

An IBM Planning Analytics
Plug&Play Framework for accelerated
Configuration, Management & Maintenance
of TM1 Security

Last Updated:
April 2020

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

Copyright © 2015 by International Business Machines Corporation (IBM). No part of this document may be reproduced or transmitted in any form without written permission from IBM.

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
11/9/2014	1.0	Andreas Kugelmeier	
11/18/2014	1.1	Andreas Kugelmeier	Fix 'No of Dimension' Attribute and Rules for Dim 'TM1 Cube'
06/07/2015	1.11	Andreas Kugelmeier	Misc. minor changes to sample objects
01/26/2016	1.12	Andreas Kugelmeier	Fix typo: "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"
12/19/2017	1.2	Andreas Kugelmeier	Add information on Default/Sample Security Pre-Configurations
05/28/2019	1.3	Andreas Kugelmeier	Documentation for process SYS_IBM_SECURITY_- _SetElementSecurityAsPerDescendantsOrParents.pro Documentation for process SYS_IBM_SECURITY_- _Process_Defaults_For_Model_Security_Staging_-_Dimensions.pro
04/09/2020	1.4	Andreas Kugelmeier	Misc. updates

Table of Contents

1. About this Document	6
2. Introduction to TM1 Security	7
2.1 TM1 Object Level Security	7
2.2 Interaction of different security rights	8
2.3 Interaction of different security rights	10
2.4 Maintaining Security Metadata in TM1 security objects	11
2.5 Securing Cell-Level Data	11
2.5.1 Cell Level Security Rules	11
2.5.2 When to apply Cell Security	11
2.5.3 Using cell security rules to prevent the merging of security credentials across groups/roles	12
2.6 Notes on using TM1 Element Security in Cognos BI	13
2.7 Changes to TM1 Security in Version 10.2	13
3. A TM1 Security Maintenance Framework	14
3.1 Security Maintenance Model: Introduction	14
3.2 Security Maintenance Model Components: Overview	15
3.3 Security Staging Model and TM1 Security Model: Process Flow diagram	16
3.4 Security Staging Model and TM1 Security Model: Object Overview	17
3.4.1 Configure Security for new TM1 objects	17
3.4.2 Apply the new security metadata to TM1 Security	17
3.4.3 Apply Cube Cell Security (if applicable) and new Security (Staging) for new cubes	17
3.4.4 Apply Dimension Element Security	17
3.5 Security Staging Model and TM1 Security Model: Security Processing	18
3.5.1 Security Staging for TM1 Groups/Roles via Staging Groups	18
3.5.2 Security Staging for TM1 Cubes, Dimensions, Processes, Application Entries	18
3.5.3 Populating Security_Staging_ -_Dimensions	19
3.5.4 Applying Security Staging data to TM1 Security	19
3.5.5 Automation of Element Security Implementation for Dimensions	24
3.5.6 Element Security Maintenance Model: Details & Options	27
3.5.7 Element Security processing for Dependent Dimensions	30
3.5.8 Other security processes	31
3.5.9 Version Security	32
3.6 Securing Cell-Level Data	33
3.6.1 Cell Level Security Rules	33
3.6.2 When to apply Cell Security	33
3.6.3 Automation of (Cell)Security maintenance for cubes	33
4. Security Framework Objects incl. Sample Pre-Configuration	37



4.1	Security Framework Objects _____	37
4.2	Default (Sample) Pre-Configuration _____	37

1. About this Document

This document describes a TM1 plug&play asset framework for the configuration, maintenance and management of TM1 security. In Section 2, the document introduces standard features and concepts of TM1 security. Section 3 documents the aforementioned TM1 asset framework and how the asset may be configured to serve particular security configuration, maintenance and management needs.

Advantages of the TM1 security asset:

- semi- to fully-automated security maintenance
- fast performance
- ability to target specific security groups/roles and objects (dimensions) only
- security configuration staging: re-configuration can be wide-ranging/far-reaching & the reconfiguration may only be applied to TM1 once all changes have been completed and validated
- significantly easier configuration & maintenance for hierarchy based element security

Section 4 includes the objects to implement the security framework including sample configurations.

Note: the ElementSecurity framework does not yet support processing C-level element for PA V2.0 hierarchies (other than the 'legacy' default hierarchy).

2. Introduction to TM1 Security

TM1 Security is group-based¹. 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.

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.

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.

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

- 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 or less restrictive than the corresponding cube and/or element access credentials: Dimension and Element Access credentials may be overruled by cell security credentials in that cell security may be less or more restrictive than cube 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.³

Sample Scenarios for Object Security configuration and interaction:

Scenario 1: Assign a user

- READ access to a cube and
- WRITE access to the corresponding dimensions/elements.

In this scenario, the READ access of the cube overrides the WRITE access of the elements, and the user can view cube data but cannot update any data in the cube.

Scenario 2: A cube contains the following dimensions: Cost Center, Account, Time Period, Version, Currency, Measures. Suppose a user has WRITE access to the cube, READ access to all of the elements in the Currency dimensions, and WRITE access to all of the elements in the other dimensions. The elements in the Currency dimensions 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

- WRITE access to all Elements underneath Cost Center Hierarchy node R&D,
- WRITE access to Version Plan,
- WRITE access to Accounts under Node 'Misc.

If we now look at a cube that contains all the aforementioned dimensions, it follows that the user will only get access to intersections associated with R&D and Version Plan and 'Misc' Accounts.

Scenario 4: Several regional groups of users need to read all data in a cube. You also want each group to update data for their own Cost Centers. To implement this security scheme, you could:

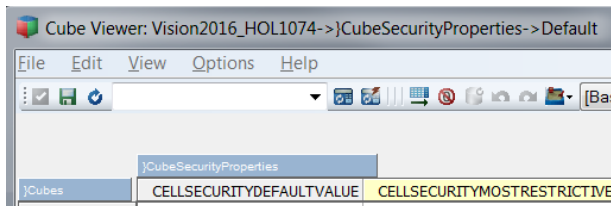
- a) Create groups that reflect Contributors by Cost Center, like Groups '<Group> <Region> Contributors', with WRITE access to the corresponding Cost Centers (ElementSecurity).
- b) Grant each group READ access to the other Cost Centers (the ones that they should be able to read only) or create groups called '<Group> <Region> READ' that have READ access to all corresponding elements (ElementSecurity).
- c) Grant each group WRITE access to the cube or create a separate group called '<CubeName> Input Reader'.
- d) Add users to the appropriate groups: each user will get groups '<Group> <Region> Contributors' & '<Group> <Region> READER'.

³ 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 (NONE or empty), running a Perspectives query via DBRW formula and hard-coding the element into the DBRW formula will retrieve the desired value from the cube.

Scenario 5: A user has access to Cost Centers from the US and the APAC region, but not to EMEA. You want the user to be able to roll up data for US and APAC. Your hierarchy contains a Level called 'All Regions'. If you give the user access to element 'All Regions', the user will be able to retrieve the 'All Regions' Value (which will include EMEA). 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 'All Regions'. Instead, have the user create a custom, user based (private) hierarchy rollup for US and APAC or create an additional rollup in the hierarchy for the those two regions.

2.3 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. This is the default behavior. At the cell (cube intersection level), this behavior can be changed. See Using cell security rules to prevent the merging of security credentials across groups/roles for details.
- 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 than 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.



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

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.

The security staging model allows the processing of security changes via TI, thereby providing a scalable rapid way for refreshing security.

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 process⁴) 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

⁴ 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.

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 can be prevented.

Example logic for such a cell security rule:

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

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).

2.6 Notes on using TM1 Element Security in Cognos BI

Inaccessible root members – parent-child hierarchies: In a parent-child hierarchy, if a root element is inaccessible, the highest level accessible descendants effectively become root elements, with level ordinal 0, and the level ordinals of their descendants are adjusted accordingly. This may result in the hierarchy becoming unbalanced.

BUT: As of TM1 10.2 FP1 and BI 10.2.1 FP1 updater kit one can have Cognos BI use so-called 'Filler Members' to 'fill' in for the TM1 members that the user does not have access to. Hierarchy Level Ordinals for the visible members will be the same as if the user were allowed to see all elements.⁵

Inaccessible non-root members: If a member is not accessible, but its parent is accessible, then all of the descendants will effectively become inaccessible.

Members with no accessible children: If a member is accessible, but none of its children are accessible, then the hierarchy is in effect unbalanced

2.7 Changes to TM1 Security in Version 10.2

Please refer to the following Links for information on changes and enhancements of TM1 10.2 Security:
[Security in TM1 10.2 Applications](#)
[Changes to TM1 Application deployment and rights-saving in TM1 10.2](#)
[Element Security and TM1 10.2 applications](#)

⁵ See <http://www-01.ibm.com/support/docview.wss?uid=swg21660287> for details
TM1 Asset Framework for Security Configuration, Maintenance & Management

3. A TM1 Security Maintenance Framework

3.1 Security Maintenance Model: Introduction

How can we interact with TM1 Security cubes?

