, Maximo®, MQIntegrator®, MQSeries®, Netcool®, OMEGAMON, OpenPower, PureAnalytics™, PureApplication®, pureCluster™, PureCoverage®, PureData®, PureExperience®, PureFlex®, pureQuery®, pureScale®, PureSystems®, QRadar®, Rational®, Rhapsody®, Smarter Commerce®, SoDA, SPSS, Sterling Commerce®, StoredIQ, Tealeaf®, Tivoli®, Trusteer®, Unica®, urban{code}®, Watson, WebSphere®, Worklight®, X-Force® and System z® Z/OS, are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at: www.ibm.com/legal/copytrade.shtml.

- IBM's statements regarding its plans, directions, and intent are subject to change or withdrawal without notice at IBM's sole discretion.
- Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision.
- The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract.
- The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated here.

Document Version History

Date	Version	Author	Description
Dec. 2013	1.0	Andreas Kugelmeier	
Jan. 2014	1.1	Andreas Kugelmeier	Minor layout changes
Oct. 2015	1.11	Andreas Kugelmeier	Minor corrections
Jan 2016	1.12	Andreas Kugelmeier	Fix typo on page 7: “If rules [not TI] are used {in ElementSecurity etc.), a security metadata change - for example due to a hierarchy change (with corresponding/resulting security changes for parent and/or child nodes) or due to a new element being added to a hierarchy (like a new archived version for which READ access now to be granted to all applicable groups) – will always require running the ‘SecurityRefresh()’ command in TM1”
April 2016	1.13	Andreas Kugelmeier	Minor corrections (typographical errors)
5/19/2016	1.2	Andreas Kugelmeier	Important updates to section on ‘interaction of different security rights’
04/09/2020	1.3	Andreas Kugelmeier	Additional information re default security behaviour where security credentials from different roles/groups are ‘merged’ vs. cell security rules that can be set up to block/avoid the merging of security credentials.

Table of Contents

1. About this Document

This document introduces TM1 security, proven practices around TM1 security maintenance and related architectural concepts

2. Introduction to TM1 Security

TM1 Security is group-based[1]. User can be assigned to one or more groups.

By default - i.e. without special security rules - Planning Analytics (TM1) security credentials are determined based on the maximum access a user is granted across all the groups a user belongs to: if a user belongs to more than one group, user access credentials by default are determined by the intersection of the different group access credentials. Example: If user is granted READ access to an element or cell via one group but WRITE access via another group, the user will be granted WRITE access. If user is not granted access to an element or cell via one group but READ access via another group, the user will be granted READ access. It is important to note that this default behavior (the 'merging' or security credentials across groups) can be blocked because TM1 allows the definition of CellSecurity rules 'by group': one easily apply CellSecurity rules that prevent the 'merging' of credentials via rules that will block access to cells unless the user has been granted access to the cells as per credentials of one role/group. For information on how to implement such rules please refer to section 2.5.3 below.

Within a TM1 security group, access credentials are granted at the TM1 Object Level:

2.1 TM1 Object Level Security

You can assign object-level security for any non-administrative user group in TM1. That means you cannot assign security rights for the ADMIN, DataAdmin or SecurityAdmin groups. The rights for these groups are predefined and appear disabled in the TM1 Security Assignments dialog box.[2]

The object-level security rights for TM1 groups are:
Admin: Group has complete access to a cube, element, dimension, or other object.
Lock: Group can view and edit a cube, element, dimension, or other object and can permanently lock objects to prevent other users from updating them.
Read: Group can view a cube, element, dimension, process, or chore, but cannot perform operations on the object.
Reserve: Group can view and edit a cube, element, dimension, or other object, and can temporarily reserve objects to prevent other users from updating them.
Write: Group can view and update a cube, element, dimension, process, or chore.
None: Group cannot see a cube, element, dimension, process, or chore, and cannot perform operations on the object.

What does it mean when I assign ***Admin*** Rights to a
Cube: Members of the group can read, write, reserve, lock, and delete the cube. They can save public cube views. They can also grant security rights to other users for this object.
Element: Members of the group can access, update, reserve, lock, and delete the element. They can also grant security rights to other users for this object.

[1] User-based security can be implemented by putting user-specific groups in place

[2] The Users that belong to the ADMIN group have full Administrator rights on the TM1 instance. Users that belong to the DataAdmin group have full data-related administrative rights on the TM1 instance but cannot make security related changes. Users that belong to the SecurityAdmin group have Security Admin rights on the TM1 Instance but cannot view any non-security related data in the instance. Security Admins also cannot change their own security credentials to include non-Security data.

