Beyond Best Effort: Routing with Requirements

Brad Smith[1], Paul Tatarsky[2]
[1] Baskin Engineering, University of California Santa Cruz (USA)
[2] Tatarsky.com (USA)
Contact email: brad@soe.ucsc.edu

Resume

Brad Smith received the master's and Ph.D. degrees in Computer Science from the University of California Santa Cruz in 1997 and 2003, respectively. He has held an Adjunct Faculty position in the Computer Science & Engineering Department of the Baskin School of Engineering since 2006, and has served in leadership positions in the Information Technology Division at UCSC including Director of Computing for the School of Engineering, Director of Core Technologies, and Vice Chancellor Information Technology from 1990 to 2016.

Research Interests & Current Projects

Brad is working in the areas of policy-based routing and the application of these technologies to the improvement of the security and robustness of Internet-based systems. He was a part of the Computer Communication Research Group (CCRG) led by Prof JJ Garcia-Luna. Working with Paul Tatarsky (co-author) he developed the technology presented here with a National Science Foundation Small Business Innovative Research (SBIR) grant. He is looking to build a team to develop a production-grade implementation and look at commercialization opportunities. He is also working on developing open source implementations of state-of-the-art routing protocols that were developed in the CCRG.

The Need - Effective Resource Control

Control *who* can transmit *what traffic* to *whom* over *what parts of the network* and *how the traffic should be handled*

Routing with Requirements developed to solve this!

Problems with the original Internet

Proposed Solution
Routing with requirements

How get *best set of paths* & Benefits

Testbed & Summary

Problems with the original Internet

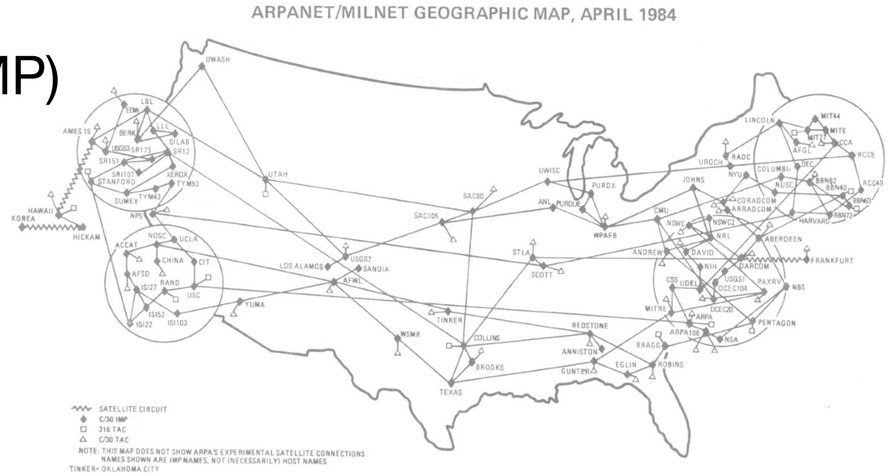
Proposed Solution
Routing with requirements

How get *best set of paths* & Benefits

Testbed & Summary

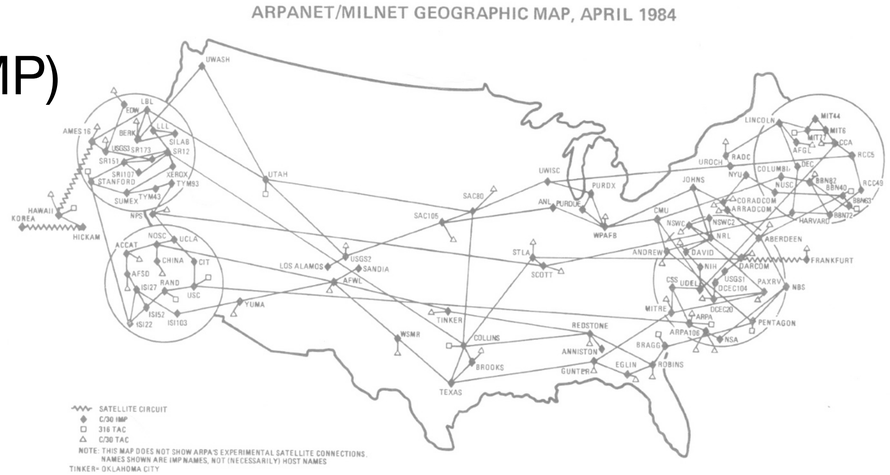
Current Internet - Features & Limitations

- Simple architecture
  - Single best path (generalized to be sbp *weight* w/ ECMP)
  - Universal <u>reachability</u>
  - Best effort delivery - all packets handled the same
  - Hugely successful - “*works over anything!*”



Current Internet - Features & Limitations

- Simple architecture
  - Single best path (generalized to be sbp *weight* w/ ECMP)
  - Universal reachability
  - Best effort delivery - all packets handled the same
  - Hugely successful - "*works over anything!*"



- Challenged by new requirements
  - Performance & policy control **vs** universal reachability, best effort delivery
  - High reliability **vs** low level network abstraction (VLANs/queues/priorities)
  - Ease of configuration **vs** complex technology (firewalls/network segmentation/etc)
  - Efficient use of resources **vs** single path/ECMP routing (30-40% link utilization)

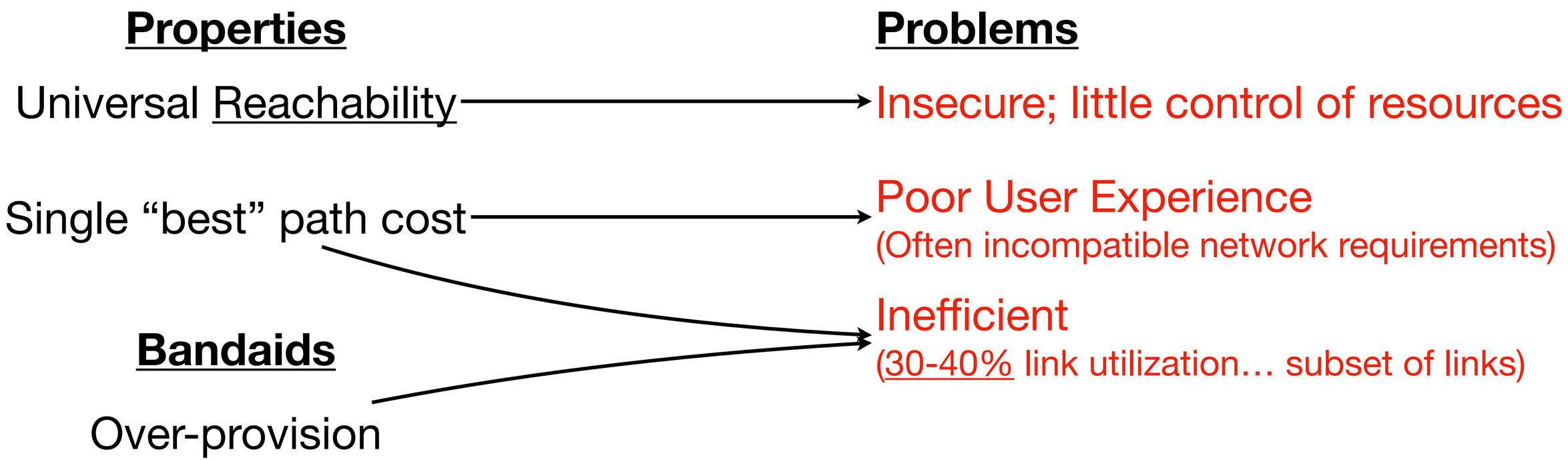
Summary

Properties	Problems
Universal Reachability	Insecure; little control of resources

Summary

Properties	Problems
Universal Reachability	Insecure; little control of resources
Single “best” path cost	Poor User Experience (Often incompatible network requirements)

Summary


Properties
Problems
Universal Reachability
Insecure; little control of resources
Single “best” path cost
Poor User Experience
(Often incompatible network requirements)
Bandaids
Inefficient
(30-40% link utilization… subset of links)
Over-provision

Summary

Properties

Universal Reachability → Insecure; little control of resources

Single "best" path cost → Poor User Experience
(Often incompatible network requirements)

Bandaids

Over-provision → Inefficient
(30-40% link utilization… subset of links)

Add-on functions → Cost and complexity
(firewalls, policy routing, queuing disciplines)
("how" implement desired properties)

Attempts to Mitigate

- Current solutions:
  - In network (core) – *single-flow* (MPLS) or *single-traffic class* (SR) are inefficient and complex
  - On network (overlay) – add-ons (firewalls, segmentation, …) are *expensive*, *complex*, *fragile*
  - Virtualization – magnify resource utilization, *low level network abstraction*
  - Programmability (SDN) – magnify expertise/labor, *low level network abstraction*
- Problems
  - *Inefficient* (single-flow (MPLS)/traffic class (SR) forwarding state)
  - *Fragile* (semi-static core, and overlay functionality not aware of topology)
  - *Insecure* (universal reachability, complex and error-prone configuration)
  - *Expensive* (complex overlay functions, expertise and labor intensive)
  - *Complex* network configuration (low level network abstraction)

Comparing solutions…

- Four Issues
  - *Single weight* vs *best set of weights*

	Path Weights Computed?
Original Routing	**Single Best Weight** (for a single traffic class)
MPLS	**Single Best Weight** (per configured traffic class)
SR	**Single Best Weight** (per configured traffic class)
RwR	**Best Set of Weights** (for ***all traffic classes***!!)

Comparing solutions…

- Four Issues
  - *Single weight* vs *best set of weights*
  - *Fate sharing* vs *Distributed state*

	Path Weights Computed?	Fate Sharing?
Original Routing	**Single Best Weight** (for a single traffic class)	**Yes** (path state in ***router***)
MPLS	**Single Best Weight** (per configured traffic class)	**No** (path state in ***each hop***)
SR	**Single Best Weight** (per configured traffic class)	**Yes** (path state in ***packets***)
RwR	**Best Set of Weights** (for ***all traffic classes***!!)	**Yes** (path state in ***router***)

Comparing solutions…

- Four Issues
  - *Single weight* vs *best set of weights*
  - *Fate sharing* vs *Distributed state*
  - Supports *partial* vs requires *full* topology

	Path Weights Computed?	Fate Sharing?	Partial Topology?
Original Routing	**Single Best Weight** (for a single traffic class)	**Yes** (path state in ***router***)	**Yes**
MPLS	**Single Best Weight** (per configured traffic class)	**No** (path state in ***each hop***)	**No**
SR	**Single Best Weight** (per configured traffic class)	**Yes** (path state in ***packets***)	**No**
RwR	**Best Set of Weights** (for ***all traffic classes***!!)	**Yes** (path state in ***router***)	**Yes**

Comparing solutions…

- Four Issues
  - *Single weight* vs *best set of weights*
  - *Fate sharing* vs *Distributed state*
  - Supports *partial* vs requires *full* topology
  - *P2MP* vs *P2P* flow state)

	Path Weights Computed?	Fate Sharing?	Partial Topology?	P2MP?
Original Routing	**Single Best Weight** (for a single traffic class)	**Yes** (path state in ***router***)	**Yes**	**Yes**
MPLS	**Single Best Weight** (per configured traffic class)	**No** (path state in ***each hop***)	**No**	**No** (P2P ***per flow!!***)
SR	**Single Best Weight** (per configured traffic class)	**Yes** (path state in ***packets***)	**No**	**Yes** (per traffic class)
RwR	**Best Set of Weights** (for ***all traffic classes***!!)	**Yes** (path state in ***router***)	**Yes**	**Yes** (for ***all*** traffic classes!!)

Comparing solutions…

- Four Issues
  - *Single weight* vs *best set of weights*
  - *Fate sharing* vs *Distributed state*
  - Supports *partial* vs requires *full* topology
  - *P2MP* vs *P2P* flow state)

	Path Weights Computed?	Fate Sharing?	Partial Topology?	P2MP?	Comments
Original Routing	**Single Best Weight** (for a single traffic class)	**Yes** (path state in ***router***)	**Yes**	**Yes**	Scalable, robust and efficient, but only one traffic class.
MPLS	**Single Best Weight** (per configured traffic class)	**No** (path state in ***each hop***)	**No**	**No** (P2P ***per flow!!***)	Not scalable, robust or efficient, multiple traffic classes.
SR	**Single Best Weight** (per configured traffic class)	**Yes** (path state in ***packets***)	**No**	**Yes** (per traffic class)	Somewhat scalable and efficient, robust, multiple traffic classes.
RwR	**Best Set of Weights** (for ***all traffic classes***!!)	**Yes** (path state in ***router***)	**Yes**	**Yes** (for ***all*** traffic classes!!)	**Scalable, robust, efficient, *all possible traffic classes!!***

Comparing solutions…

- Four Issues
  - *Single weight* vs *best set of weights*
  - *Fate sharing* vs *Distributed state*
  - Supports *partial* vs requires *full* topology
  - *P2MP* vs *P2P* flow state)

- SR – *person in the loop*
  - Per-traffic class QoS and TE requirements
  - Heavily engineered networks; advanced skills
- RwR – *one-time config of needed parameters*
  - ***Network autonomically satisfies requirements and minimizes contestion***

	Path Weights Computed?	Fate Sharing?	Partial Topology?	P2MP?	Comments
Original Routing	**Single Best Weight** (for a single traffic class)	**Yes** (path state in ***router***)	**Yes**	**Yes**	Scalable, robust and efficient, but only one traffic class.
MPLS	**Single Best Weight** (per configured traffic class)	**No** (path state in ***each hop***)	**No**	**No** (P2P ***per flow!!***)	Not scalable, robust or efficient, multiple traffic classes.
SR	**Single Best Weight** (per configured traffic class)	**Yes** (path state in ***packets***)	**No**	**Yes** (per traffic class)	Somewhat scalable and efficient, robust, multiple traffic classes.
RwR	**Best Set of Weights** (for ***all traffic classes***!!)	**Yes** (path state in ***router***)	**Yes**	**Yes** (for ***all*** traffic classes!!)	**Scalable, robust, efficient, *all possible traffic classes!!***

Why is this important?

Future Internet target use cases

- Health-care
- Transportation
- Manufacturing
- Agriculture
- Vehicle-to-vehicle
- Building automation
- Infrastructure
- Military

Why is this important?

Future Internet target use cases

- Health-care
- Transportation
- Manufacturing
- Agriculture
- Vehicle-to-vehicle
- Building automation
- Infrastructure
- Military

With “hard” requirements

- Life-safety
- Real-time
- Security
- Ubiquitous

Why is this important?

Future Internet target use cases

- Health-care
- Transportation
- Manufacturing
- Agriculture
- Vehicle-to-vehicle
- Building automation
- Infrastructure
- Military

With “hard” requirements

- Life-safety
- Real-time
- Security
- Ubiquitous

Network engineers do not know how they’re going to implement them!

Problems with the original Internet

Proposed Solution:
Routing with requirements

How get *best set of paths* & Benefits

Testbed & Summary

Routing with requirements

Ordered, ***performance requirements*** (delay, bandwidth, etc.)