A) via the TM1 UI

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

B) input directly into TM1 security cubes

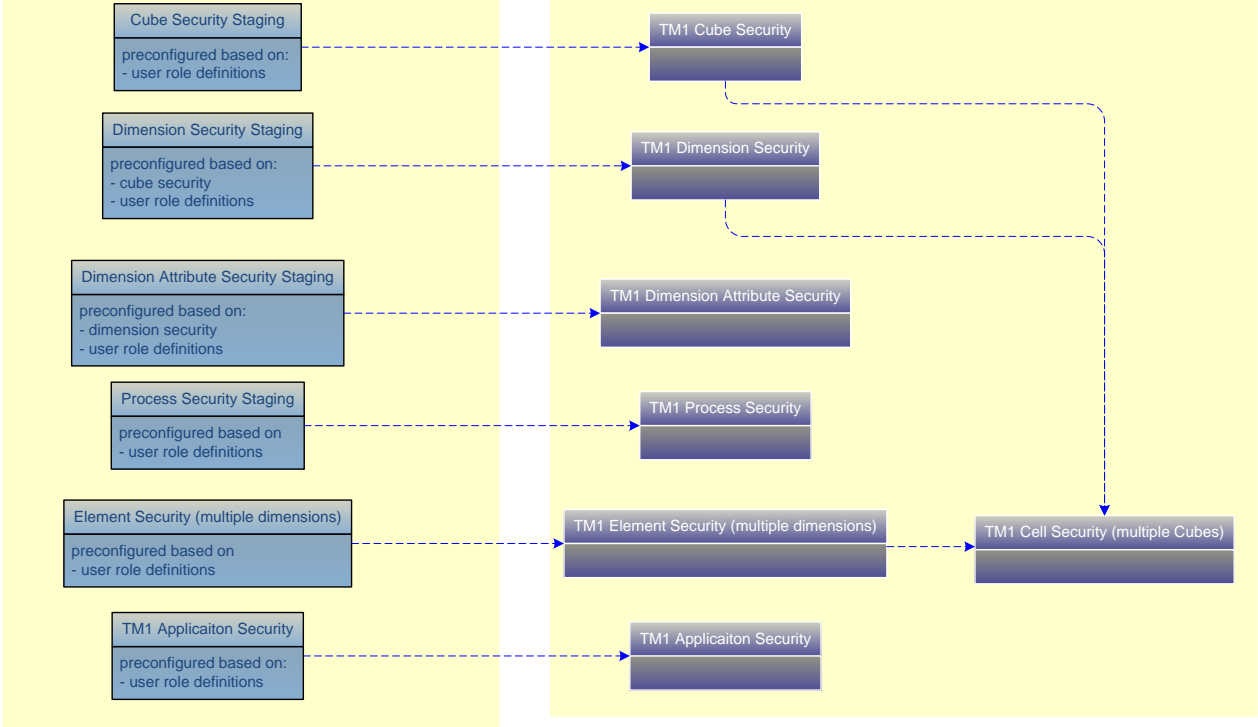
- advantage: can use cube views
- disadvantage: cumbersome to use; time-intensive maintenance especially for element level security on larger dimensions with many hierarchy levels; no automation

C) **Security Maintenance Model**

- advantages:
 - semi- to fully-automated security maintenance
 - fast performance & no need for purging the TM1 security cache (no need for running `SecurityRefresh()`)
 - ability to only target specific security groups/roles and objects (dimensions) that are to be updated
 - re-configuration can be wide-ranging/far-reaching & the re-configuration may only be applied to TM1 once all changes have been completed and validated
 - easier configuration for hierarchy based element security
- disadvantage: requires addtl. processes & Security Maintenance Model to be configured (but: configuration effort small, and: once Model & processes have been configured, ongoing maintenance is expected to be minimal)

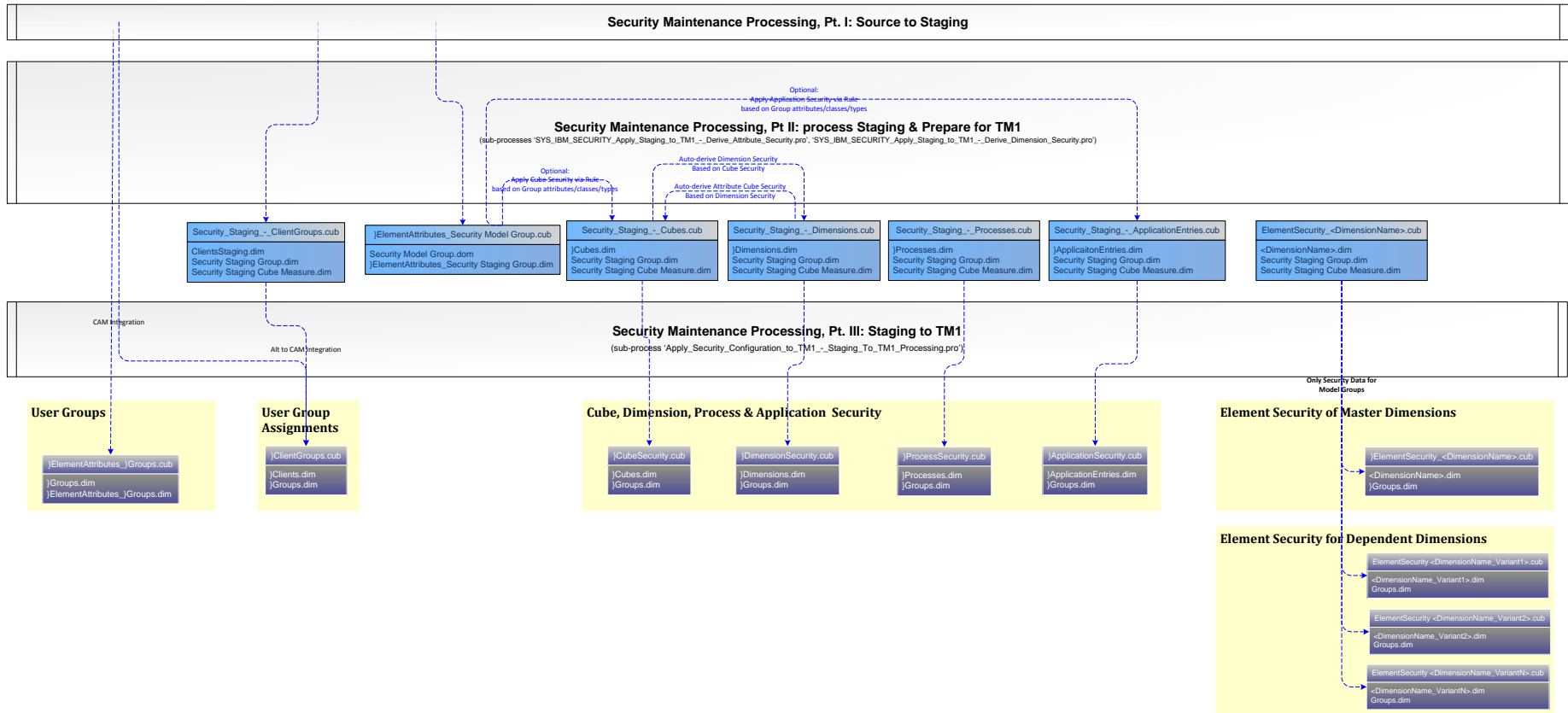
3.2 Security Maintenance Model Components: Overview

Security Staging Model: Allows (semi-)automatic security configuration based on security role definitions and/or external security metadata => Security is predefined in the staging model and when ready, it is pushed to TM1 security via one mouse click either as a whole or per section. Advantages: high flexibility, low maintenance, high performance for security changes





3.3 Security Staging Model and TM1 Security Model: Process Flow diagram



3.4 Security Staging Model and TM1 Security Model: Object Overview

3.4.1 Configure Security for new TM1 objects

- 'TM1 Dimension.dim': contains dimensions that are to be maintained in the context of the Dimension (Security) Maintenance Model.
- '}ElementAttributes_TM1 Dimension.cub': attributes cube to 'TM1 Dimension.dim'; used to configure element security for new or existing dimensions.
- 'Tm1 Cube.dim': contains cubes that are to be maintained in the context of the Cube (Security) Maintenance Model.
- '}ElementAttributes_TM1 Cube.cub': attributes cube to 'TM1 Cube.dim'; used to configure cube and cell security for new or existing cubes.