Dimension: Members of the group can add, remove, and reorder elements in the dimension, and can reserve or lock the dimension. They can save public dimension subsets. They can also grant security rights to other users for this object.

Application: Members of the group can see the application, use references within the application, and create both public and private references in the application. When a group has Admin privilege to an application, members of the group can set security privileges for all references and sub-applications within the application for other groups but not their own group.

Reference: Members of the group can use the reference, as well as update or delete the reference. They can publish private references, and privatize public references

What does it mean when I assign ***Reserve*** Rights to a

Cube: Members of the group have all privileges implied by Write permission, and can also reserve the cube to prevent other users from applying edits. The reservation can be removed either by the user who reserved the cube or by users who have Admin rights for the cube.*

Element: Members of the group have all privileges implied by Write permission, and can also reserve the element to prevent other users from updating cube cells identified by the element. The reservation can be removed either by the user who reserved the element or by users who have Admin rights for the element.*

Dimension: Members of the group have all privileges implied by Write permission, and can also reserve the dimension to prevent other users from redefining the dimension. The reservation can be removed either by the user who reserved the dimension or by users who have Admin rights for the dimension.*

*: A reservation expires automatically when the reserving user disconnects from the remote server or when the server shuts down.

What does it mean when I assign ***'None'*** Rights to a

Cube: Members of the group cannot see the cube in the Server Explorer, and thus cannot browse the cube.

Cell: Members of the group cannot see data for the cell.

Element: Members of the group cannot see the element in the Subset Editor or Dimension Editor, and cannot see the cells identified by the element when browsing a cube.

Dimension: Members of the group cannot see the dimension in the Server Explorer, and cannot browse a cube that contains the dimension.

Process: Members of the group cannot see the process in the Server Explorer, and thus cannot execute the process. Note: Privileges assigned to processes are ignored when a process is executed from within a chore.

Application: Members of the group cannot see the application or its contents in the Server Explorer.

Chore: Members of the group cannot see the chore in the Server Explorer, and thus cannot execute the chore.

Reference: Members of the group cannot see the reference in the Server Explorer.

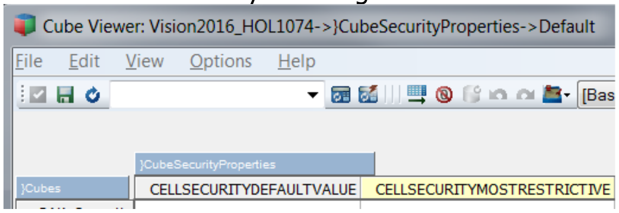
TM1 Object level security access credentials are stored in TM1 Security cubes:

- **Object Security Cubes:** }<ObjectType>Security.cub, i.e. }ApplicationSecurity.cub, }ChoreSecurity.cub, }CubeSecurity.cub, }DimensionSecurity.cub, }ProcessSecurity.cub
- **Element Security Cubes:** }ElementSecurity_<DimensionName>.cub
- **Cell Security Cubes:** }CellSecurity_<CubeName>.cub

2.2 Interaction of different security rights

- Rollup values remain unchanged by Security: the value of a non-leaf member does not change if some or all of its descendants are inaccessible.
- As mentioned above, if a user belongs to more than one group, user access credentials are determined by the intersection of the different group access credentials. Example: If user is granted READ access to an element or cell via one group but WRITE access via another group, the user will be granted WRITE access. If user is not granted access to an element or cell via one group but READ access via another group, the user will be granted READ access.
- Cube access credentials overrule cell security credentials in that cell security may not be less restrictive than cube security. Example: if a user is granted READ access to a cube, WRITE Access cannot be granted via CellSecurity.
- Cell level security may be more restrictive then the corresponding cube access credentials

