NAMED DATA NETWORKING

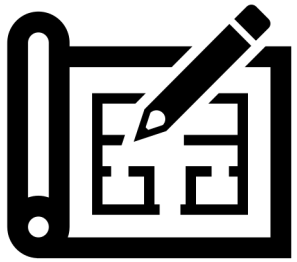
ACM ICN Paris 2021

VerSec Language:
Trust Schema Writing Tutorial

Proyash Podder (FIU)

Tutorial: Power of Trust Schemas for Easy and Secure Deployment of NDN Applications

VerSec: Declarative Language

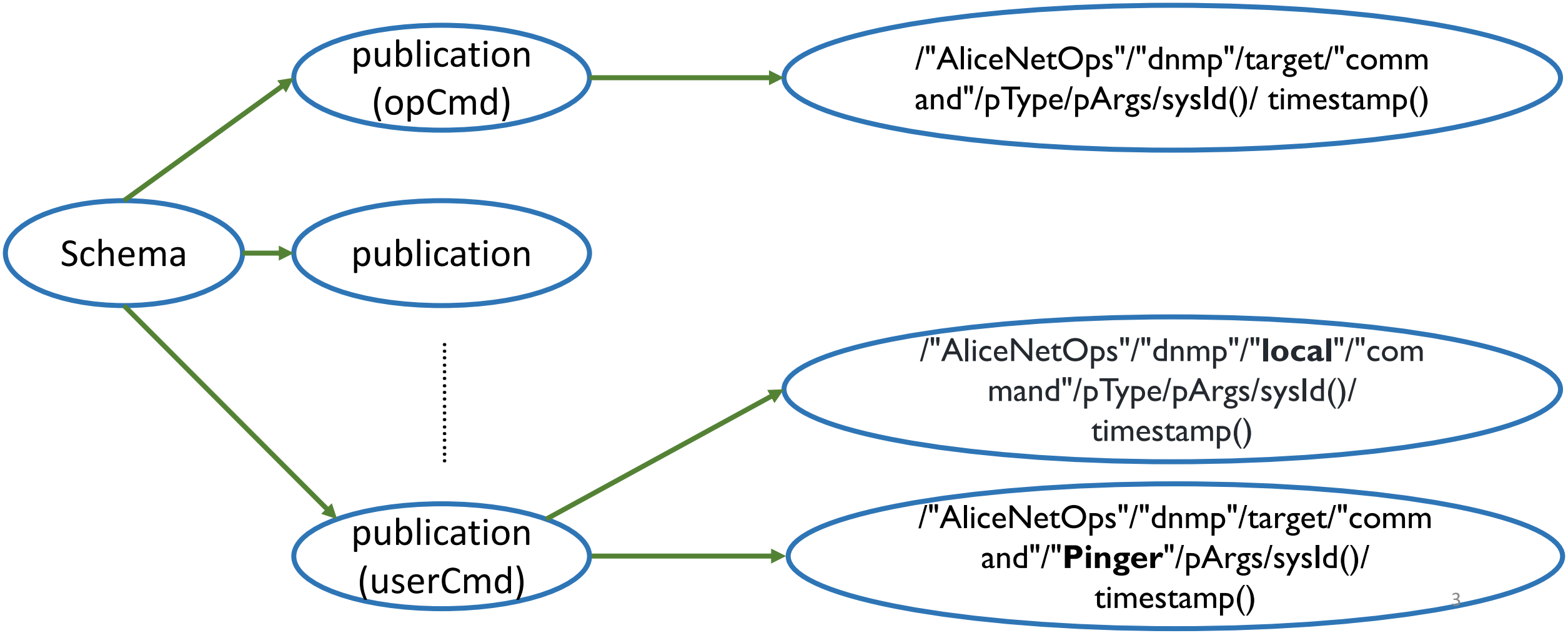


Declare Statements About Publications

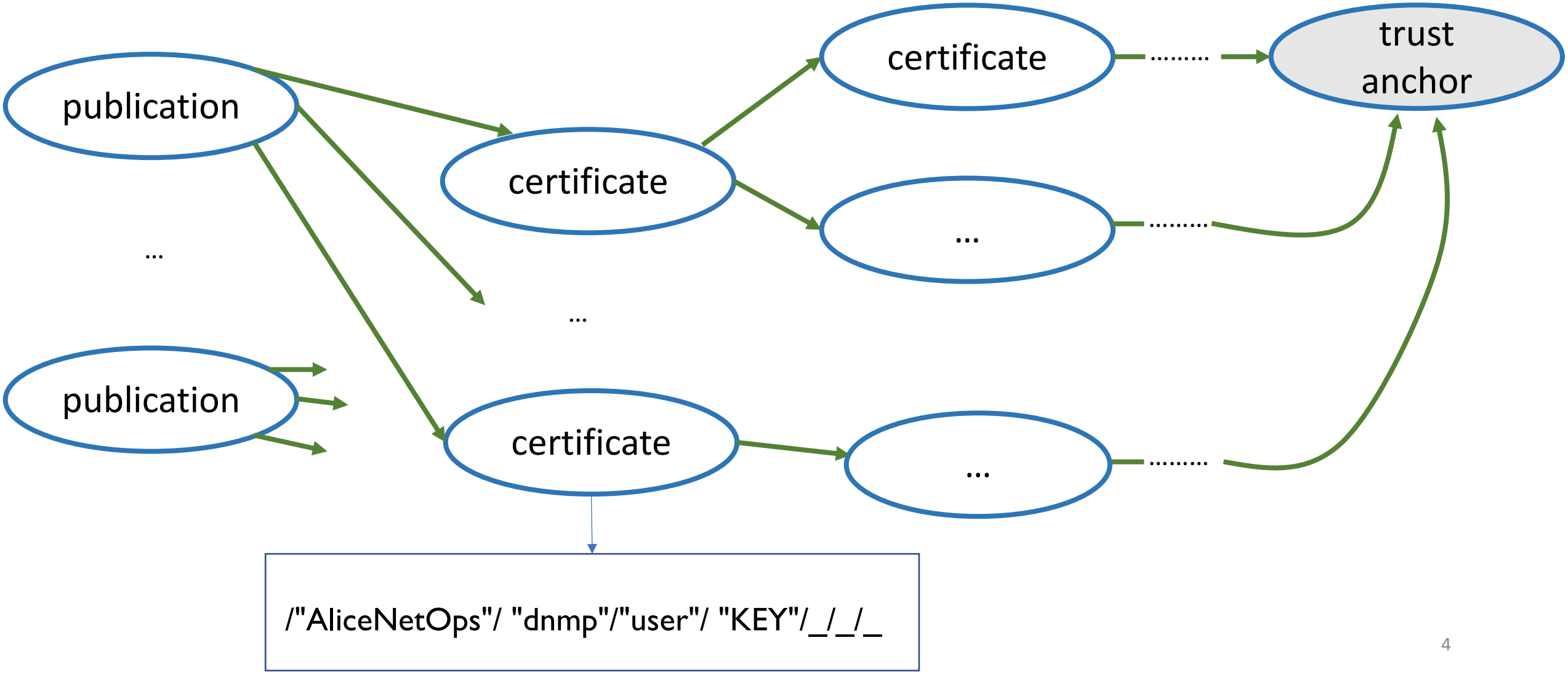
Declare constraints on Name layout and components of names