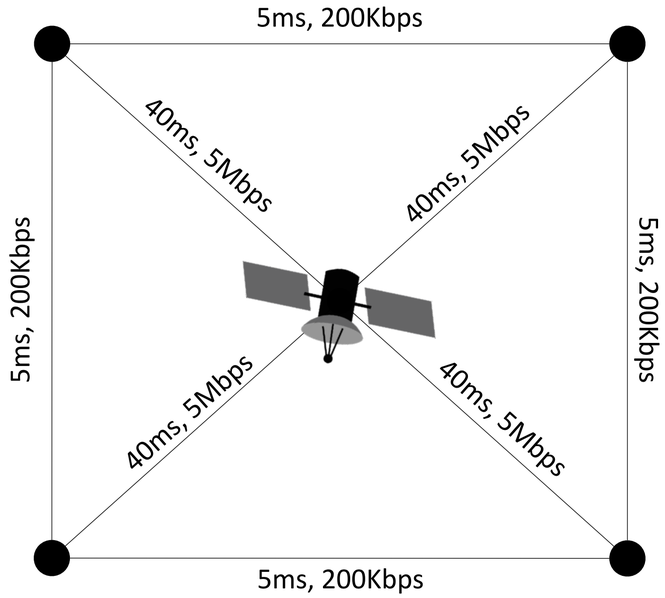
Routing with requirements

Ordered, ***performance requirements*** (delay, bandwidth, etc.)

Categorical requirements with Boolean expressions

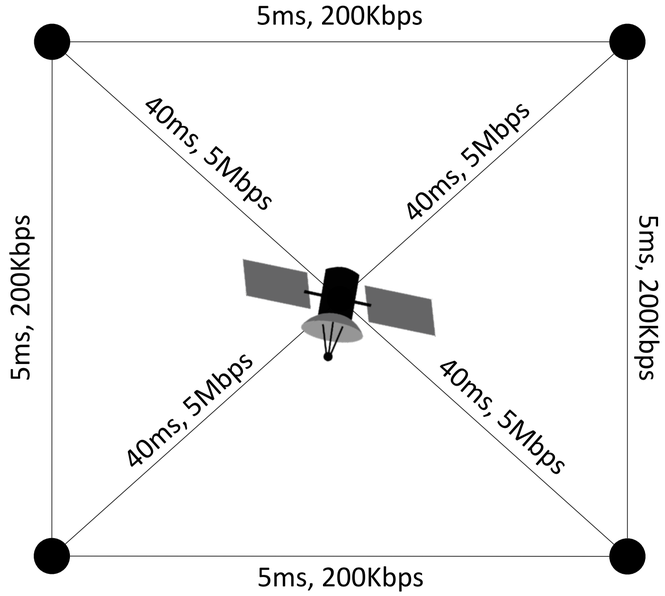
Quality of Service - video and VoIP traffic

- ***Link metrics***: delay, bandwidth



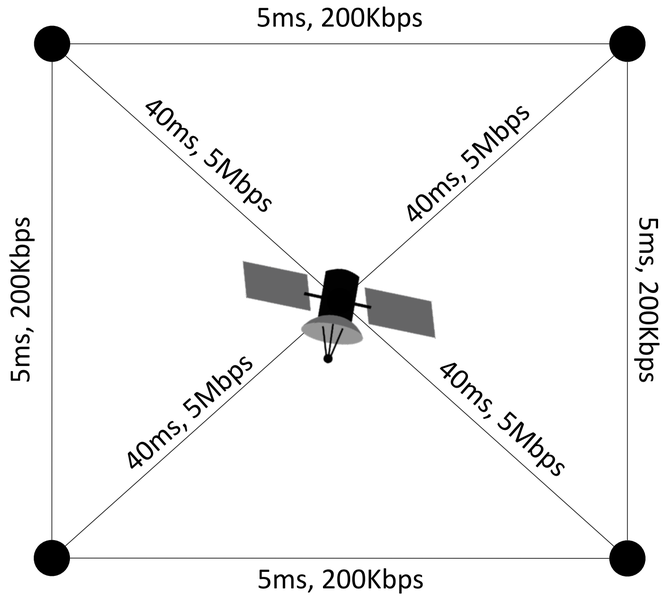
Quality of Service - video and VoIP traffic

- ***Link metrics***: delay, bandwidth
- ***Performance requirements***
  - VoIP: ≤ `50ms` delay, ≥ `100Kbps` bandwidth
  - Video streaming: ≤ `500ms` delay, ≥ `3Mbps` bandwidth



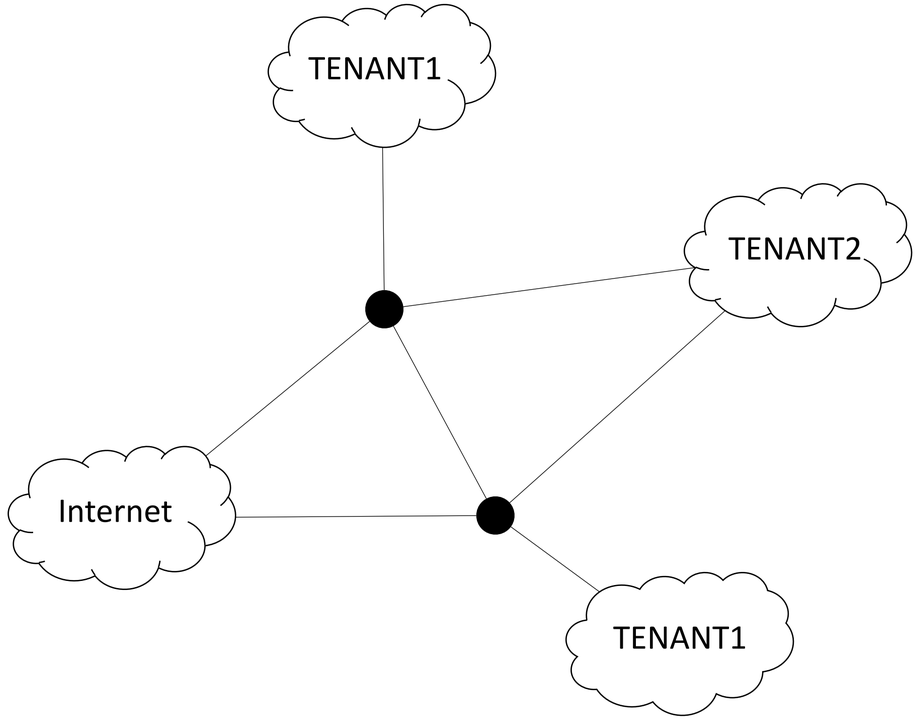
Quality of Service - video and VoIP traffic

- ***Link metrics***: delay, bandwidth
- ***Performance requirements***
  - VoIP: ≤ `50ms` delay, ≥ `100Kbps` bandwidth
  - Video streaming: ≤ `500ms` delay, ≥ `3Mbps` bandwidth
- *Traffic is forwarded over path that satisfies flow performance constraints; if more than one, use least congested*



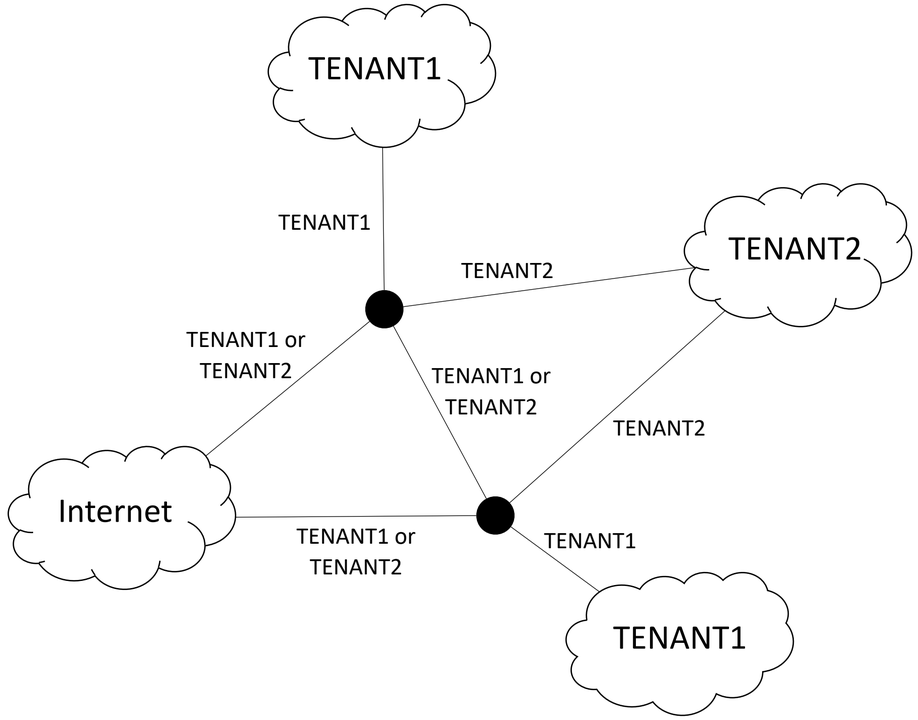
Multitenant

- ***Flow variables***: `TENANT1`, `TENANT2`



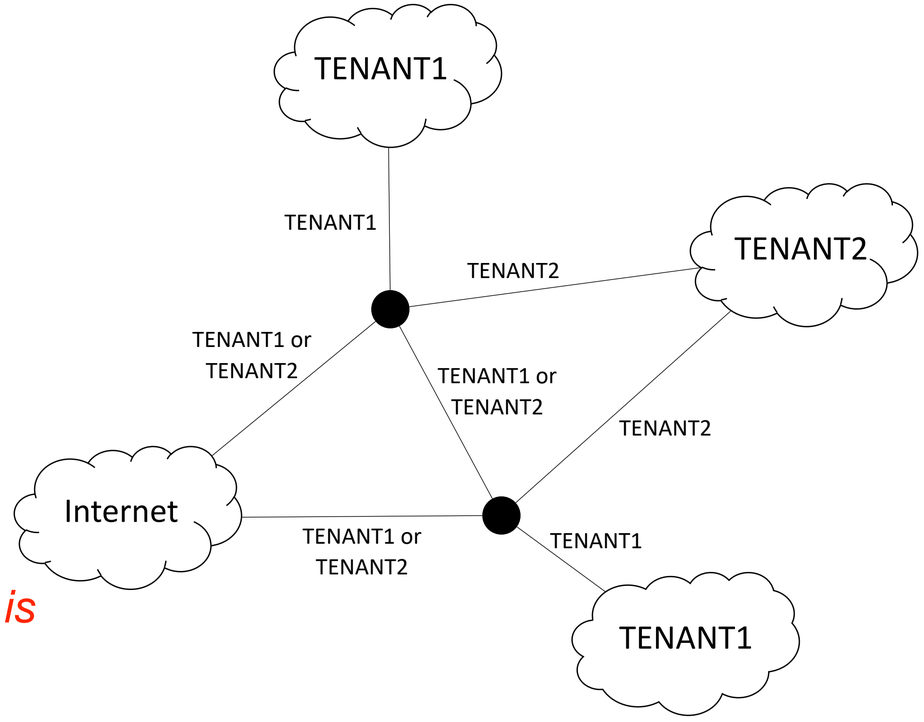
Multitenant

- ***Flow variables***: `TENANT1`, `TENANT2`
- ***Flow requirements***
  - `TENANT1`
  - `TENANT2`
  - `TENANT1 or TENANT2`



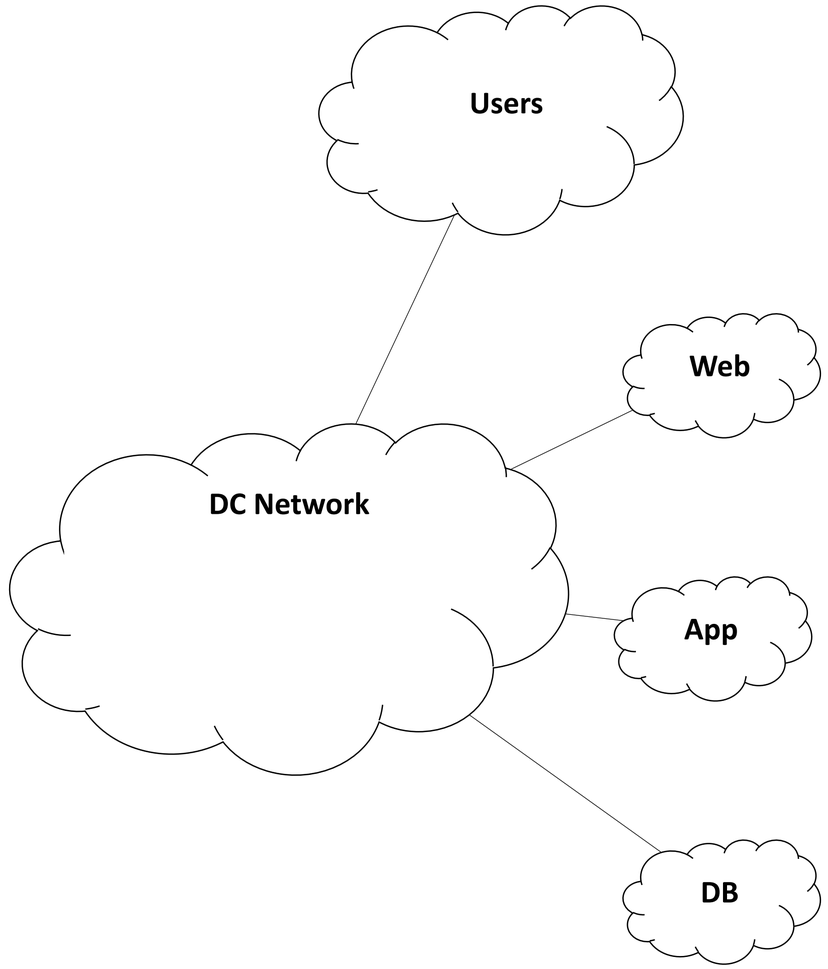
Multitenant

- ***Flow variables***: `TENANT1`, `TENANT2`
- ***Flow requirements***
  - `TENANT1`
  - `TENANT2`
  - `TENANT1 or TENANT2`
- *Traffic is forwarded over path with fewest hops that tenant is authorized to use; if more than one, use least congested*



Three Tier Web Architecture using Zero Trust

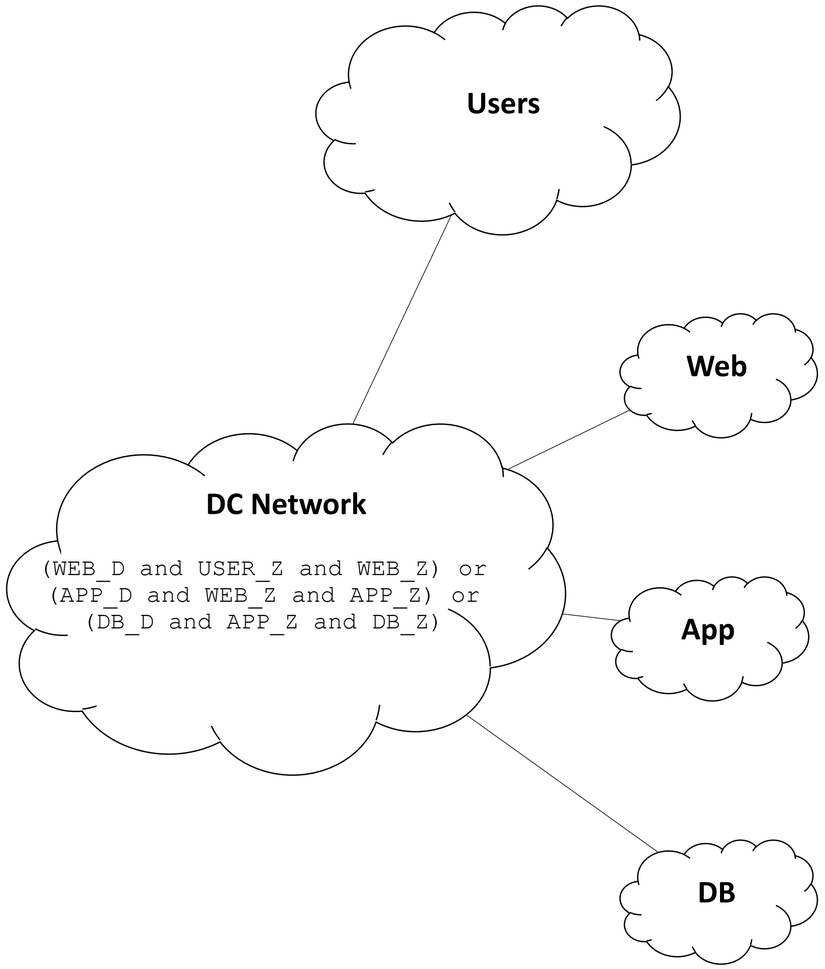
- ***Flow variables***
  - Flow data types: `WEB_D`, `APP_D`, `DB_D`
  - Endpoint zone: `USER_Z`, `WEB_Z`, `APP_Z`, `DB_Z`