3.4.2 Apply the new security metadata to TM1 Security

- Process 'SYS_IBM_SECURITY_Apply_Staging_To_TM1.pro', based on settings in lookup & configuration cube '}ElementAttributes_TM1 Dimension.cub'

'SYS_IBM_SECURITY_Apply_Staging_To_TM1.pro' leverages the following sub-processes:

- 'SYS_IBM_SECURITY_Apply_Staging_To_TM1_-_Copy_Staging_Groups_Attributes.pro': copies Staging Group attribute values to the corresponding TM1 group attribute.
- 'SYS_IBM_SECURITY_Apply_Staging_To_TM1_-_Derive_Attribute_Security.pro': derives attribute cube security based on dimension access rights and based on the dimension access to attribute access mapping in the parameters of the process.
- 'SYS_IBM_SECURITY_Apply_Staging_To_TM1_-_Derive_Dimension_Security.pro': derives dimension access based on cube access rights. Process runs 3 times: (1) assigning NONE access to dimensions where cube access is NONE, (2) assigning READ access to dimensions where cube access is READ, (3) assigning WRITE access to dimensions where cube access is WRITE.
- 'SYS_IBM_SECURITY_Apply_Staging_To_TM1_-_Ancestor_Processing.pro': process to push Ancestor-Security-based security metadata from the Ancestor Security Staging Model 'ElementSecurity_Staging_AncestorBased.cub' to the corresponding Element Security Staging Model 'ElementSecurity_Staging_<DimensionName>.cub' cube.
- 'SYS_IBM_SECURITY_Apply_Staging_To_TM1_-_Staging_To_TM1_Processing.pro': process to push the security metadata from the staging model over to the TM1 security cubes.

3.4.3 Apply Cube Cell Security (if applicable) and new Security (Staging) for new cubes

- Process 'Manage Cube - Apply Security.pro', based on settings in lookup & configuration cube '}ElementAttributes_TM1 Cube.cub'.

'Manage Cube - Apply Security.pro' leverages the following sub-processes:

- Manage Cube - Apply Security - Construct Default Cell Security Rule
- Manage Cube - Apply Security - From Cube Security Staging to TM1 Security
- Manage Cube - Apply Security - From Reference TM1 Security Object To Cube Security Staging

3.4.4 Apply Dimension Element Security

- Process 'Manage Dimension - Apply Element Security.pro', based on settings in lookup & configuration cube '}ElementAttributes_TM1 Dimension.cub'.

'Manage Dimension - Apply Element Security.pro' leverages the following sub-processes:

- 'Manage Dimension - Apply Element Security - Construct Default Element Security Staging Rules.pro'
- 'Manage Dimension - Apply Element Security - Create or Destroy Element Security Staging Cube.pro'
- 'Manage Dimension - Apply Element Security - Create or Destroy ElementSecurity Cube.pro'

3.5 Security Staging Model and TM1 Security Model: Security Processing

3.5.1 Security Staging for TM1 Groups/Roles via Staging Groups

The security staging model contains staging groups. Security rules and configurations are performed against the staging groups (which exclude TM1 Admin, DataAdmin and Security Admin groups). Security Staging Groups (elements in 'Security Staging Group.dim') are mapped to TM1 CAM group IDs or TM1 User Groups or AD Groups in '}Groups.dim') via Staging Model Attribute 'CAM Security Group'. This attribute value is to be maintained manually or (once, if available in an external database) via TI process

Note: the 'Security staging Group' attribute 'Is Security Staging Group' is used to define 'true' staging groups. Only Staging Groups that are flagged with 'Is Security Staging Group' = Y will be processed when applying security staging data to TM1 security. => set 'Is Security Staging Group' to N for staging groups you do not want to process!

3.5.2 Security Staging for TM1 Cubes, Dimensions, Processes, Application Entries

- Cube Security is staged in cube 'Security_Staging_-_Cubes.cub' and processed to '}CubeSecurity.cub'
- Dimension Security is staged in cube 'Security_Staging_-_Dimensions.cub' and processed to '}DimensionSecurity.cub'
- Process Security is staged in cube 'Security_Staging_-_Processes.cub' and processed to '}ProcessSecurity.cub'
- Application Security is staged in cube 'Security_Staging_-_ApplicationEntries.cub' and processed to '}ApplicationSecurity.cub'
- Dimension Element Security is staged in cubes 'ElementSecurity_Staging_<MasterDimensionName>.cub' and processed to '}ElementSecurity_<MasterDimensionName>.cub'
- Ancestor-based Dimension Element Security is staged first in the cube 'ElementSecurity_Staging_AncestorBased.cub' and then automatically processed to cubes 'ElementSecurity_Staging_<MasterDimensionName>.cub' and from there processed to '}ElementSecurity_<MasterDimensionName>.cub'

3.5.3 Populating Security_Staging_-_Dimensions

It is not recommended to use rules for security defaults in Security_Staging_-_Dimensions, as this can lead to initial locks during dependency establishment (the existence of rules in this model may in some scenarios cause a lock on the }Dimensions dimension when the dependency is established the first time).

To address this, the process

SYS_IBM_SECURITY_-_Process_Defaults_For_Model_Security_Staging_-_Dimensions.pro

was put in place. Default security 'rules' are applied via processing (rather than cube rules), hence avoiding dependency locks. The process can be applied to all groups or a specific group (parameter pGroup) and/or for all dimensions or specific dimensions (parameter pDimension). Please refer to the TI-process comments for default security rules.

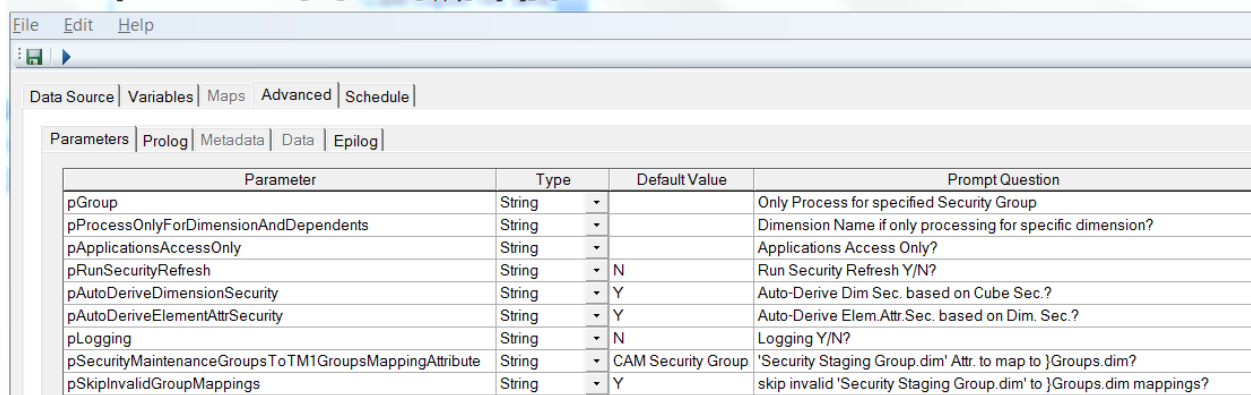
3.5.4 Applying Security Staging data to TM1 Security

Running the Process to Apply Security Staging Data / Update TM1 Security from Staging

Security Staging data is processed to TM1 security via the Process

'SYS_IBM_Security_Apply_Staging_to_TM1.pro' and its subsequent sub-processes

Turbo Integrator: Framework->SYS_IBM_SECURITY_Apply_Staging_to_TM1



Parameter	Type	Default Value	Prompt Question
pGroup	String		Only Process for specified Security Group
pProcessOnlyForDimensionAndDependents	String		Dimension Name if only processing for specific dimension?
pApplicationsAccessOnly	String		Applications Access Only?
pRunSecurityRefresh	String	N	Run Security Refresh Y/N?
pAutoDeriveDimensionSecurity	String	Y	Auto-Derive Dim Sec. based on Cube Sec.?
pAutoDeriveElementAttrSecurity	String	Y	Auto-Derive Elem.Attr.Sec. based on Dim. Sec.?
pLogging	String	N	Logging Y/N?
pSecurityMaintenanceGroupsToTM1GroupsMappingAttribute	String	CAM Security Group	'Security Staging Group.dim' Attr. to map to }Groups.dim?
pSkipInvalidGroupMappings	String	Y	skip invalid 'Security Staging Group.dim' to }Groups.dim mappings?