- Dimension and Element Access credentials may be overruled by cell security credentials in that cell security may be less restrictive than element and dimension security. Example: If a user is granted READ access to an element via element security or just via dimension security (no element security in place) but WRITE Access to corresponding intersections in a cube via Cell Level Security, then WRITE Access is granted for this specific cube in accordance with the cell security configuration. Such a configuration is of particular use if for one and the same user group, write access to an element is only to be granted in some cases (in some cubes) or if in certain cases the write access is to be overruled with READ access only. Cell level security is typically driven by TM1 Cube Rules that are applied in the CellSecurity cube.
  -> Caution: Consequently, while one might not be able to see an element, cell level security can be wrongfully configured such that access to the data elements is granted regardless: if cell level security grants READ access to an intersection BUT corresponding element access is not granted (ElementSecurity value NONE or empty = no access), running a Perspectives query via DBRW formula and hard-coding the element into the DBRW formula will retrieve the desired value from the cube.
- The aforementioned behavior where Dimension and Element Access credentials may be overruled by Cell Security in either direction (more or less restrictive) can be changed per cube: In cube }CubeSecurityProperties, changing the value for measure 'CELLSECURITYMOSTRESTRICTIVE' to Yes for a cube will ensure that CellSecurity may restrict existing security further but will prevent CellSecurity from being able to remove/loosen restrictions. The setting can therefore be useful in that it allows simpler CellSecurity rules because it prevents CellSecurity from being able to overrule/override Element Security and hence element security checks may not have to be performed within the CellSecurity rules logic.

  
  Cube Viewer: Vision2016_HOL1074->}CubeSecurityProperties->Default
File
Edit
View
Options
Help
[Bas
}CubeSecurityProperties
}Cubes
CELLSECURITYDEFAULTVALUE
CELLSECURITYMOSTRESTRICTIVE

  But: if the setting is empty (default), CellSecurity will overrule element security in any direction

2.3 Sample Scenarios for Object Security configuration and interaction

Scenario 1: Assign a user

- READ access to the P&L cube and
- WRITE access to the dimensions/elements used in the cube.

In this scenario, the Read access of the P&L Cube overrides the Write access of the elements, and the user can view P&L cube data but cannot update the P&L cube data.

Scenario 2: The P&L What If Analysis cube contains the following dimensions: Account, Company, Cost Center, Geography, Version, Time Period, Currency. Suppose a user has WRITE access to the P&L What If Analysis cube, READ access to all of the elements in the Currency dimension, and WRITE access to all of the elements in the other dimensions. The elements in the Currency dimension identify every cell in the cube, and therefore – in the absence of cell security rules that could overrule the read access restriction - the user cannot update any cube data.

Scenario 3: A User has

- READ access to all Elements underneath Org Hierarchy node A (dimension Cost Center/Org),
- READ Access to Company 1 (dimension Company),
- READ Access to Geography Ohio (dimension Geography).

The P&L cube contains all the aforementioned dimensions. It follows that the user will only get access to intersections associated with Company 1 and Geography Ohio and Org node A. If Org node A contains data elements that are not associated with Company 1 and Geography Ohio, the user will not have access to the data.

Scenario 4: Several regional groups of users to read all data in the P&L What If Analysis cube. You also want each group to update data for their own BUs and Product IDs. To implement this security scheme, you could:

a) Create groups that reflect Contributors by BU & PID, like Groups '<BU> Contributors' & '<PID> Contributors', with Write access to the corresponding Customers and Products (ElementSecurity).
b) Grant each group Read access to the other Bus and PIDs (the ones that they should be able to read only) or create groups called 'All BUs READ' & 'All Product IDs READ' that have read access to all corresponding elements (ElementSecurity).
c) Grant each group Write access to the P&L What If Analysis cube or create a separate group called 'P&L What If Analysis Contributor'.
d) Add users to the appropriate groups: each user will get groups '<BU> Contributor', 'P&L What If Analysis Contributor', 'All Product IDs Read> and 'All BUs READ'.

Scenario 5: A user has access to customers from the San Francisco and the Oakland Region, but not to LA. You want the user to be able to roll up data for San Francisco and Oakland. Your hierarchy contains a Level called CA with children SF, Oakland & LA. It follows that if you give the user access to element CA, the user will be able to retrieve the Total CA number (which will include LA). TM1 hierarchy levels will always display the full value according to the hierarchical rollup. i.e. a consolidation is generally determined exclusively by the value of the immediate children/descendants of the parent and the element weight of the children. The value of a non-leaf element does not change if a user does not have access to some or all of its descendants.
=> Do not give the user access to node CA. Instead, have the user create a custom, user based (private) hierarchy rollup for San Francisco and Oakland or create an additional rollup in the hierarchy for the SF and Oakland region