Three Tier Web Architecture using Zero Trust

- ***Flow variables***
  - Flow data types: `WEB_D`, `APP_D`, `DB_D`
  - Endpoint zone: `USER_Z`, `WEB_Z`, `APP_Z`, `DB_Z`
- ***Global requirements***

```
(WEB_D and USER_Z and WEB_Z) or
(APP_D and WEB_Z and APP_Z) or
(DB_D and APP_Z and DB_Z)
```

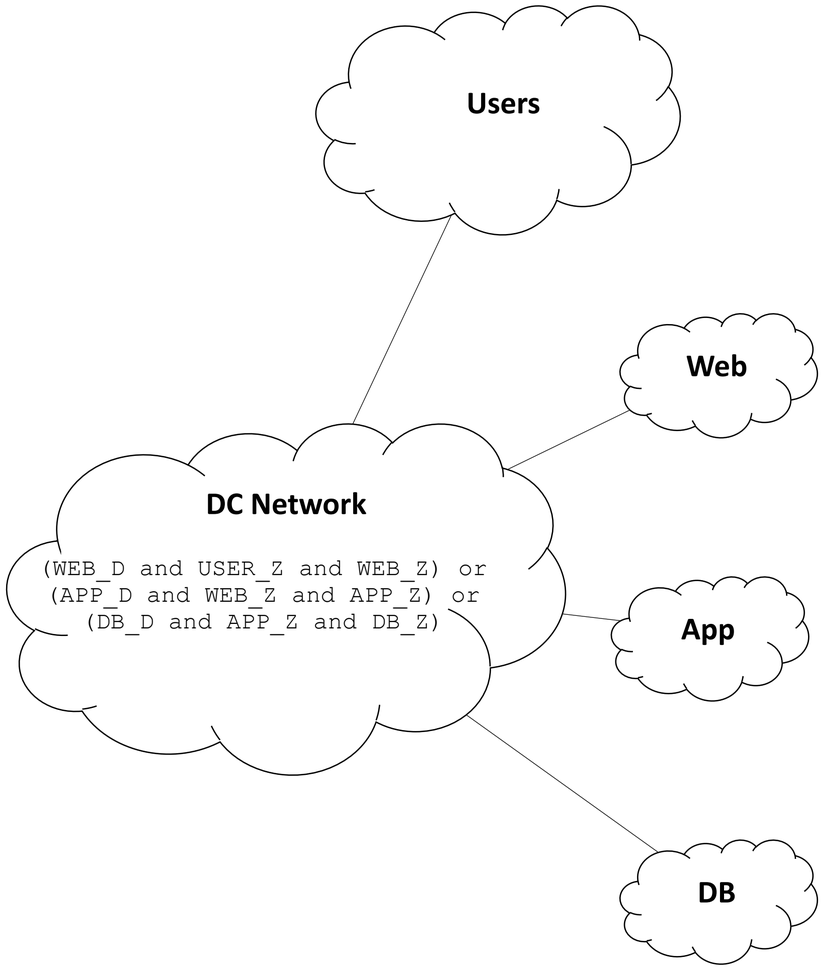


Three Tier Web Architecture using Zero Trust

- ***Flow variables***
  - Flow data types: `WEB_D`, `APP_D`, `DB_D`
  - Endpoint zone: `USER_Z`, `WEB_Z`, `APP_Z`, `DB_Z`
- ***Global requirements***

```
(WEB_D and USER_Z and WEB_Z) or
(APP_D and WEB_Z and APP_Z) or
(DB_D and APP_Z and DB_Z)
```

- *Traffic at each layer is isolated.*



Autonomic control with Boolean variables

- Boolean variables provide control points for network policy
  - Autonomic control of Boolean variable value

Autonomic control with Boolean variables

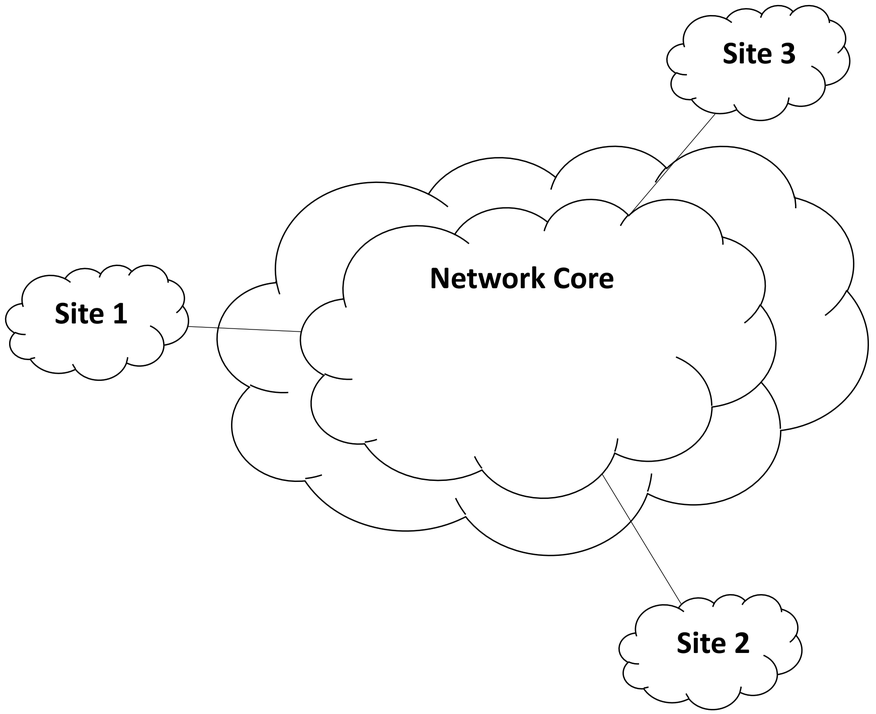
- Boolean variables provide control points for network policy
  - Autonomic control of Boolean variable value
- Two classes
  - ***Controller managed variables***
    - Date/time, interface info (speed, media type, drop statistics, encryption, etc.), …

Autonomic control with Boolean variables

- Boolean variables provide control points for network policy
  - Autonomic control of Boolean variable value
- Two classes
  - ***Controller managed variables***
    - Date/time, interface info (speed, media type, drop statistics, encryption, etc.), …
  - ***Programmatically controlled variables***
    - External systems control variables using API
      - Examples: threat status, user/host information from “network access control” (NAC) systems
    - Prototype implemented a REST interface

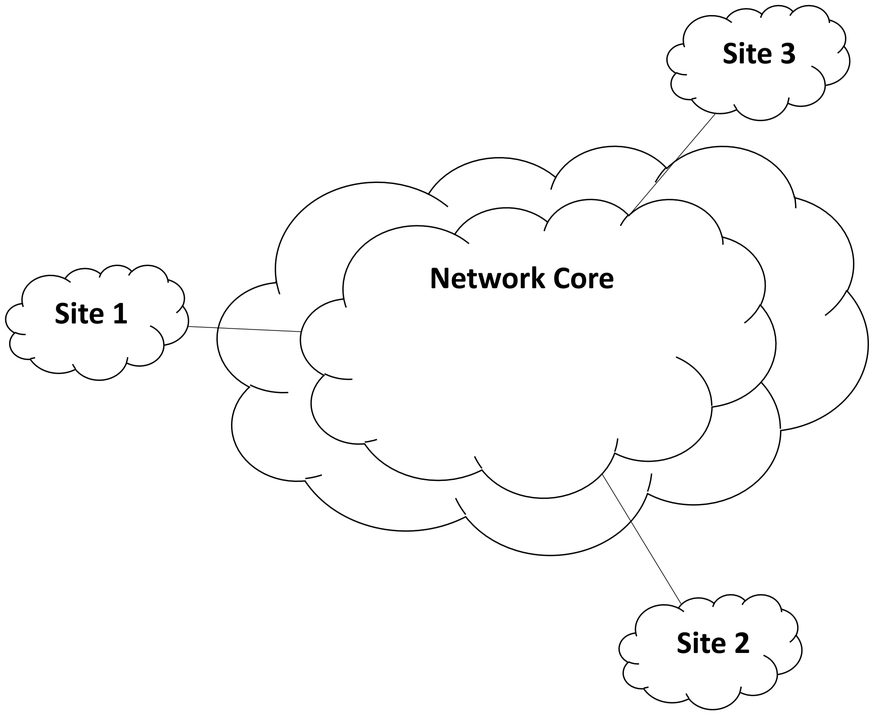
Time of Day Backups - *controller* variables

- ***Flow variable***
  - `BACKUP`



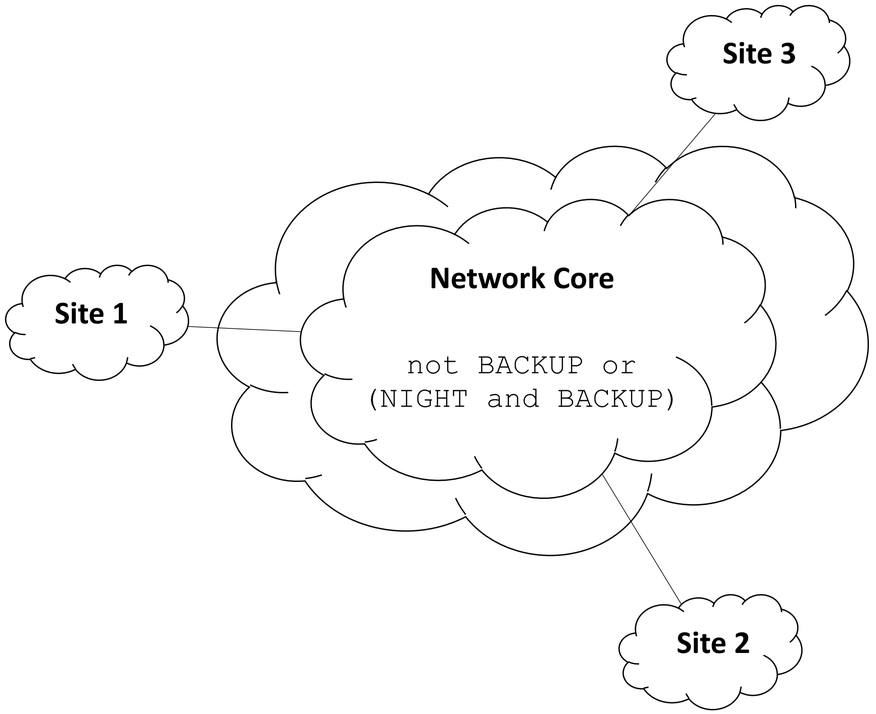
Time of Day Backups - *controller* variables

- ***Flow variable***
  - `BACKUP`
- ***Environment variable***
  - `NIGHT` - *value set by controller!*



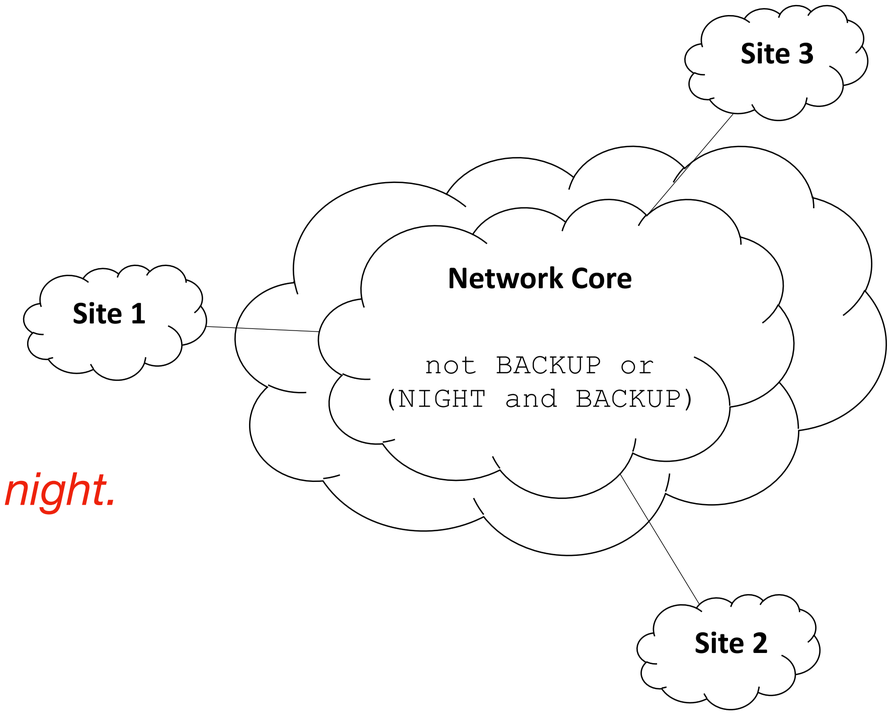
Time of Day Backups - *controller* variables

- ***Flow variable***
  - `BACKUP`
- ***Environment variable***
  - `NIGHT` - *value set by controller!*
- ***`Link requirements`***
  - `not BACKUP or (NIGHT and BACKUP)`



