

Client Identifying Data (CID) Requirements Specification for banks in Switzerland

Language: Z Notation

Developed By Serge (Siarhei Vinahradau, vinahradau@yahoo.de)

Specification, further referred to as FINMA:

<https://www.finma.ch/de/~media/finma/dokumente/rundschreiben-archiv/finma-rs200821---30-06-2017.pdf>

Specification requirements:

- CID data classification (FINMA 10*)
 - DATACATEGORY
 - CIDCATEGORIES
 - METADATA
- CID data owner (FINMA 13*)
 - ENTITY
 - DOMAIN
- all nodes with CID data stored should be recorded (FINMA 15*)
 - CIDSTORINGNODESAUDITLOG
- CID protection risks are country specific (FINMA 20*)
 - COUNTRY
- no node outside Switzerland should have unprotected CID data stored (FINMA 20*)
 - CONTENT
 - NODE
 - AddNodeData
- CID data accessed by users from outside Switzerland has to be protected (FINMA 20*)
 - AccesNodeData
- role and function based authorisation system in place (FINMA 22*)
 - ROLE
 - USER
 - DOMAIN
- List of users with bulk CID access (FINMA 34*)
 - BulkCIDAccessUsersList
- logs for bulk CID access (FINMA 40*)
 - CIDBULKLOG
- an internal employee has to be responsible for the compliance of outsourced CID activities (FINMA 50*)
 - DOMAIN
 - USER
 - AddUser
 - AddInternalUser
 - AddExternalUser

—
DATACATEGORY ::= DIRECT | INDIRECT | POTENTIALLYDIRECT | PROTECTED | NONCID
CIDCATEGORIES == {DIRECT, INDIRECT, POTENTIALLYDIRECT}
COUNTRY ::= SWITZERLAND | UK | USA | GERMANY
METADATA ::= CUSTOMERNAME | CUSTOMERADDRESS | ISVIPCUSTOMER
CONTENT ::= MUSTERMANN | SEESTRASSE | YES | NO | XXXXX
ENTITY ::= ENTITY1 | ENTITY2 | ENTITY3
USER ::= USER1 | USER2 | USER3
ROLE ::= ROLEGUICIDUSER | ROLEGUIUSER | ROLEBULKCID | ROLEBULK | ROLE1
CIDROLES == {ROLEGUICIDUSER, ROLEBULKCID}
NODEID ::= NODE1 | NODE2 | NODE3
L

```

┌ NODE
  nodeId: NODEID
  nodeCountry: COUNTRY
  nodeDataCategories: METADATA → DATACATEGORY
  nodeDataContents: METADATA → CONTENT
  nodeMetadata: P METADATA
  nodeContentsMetadata: P METADATA
├
  nodeCountry = SWITZERLAND ∨ (∀ c : ran nodeDataCategories • c ∉ CIDCATEGORIES)
  dom nodeDataContents ⊆ dom nodeDataCategories
  nodeMetadata = dom nodeDataCategories
  nodeContentsMetadata = dom nodeDataContents
└

┌ DOMAIN
  dataClassification: METADATA → DATACATEGORY
  dataOwner: METADATA → ENTITY
  roles: ROLE ↔ METADATA
  userAccessRights: USER ↔ ROLE
  teams: ENTITY ↔ USER
  internalUsers: P USER
  externalUsers: P USER

  classificationMetadata: P METADATA
  dataOwnerMetadata: P METADATA
  rolesRoles: P ROLE
  teamsTeams: P ENTITY
├
  ∀ u : USER • ¬(u ∈ internalUsers ∧ u ∈ externalUsers)
  ∀ u : dom userAccessRights • u ∈ ran teams
  ∀ u : dom userAccessRights • u ∈ internalUsers ∨ u ∈ externalUsers
  ∀ u : externalUsers • ¬(userAccessRights({u}) ∩ CIDROLES ≠ ∅ ∧ teams(dom (teams ▷ {u})))
  ∩ internalUsers = ∅

  classificationMetadata = dom dataClassification
  dataOwnerMetadata = dom dataOwner
  rolesRoles = dom roles
  dom dataClassification ⊆ dom dataOwner
  teamsTeams = dom teams
  #(dom dataClassification) < 6
  #(dom dataOwner) < 6
└

┌ CIDSTORINGNODESAUDITLOG
  cidStoringNodesIds: P NODEID
  NODE
├
  #(cidStoringNodesIds) < 6
  ∀ cidDataCategory : ran nodeDataCategories •
    cidDataCategory ∈ CIDCATEGORIES ⇒ nodeId ∈ cidStoringNodesIds
└

┌ CIDBULKLOG
  cidBulkAccess: USER ↔ NODEID
  cidBulkAccessUsers: P USER
├
  cidBulkAccessUsers = dom cidBulkAccess
  #(cidBulkAccess) < 6
└