Parameters:

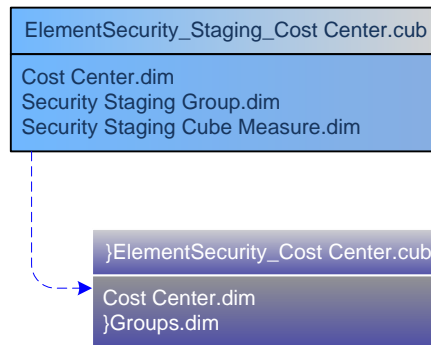
- **pGroup:**

Leave empty to process for all groups. Enter a group name to process for only the one group.

- **pProcessOnlyForDimension:**

Leave empty to update TM1 security for all security objects/dimensions. Enter the name of a master dimension to only update the security data for the master dimension and its dependent dimensions. Enter the name of a dependent dimension to only update the security data for the dependent dimension. Example: if pProcessOnlyForDimension is set to 'Cost Center', only the following processing will occur or Cost Center has no dependent dimensions⁶:

⁶ pAutoDeriveDimensionsecurity and pAutoDeriveElementAttributeSecurity both need to be set to N to only process the specified dimension



- **pApplicationsAccessOnly:**

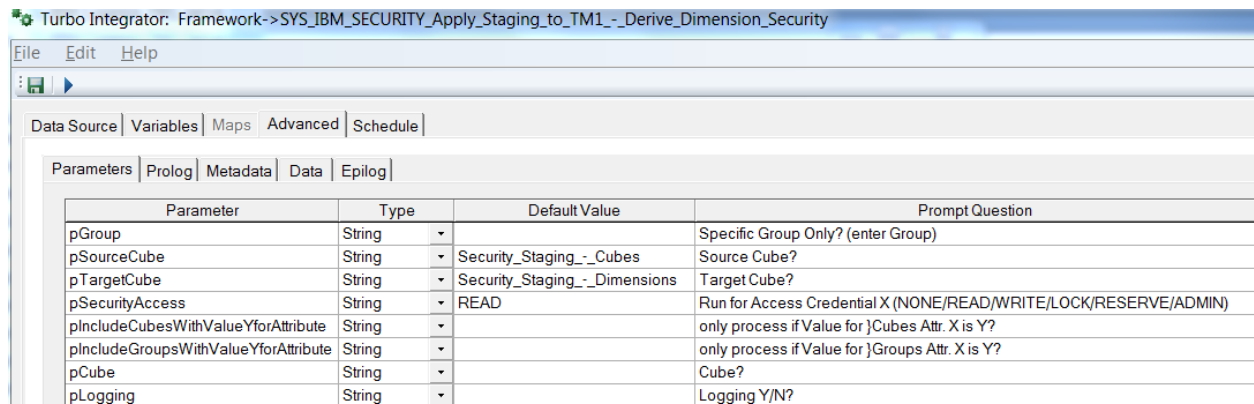
Set to Y to only update Applications Access and ignore all other (ElementSecurity, Dimension Security, Cube Security etc.) security objects. If Parameter is set to Y, parameter values for pProcessOnlyForDimension will be ignored.

- **pRunSecurityRefresh:**

Default = N, only set to Y if a complete purging of security cache information is desired/needed. Normally, leave this parameter at value N.

- **pAutoDeriveDimensionSecurity:**

Auto-Derive access to dimensions based on access credentials to the cubes in which the dimensions are used? If set to Y, the process will automatically derive Dimension Security (based on cube security) by invoking sub-process 'SYS_IBM_Security_Apply_Staging_to_TM1_-_Derive_Dimension_Security.pro':



Parameters:

- **pGroup:** Leave empty to process for all groups. Enter a group name to process for only the one group. pGroup is passed to sub-process from master process 'SYS_IBM_Security_Apply_Staging_to_TM1.pro'
- **pSourceCube:** set to staging cube for cube security
- **pTargetCube:** set to staging cube for dimension security
- **pSecurityAccess:** the cube security access credential to process to its equivalent dimension security access credential.

The Master process 'SYS_IBM_Security_Apply_Staging_to_TM1.pro' will run the sub-process 'SYS_IBM_Security_Apply_Staging_to_TM1_-_Derive_Dimension_Security.pro' three times in a row to first (re-)process dimensions security for cubes that a group does not have access

to, then to 'overwrite'/update the dimension security credentials for dimensions belonging to cubes that the group has READ access to (i.e. updating NONE with READ if applicable) and then to 'overwrite'/update the dimension security credentials for dimensions belonging to cubes that the group has WRITE access to (i.e. updating NONE & READ with WRITE if applicable) and then

- 1) pSecurityAccess = NONE
 - 2) pSecurityAccess = READ
 - 3) pSecurityAccess = WRITE
- **pCube:** enter a cube name to only process for a specific cube. If the sub-process 'SYS_IBM_Security_Apply_Staging_to_TM1_-_Derive_Dimension_Security.pro' is invoked by 'SYS_IBM_Security_Apply_Staging_to_TM1.pro', the parameter will be set to '' (empty)
 - **pIncludeGroupsWithValueYForAttribute:** Will only auto-derive element attribute cube security for groups with value Y for the attribute entered as pIncludeGroupsWithValueYForAttribute. Enter an attribute and populate the attribute values accordingly to limit auto-attribute security processing to specific Security Staging Group only.
 - **pIncludeCubesWithValueYForAttribute:** Will only auto-derive element attribute cube security for dimensions with value Y for attribute 'Allow Auto-Config of Dimension Security'. To process dimension security for all cubes, remove the value from pIncludeCubesWithValueYForAttribute (i.e. set pIncludeCubesWithValueYForAttribute to empty and save the process).
 - **pLogging:** will generate detailed debug log if set to Y
- **pAutoDeriveElementAttributeSecurity:** Auto-Derive Access to element attributes cubes based on access credentials to the corresponding dimension?

If set to Y, sub-process 'SYS_IBM_Security_Apply_Staging_to_TM1_-_Derive_Attribute_Security.pro' will be invoked:

Turbo Integrator: Framework->SYS_IBM_SECURITY_Apply_Staging_to_TM1_-_Derive_Attribute_Security

Parameter	Type	Default Value	Prompt Question
pGroup	String		Specific Group Only? (enter Group)
pSourceCube	String	Security_Staging_-_Dimensions	Source Cube?
pTargetCube	String	Security_Staging_-_Cubes	Target Cube?
pDimReadLeadsToAttribute	String	READ	Overwrite TM1 Default with X
pDimWriteLeadsToAttribute	String	READ	Overwrite TM1 Default with X
pDimLockLeadsToAttribute	String	READ	Overwrite TM1 Default with X
pDimReserveLeadsToAttribute	String	READ	Overwrite TM1 Default with X
pDimAdminLeadsToAttribute	String	READ	Overwrite TM1 Default with X
pIncludeGroupsWithValueYforAttribute	String		only process if Value for JGroups Attr. X is Y?
pLogging	String	N	Logging N or Y?
pIncludeDimensionsWithValueYForAttribute	String		only process if Value for JDimensions Attr. X is Y?

Parameters:

- **pGroup:** Leave empty to process for all groups. Enter a group name to process for only the one group. pGroup is passed to sub-process from master process 'SYS_IBM_Security_Apply_Staging_to_TM1.pro'

- **pSourceCube:** set to staging cube for dimension security
- **pTargetCube:** set to staging cube for cube security
- pDimREADLeadsToAttribute, pDimWriteLeadsToAttribute, pDimLockLeadsToAttribute, pDimReserveLeadsToAttribute, pDimAdminLeadsToAttribute: change according to desired defaults:
 - **pDimREADLeadsToAttribute:** the attributes cube security credentials if a group has READ credentials to the dimension
 - **pDimWriteLeadsToAttribute:** the attributes cube security credentials if a group has WRITE credentials to the dimension
 - **pDimLockLeadsToAttribute:** the attributes cube security credentials if a group has LOCK credentials to the dimension
 - **pDimReserveLeadsToAttribute:** the attributes cube security credentials if a group has RESERVE credentials to the dimension
 - **pDimAdminLeadsToAttribute:** the attributes cube security credentials if a group has ADMIN credentials to the dimension
- **pIncludeGroupsWithValueYForAttribute:** Will only auto-derive element attribute cube security for groups with value Y for the attribute entered as pIncludeGroupsWithValueYForAttribute. Enter an attribute and populate the attribute values accordingly to limit auto-attribute security processing to specific Security Staging Groups only.
- **pIncludeDimensionsWithValueYForAttribute:** Will only auto-derive element attribute cube security for dimensions with value Y for attribute 'Allow Auto-Config of ElementAttributes Security'. To process element attributes security for all dimensions, remove the value from pIncludeDimensionsWithValueYForAttribute (i.e. set pIncludeDimensionsWithValueYForAttribute to empty and save the process)
- **pLogging:** will generate detailed debug log if set to Y
- **pSecurityMaintenanceGroupsToTM1GroupsMappingAttribute:** if <> ' will use the value for 'Security staging group' attribute <pSecurityMaintenanceGroupsToTM1GroupsMappingAttribute> to map a security staging group (= staging group) to a CAM Group (= TM1 CAM Security Group).
- **pSkipInvalidGroupMappings:** if set to Y, will not attempt to process Staging Groups that are not mapped to a CAM/TM1 Security Group. Logfile Output will be created, but not errors will be generated. Is not set to Y, any incorrect mapping will lead to an error output.