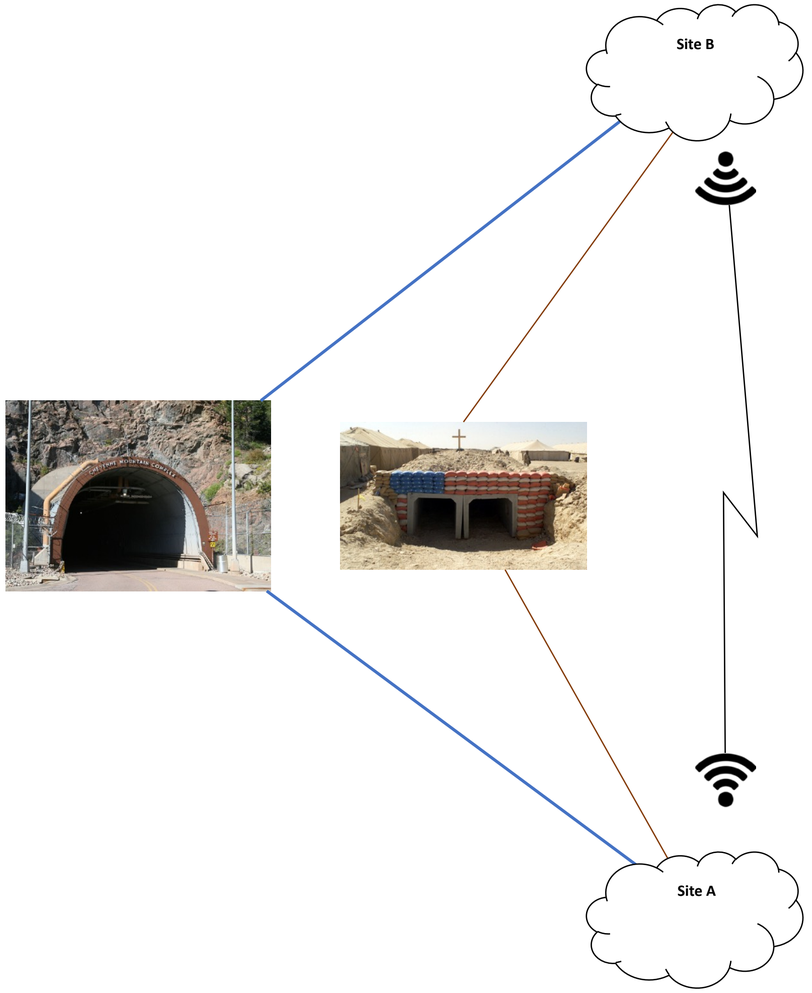
Time of Day Backups - *controller* variables

- ***Flow variable***
  - `BACKUP`
- ***Environment variable***
  - `NIGHT` - *value set by controller!*
- ***`Link requirements`***
  - `not BACKUP or (NIGHT and BACKUP)`
- *Backup traffic only allowed on network core at night.*



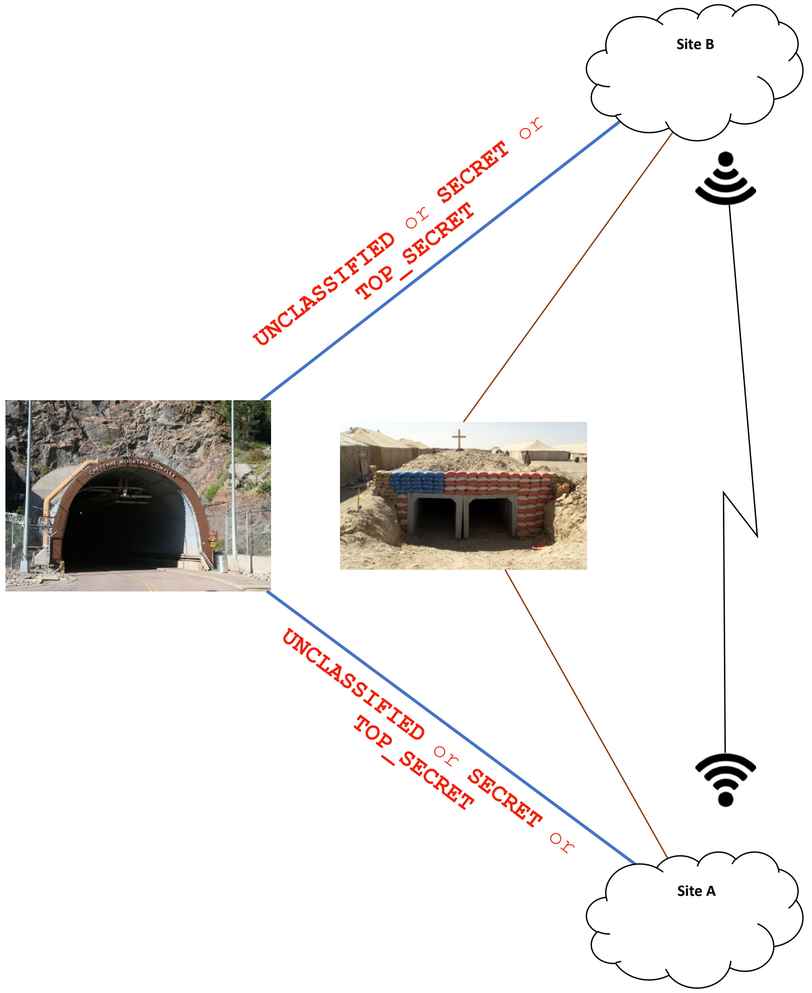
Multi-Layer Security

- ***Flow variables***
  - `TOP_SECRET`
  - `SECRET`
  - `UNCLASSIFIED`



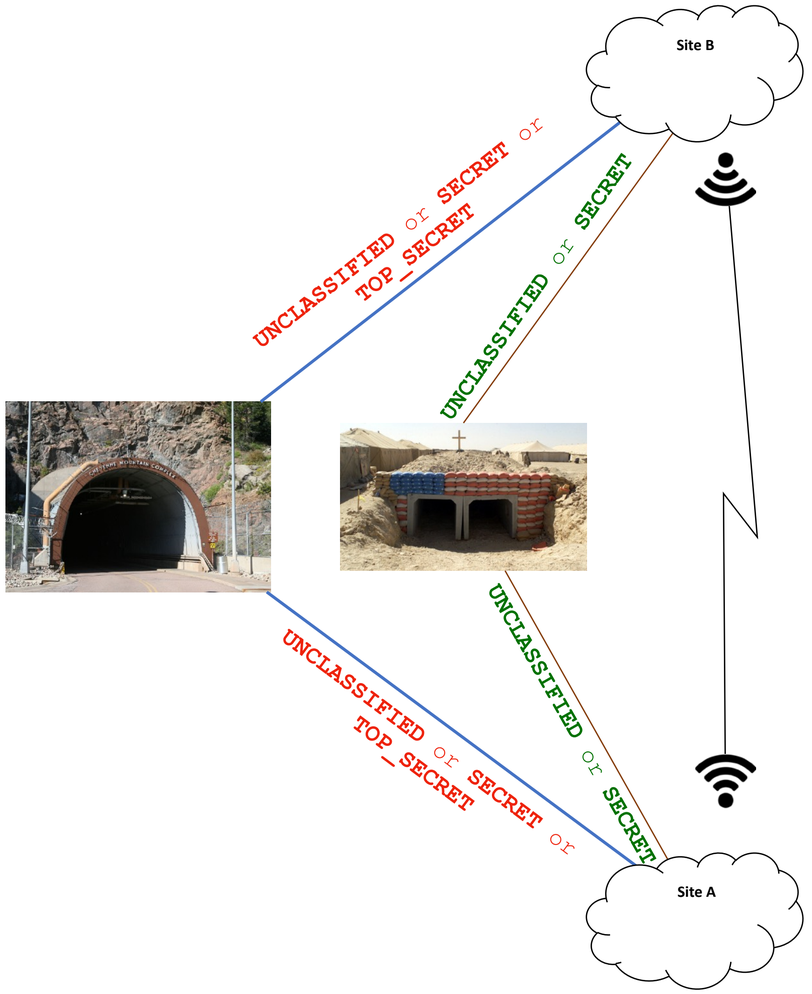
Multi-Layer Security

- ***Flow variables***
  - `TOP_SECRET`
  - `SECRET`
  - `UNCLASSIFIED`
- ***Link requirements***
  - HIGHLY SECURED: `UNCLASSIFIED or SECRET or TOP_SECRET`



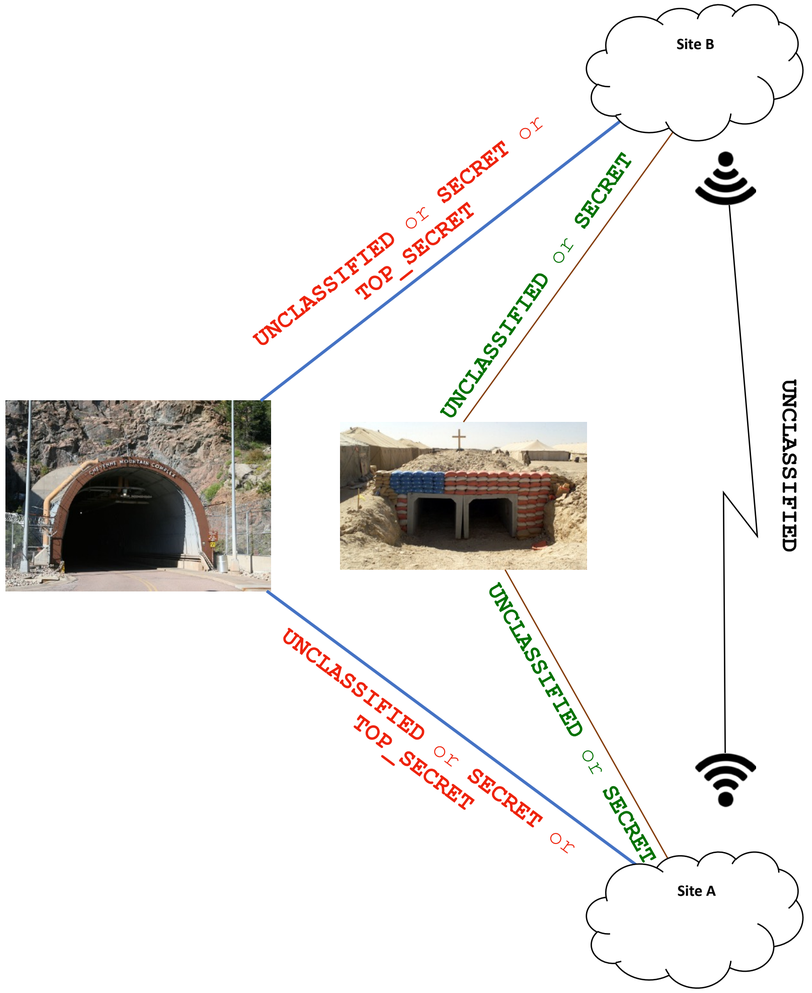
Multi-Layer Security

- ***Flow variables***
  - `TOP_SECRET`
  - `SECRET`
  - `UNCLASSIFIED`
- ***Link requirements***
  - HIGHLY SECURED: `UNCLASSIFIED or SECRET or TOP_SECRET`
  - SECURED: `UNCLASSIFIED or SECRET`



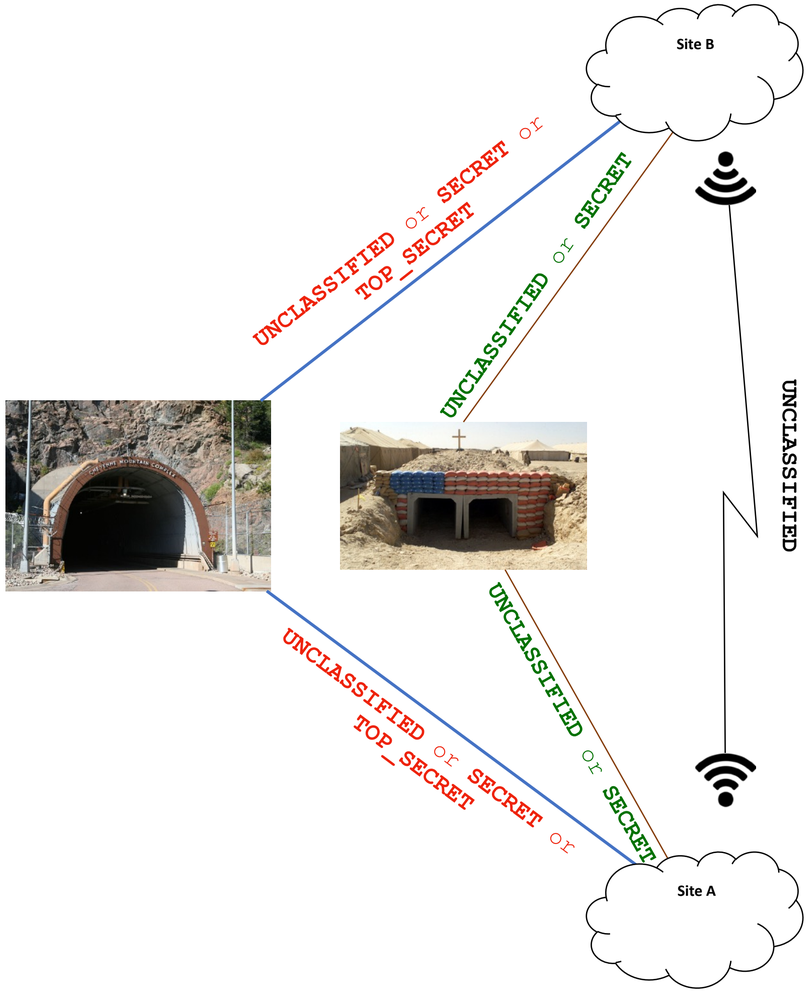
Multi-Layer Security

- ***Flow variables***
  - `TOP_SECRET`
  - `SECRET`
  - `UNCLASSIFIED`
- ***Link requirements***
  - HIGHLY SECURED: `UNCLASSIFIED or SECRET or TOP_SECRET`
  - SECURED: `UNCLASSIFIED or SECRET`
  - UNSECURED: `UNCLASSIFIED`



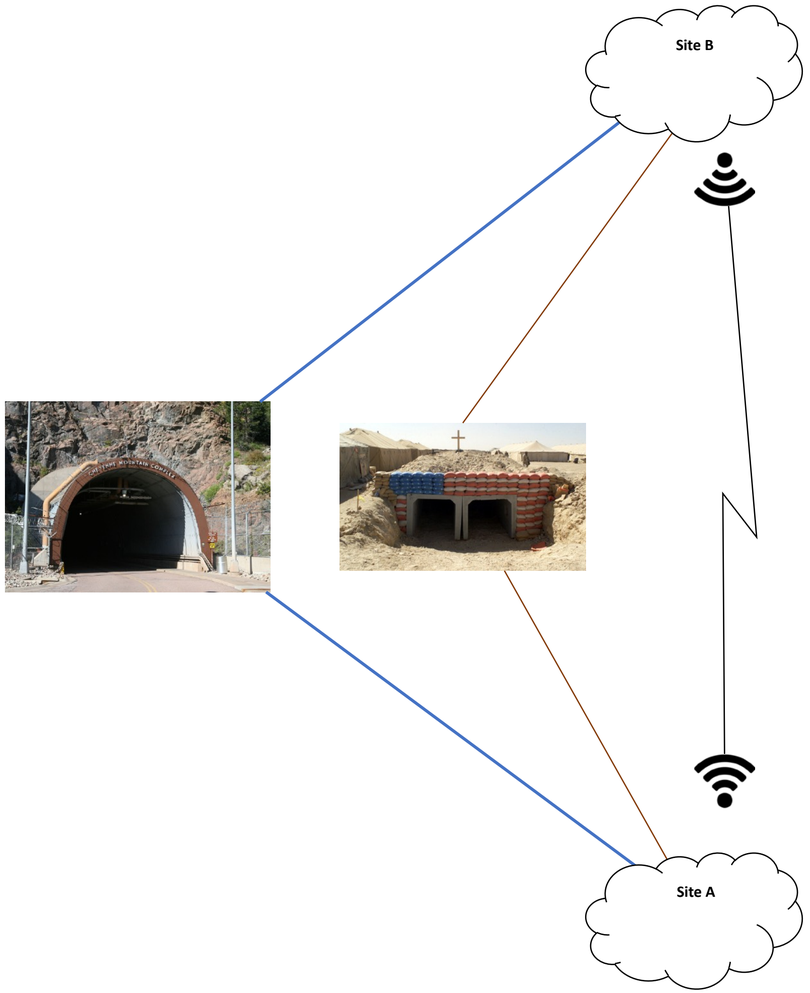
Multi-Layer Security

- ***Flow variables***
  - `TOP_SECRET`
  - `SECRET`
  - `UNCLASSIFIED`
- ***Link requirements***
  - HIGHLY SECURED: `UNCLASSIFIED or SECRET or TOP_SECRET`
  - SECURED: `UNCLASSIFIED or SECRET`
  - UNSECURED: `UNCLASSIFIED`
- *Traffic is routed over appropriately secure paths.*