```



```

└ InitDomain
  DOMAIN '
  NODE '
  CIDSTORINGNODESAUDITLOG '
  CIDBULKLOG '
|
  dataOwnerMetadata' = ∅
  classificationMetadata' = ∅
  userAccessRights' = ∅
  teams' = ∅
  internalUsers' = ∅
  externalUsers' = ∅
  nodeMetadata' = ∅
  cidStoringNodesIds' = ∅
  nodeId' = NODE1
  cidBulkAccess' = ∅
└

```

```

┌ AddRole
  ΔDOMAIN
  role?: ROLE
  metadata?: METADATA
|
  roles' = roles ∪ {(role?, metadata?)}
  dataClassification' = dataClassification
  teams' = teams
  internalUsers' = internalUsers
  externalUsers' = externalUsers
  dataOwner' = dataOwner
  userAccessRights' = userAccessRights
└

┌ AddUser
  ΔDOMAIN
  user?: USER
  entity?: ENTITY
|
  teams' = teams ∪ {(entity?, user?)}
  userAccessRights' = userAccessRights
  roles' = roles
  internalUsers' = internalUsers
  externalUsers' = externalUsers
  dataClassification' = dataClassification
  dataOwner' = dataOwner
└

┌ AddExternalUser
  ΔDOMAIN
  user?: USER
|
  externalUsers' = externalUsers ∪ {user?}
  internalUsers' = internalUsers
  teams' = teams
  userAccessRights' = userAccessRights
  roles' = roles
  dataClassification' = dataClassification
  dataOwner' = dataOwner
└

┌ AddInternalUser
  ΔDOMAIN
  user?: USER
|
  internalUsers' = internalUsers ∪ {user?}
  externalUsers' = externalUsers
  teams' = teams
  userAccessRights' = userAccessRights
  roles' = roles
  dataClassification' = dataClassification
  dataOwner' = dataOwner
└

```

```

┌ AddUserAccessRight
  ΔDOMAIN
  user?: USER
  role?: ROLE
|
  userAccessRights' = userAccessRights ∪ {(user?, role?)}
  teams' = teams
  internalUsers' = internalUsers
  externalUsers' = externalUsers
  roles' = roles
  dataClassification' = dataClassification
  dataOwner' = dataOwner
└

```

```

┌ RemoveUserAccessRight
  ΔDOMAIN
  user?: USER
  role?: ROLE
|
  userAccessRights' = userAccessRights \ {(user?, role?)}
  roles' = roles
  teams' = teams
  internalUsers' = internalUsers
  externalUsers' = externalUsers
  dataClassification' = dataClassification
  dataOwner' = dataOwner
└

```

```

┌ AddNodeData
  ΔNODE
  ΔCIDSTORINGNODESAUDITLOG
  ∃DOMAIN
  nodeIdInput?: NODEID
  nodeCountryInput?: COUNTRY
  nodeMetadataInput?: METADATA
  nodeDataContentInput?: CONTENT
  |
  nodeCountry' = nodeCountryInput?
  ∧ nodeId' = nodeIdInput?
  ∧
  (
    (nodeCountryInput? = SWITZERLAND ∧ (dataClassification nodeMetadataInput?) ∈
CIDCATEGORIES
    ∧ cidStoringNodesIds' = cidStoringNodesIds ∪ {nodeIdInput?}
    ∧ nodeDataContents' = nodeDataContents ⊕ {nodeMetadataInput? ↦
nodeDataContentInput?}
    ∧ nodeDataCategories' = nodeDataCategories ⊕ {nodeMetadataInput? ↦ (dataClassification
nodeMetadataInput?)})
    ∨
    ((dataClassification nodeMetadataInput?) ∉ CIDCATEGORIES
    ∧ cidStoringNodesIds' = cidStoringNodesIds
    ∧ nodeDataContents' = nodeDataContents ⊕ {nodeMetadataInput? ↦
nodeDataContentInput?}
    ∧ nodeDataCategories' = nodeDataCategories ⊕ {nodeMetadataInput? ↦ (dataClassification
nodeMetadataInput?)})
    ∨
    (nodeCountryInput? ≠ SWITZERLAND ∧ (dataClassification nodeMetadataInput?) ∈
CIDCATEGORIES
    ∧ cidStoringNodesIds' = cidStoringNodesIds
    ∧ nodeDataContents' = nodeDataContents ⊕ {nodeMetadataInput? ↦ XXXXX}
    ∧ nodeDataCategories' = nodeDataCategories ⊕ {nodeMetadataInput? ↦ PROTECTED})
  )
└