2.4 Maintaining Security Metadata in TM1 security objects

Cube Security ('}CubeSecurity.cub'), Dimension Security ('}DimensionSecurity.cub'), Process Security ('}ProcessSecurity.cub'), and Element Security Data ('}ElementSecurity_<Dimension>.cub') should always be processed via TI instead of cube rules. If rules are used, a security metadata change - for example due to a hierarchy change (with corresponding/resulting security changes for parent and/or child nodes) or due to a new element being added to a hierarchy (like a new archived version for which READ access now to be granted to all applicable groups) – will always require running the 'SecurityRefresh()' command in TM1, effectively rendering all cached security settings invalid and hence renewing/refreshing all security credentials. A security refresh on large models will typically lead to a multi- to many minute lock of all user activity due to TM1 refreshing security access credentials for all active users and groups. If security is manually entered or processed via TI (and hence directly stored in the corresponding security cube), a security refresh is not necessary for such security changes. The security changes will propagate automatically and with only very short locks.

2.5 Securing Cell-Level Data

2.5.1 Cell Level Security Rules

TM1 Cell Level security is applied individually per cube via the content of a cube-specific cell level security cube with values like READ, WRITE, NONE for the security groups & cube cell intersections & subsections that are to be secured accordingly. The data content typically is determined via rules (rather than written to the cell level security cube via TI process[3]) in the Cell Security Cube. Cell Security Cube rules are re-evaluated automatically once a user refreshes a query; i.e. contrary to Cube, Dimension, Element Security etc., a SecurityRefresh() is not required to refresh credentials established by cell security rules.

2.5.2 When to apply Cell Security

IF the security requirements can be just as easily be met using cube, dimension & element security only then cell level security should be avoided. Particularly with very large cubes, cell level security can impose a query performance overhead depending on cell security requirements and cube size. With the release of TM1 10.2 however, cell security performance has greatly improved due to the ability to customize a cell security cube such that it only contains the dimensions that are used in determining cell level security (hence allowing for significantly faster retrieval of cell security metadata): Prior to TM1 10.2, cell level security cubes contained the same # of dimensions as the target cube plus the }Groups dimension. As a result, a comprehensive cell level security schema against very large cubes could have a significant effect on performance (by slowing down queries due to the large cell level security metadata having to be evaluated). As TM1 10.2, cell level security can be defined against a subset – as in a select # of dimensions - of the target cube, potentially resulting in significantly faster cell level security processing time.

In a scenario where for example the cost center dimension is used to primarily define security, cell security rules would need to be implemented on only a small subset of each cube(s) dimensions: READ-Only cubes will only need to be secured against their respective cost center dimension, meaning the Cell Security cube for READ only cubes will only have two dimensions ('<CostCenterDimension>.dim' and '}Groups.dim'). Write-Back cubes for What-If analysis need only to be secured against Cost Center, Time Period & Version resulting in a cell security cube with 4 dimensions ('<CostCenterDimension>.dim', '<TimePeriodDimension>.dim', '<VersionDimension>.dim' and '}Groups.dim').

2.5.3 Using cell security rules to prevent the merging of security credentials across groups/roles

Normally - i.e. without special security rules - Planning Analytics (TM1) security credentials are determined based on the maximum access a user is granted across all the groups a user belongs to. By implementing cell security, and by applying cell security rules that will determine security by group and that will block cell access to intersections that are not accessible to the group, this 'merging' of security credentials across a user's groups ca be prevented.

Example logic for such a sell security rule:

> IF Access to Product Member = READ AND Access to Account = READ in SAME Group, THEN Access =READ, otherwise NONE)

[3] The corresponding data volume would be very high if one were to process cell level security via TI. Also, because all or at a minimum a very large # of cells would have to be processed, the TI process would suffer from a very long runtime. More importantly, a security change at the element security level (which typically warrants corresponding changes at the cell security level) would require re-processing of the cell security credentials in each applicable cube, causing long processing times and corresponding user locks after only minor security changes.