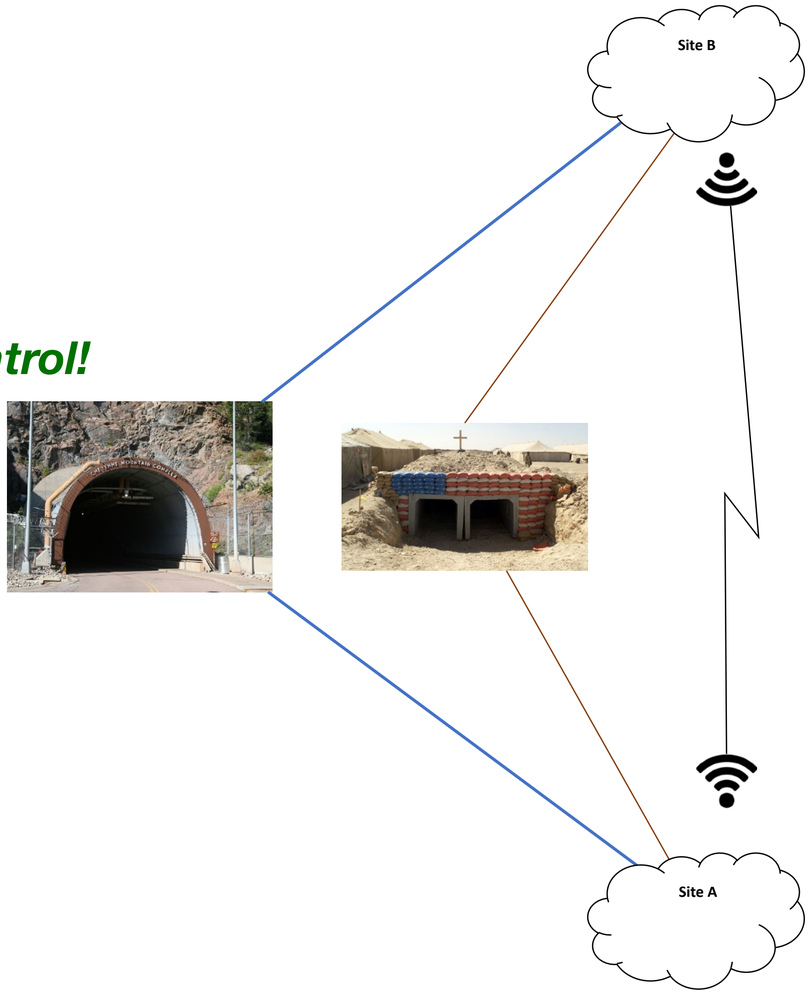
DEFCON/MLS - *programmatic* variables

- ***Flow variables***
  - **TS**
  - **SEC**
  - **UNC**



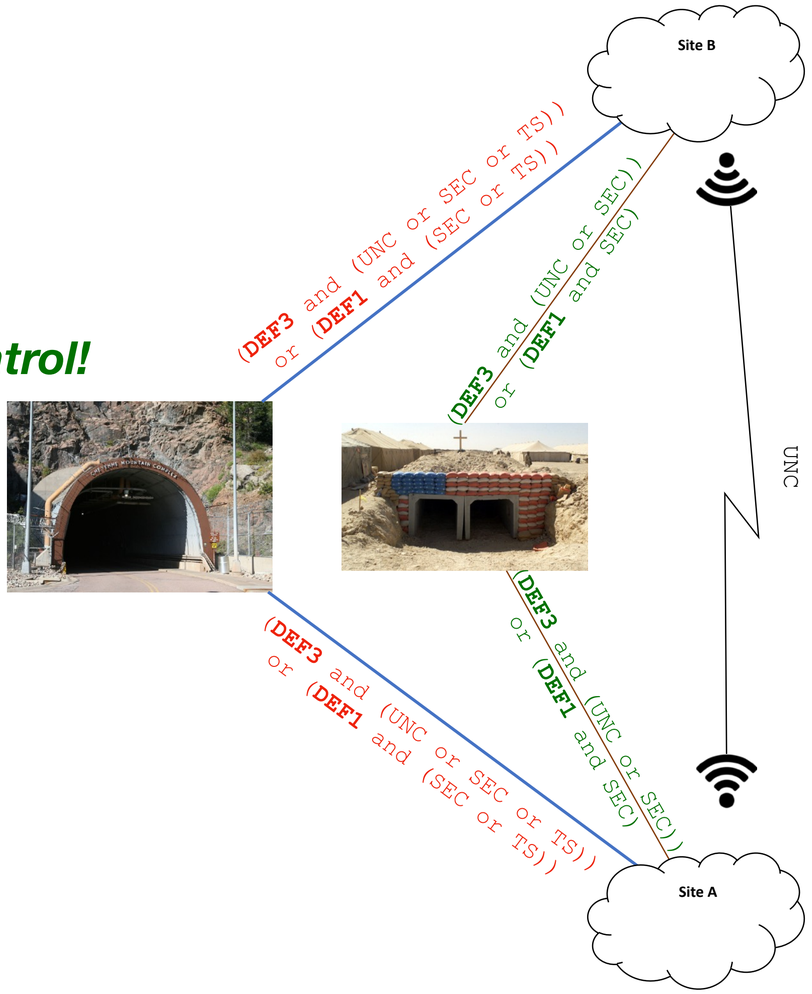
DEFCON/MLS - *programmatic* variables

- ***Flow variables***
  - **TS**
  - **SEC**
  - **UNC**
- ***Environmental requirements*** - ***remote programmatic control!***
  - **DEF1**
  - **DEF3**



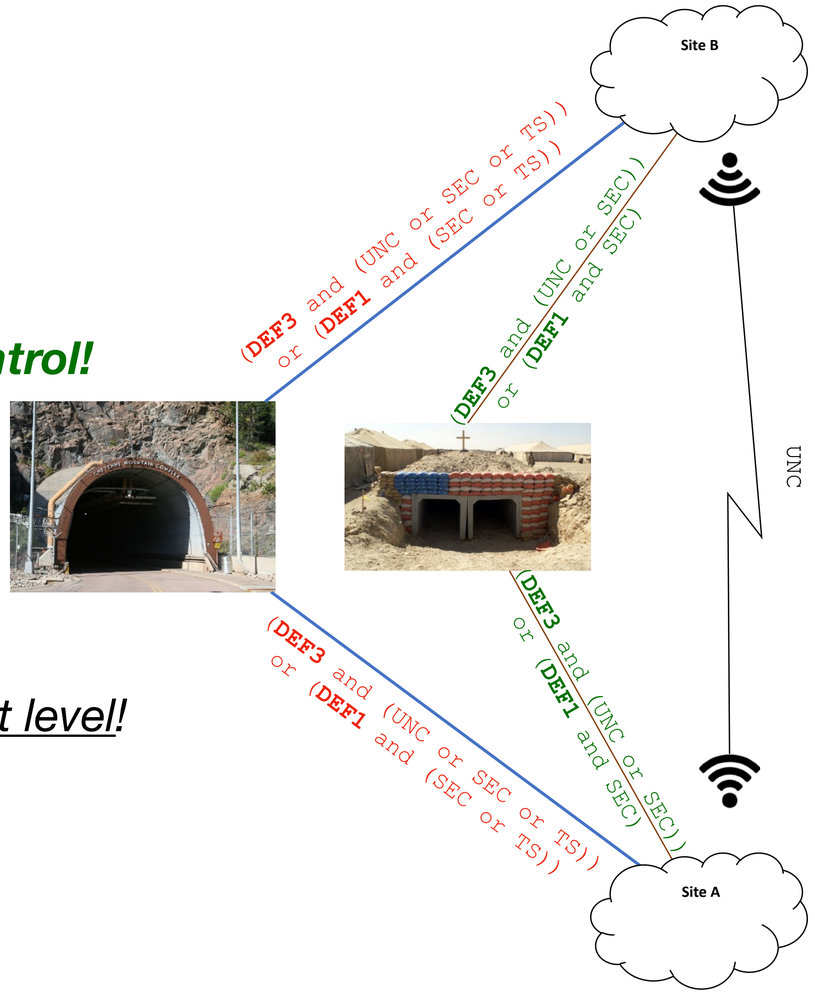
DEFCON/MLS - *programmatic* variables

- ***Flow variables***
  - **TS**
  - **SEC**
  - **UNC**
- ***Environmental requirements*** - ***remote programmatic control!***
  - **DEF1**
  - **DEF3**
- ***Link requirements***
  - (**DEF3** and (UNC or SEC or TS)) or (**DEF1** and (SEC or TS))
  - (**DEF3** and (UNC or SEC)) or (**DEF1** and SEC)
  - **UNC**



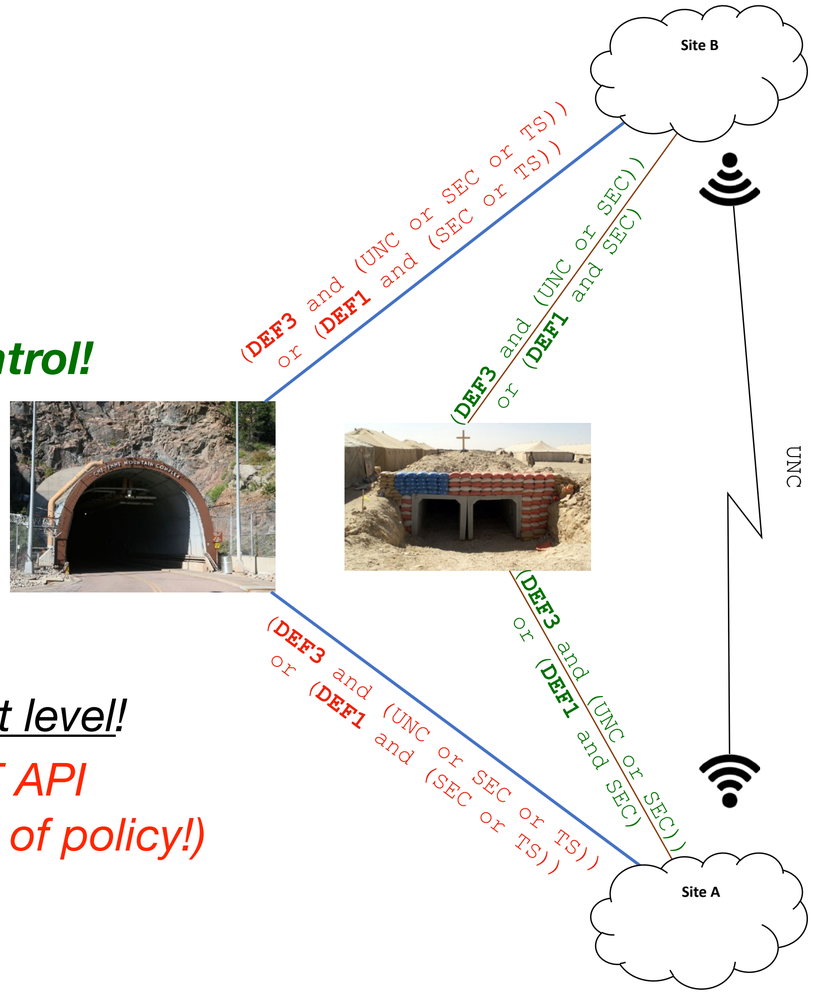
DEFCON/MLS - *programmatic* variables

- ***Flow variables***
  - **TS**
  - **SEC**
  - **UNC**
- ***Environmental requirements*** - ***remote programmatic control!***
  - **DEF1**
  - **DEF3**
- ***Link requirements***
  - (**DEF3** and (UNC or SEC or TS)) or (**DEF1** and (SEC or TS))
  - (**DEF3** and (UNC or SEC)) or (**DEF1** and SEC)
  - **UNC**
- *Traffic is routed over paths with security appropriate to threat level!*



DEFCON/MLS - *programmatic* variables

- ***Flow variables***
  - **TS**
  - **SEC**
  - **UNC**
- ***Environmental requirements*** - ***remote programmatic control!***
  - **DEF1**
  - **DEF3**
- ***Link requirements***
  - (**DEF3** and (UNC or SEC or TS)) or (**DEF1** and (SEC or TS))
  - (**DEF3** and (UNC or SEC)) or (**DEF1** and SEC)
  - **UNC**
- *Traffic is routed over paths with security appropriate to threat level!*
- *In prototype we implement environment variables with REST API (programmatic control of DEFCON enables external steering of policy!)*



Problems with the original Internet

Proposed Solution
Routing with requirements

How get “best set of paths” & Benefits

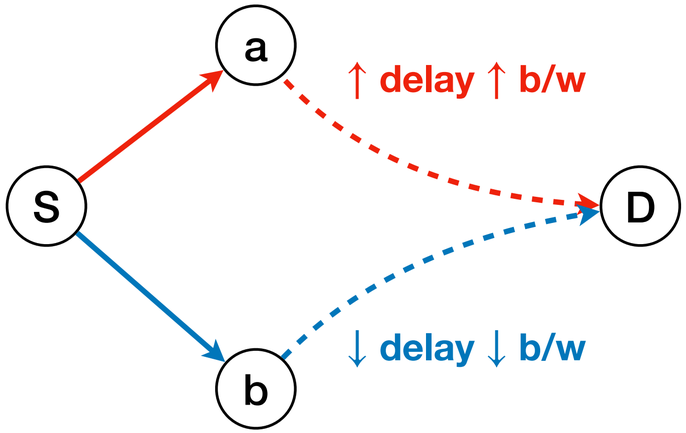
Testbed & Summary

A single weight is very limiting

- A *single weight* per dest (single path or ECMP) *limits usefulness of paths*

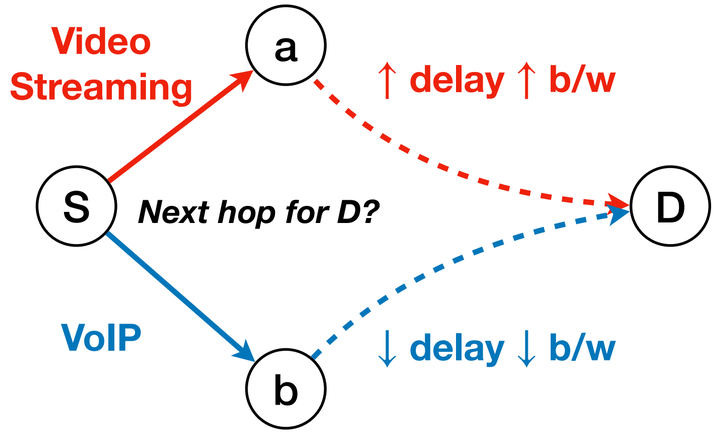
A single weight is very limiting

- A *single weight* per dest (single path or ECMP) *limits usefulness of paths*
  - In general, a single weight *doesn't meet the needs of all applications*