```



```

┌ AccessNode
├ ∃NODE
├ ∃DOMAIN
├ user?: USER
├ userCountry?: COUNTRY
├ nodeId?: NODEID
├ accessNodeMetadata?: METADATA
├ contentOutput!:  $\mathbb{P}$  CONTENT
├
├ nodeId? = nodeId
├ ∧
├ accessNodeMetadata? ∈ roles( $\langle\langle$ userAccessRights( $\langle\{$ user? $\rangle\rangle$ ) $\rangle\rangle$ )
├ ∧
├ (
├ (nodeDataCategories( $\langle\{$ accessNodeMetadata? $\rangle\rangle$ ) ⊆ CIDCATEGORIES ∧ userCountry? ≠
SWITZERLAND
├ ∧ contentOutput! = {XXXXX})
├ ∨
├ ((nodeDataCategories( $\langle\{$ accessNodeMetadata? $\rangle\rangle$ ) ∩ CIDCATEGORIES = ∅ ∨ userCountry? =
SWITZERLAND)
├ ∧ contentOutput! = nodeDataContents( $\langle\{$ accessNodeMetadata? $\rangle\rangle$ )
├ )
├ )
├ L

```

```

┌ AccessBulk
├  $\exists$ DOMAIN
├  $\exists$ NODE
├  $\Delta$ CIDBULKLOG
├ user?: USER
├ nodeId?: NODEID
├ userCountry?: COUNTRY
├ contentOutput!:  $\mathbb{P}$  CONTENT
├
├ (
├   ROLEBULKCID  $\in$  userAccessRigths( $\{\{user?\}\}$ )
├    $\wedge$  userCountry? = SWITZERLAND
├    $\wedge$  ran nodeDataCategories  $\cap$  CIDCATEGORIES  $\neq \emptyset$ 
├    $\wedge$  cidBulkAccess' = cidBulkAccess  $\cap \{(user?, nodeId?)\}$ 
├    $\wedge$  contentOutput! = ran nodeDataContents
├    $\wedge$  nodeId? = nodeId
├ )
├  $\vee$ 
├ (
├   (ROLEBULK  $\in$  userAccessRigths( $\{\{user?\}\}$ )  $\vee$  ROLEBULKCID  $\in$  userAccessRigths( $\{\{user?\}\}$ ))
├    $\wedge$  cidBulkAccess' = cidBulkAccess
├    $\wedge$  ran nodeDataCategories  $\cap$  CIDCATEGORIES =  $\emptyset$ 
├    $\wedge$  contentOutput! = ran nodeDataContents
├    $\wedge$  nodeId? = nodeId
├ )
└ L

```

```

┌ AssignDataOwner
  ΔDOMAIN
  metadata?: METADATA
  dataOwnerInput?: ENTITY
┌
  dataOwner' = dataOwner ⊕ {metadata? ↦ dataOwnerInput?}
  roles' = roles
  userAccessRigths' = userAccessRigths
└

┌ ClassifyDataCategory
  ΔDOMAIN
  metadata?: METADATA
  dataCategory?: DATACATEGORY
┌
  dataClassification' = dataClassification ⊕ {metadata? ↦ dataCategory?}
  roles' = roles
  userAccessRigths' = userAccessRigths
└

-
┌ ImplementDataClassification == AssignDataOwner ∧ ClassifyDataCategory
└

┌ RecycleData
  ΔDOMAIN
  metadata?: METADATA
┌
  metadata? ∈ dataOwnerMetadata
  metadata? ∈ classificationMetadata
  dataClassification' = {metadata?} ⋈ dataClassification
  dataOwner' = {metadata?} ⋈ dataOwner
  roles' = roles
  teams' = teams
  userAccessRigths' = userAccessRigths
└

┌ BulkCIDAccessUsersList
  ∃DOMAIN
  ∃NODE
  BulkCIDAccessUsersList!: ℙ USER
┌
  BulkCIDAccessUsersList! = dom (userAccessRigths ▷ {ROLEBULKCID})
└

```