NOTE:

The final processing of each staging cube credentials to the corresponding TM1 security is performed by sub-process 'SYS_IBM_Security_Apply_Staging_to_TM1_-_Staging_To_TM1_Processing.pro'. This process features two alternative methodologies to update the TM1 security cubes:

- A) **parameter pZeroOutTarget = N or empty:** copy **all** Security Staging Group data (including empty cells, i.e. if will copy non-zero value suppressed data) from each applicable security staging model cube to its corresponding TM1 security cube(s). With this setting, the TM1 Security cubes are **not zeroed-out** prior to the update. Only values corresponding to existing (and mapped) staging groups are overwritten. This is an applicable methodology for as long as the security staging cubes do not grow too large and for as long as the staging data groups are in synch with the TM1 (CAM) security groups.

- B) **parameter pZeroOutTarget = Y:** copy **only non-empty** Security Staging Group data (excluding empty cells) from each applicable security staging model cube to its corresponding TM1 security cube(s). With this setting, the TM1 Security cubes **are zeroed-out** prior to the update. Unless security is updated for only a specific group, security credentials for all TM1 groups (except Admin, DataAdmin, SecuritAdmin) is overwritten regardless of the available staging groups.

When pZerooutTarget is set to Y, the following security staging group design considerations need to be taken into account: ensure the 'Value' fields in all security staging cubes are properly fed (rule Feeders), if not, implement corresponding feeders or remove skipcheck in the security staging cubes that contain rule-derived security definitions. Removing skipcheck in the security staging cubes is a viable option because the staging cubes are not queried often, are typically not very large (exception: element security staging for very large dimensions).

To change the default behaviour, set parameter pZeroOutTarget to Y or N and then save the process.

3.5.5 Automation of Element Security Implementation for Dimensions

The Process 'SYS_IBM_Security_Apply_Staging_to_TM1.pro' will update the TM1 security objects (based on their corresponding staging cubes) based on the information in lookup cube 'ElementAttributes_TM1 Dimension.cub'.

Element Security Staging cubes and 'ElementSecurity_<DimensionName>.cub' cubes do not have to be set up manually but instead are configured via corresponding entries in 'ElementAttributes_TM1 Dimension.cub' (step 1 below) which then may be applied via process 'Manage Dimension – Apply Element Security.pro' (step 2).

Configuration Attributes **for creating/maintaining security configuration for a dimension object in 'ElementAttributes_TM1 Dimension.cub'**:

Is Active:

Set to Y to process object/dimension. Security will not be processed for inactive dimensions.

Is New Dimension:

Automatically generated: Y for dimension objects that do not exist yet, N for existing objects

Is Security Object:

Set to Y to process object/dimension. Security will not be processed for inactive dimensions.

Is Ancestor-based security:

Set to Y if security is to be defined at the C-Level and automatically applied to descendants of the c-levels.

Is Master Dimension:

If set to Y, the Element Security Staging cube for measure 'Security Staging Cube' (see below) will display 'ElementSecurity_Staging_<DimensionName>.cub';

If set to N, the Element Security Staging cube for measure 'Security Staging Cube' (see below) will display 'ElementSecurity_Staging_<MasterDimensionName>.cub' (deriving the corresponding master dimension from the value for measure 'Master Dimension').

Note that at the security staging level, Element Security may only be set up for Master Dimensions. See below for details.

Is Dependent Dimension:

If set to Y, the Element Security Staging cube for measure 'Security Staging Cube' (see below) will display 'ElementSecurity_Staging_<DimensionName>.cub';

Master Dimension:

If 'Is Dependent Dimension' = Y, then the master dimension set here will be used to populate the value for measure 'Security Staging Cube' with 'ElementSecurity_Staging_<MasterDimensionName>.cub', i.e. the Master Dimension Security Staging Cube will be used to stage Element Security for the dependent dimension.

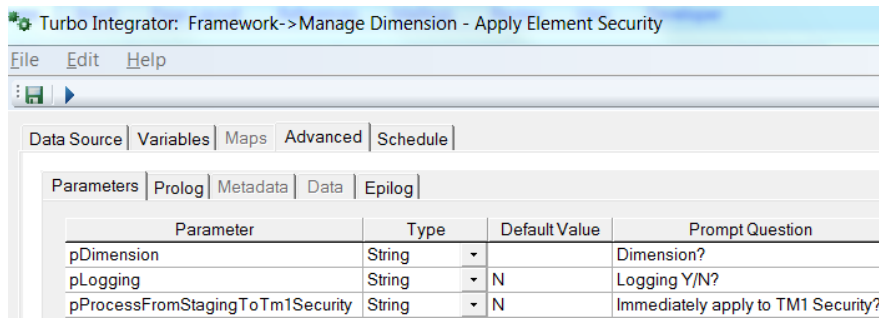
Apply ElementSecurity

Set to Y and then run process 'Manage Dimension – Apply Security.pro' to apply Element Security.

Set to N and then run process 'Manage Dimension – Apply Security.pro' to remove Element Security (should it exist, see documentation on measure 'Element Security is Applied' below).

When Element Security is removed via setting the value for this attribute to N and then running 'Manage Dimension – Apply Security.pro', existing element security cubes and element security staging cubes will be removed from the system.

'Manage Dimension - Apply Element Security.pro':



Element Security Is Applied:

Y if a cube if Element Security is applied, i.e. '}ElementSecurity_<DimensionName>.cub' exists. Set 'Apply Element Security' value to N and then run process 'Manage Dimension – Apply Security.pro' to remove Element Security.

Security Staging Cube:

Automatically generated: Source (Security Staging) Cube. Will be an 'ElementSecurity_Staging_<MasterDimension>.cub' cube for non System Dimensions. Will be a 'Security_Staging_-_<ObjectName/DimensionName>.cub' for system dimensions such as }Cubes.dim, }ApplicaitonEntries.dim, }Dimensions.diim, }Processes.dim etc.

Security Staging Cube Override:

Enter a different Element Security staging cube name to override the value that is auto-generated under attribute 'Security Staging Cube'

TM1 Security Cube:

Automatically generated: Target TM1 Security Cube. Will be an '}ElementSecurity_<Dimension>.cub' cube for non System Dimensions. Will be a '}<ObjectName/DimensionName>Security.cub' for system dimensions such as }Cubes.dim, }ApplicaitonEntries.dim, }Dimensions.diim, }Processes.dim etc.

Security Configuration is Valid:

Automatically Generated. Value N = Security Staging cube and/or TM1 Security cube do not exist;
Value Y = Security Staging cube and TM1 Security cube do exist;

Is Dependent Dimension & Master Dimension:

If 'SYS_IBM_Security_Apply_Staging_to_TM1.pro' runtime parameter 'pProcessOnlyForDimension' is used and a Master Dimension is specified as the parameter value for 'pProcessOnlyForDimension', the 'Is Dependent Dimension' value along with the value for 'Master Dimension' will be used to determine if the corresponding dependent dimension is to be processed.

If 'SYS_IBM_Security_Apply_Staging_to_TM1.pro' runtime parameter 'pProcessOnlyForDimension' is left empty, the 'Is Dependent Dimension' value will be used to determine if the dimension is to be processed. I.e. if 'Is Dependent Dimension' = Y, THEN process the dimension.

Is System Dimension:

If 'SYS_IBM_Security_Apply_Staging_to_TM1.pro' runtime parameter 'pProcessOnlyForDimension' is left empty, the 'Is System Dimension' value will be used to determine if the dimension is to be processed. I.e. if 'Is System Dimension' = Y, THEN process the dimension.