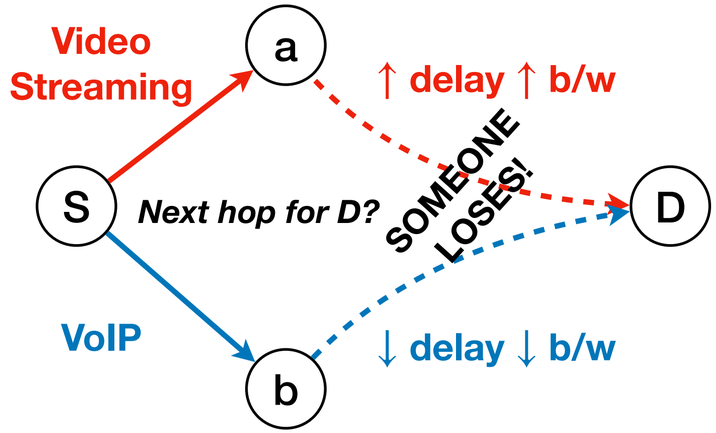
A single weight is very limiting

- A *single weight* per dest (single path or ECMP) *limits usefulness of paths*
  - In general, a single weight *doesn't meet the needs of all applications*



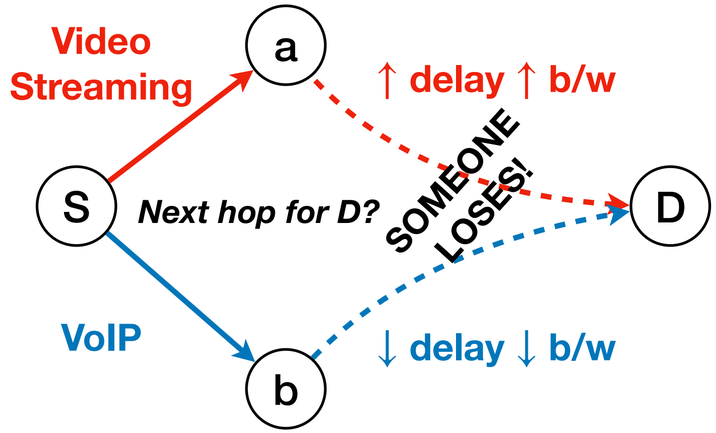
A single weight is very limiting

- A *single weight* per dest (single path or ECMP) *limits usefulness of paths*
  - In general, a single weight *doesn't meet the needs of all applications*



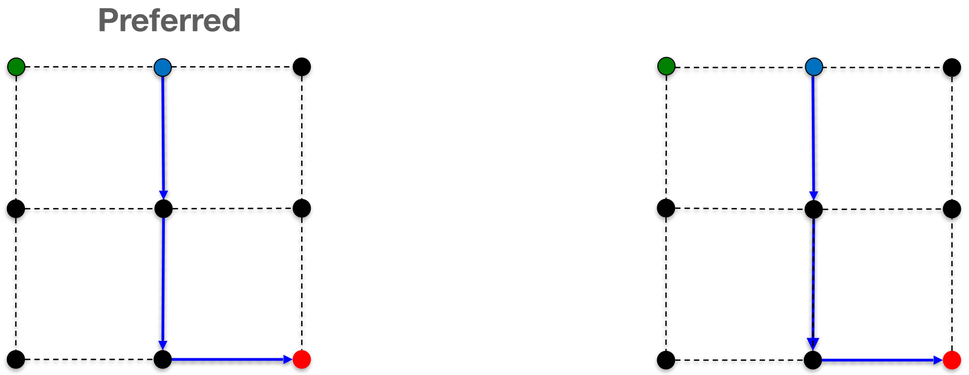
A single weight is very limiting

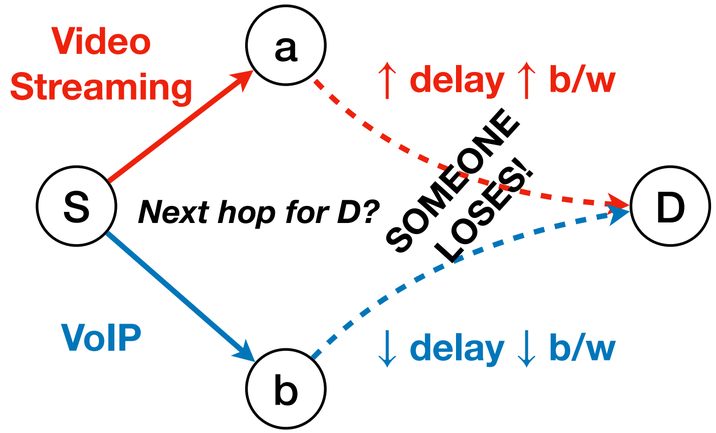
- A *single weight* per dest (single path or ECMP) *limits usefulness of paths*
  - In general, a single weight *doesn't meet the needs of all applications*
  - *Tends towards congestion* due to use of the small # of paths with the *single weight*



A single weight is very limiting

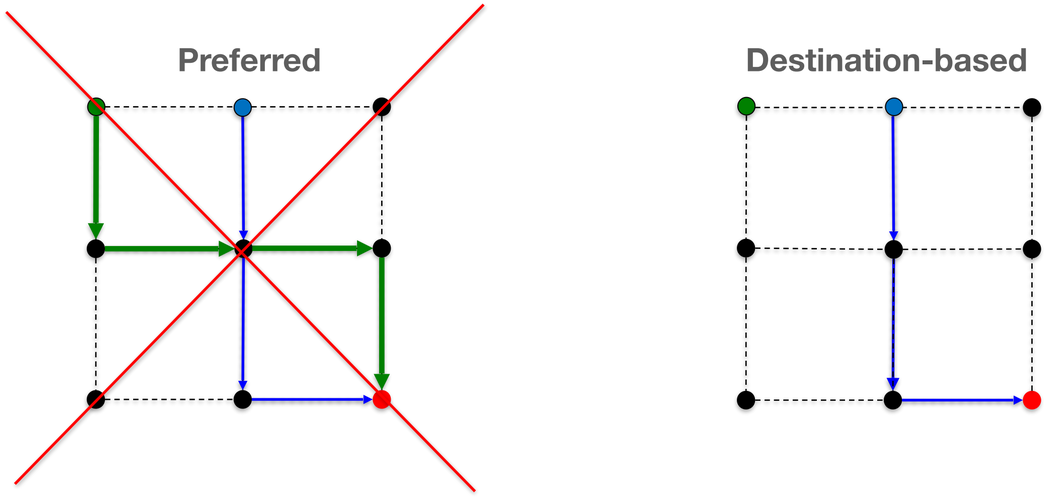
- A *single weight* per dest (single path or ECMP) *limits usefulness of paths*
  - In general, a single weight *doesn't meet the needs of all applications*
  - *Tends towards congestion* due to use of the small # of paths with the *single weight*
- *Destination-based forwarding* makes it worse…

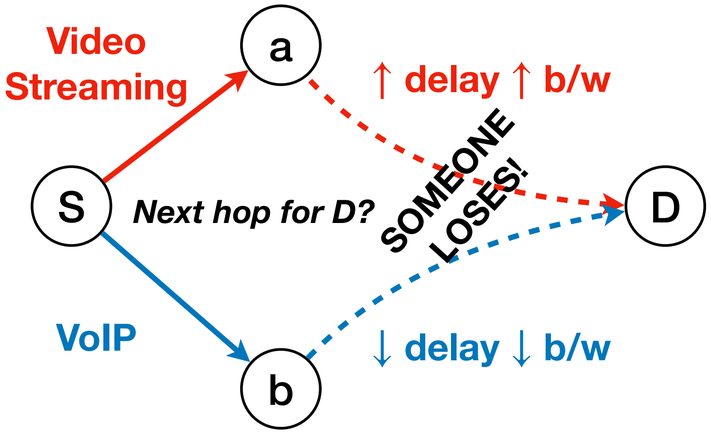




A single weight is very limiting

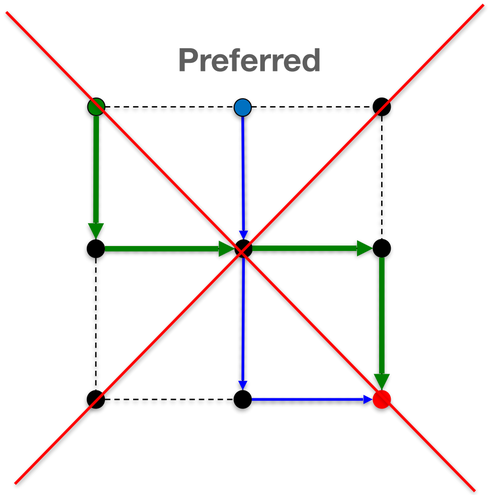
- A *single weight* per dest (single path or ECMP) *limits usefulness of paths*
  - In general, a single weight *doesn't meet the needs of all applications*
  - *Tends towards congestion* due to use of the small # of paths with the *single weight*
- *Destination-based forwarding* makes it worse…

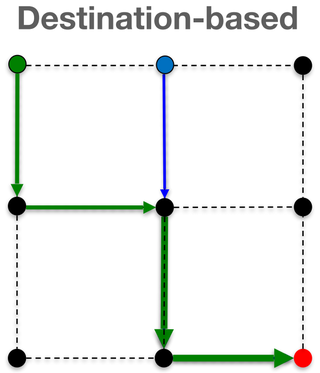


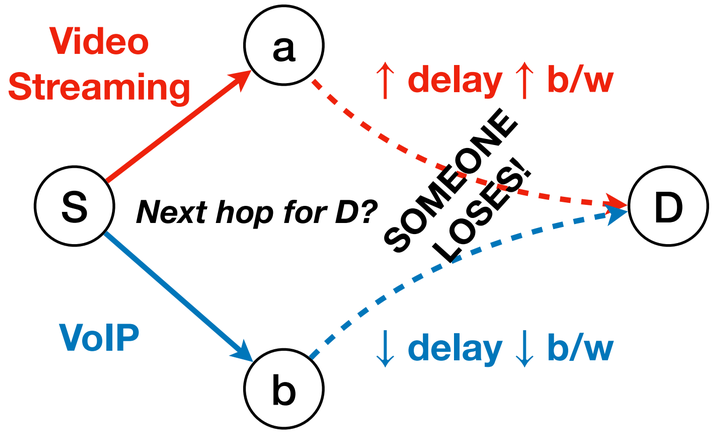


A single weight is very limiting

- A *single weight* per dest (single path or ECMP) *limits usefulness of paths*
  - In general, a single weight *doesn't meet the needs of all applications*
  - *Tends towards congestion* due to use of the small # of paths with the *single weight*
- *Destination-based forwarding* makes it worse…

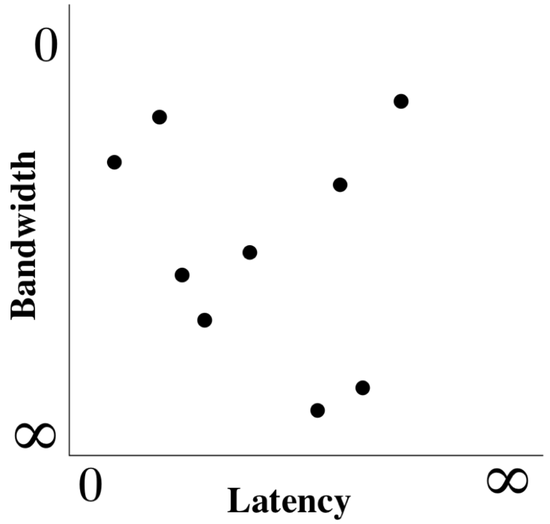






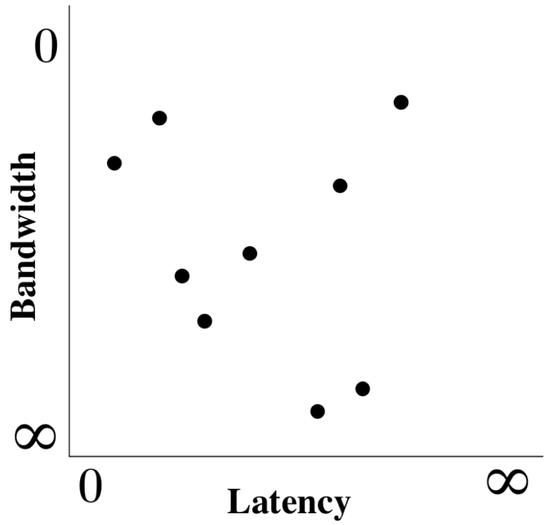
Paths to the same destination must overlap once they intersect.

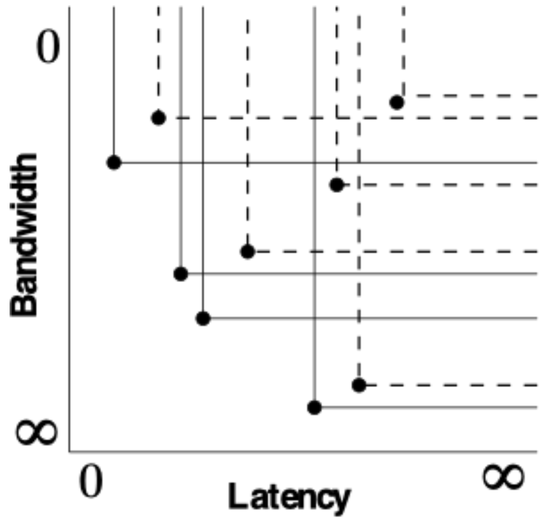
Ordered requirements - QoS b/w and latency



- Path weights are points in multi-dimensional space (best weight at origin)

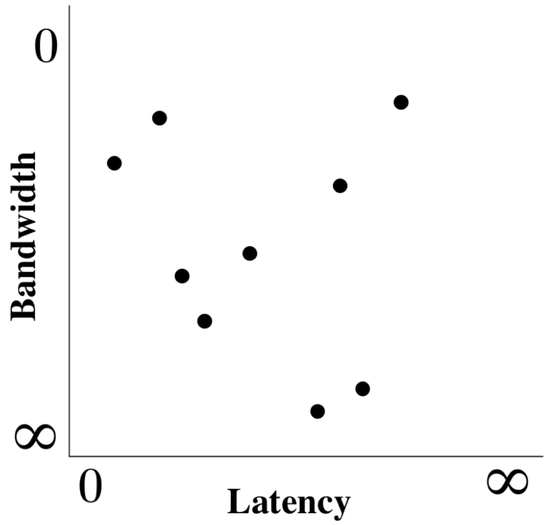
Ordered requirements - QoS b/w and latency