Example Rule / Rule Template:

```
skipcheck;
[] = S:
        IF ( DB('}ElementSecurity_Account', !Account, !}Groups) @= 'READ'
          & DB('}ElementSecurity_Product', !Product, !}Groups) @= 'READ',
                'READ',
                ‘NONE’);
```

Or

```
skipcheck;
[] = S:
        IF ( DB('}ElementSecurity_Account', !Account, !}Groups) @= 'READ'
          & DB('}ElementSecurity_Product', !Product, !}Groups) @= 'READ',
                'READ',
                continue);
```

(with default CellSecurityDefaultValue in }CubeSecurityDefaults.cub set to NONE)

Note: the corresponding Cell Security Cubes (}CellSecurity_<CubeName>) do ONLY need to contain dimensions for which such a separation is needed. If for example Version/Scenario security is identical for all user groups that a particular user may have access to, those dimensions do not need to be part of the cell security model. Limiting the dimensions in the cell security models will result in query performance improvements (vs. applying the cell security rules to all dimensions with ElementSecurity).

3. Security Maintenance: Interaction with TM1 Security objects

A) via the TM1 UI:

advantage: built in UI
disadvantage: cumbersome to use, especially for element level security on larger dimensions with many hierarchy levels; no automation

B) input directly into TM1 security cubes

advantage: can use cube views
disadvantage: cumbersome to use, especially for element level security; no automation

C) Security Maintenance Model

advantages:

- semi-automated to fully automated security maintenance updates
- fast performance, especially when targeting specific security groups/roles, and specific objects (dimensions) that are to be updated
- re-configuration can be wide-ranging/far-reaching without having to be applied to TM1 model => apply new security only when ready
- easier configuration for hierarchy based element security
  ability to target specific security areas/ groups only (speed)

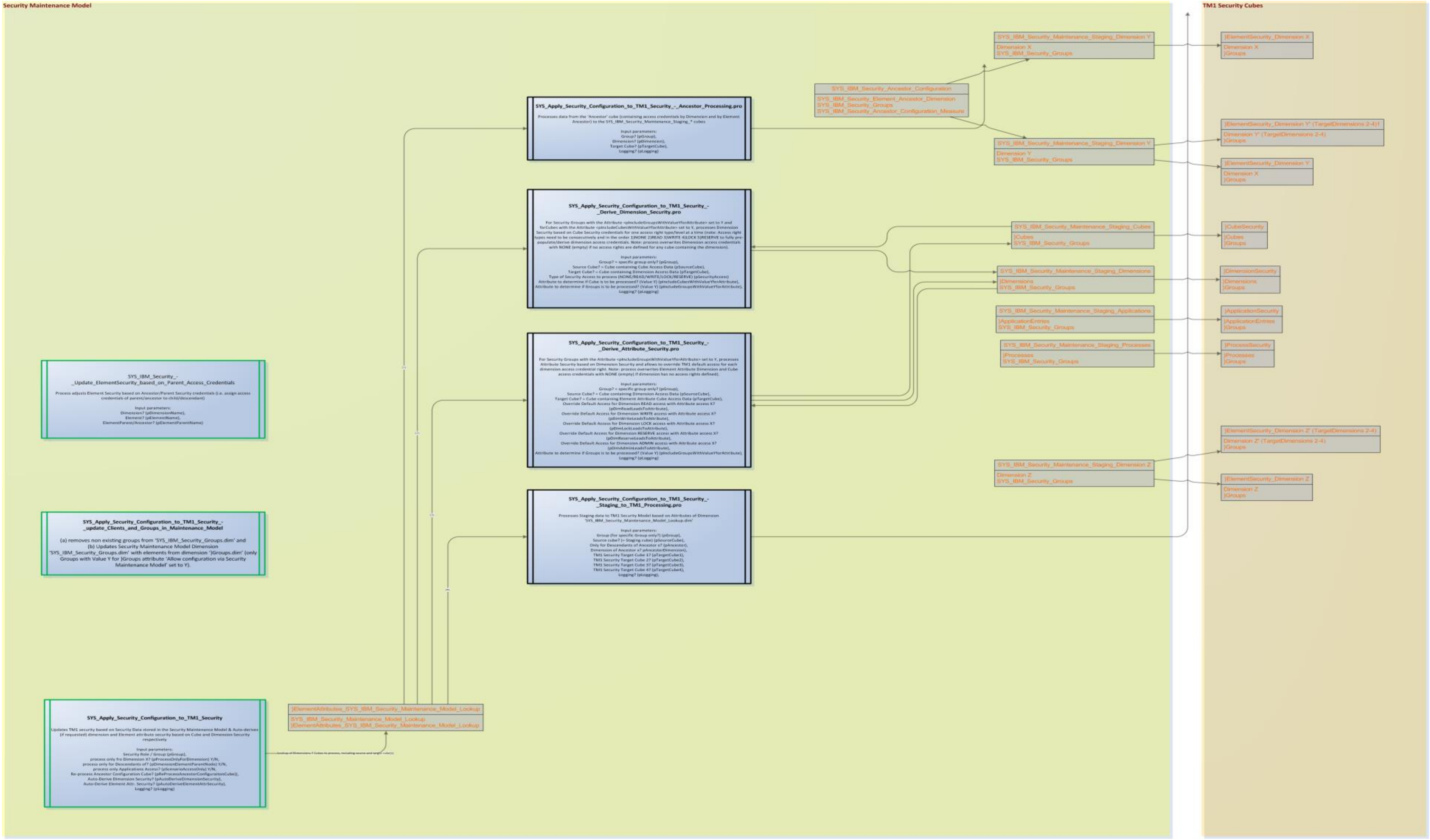
disadvantage: requires addtl. processes & Security Maintenance/Staging Model to be built.
However: Pre-built Security Maintenance and Staging Model components are available on demand.

