

Validation Rules

```
validationRules :=
    identifierDecls
    ruleDefs
    EOF

identifierDecls :=
    CONTAINERS_KW LBRACE (containerDecl)* RBRACE
    DECLARATIONS_KW LBRACE (identifierDecl)* RBRACE
    [tagDecls]

containerDecl :=
    IDENTIFIER IDENTIFIER (COMMA IDENTIFIER)* SEMICOLON

identifierDecl :=
    typeName IDENTIFIER (COMMA IDENTIFIER)* SEMICOLON

typeName :=
    PRINCIPAL_KW | TTP_KW | NAME_KW | VARIABLE_KW | KEY_KW

tagDecls :=
    TAGS_KW LBRACE (tagDecl )* RBRACE

tagDecl :=
    (TAG_KW | TAG_TYPE) idOrKey (COMMA idOrKey)* SEMICOLON

ruleDefs :=
    RULES_KW LBRACE (rule)+ RBRACE

rule :=
    RULE_KW id LPAR id RPAR
    LBRACE
    instructionDecl
    [conditions]
    [actions]
    RBRACE

instructionDecl :=
    LBRACE (instruction | NIL_KW) RBRACE
```

```

instruction :=
    id LPAR [arguments] RPAR

arguments :=
    argument (COMMA argument)*

argument :=
    (abstractData | assign | instruction | message | taggedId | asymKeyInstance)

assign :=
    id EQ (value | instruction | message | keyInstr)

keyInstr :=
    KEY_INSTR_KW LPAR IDENTIFIER COMMA IDENTIFIER RPAR |
    ASYM_KEY_INSTR_KW LPAR IDENTIFIER RPAR

value :=
    STRING_VALUE

message :=
    (LBRACE messageData RBRACE |
     ASYM_LBRACE messageData ASYM_RBRACE)
    (key | LPAR key RPAR)

messageData :=
    (taggedId (COMMA taggedId)* [COMMA abstractData] | abstractData)

key :=
    COLON (id | asymKeyInstance)

asymKeyInstance :=
    (PUB_KW | PRIV_KW) LPAR id RPAR

taggedId :=
    id [COLON (idOrKey | ANY_TAG_KW)] [LPAR arguments RPAR]

conditions :=
    [operationSource] CONDITIONS_KW LBRACE (condition)+ RBRACE

operationSource :=
    (NATIVE_KW | EXTERNAL_KW)

```

```

condition := 
    [RESET_ABSTRACT] [(EBTDIR |UBTDIR)] [BOOL]
    [LPAR operationSource RPAR]
    [ipohesis]
    thesis

ipohesis := 
    IPOTHESIS_KW LPAR orCheck RPAR SEMICOLON

thesis := 
    THESIS_KW LPAR orCheck RPAR SEMICOLON

orCheck := 
    andCheck (PIPE andCheck)*

andCheck := 
    notCheck (AMP notCheck)*

notCheck := 
    [NOT] simpleCheck

simpleCheck:
    LPAR orCheck RPAR | operation

operation := 
    [LPAR operationSource RPAR] id DOT id LPAR arguments RPAR

actions := 
    [operationSource] ACTIONS_KW LBRACE (operation SEMICOLON)+ RBRACE

idOrKey := 
    (IDENTIFIER | KEY_INSTR_KW)

id := 
    IDENTIFIER

abstractData := 
    (ABSTRACT_CONTENT_KW | ABSTRACT_TAGGED_CONTENT_KW)

```

Tabella 1: Valori dei token.

RULES_KW	rules
RULE_KW	rule
DECLARATIONS_KW	declarations
CONTAINERS_KW	containers
TAGS_KW	tags
IPOTHESIS_KW	ipohesis
THESIS_KW	thesis
CONDITIONS_KW	conditions
ACTIONS_KW	actions
NIL_KW	0
ABSTRACT_CONTENT_KW	...
ABSTRACT_TAGGED_CONTENT_KW	:::
ANY_TAG_KW	anyTag
NATIVE_KW	native
EXTERNAL_KW	external
KEY_INSTR_KW	sym-key
ASYM_KEY_INSTR_KW	asym-key
PUB_KW	Pub
PRIV_KW	Priv
CONTAINER_KW	Container
PRINCIPAL_KW	Principal
TPP_KW	TrustedThirdParty
NAME_KW	Name
VARIABLE_KW	Variable
KEY_KW	Key
TAG_KW	Tag
BOOL	true false
STRING_VALUE	"[^"]*"
EBTDIR	@exist
UBTDIR	@univ
RESET_ABSTRACT	@ra

Tabella 2: Valori dei token (continua).

NOT	!
LPAR	(
RPAR)
LBRACE	{
RBRACE	}
ASYM_LBRACE	{
ASYM_RBRACE	}
DOT	.
COLON	:
COMMA	,
PIPE	
EQ	=
SEMICOLON	;
QUOTE	"
AMP	&
IDENTIFIER	[A-Z, a-z, _] ([A-Z, a-z, 0-9, _])* (?)?
TAG_TYPE	KeyTag TypeTag NonceTag MessageTag IdentityTag CiphertextTag