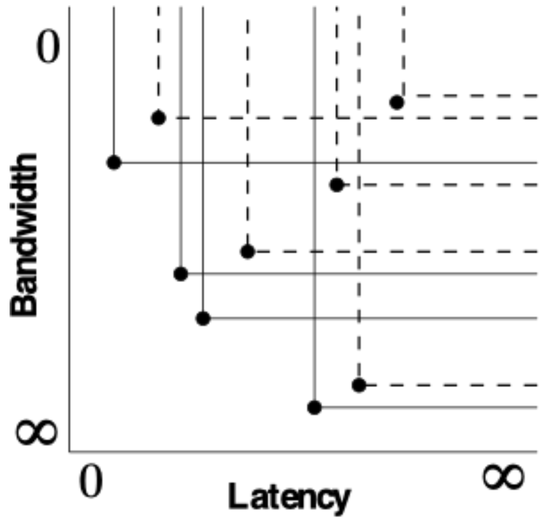


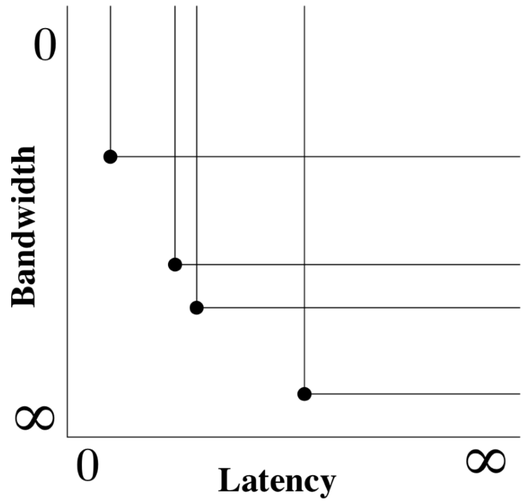


- Path weights are points in multi-dimensional space (best weight at origin)
- Some paths are “better” than others

Ordered requirements - QoS b/w and latency

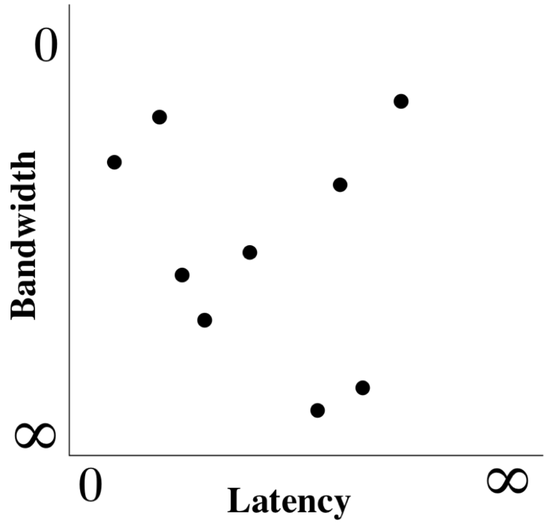


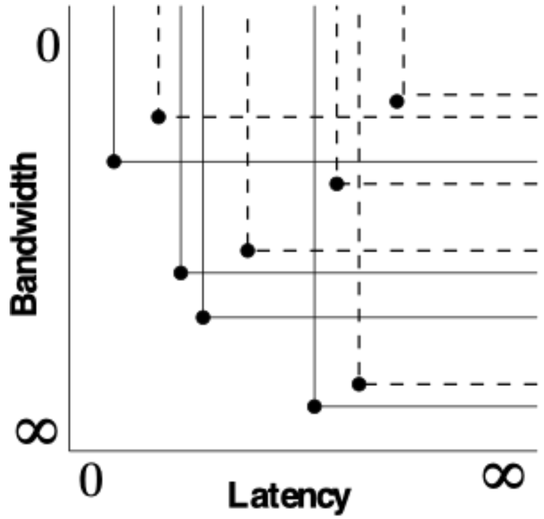


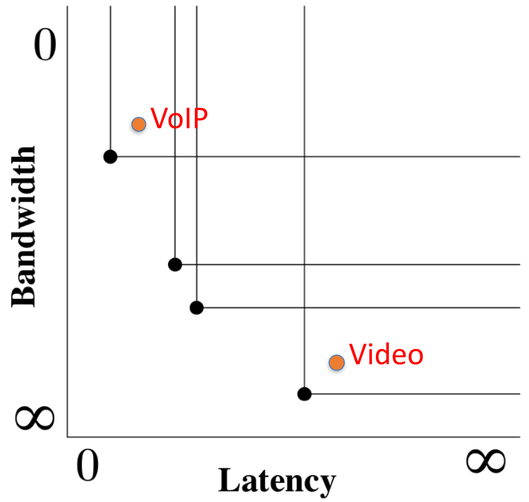


- Path weights are points in multi-dimensional space (best weight at origin)
- Some paths are “better” than others
- *Best set of paths* are those not ***dominated*** by any other path

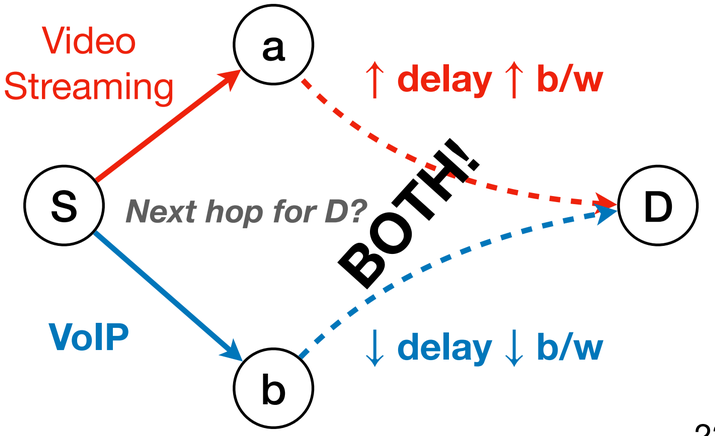
Ordered requirements - QoS b/w and latency







- Path weights are points in multi-dimensional space (best weight at origin)
- Some paths are “better” than others
- *Best set of paths* are those not ***dominated*** by any other path



Categorical requirements using Boolean expressions - *Multitent*

Some Path Expressions

	TA	TB	**TA**	**TB**	**TA or TB**	**TA and TB**	**TRUE**	**FALSE**
1	*F*	*F*	***F***	***F***	***F***	***F***	***T***	***F***
2	*F*	*T*	***F***	***T***	***T***	***F***	***T***	***F***
3	*T*	*F*	***T***	***F***	***T***	***F***	***T***	***F***
4	*T*	*T*	***T***	***T***	***T***	***T***	***T***	***F***

Categorical requirements using Boolean expressions - *Multitent*

Some Path Expressions

	TA	*TB*	***TA***	***TB***	***TA or TB***	***TA and TB***	***TRUE***	***FALSE***
1	*F*	*F*	***F***	***F***	***F***	***F***	***T***	***F***
2	*F*	*T*	***F***	***T***	***T***	***F***	***T***	***F***
3	*T*	*F*	***T***	***F***	***T***	***F***	***T***	***F***
4	*T*	*T*	***T***	***T***	***T***	***T***	***T***	***F***

Categorical requirements using Boolean expressions - *Multitent*

Some Path Expressions

	TA	TB	TA	TB	TA or TB	TA and TB	TRUE	FALSE
1	F	F	F	F	F	F	T	F
2	F	T	F	T	T	F	T	F
3	T	F	T	F	T	F	T	F
4	T	T	T	T	T	T	T	F

Categorical requirements using Boolean expressions - *Multitent*

Some Path Expressions

	TA	TB	TA	TB	TA or TB	TA and TB	TRUE	FALSE
1	F	F	F	F	F	F	T	F
2	F	T	F	T	T	F	T	F
3	T	F	T	F	T	F	T	F
4	T	T	T	T	T	T	T	F

Categorical requirements using Boolean expressions - *Multitent*

Some Path Expressions

	TA	TB	TA	TB	TA or TB	TA and TB	TRUE	FALSE
1	F	F	F	F	F	F	T	F
2	F	T	F	T	T	F	T	F
3	T	F	T	F	T	F	T	F
4	T	T	T	T	T	T	T	F

Path Access of flow between TA and some node that is not in TB

	TA	TB	TA and ¬TB	TA	TB	TA or TB	TA and TB	TRUE	FALSE
1	F	F	F	F	F	F	F	T	F
2	F	T	F	F	T	T	F	T	F
3	T	F	T	T	F	T	F	T	F
4	T	T	F	T	T	T	T	T	F

Categorical requirements using Boolean expressions - *Multitent*

Some Path Expressions

	TA	TB	TA	TB	TA or TB	TA and TB	TRUE	FALSE
1	F	F	F	F	F	F	T	F
2	F	T	F	T	T	F	T	F
3	T	F	T	F	T	F	T	F
4	T	T	T	T	T	T	T	F

Path Access of flow between TA and some node that is not in TB

	TA	TB	TA and ¬TB	TA	TB	TA or TB	TA and TB	TRUE	FALSE
1	F	F	F	F	F	F	F	T	F
2	F	T	F	F	T	T	F	T	F
3	T	F	T	T	F	T	F	T	F
4	T	T	F	T	T	T	T	T	F

Categorical requirements using Boolean expressions - *Multitent*

Some Path Expressions

	TA	TB	TA	TB	TA or TB	TA and TB	TRUE	FALSE
1	F	F	F	F	F	F	T	F
2	F	T	F	T	T	F	T	F
3	T	F	T	F	T	F	T	F
4	T	T	T	T	T	T	T	F

Path Access of flow between TA and some node that is not in TB

	TA	TB	TA and ¬TB	TA	TB	TA or TB	TA and TB	TRUE	FALSE
1	F	F	F	F	F	F	F	T	F
2	F	T	F	F	T	T	F	T	F
3	T	F	T	T	F	T	F	T	F
4	T	T	F	T	T	T	T	T	F

Categorical requirements using Boolean expressions - *Multitent*

Some Path Expressions

	TA	TB	TA	TB	TA or TB	TA and TB	TRUE	FALSE
1	F	F	F	F	F	F	T	F
2	F	T	F	T	T	F	T	F
3	T	F	T	F	T	F	T	F
4	T	T	T	T	T	T	T	F

Path Access of flow between TA and some node that is not in TB

	TA	TB	TA and ¬TB	TA	TB	TA or TB	TA and TB	TRUE	FALSE
1	F	F	F	F	F	F	F	T	F
2	F	T	F	F	T	T	F	T	F
3	T	F	T	T	F	T	F	T	F
4	T	T	F	T	T	T	T	T	F

- A constraint must be true for *at least one* of a path constraint's satisfying truth assignments.

Categorical requirements using Boolean expressions - *Multitent*

Some Path Expressions

	TA	TB	TA	TB	TA or TB	TA and TB	TRUE	FALSE
1	F	F	F	F	F	F	T	F
2	F	T	F	T	T	F	T	F
3	T	F	T	F	T	F	T	F
4	T	T	T	T	T	T	T	F

Path Access of flow between TA and some node that is not in TB

	TA	TB	TA and ¬TB	TA	TB	TA or TB	TA and TB	TRUE	FALSE
1	F	F	F	F	F	F	F	T	F
2	F	T	F	F	T	T	F	T	F
3	T	F	T	T	F	T	F	T	F
4	T	T	F	T	T	T	T	T	F

- A constraint must be true for *at least one* of a path constraint's satisfying truth assignments.
- Flow constraint *(TA and ¬TB)* is allowed for path expressions *TA*, *(TA or TB)*, and *TRUE*

Categorical requirements using Boolean expressions - *Multitent*

Some Path Expressions

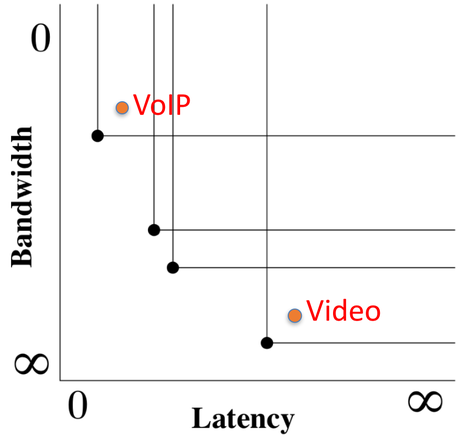
	TA	TB	TA	TB	TA or TB	TA and TB	TRUE	FALSE
1	F	F	F	F	F	F	T	F
2	F	T	F	T	T	F	T	F
3	T	F	T	F	T	F	T	F
4	T	T	T	T	T	T	T	F

Path Access of flow between TA and some node that is not in TB

	TA	TB	TA and ¬TB	TA	TB	TA or TB	TA and TB	TRUE	FALSE
1	F	F	F	F	F	F	F	T	F
2	F	T	F	F	T	T	F	T	F
3	T	F	T	T	F	T	F	T	F
4	T	T	F	T	T	T	T	T	F

- A constraint must be true for *at least one* of a path constraint's satisfying truth assignments.
- Flow constraint *(TA and ¬TB)* is allowed for path expressions *TA*, *(TA or TB)*, and *TRUE*
- A flow can use a path if *the conjunction ('and'ing) of the flow and path constraints is* ***satisfiable***

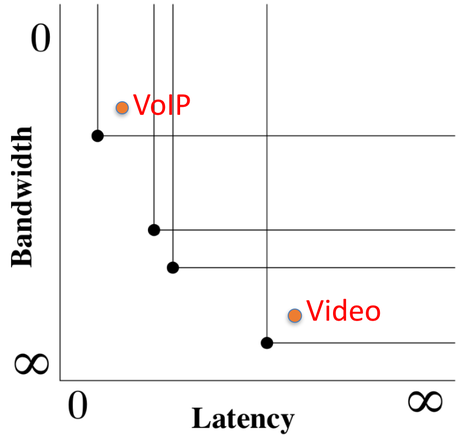
Routing with Requirements… pulling it all together



Path Access of flow between TA and some node that is not in TB

	TA	TB	TA and ¬TB	TA	TB	TA or TB	TA and TB	TRUE	FALSE
1	F	F	F	F	F	F	F	T	N
2	F	T	F	F	T	T	F	T	N
3	T	F	T	T	F	T	F	T	N
4	T	T	F	T	T	T	T	T	N

Routing with Requirements… pulling it all together



X

Path Access of flow between TA and some node that is not in TB

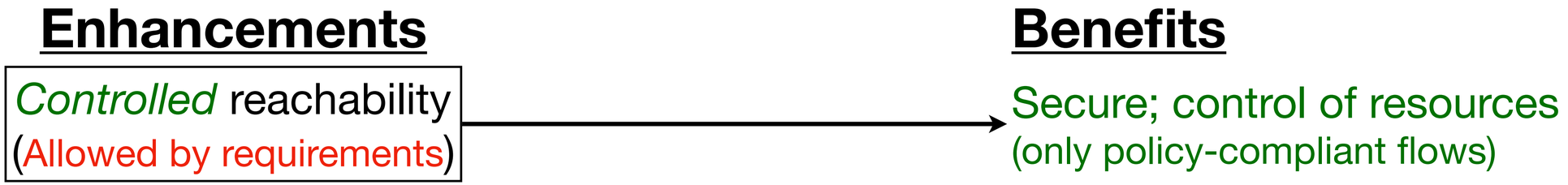
	TA	*TB*	*TA and ¬TB*	***TA***	***TB***	***TA or TB***	***TA and TB***	***TRUE***	***FALSE***
1	*F*	*F*	***F***	***F***	***F***	***F***	***F***	***T***	N
2	*F*	*T*	***F***	***F***	***T***	***T***	***F***	***T***	N
3	*T*	*F*	***T***	***T***	***F***	***T***	***F***	***T***	N
4	*T*	*T*	***F***	***T***	***T***	***T***	***T***	***T***	N

In effect, compute and use the ***dominant set of paths…*** for each satisfying truth assignment.