3.1 Custom TM1 Security Staging & Maintenance Model

A Security Staging & Maintenance Model is designed to be

- a staging ground or mirror of the TM1 security model
- with additional cubes/features/external data loads that feed into the staging model to simplify security maintenance

=> we use the security staging & maintenance model to configure security and then push the data from the Security Maintenance Model to the TM1 Security Model



SYS_Apply_Security_Configuration_to_TM1_Security_-_Ancestor_Processing.pro

Processes data from the 'Ancestor' cube (containing access credentials by Dimension and by Element Ancestor) to the SYS_IBM_Security_Maintenance_Staging_* cubes

Input parameters:
Group? (pGroup),
Dimension? (pDimension),
Target Cube? (pTargetCube),
Logging? (pLogging)

SYS_Apply_Security_Configuration_to_TM1_Security_-_Derive_Dimension_Security.pro

For Security Groups with the Attribute <pIncludeGroupsWithValueYforAttribute> set to Y and forCubes with the Attribute <pIncludeCubesWithValueYforAttribute> set to Y, processes Dimension Security based on Cube Security credentials for one access right type/level at a time (note: Access right types need to be consecutively and in the order 1)NONE 2)READ 3)WRITE 4)LOCK 5)RESERVE to fully pre-populate/derive dimension access credentials. Note: process overwrites Dimension access credentials with NONE (empty) if no access rights are defined for any cube containing the dimension).

Input parameters:
Group? = specific group only? (pGroup),
Source Cube? = Cube containing Cube Access Data (pSourceCube),
Target Cube? = Cube containing Dimension Access Data (pTargetCube),
Type of Security Access to process (NONE/READ/WRITE/LOCK/RESERVE) (pSecurityAccess)
Attribute to determine if Cube is to be processed? (Value Y) (pIncludeCubesWithValueYforAttribute),
Attribute to determine if Groups is to be processed? (Value Y) (pIncludeGroupsWithValueYforAttribute),
Logging? (pLogging)

SYS_Apply_Security_Configuration_to_TM1_Security_-_Derive_Attribute_Security.pro

For Security Groups with the Attribute <pIncludeGroupsWithValueYforAttribute> set to Y, processes Attribute Security based on Dimension Security and allows to override TM1 default access for each dimension access credential right. Note: process overwrites Element Attribute Dimension and Cube access credentials with NONE (empty) if dimension has no access rights defined).

Input parameters:
Group? = specific group only? (pGroup),
Source Cube? = Cube containing Dimension Access Data (pSourceCube),
Target Cube? = Cube containing Element Attribute Cube Access Data (pTargetCube),
Override Default Access for Dimension READ access with Attribute access X? (pDimReadLeadsToAttribute),
Override Default Access for Dimension WRITE access with Attribute access X? (pDimWriteLeadsToAttribute),
Override Default Access for Dimension LOCK access with Attribute access X? (pDimLockLeadsToAttribute),
Override Default Access for Dimension RESERVE access with Attribute access X? (pDimReserveLeadsToAttribute),
Override Default Access for Dimension ADMIN access with Attribute access X? (pDimAdminLeadsToAttribute),
Attribute to determine if Groups is to be processed? (Value Y) (pIncludeGroupsWithValueYforAttribute),
Logging? (pLogging)

SYS_Apply_Security_Configuration_to_TM1_Security_-_Staging_to_TM1_Processing.pro

Processes Staging data to TM1 Security Model based on Attributes of Dimension 'SYS_IBM_Security_Maintenance_Model_Lookup.dim'