Example Configuration:

- o 'Cost Center.dim' as a Master Dimension with ancestor-based element security
- o 'Cost Center Group.dim' as a dependent dimension (dependent on Master Dimension 'Cost Center.dim'), but w/o security applied
- o 'Version.dim' as a Master Dimension with element security

All TM1-Dimension.dim attributes:

TM1 Dimension	Version	Cost Center	Cost Center Group	ApplicationEntries	Cubes	Dimensions	Processes
Is New Dimension	N	N	N	N	N	N	N
Is Active	Y	Y	Y	Y	Y	Y	Y
Is System Dimension	N	N	N	Y	Y	Y	Y
Is Master Dimension	Y	Y	N	N	N	N	N
SORTELEMENTSTYPE	ByHierarchy	ByHierarchy	ByHierarchy				
Is Dependent Dimension	N	N	Y	N	N	N	N
Master Dimension			Cost Center				
Hierarchy Level Attribute			Hierarchy Level				
Hierarchies			Total Company				
Weight for Descendants of APEX Nodes	0	0	0	1	0	0	0
Start Level	0	0	0	0	0	0	0
End Level	0	0	0	2	0	0	0
Is Security Object	Y	Y	N	Y	Y	Y	Y
Security Configuration is Valid	Y	Y	N	Y	Y	Y	Y
Is Ancestor-Based Security	Y	Y	N	N	N	N	N
Element Security Is Applied	Y	Y	N	N	N	N	N
Apply ElementSecurity	Y	Y	N	N	N	N	N
Security Staging Cube	ElementSecurity_Staging_Version	ElementSecurity_Staging_Cost Center	ElementSecurity_Staging_Cost Center	Security_Staging_-_ApplicationEntries	Security_Staging_-_Cubes	Security_Staging_-_Dimensions	Security_Staging_-_Processes
Security Staging Cube Override							
TM1 Security Cube	ElementSecurity_Version	ElementSecurity_Cost Center	ElementSecurity_Cost Center Group	ApplicationSecurity	CubeSecurity	DimensionSecurity	ProcessSecurity

Attributes applicable to security configuration only:

Security Default	Version	Cost Center	Cost Center Group	ApplicationEntries	Cubes	Dimensions	Processes
Is New Dimension	N	N	N	N	N	N	N
Is Active	Y	Y	Y	Y	Y	Y	Y
Is System Dimension	N	N	N	Y	Y	Y	Y
Is Master Dimension	Y	Y	N	N	N	N	N
Is Dependent Dimension	N	N	Y	N	N	N	N
Master Dimension			Cost Center				
Is Security Object	Y	Y	N	Y	Y	Y	Y
Security Configuration is Valid	Y	Y	N	Y	Y	Y	Y
Is Ancestor-Based Security	Y	Y	N	N	N	N	N
Element Security Is Applied	Y	Y	N	N	N	N	N
Apply ElementSecurity	Y	Y	N	N	N	N	N
Security Staging Cube	ElementSecurity_Staging_Version	ElementSecurity_Staging_Cost Center	ElementSecurity_Staging_Cost Center	Security_Staging_-_ApplicationEntries	Security_Staging_-_Cubes	Security_Staging_-_Dimensions	Security_Staging_-_Processes
Security Staging Cube Override							
TM1 Security Cube	ElementSecurity_Version	ElementSecurity_Cost Center	ElementSecurity_Cost Center Group	ApplicationSecurity	CubeSecurity	DimensionSecurity	ProcessSecurity

3.5.6 Element Security Maintenance Model: Details & Options

At the security staging level, Element Security may only be set up for Master Dimensions. Element Security is maintained at the element security staging level in Element Security Staging cubes named 'ElementSecurity_Staging_<MasterDimension>.cub'. Once maintenance has occurred at the security staging level it may be processed into (applied to) TM1 Security. If Element Security is needed for a dependent dimension (i.e. a dimension that per Dimension Maintenance Model 'ElementAttributes_TM1 Dimension.cub' is a dimension dependent on a specified master dimension), the corresponding 'ElementSecurity_<DependentDimensionName>.cub' will receive its data from the master dimension staging cube 'ElementSecurity_Staging_<MasterDimension>.cub' because a master dimension contains all elements – and hence all security information – of the dependent dimension.

Example: Cost Center Cube 'ElementSecurity_Staging_Cost Center.cub' contains dimensions

- Cost Center.dim: cost center master dimension
- Security Staging Group.dim: dimension with Security Staging Group.
- Measure dimension with
 - 'Value' as a string element and with values '' (empty), 'READ', 'WRITE' or others (values other than READ or WRITE will be translated into 'READ')
 - 'ValueFlag' as a numeric ('simple') element to perform calculations that will aide in further processing the security metadata for use in the TM1 Security Model. This is described below under (4)

Options for Defining Element Security via Element Security Staging cube:

Option 1: Define element access via cube rules in staging model.

An Example can be found in the configuration of Element Security for dimension 'Version.dim'

Option 2: Enter or process (from a DB) element access rights directly into staging model for both N- and C-Level elements

Option 3: Enter or process (from a DB) element access rights directly into staging model for N-level elements and leverage the rules template in the staging model to automatically derive C-Level access based on child access: if a group has access to all immediate children of a node (C-Level element), it will automatically gain access to the parent. Rule Template design: For each C-Level element, a rule against security staging cube measure 'ValueFlag' will determine if a Group has access to all immediate children of the C-Level Cost Center. If the answer is yes, READ access will be granted (per rule) to the C-Level ancestor. Element security is then completely defined for each group and now can be processed to the TM1 security Model cubes.

An Example can be found in the configuration of Element Security for dimension 'Operating Accounts.dim':

Value	Application Admin	Security Framework Admin	Sample Group 1	Sample Group 2
Gross Profit	READ	READ	READ	
Revenue	WRITE	WRITE	WRITE	
Cost of Goods Sold	WRITE	WRITE	WRITE	
Total Operating Expense	READ	READ		
Salaries & Wages	READ	READ		
Salaries	WRITE	WRITE		
Wages	WRITE	WRITE		
PERSONNEL EXPENSES	READ	READ	READ	
Misc. Employee Expense	WRITE	WRITE	READ	
Automobiles	WRITE	WRITE	WRITE	
TRAVEL EXPENSE	READ	READ		
Travel	WRITE	WRITE		WRITE
Accommodation	WRITE	WRITE		WRITE
Meal Allowance	WRITE	WRITE		
MARKETING EXPENSE	READ	READ		
Advertising	WRITE	WRITE		
Other Marketing Exp	WRITE	WRITE		
CORPORATE OVERHEADS	READ	READ		
IT Costs	WRITE	WRITE		
Communications	WRITE	WRITE		

Option 4: Enter or process (from a DB) element access rights for specific C-level elements directly into the 'ancestor' staging model and leverage the algorithm of the TI process 'SYS_IBM_SECURITY_Apply_Staging_To_TM1_-_Ancestor_Processing.pro' to apply the corresponding security access to all descendants of the C-Level node. The TI sub-process will push Ancestor-Security-based security metadata from the Ancestor Security Staging Model 'ElementSecurity_Staging_AncestorBased.cub' to the corresponding Element Security Staging Model 'ElementSecurity_Staging_<DimensionName>.cub' cube.

An Example can be found in the configuration of Element Security for dimension 'Cost Center.dim', with the entries in the 'AncestorBased' staging cube like

Cube Viewer: Framework->ElementSecurity_Staging_AncestorBased->Default

Security Staging Group	Ancestor1	Access1	Ancestor2	Access2	Ancestor3	Access3
Application Admin	Total	WRITE				
Security Framework Admin	Total Company	WRITE				
Sample Group 1	Corporate	WRITE	Total Company	READ		
Sample Group 2	Legal Entity A	WRITE	North America	READ		
Everyone						

leading to Element Security Staging like

Cube Viewer: Framework->ElementSecurity_Staging_Cost Center->Default

Security Staging Group	-- Total Company	+ Holding Company	-- North America	+ Legal Entity A	+ Legal Entity B	+ Legal Entity C
Application Admin	WRITE	WRITE	WRITE	WRITE	WRITE	WRITE
Security Framework Admin	WRITE	WRITE	WRITE	WRITE	WRITE	WRITE
Sample Group 1	READ	READ	READ	READ	READ	READ
Sample Group 2			READ	WRITE	READ	READ
Everyone						

and corresponding ElementSecurity

Options for processing security metadata / element access rights from a DB: For dimensions that are used to define primary security access (such as 'Cost Center', 'Customer', 'Product', ...), load security metadata per group/role and per dimension based on security role definitions from a Data Warehouse / Database. Such an approach will ensure that both DW and TM1 will share the same element access security credentials.

Option 1: build a custom TI process to load element security credentials into the staging cube as needed based on options 1-4 above.