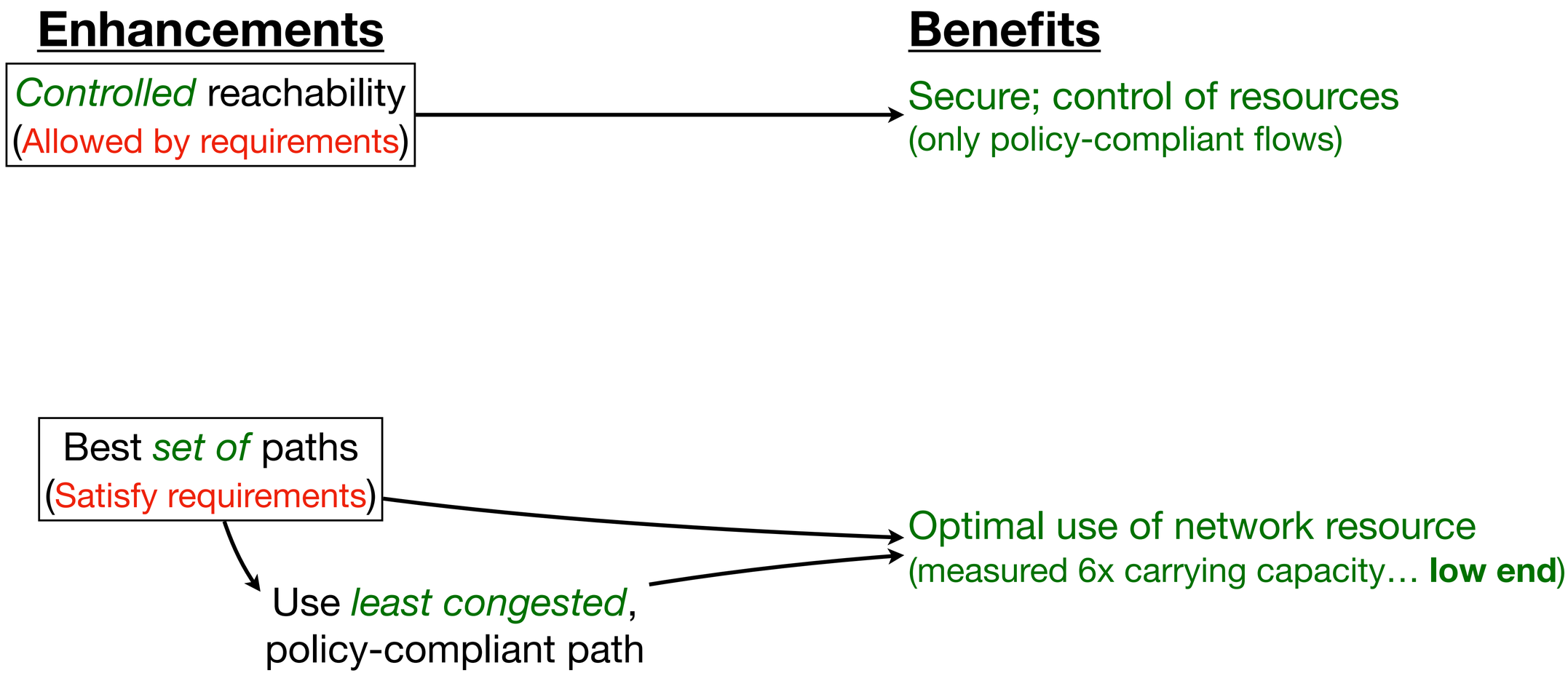
Benefits of Routing with Requirements

Enhancements	Benefits

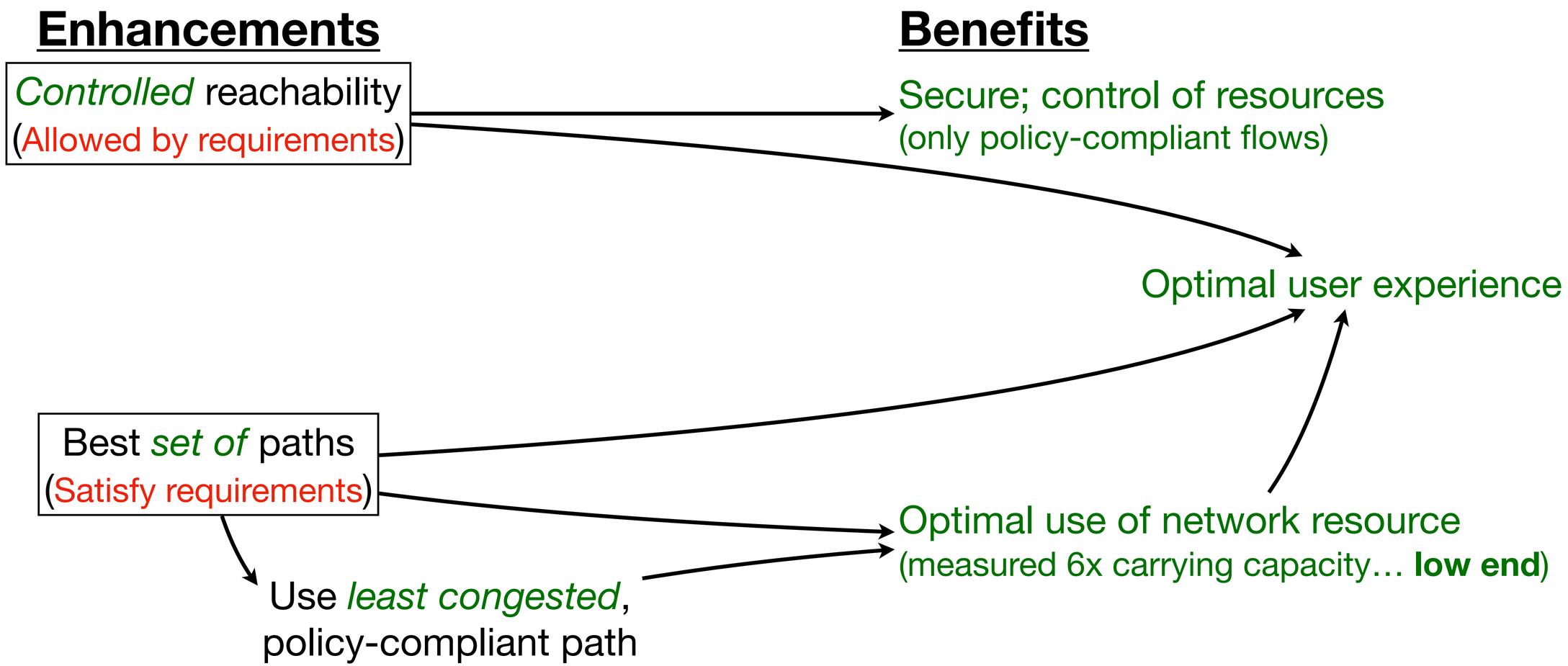
Benefits of Routing with Requirements



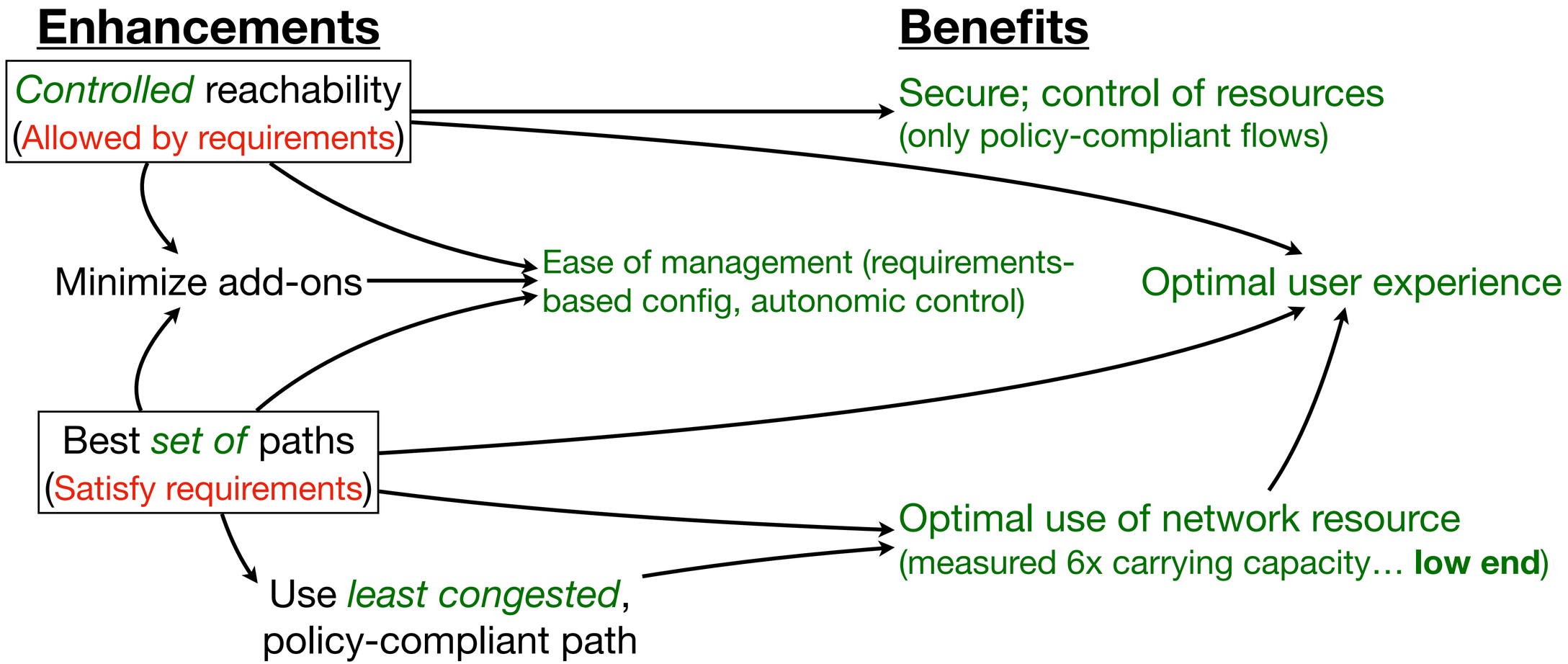
Benefits of Routing with Requirements



Benefits of Routing with Requirements



Benefits of Routing with Requirements



Problems with the original Internet

Proposed Solution
Routing with requirements

Benefits

Testbed & Summary

Test Lab results *untuned* Prototype (3x3 torus)

"Evaluation of NwC features like QoS, Zero Trust, and Network Segmentation – verified to meet expected results. The NwC controller efficiently uses the flow table resources by deleting flow table entries that haven't been used for a period of time."

Test Lab results *untuned* Prototype (3x3 torus)

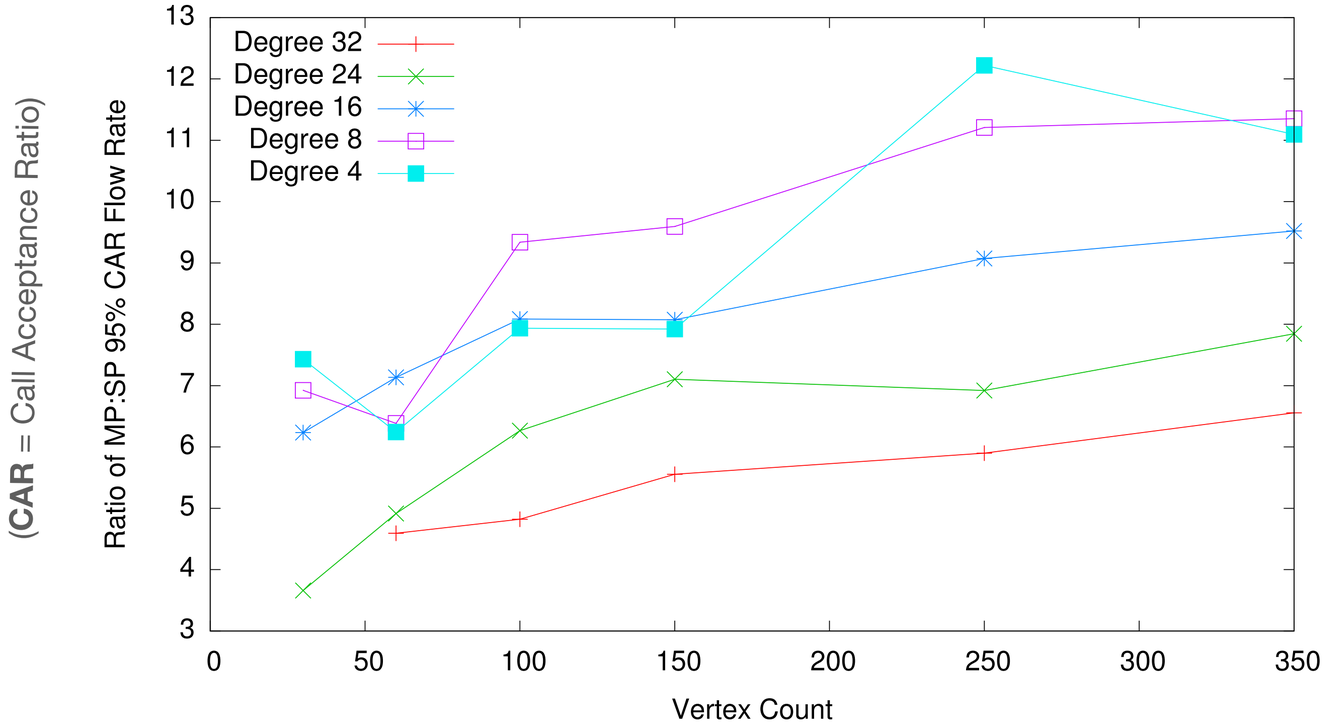
"Evaluation of NwC features like QoS, Zero Trust, and Network Segmentation – verified to meet expected results. The NwC controller efficiently uses the flow table resources by deleting flow table entries that haven't been used for a period of time."

"For TCP traffic, NwC delivers up to 11% greater throughput than RSTP at 6x the load. For UDP traffic, NwC delivers similar amount of goodput and average loss rate on total bandwidth as RSTP, but at 6x the load than RSTP"

Test Lab results *untuned* Prototype (3x3 torus)

"Evaluation of NwC features like QoS, Zero Trust, and Network Segmentation – verified to meet expected results. The NwC controller efficiently uses the flow table resources by deleting flow table entries that haven't been used for a period of time."

"For TCP traffic, NwC delivers up to 11% greater throughput than RSTP at 6x the load. For UDP traffic, NwC delivers similar amount of goodput and average loss rate on total bandwidth as RSTP, but at 6x the load than RSTP"



Summary - Routing with requirements

- Requirements defined by user, application, network admin

Summary - Routing with requirements

- Requirements defined by user, application, network admin
- Works over anything… *exploits special functionality when it's available!*

Summary - Routing with requirements

- Requirements defined by user, application, network admin
- Works over anything… *exploits special functionality when it's available!*
- Benefits
  - Secure
  - Optimize user experience (policy-compliant flows)
  - Optimize use of network resources
  - Intuitive configuration
  - Autonomic control (Boolean variables)… threat level, roles, schedule, etc.
  - Many applications… Zero Trust, network segmentation, 5G slices, IPMS, …

Summary - Routing with requirements

- Requirements defined by user, application, network admin
- Works over anything… *exploits special functionality when it's available!*
- Benefits
  - Secure
  - Optimize user experience (policy-compliant flows)
  - Optimize use of network resources
  - Intuitive configuration
  - Autonomic control (Boolean variables)… threat level, roles, schedule, etc.
  - Many applications… Zero Trust, network segmentation, 5G slices, IPMS, …
- Implementation options… network/link layer, controller/distributed

For questions, to express interest in collaborating, or to request online access to the prototype (only requires a web browser) contact me at:

brad@curatednetworks.com