Declare constrains on structural and signing relationships between names

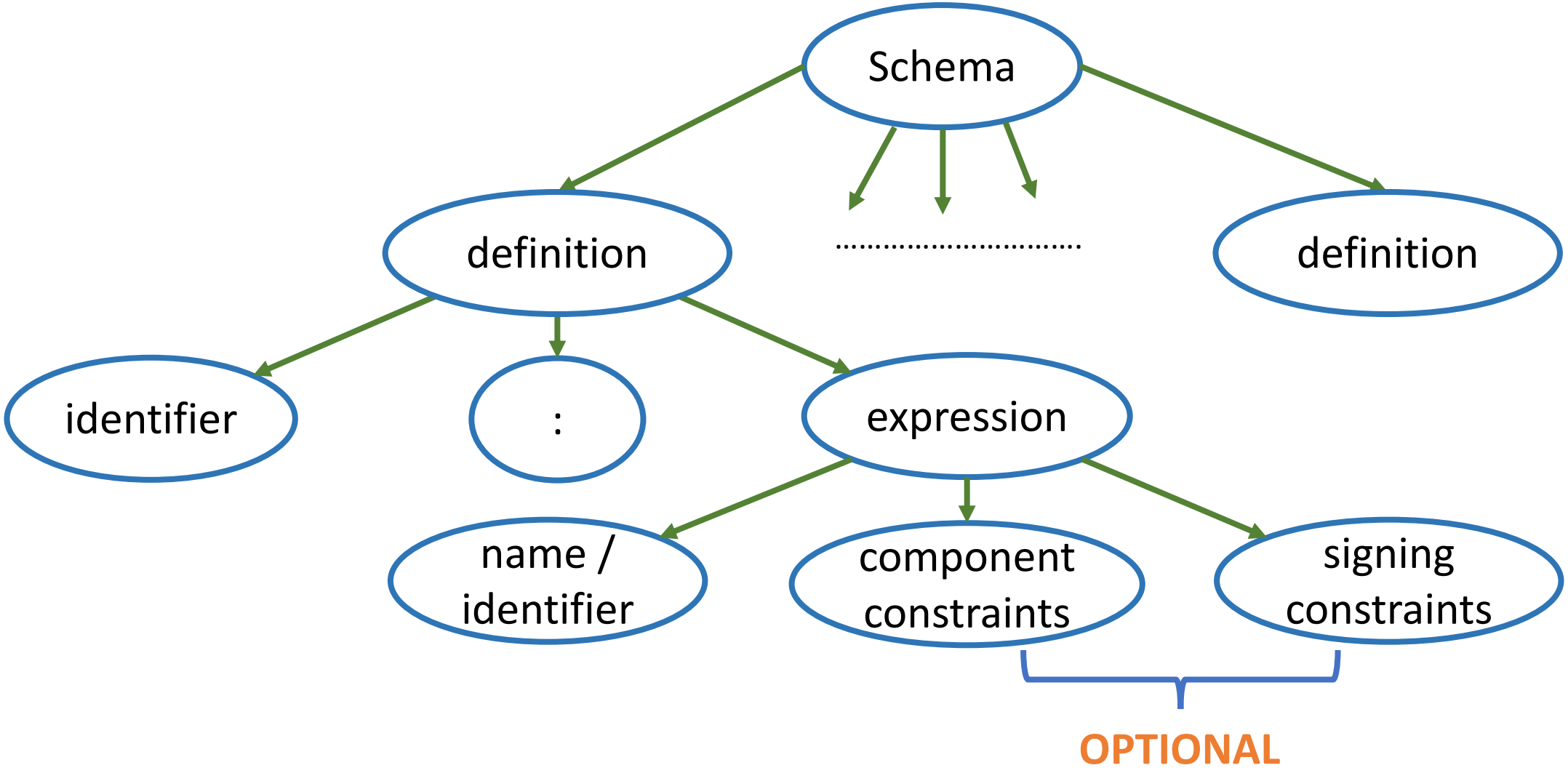
VerSec Goal: Publication Templates



VerSec Goal: Certificate DAGs



VerSec Language Structure



Name

- *Names* consist of a sequence of *components* separated by slashes /
  - /"AliceNetOps"/"dnmp"/target/"command"/pType/pArgs/sysId()/timestamp()

- Each component is an expression that can be:
  - a literal string
    - Actual value for the name component
  - an internal function *call*
    - *Actual value at runtime*
  - an *identifier*
    - Details on the next slide
  - an *expression* enclosed in parenthesis
  - two expressions separated by a vertical bar (|).

Identifier

- *Identifiers* are like variables
- VerSec defines the following identifier types
  - _-identifier (starts with underscore)
    - Variable's value must be derived from the schema rules (compile time)
    - Example
      - _network: "Foobar"
        #command: _**network**/topic/"Field"/func()
      - => #command:"**Foobar**"/topic/"Field"/func()
  - #-identifier
    - Target publication template (targets of VerSec compilation)
      - #command, #mypub, …
  - Regular identifier
    - Parameter that must be supplied at run time
      - "topic" in the above example must be supplied at run-time

Component Constraints

- A *component constraint* is an **open brace** followed by one or more **constraint terms** followed by a **closing brace**.
- Multiple component constraints can be given, separated by | or &.

#mypub: /_domain/_topic/param

#mypub & {_topic: "req", param: "status"} | {_topic: "cmd", param: "start"}

/_domain/"req"/"status"

/_domain/"cmd"/"start"

Component Constraints Combinations

- #mypub: /_topic/_param

- #mypub & {_topic: "req", _param: "status"} | {_topic: "cmd", _param: "start"}
  - /"req"/"status"
  - /"cmd"/"start"

- #mypub & {_topic: "req"|"cmd", _param:"status"|"start"}
  - /"req"/"status"
  - /"req"/"start"
  - /"cmd"/"status"
  - /"cmd"/"start"

Signing Constraints

- A *signing constraint* consists of a <= (signed-by operator) followed by one or more *definition identifiers* separated by | operators.

cmd: #mypub & {_topic: "cmd"} <= opCert

req: #mypub & {_topic: "req"} <= opCert | userCert

implies that

cmd publications must be signed by an opCert
while req publications can be signed by either an opCert or a userCert