Option 2: use the include master- and meta-data update framework to load security metadata from an external data source:

- Step 1: configure cube and data source via the cube to data source mapping utility 'TM1 Cube to Data Source Mapping.cub'. For detailed information on how to map a cube to a data source, please see the document 'TM1Asset_CubeDimensionAndDataLoad_Framework.docx'.
- Step 2: run process 'SYS_IBM_Cube_Update.pro' (generic process) against the SQL Data source and the element security staging cube 'ElementSecurity_Staging_<MasterDimensionName>.cub':

Parameters:

pCube: cube to update

pCubeDataSource: SQL or File or Cube Data Source to use for the cube update

Note: pCube and pCubeDataSource need to be mapped in 'TM1 Cube to Data Source Mapping.cub'

pLogging: set to Y for DEBUGGING only

pZeroOutTarget:

Y => Target Cube (the element security staging cube) will be zeroed out prior to update/load with data from data source

N => Target Cube (the element security staging cube) will not be zeroed out prior to update/load with data from data source

pAddElements: if set to Y, will add elements from the data source feed that may not exist in the dimension as per DimensionElementInsertDirect. If set to N, will skip load of records containing elements that do not exist.

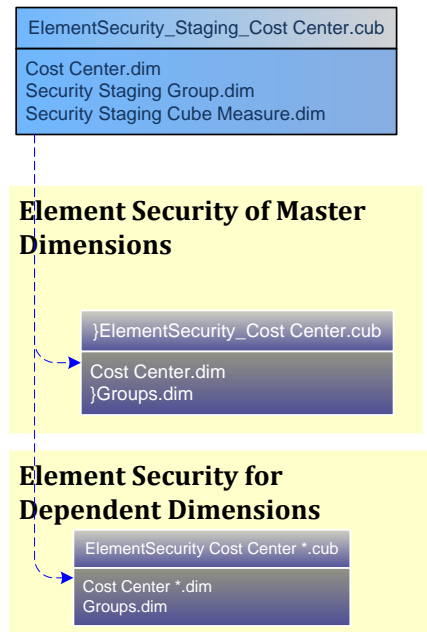
pUpdateData: if set to Y, will load/update cube data, if set to N, will skip data processing (hence process can be used only to add elements and/or to zero-out data)

pAttachOrphansToOrphansNode: If set to Y, will attach new elements to the default orphans node as specified in 'SYS_IBM_Control.cub'

3.5.7 Element Security processing for Dependent Dimensions

Dependent dimensions are directly related to master dimensions in that a dependent dimension's elements are all contained in the master dimension & the dependent dimension is directly derived/processed based on the Elements and Hierarchies in the Master dimension. A Dependent Dimension may not need to have ElementSecurity defined: Because the elements of a Dependent Dimension are all contained in the Master Dimension, Cell Security rules can be built against the ElementSecurity cube of the master dimension if applicable, i.e. if users still are to gain READ access to elements of the dimension but shall get restricted access at the cell (cube) level only). The cell-level security configuration utility based on '}ElementAttributes_TM1 Cube.cub' and applied using TI Process 'Manage Cubes – Apply Security.pro' will take this into consideration when automatically building a cell-security rule: if the cube contains a dependent cost center dimension but there is no '}ElementSecurity_<DependentCostCenterDimension>.cub', the cell security rules will reference the master cost center dimension '}ElementSecurity Cube '}ElementSecurity_<CostCenterMaster>.cub', if however '}ElementSecurity_<DependentCostCenterDimension>.cub' exists, the cell security cube rules will reference '}ElementSecurity_<DependentCostCenterDimension>.cub'.

For every dependent dimension with TM1 Element Security in place, ElementSecurity will be processed based on the security metadata in the master dimension as displayed in the following diagram:



3.5.8 Other security processes

- 'SYS_IBM_SECURITY_-_Update_ElementSecurity_based_On_Parent_Access_Credentials.pro'**: Process to run for a dimension if an element was added (for example a new historical version was created) and access should be granted based on Parent credentials. Process is used to update security during version management such that users automatically receive READ access to a new historical snapshot or version.
- 'SYS_IBM_SECURITY_-_Validate_Element_Access.pro'**: Process is used as a sub-process of any end- user-triggered TI process that performs a write-operation to determine if the user has proper access rights to the corresponding elements. Process will (1) check if the user is an Admin or DataAdmin and if yes, it will discontinue processing b/c access in this case is granted. Otherwise, it (2) will (i) create a subset of all groups (via 'SYS_IBM_SECURITY_-_Validate_Element_Access.pro') that have the access (pElementAccess) for the dimension (pDimensioName) element (pElementName) in '})ElementSecurity_<DimensionName>.cub' and (ii) then will check if the user belongs to any of the groups identified in (i). If yes, the process will end normally; if the user does not belong to any of the groups from (i) the process will end with an error.

Sample implementation of 'SYS_IBM_SECURITY_-_Validate_Element_Access.pro' as a sub-process:

Prolog of user-facing TI process

```
sSecurityAccess = "";
IF ( ExecuteProcess ( 'SYS_IBM_Security_Validate_Element_Access', pDimensionName, <DimensionName>, pElement,
                    <ElementName>, pElementAccess, 'WRITE')
    <> processexitnormal());
sSecurityAccess = 'N';
processbreak;
ENDIF;
```

First line of Epilog:

```
IF ( sSecurityAccess = 'N' );
    sSecurityAccessViolationMessage = 'TM1 User does not have sufficient access credentials to dimension "'
        | <DimensionName>
        | " element "'
        | <ElementName>
        | "'";
    # write to process log cube to allow access of error message via UI if needed #
    CellPutS ( sSecurityAccessViolationMessage, 'SYS_IBM_Process_Log', ProcessName, TM1ClientID, 'Last Status Message' );
    itemreject ( sSecurityAccessViolationMessage );
ENDIF;
```

- 'SYS_IBM_SECURITY_-_SetElementSecurityAsPerDescendantsOrParents.pro'**:

Generic Process for processing of

- Parent-based Security: in '})ElementSecurity_<DimensionName>.cub, apply Parent Access credentials to all descendants of the parent node.
- Leaf-Element-based Security: in '})ElementSecurity_<DimensionName>.cub, if a group has access to all leafs below a parent, allow the group to read the parent node.
- Parameters:
 - pTargetDimension: Dimension for which to process
 - pHierarchy: Dimension Hierarchy Container. If empty, then the default hierarchy container is used.
 - pGroup: Group to process for. If left empty, processing will occur for all groups.
 - pParent: Parent node to process for. If left empty, processing will occur for all parents.

- pParentNodeOrdinalFilter: optional, to apply only to parents of a / up to / a specified hierarchy level ordinal. Allowable values are = or < or > or <= or >= N (with N = a number), like >2 or <2 or <=2 or >=2 or =2. If left empty, no parent filter is applied (all parents will be processed).
 - pLogging: Y/N, for debug logging only
 - pLogFileDirectoryPath: path for debug log file
 - pDescendantsAccessAsPerParentAccess: set to Y to process descendants as per parent access. For each parent in a control subset, all (leaf&Parent) descendants will be processed via a parent-specific subset (new subset for each parent). IF Parent Access is <> NONE and a leaf has access = NONE => process parent access to leaf. IF Parent Access is = NONE => and a leaf has access <> NONE => no action, because access granted either via prior process or via prior parent
- pParentAccessAsPerDescendantsAccess: set to Y to process parents as per leaf-descendants access. For each parent in a control subset, all leaf descendants will be evaluated via a parent-specific subset (new subset for each parent). Only if all leaf level descendants have access <> NONE will parent access be set to 'READ'

3.5.9 Version Security

As per Staging cube rules in ElementSecurity_Staging_Version,

- all groups get READ access to version 'Actual',
- for FCST and Plan, WRITE access for all groups
- No default for what-if versions
- READ for all other versions (historical versions etc.)

3.6 Securing Cell-Level Data

3.6.1 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 content typically is to determined via rules (rather than written to the cell level security cube via TI process⁷) 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.

3.6.2 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 increased due to the ability with 10.2 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.

3.6.3 Automation of (Cell)Security maintenance for cubes

The security framework allows to auto-assign, re-assign (rebuild) or destroy cell security for new and existing cubes by means of

- a) a lookup cube ' }ElementAttributes_TM1 Cube.cub', allowing configuration of new cubes (naming, dimensionality, security regime) and**
- b) a TI process 'Manage Cubes – Apply Security.pro' that will be leveraging the lookup cube to determine runtime parameters and apply security accordingly.**

The lookup cube ' }ElementAttributes_TM1 Cube.cub' includes the configuration attributes:

Is new Cube:

automatically derived. Y => cube does not exist (yet); N => Cube exists

Cube Security:

Picklist with (1) Custom (2) based on Reference Security Object and Element, (3) based on Universal Access Credentials,

Reference Security Object:

Allows selecting a security object to use as a template for configuring security for the new cube. Picklist with Security Staging Cubes for (1) Cube Security, (2) Dimension Security, (3) Process Security, (4) Element Security according to }Cubes.dim subset 'Security Objects' (if the subset does not exist, please create the subset with the aforementioned security staging cube names).