Input parameters:
Group (for specific Group only?) (pGroup),
Source cube? (= Staging cube) (pSourceCube),
Only for Descendants of Ancestor x? (pAncestor),
Dimension of Ancestor x? pAncestorDimension),
TM1 Security Target Cube 1? (pTargetCube1),
TM1 Security Target Cube 2? (pTargetCube2),
TM1 Security Target Cube 3? (pTargetCube3),
TM1 Security Target Cube 4? (pTargetCube4),
Logging? (pLogging),

SYS_IBM_Security_-_Update_ElementSecurity_based_on_Parent_Access_Credentials

Process adjusts Element Security based on Ancestor/Parent Security credentials (i.e. assign access credentials of parent/ancestor to child/descendant)

Input parameters:
Dimension? (pDimensionName),
Element? (pElementName),
ElementParent/Ancestor? (pElementParentName)

SYS_Apply_Security_Configuration_to_TM1_Security_-_update_Clients_and_Groups_in_Maintenance_Model

(a) removes non existing groups from 'SYS_IBM_Security_Groups.dim' and (b) Updates Security Maintenance Model Dimension 'SYS_IBM_Security_Groups.dim' with elements from dimension '}Groups.dim' (only Groups with Value Y for }Groups attribute 'Allow configuration via Security Maintenance Model' set to Y).

SYS_Apply_Security_Configuration_to_TM1_Security

Updates TM1 security based on Security Data stored in the Security Maintenance Model & Auto-derives (if requested) dimension and Element attribute security based on Cube and Dimension Security respectively

Input parameters:
Security Role / Group (pGroup),
process only fro Dimension X? (pProcessOnlyForDimension) Y/N,
process only for Descendants of? (pDimensionElementParentNode) Y/N,
process only Applications Access? (pScenarioAccessOnly) Y/N,
Re-process Ancestor Configuration Cube? (pReProcessAncestorConfiguraitonCube)),
Auto-Derive Dimension Security? (pAutoDeriveDimensionSecurity),
Auto-Derive Element Attr. Security? (pAutoDeriveElementAttrSecurity),
Logging? (pLogging)

Lookup of Dimensions ? Cubes to process, including source and target cube(s)
}ElementAttributes_SYS_IBM_Security_Maintenance_Model_Lookup
SYS_IBM_Security_Maintenance_Model_Lookup
}ElementAttributes_SYS_IBM_Security_Maintenance_Model_Lookup
SYS_IBM_Security_Ancestor_Configuration
SYS_IBM_Security_Element_Ancestor_Dimension
SYS_IBM_Security_Groups
SYS_IBM_Security_Ancestor_Configuration_Measure
SYS_IBM_Security_Maintenance_Staging_Dimension Y
Dimension X
SYS_IBM_Security_Groups
SYS_IBM_Security_Maintenance_Staging_Dimension Y
Dimension Y
SYS_IBM_Security_Groups
SYS_IBM_Security_Maintenance_Staging_Cubes
}Cubes
SYS_IBM_Security_Groups
SYS_IBM_Security_Maintenance_Staging_Dimensions
}Dimensions
SYS_IBM_Security_Groups
SYS_IBM_Security_Maintenance_Staging_Applications
}ApplicationEntries
SYS_IBM_Security_Groups
SYS_IBM_Security_Maintenance_Staging_Processes
}Processes
SYS_IBM_Security_Groups
SYS_IBM_Security_Maintenance_Staging_Dimension Z
Dimension Z
SYS_IBM_Security_Groups
}ElementSecurity_Dimension X
Dimension X
}Groups
}ElementSecurity_Dimension Y' (TargetDimensions 2-4)}
Dimension Y' (TargetDimensions 2-4)
}Groups
}ElementSecurity_Dimension Y
Dimension X
}Groups
}CubeSecurity
}Cubes
}Groups
}DimensionSecurity
}Dimensions
}Groups
}ApplicationSecurity
}ApplicationEntries
}Groups
}ProcessSecurity
}Processes
}Groups
}ElementSecurity_Dimension Z' (TargetDimensions 2-4)
Dimension Z' (TargetDimensions 2-4)
}Groups
}ElementSecurity_Dimension Z
Dimension Z
}Groups

Figure 1: Sample Architecture of a Security Maintenance Model