⁷ 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.

Reference Security Dimension:

Allows selecting a dimension from the reference security object to use as a template for configuring security for the new cube: First Dimension of the reference Security Object.

Reference Security Element:

Allows selecting an element from the reference security object to use as a template for configuring security for the new cube: Picklist with elements from first dimension of the Reference Security object picked before. Example: if object }CubeSecurity.cub was selected, show list of cubes

Note: All 3 'Reference Security *' values must be populated in order to configure security based on a reference element. Example: you want to create a new cube 'SGA Input T&E' to model specific SGA input requirements for T&E. The cube is not part of any security access rights assigned to Subject Area Groups, i.e. you need to define new cube security settings for this cube. However, you do know that the cube shall have the same security as cube 'SGA Input' => As the reference security object, pick the cube security staging cube 'Security_Staging_-_Cubes', then - as the reference security dimension - pick dimension '}Cubes', and then - as the reference security object - pick the cube element 'SGA Input'.

Cell Security can be applied

Y if any dimension N was set to Y for 'use Dimension N for CellSecurity'

Cell Security Is Applied

Y/N (automatically generated if cell security cube exists)

Cell Security (User input as per Pick List):

N or empty => do not apply cell security

Custom => custom cell security rule

Additional options:

Other Cell Security 'Defaults' could be entered/allowed by modifying the rule in the picklist cube }PickList_}ElementAttributes_TM1 Cube, for example, instead of

```
['Cell Security'] = S: ('static:N:Custom:');
```

The picklist rule could read:

```
['Cell Security'] = S: ('static:N:Custom:Default Read-Only:Default Read/Write:Default Cost Center Read-Only:Default Cost Center Read/Write');
```

IF such default cell security rules shall be created, modify the process 'Manage Cube - Apply Security.pro' (see below) as per comment in the process prolog:

```
IF ( sCellSecurity @= 'Custom' );
```

```
    ### add code to run sub-process 'Manage Cube - Apply Security - Construct Default Cell Security Rule.pro' to construct default cell security rules
```

```
ENDIF;
```

Then, modify 'Manage Cube - Apply Security - Construct Default Cell Security Rule.pro' to adhere to the 'Cell Security' type settings. Note that 'Manage Cube - Apply Security - Construct Default Cell Security Rule.pro' contains sample code & methodologies to create default cell security rules.

No of Dimensions

automatically derived & for existing cubes only – not for new cubes

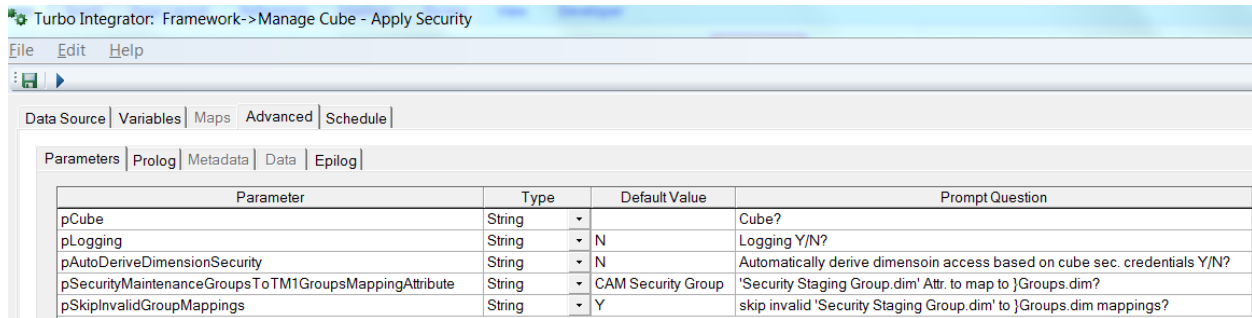
Dimension N (1-16)

automatically derived for existing cubes / User input as per Pick List for new cubes: Pick the Nth dimension for a new cube / Nth dimension of an existing cube

use Dimension N (1-16) for CellSecurity

Y/N picklist. Set to Y for dimensions that are to be used for defining CellSecurity

Note that a cube needs to exist in order to apply the desired security configurations. Once a cube has been created the 'Is New cube' flag in '})ElementAttributes_TM1 Cube.cub' will display the value 'N', meaning one can apply security to the new cube: run process 'Manage Cube – Apply Security.pro' against the cube configuration in '})ElementAttributes_TM1 Cube.cub' and Cell Security will be applied accordingly. **In order to remove cell security for an existing cube, set the value for 'Cell Security' in '})ElementAttributes_TM1 Cube.cub' to either empty or = 'N' and then run 'Manage Cube – Apply Security.pro': cell security for the cube will be removed (should it exist).** Please remove cell security for a cube prior to changing cell security for a cube (as in changing a cube from Default READ-Only Default READ/WRITE).



Parameters for process '**Manage Cube – Apply Security.pro**':

- **pCube:** Cube for which to apply security
- **Other parameters:** same as for '**SYS_IBM_Security_Apply_Staging_to_TM1.pro**'
- **Note re: pAutoderiveDimensionSecurity:** only the dimension security staging cube will be re-processed for the cube specified in parameter pCube. Please run '**SYS_IBM_Security_Apply_Staging_to_TM1.pro**' to process new dimension security from staging to TM1 security.

In the following sample configuration, Cube 'Operating Expense Allocations' is configured for Custom Cell Security Access and shall receive the same default security credentials as cube 'Operating Revenue & Expense':

Cube Viewer: Framework->ElementAttributes_TM1 Cube->Default

File Edit View Options Help

Default [Base]

ElementAttributes_TM1 Cube:Default	Operating Expense Allocations	Operating Revenue & Expense
Is new Cube	N	N
Cube Security	based on Reference Security Object and Element	Custom
Reference Security Dimension		n/a
Reference Security Object	}Cubes	n/a
Reference Security Element	Operating Revenue & Expense	n/a
Cell Security Can Be Applied	Y	Y
Cell Security is Applied	N	N
Cell Security	N	Custom
No of Dimensions		6
Dimension 1	Operating Accounts	Operating Accounts
Dimension 2	Cost Center	Cost Center
Dimension 3	Product	Product
Dimension 4	Version	Version
Dimension 5	Time Period	Time Period
Dimension 6	Allocations Measure	
Dimension 7		
Dimension 8		
Dimension 9		
Dimension 10		
Dimension 11		
Dimension 12		
Dimension 13		
Dimension 14		
Dimension 15		
Dimension 16		
use Dimension 1 for CellSecurity		N
use Dimension 2 for CellSecurity		Y
use Dimension 3 for CellSecurity		N
use Dimension 4 for CellSecurity		Y
use Dimension 5 for CellSecurity		Y
use Dimension 6 for CellSecurity		
use Dimension 7 for CellSecurity		
use Dimension 8 for CellSecurity		
use Dimension 9 for CellSecurity		
use Dimension 10 for CellSecurity		
use Dimension 11 for CellSecurity		
use Dimension 12 for CellSecurity		
use Dimension 13 for CellSecurity		
use Dimension 14 for CellSecurity		
use Dimension 15 for CellSecurity		
use Dimension 16 for CellSecurity		

4. Security Framework Objects incl. Sample Pre-Configuration

4.1 Security Framework Objects

Click [here](#)

4.2 Default (Sample) Pre-Configuration

The Security Framework includes pre-configured group/role types which are assigned as per 'Security Staging Group' attribute:

Staging Groups with Attribute '**Is Application Admin Group**'= 'Y': Application Admin group(s), with read/write access to all non-security configuration models and corresponding dimensions, as well as to all applicable (master-) processes (sub-processes may not be directly readable/accessible for Application Admin Groups).

Staging Groups with Attribute '**Is Security Framework Admin Group**'= 'Y': Can write to (as in 'configure') security staging cube configurations and read TM1 Security cube configurations (which are populated based on staging configurations), and run processes to update security as per Security Staging Model.

Staging Groups with Attribute '**Is Security Framework Operator Group**'= 'Y': Can read security staging cube and TM1 Security cube configurations and run processes to update security as per Security Staging Model. Cannot change security properties.

Staging Groups with Attribute '**Is Framework Modeler Group**' = 'Y': Can write to (as in 'configure') the Metadata-based Cube-, Dimension-, Hierarchy-, and Data-Load Framework.

Staging Groups with Attribute '**Is Framework Reader Group**' = 'Y': Can configurations pertaining to the Metadata-based Cube-, Dimension-, Hierarchy-, and Data-Load Framework.