Detecting Configuration Errors Via Pattern Mining

Jyotirmay Chauhan, Devon Lee,
Emily Yu, Aaron Gember-Jacobson

Configurations are complex

```
interface iface1
  description To Core
  address 1.0.1.1/24
  access-list filterA in
```

- Many attributes

Configurations are complex

```
interface iface1
  description To Core
  address 1.0.1.1/24
  access-list filterA in

interface iface2
  description Floor 1
  switchport allowed vlans 10, 20

interface iface3
  description Floor 2
  switchport allowed vlans 10
```

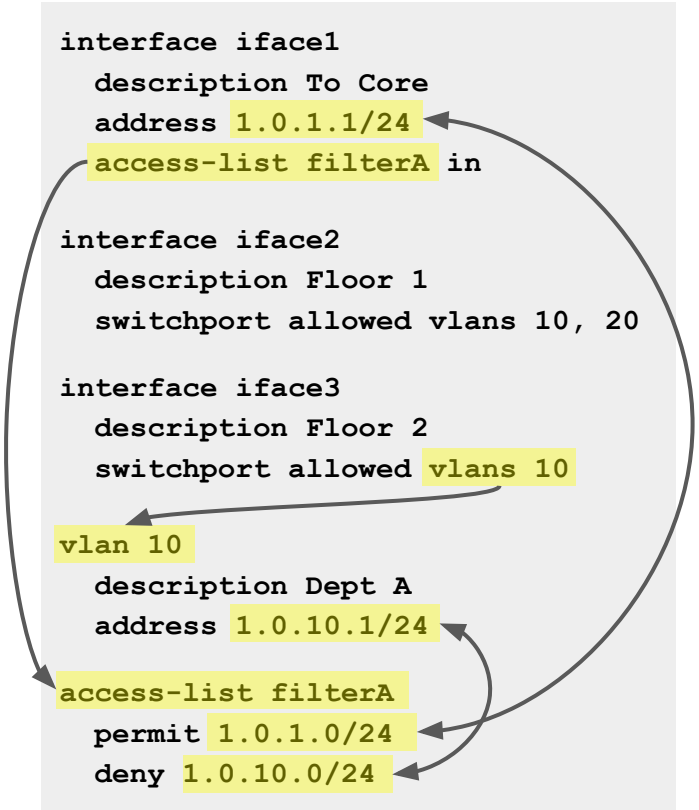
- Many attributes
- Many stanzas

Configurations are complex

```
interface iface1
  description To Core
  address 1.0.1.1/24
  access-list filterA in

interface iface2
  description Floor 1
  switchport allowed vlans 10, 20

interface iface3
  description Floor 2
  switchport allowed vlans 10

vlan 10
  description Dept A
  address 1.0.10.1/24

access-list filterA
  permit 1.0.1.0/24
  deny 1.0.10.0/24
```

- Many attributes
- Many stanzas
- Multiple types of stanzas

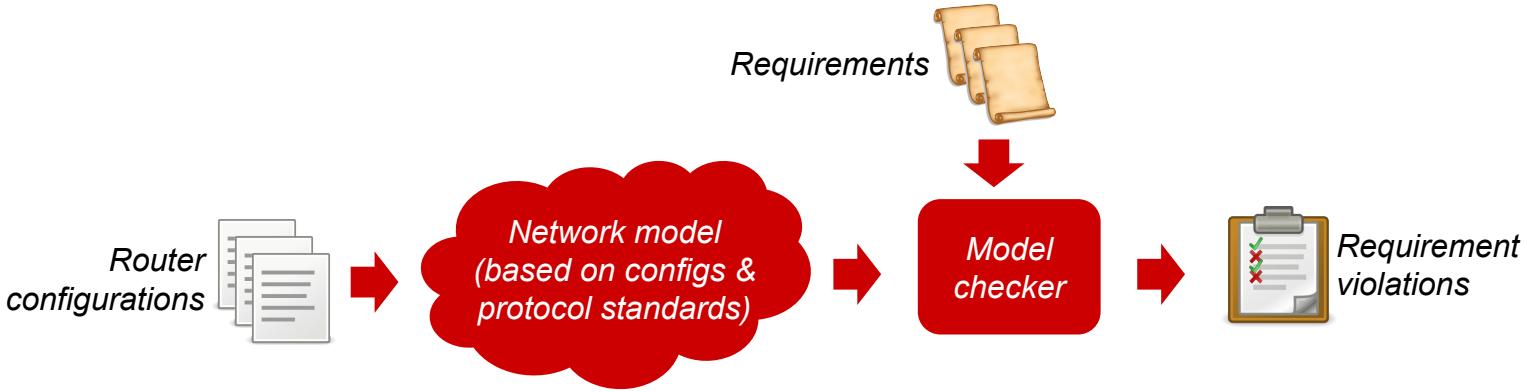
Configurations are complex



- Many attributes
- Many stanzas
- Multiple types of stanzas
- Complex relationships between stanzas

Misconfigurations are hard to find

Model checking (e.g., Minesweeper, Plankton, Tiramisu)



✘ Hard to construct an accurate/efficient network model

✘ Hard to enumerate requirements

✘ Hard to relate violations to specific lines of configuration

Pattern mining (e.g., SelfStarter, Minerals)

- Routers within same network have similar configurations
- Identify patterns → flag outliers as potential bugs

✔ Don’t need to specify requirements or construct a model

✔ Errors are localized to specific lines configuration

Association rule mining (Minerals)

- Find common combinations of attributes within a specific type of stanza

`access-list in ⇒ access-list out`

```
interface iface1
  description Dept A
  address 1.0.1.1/24
  access-list filterA in
  access-list filterZ out

interface iface2
  description Dept B
  address 1.0.2.1/24
  access-list filterB in
  access-list filterZ out

interface iface3
  address 1.0.3.1/24
  access-list filterC in
```

Template inference (SelfStarter)

- Find parameterized-patterns within ACLs, route filters, etc.

```
permit 1.0.__.0/24 any
deny any any
```

```
access-list filterA
  permit 1.0.1.0/24 any
  deny any any

access-list filterB
  permit 1.0.22.0/24 any
  deny any any

access-list filterC
  permit 1.0.3.0/24 any
  deny any any

access-list filterD
  permit 1.0.4.0/23 any
  deny any any
```

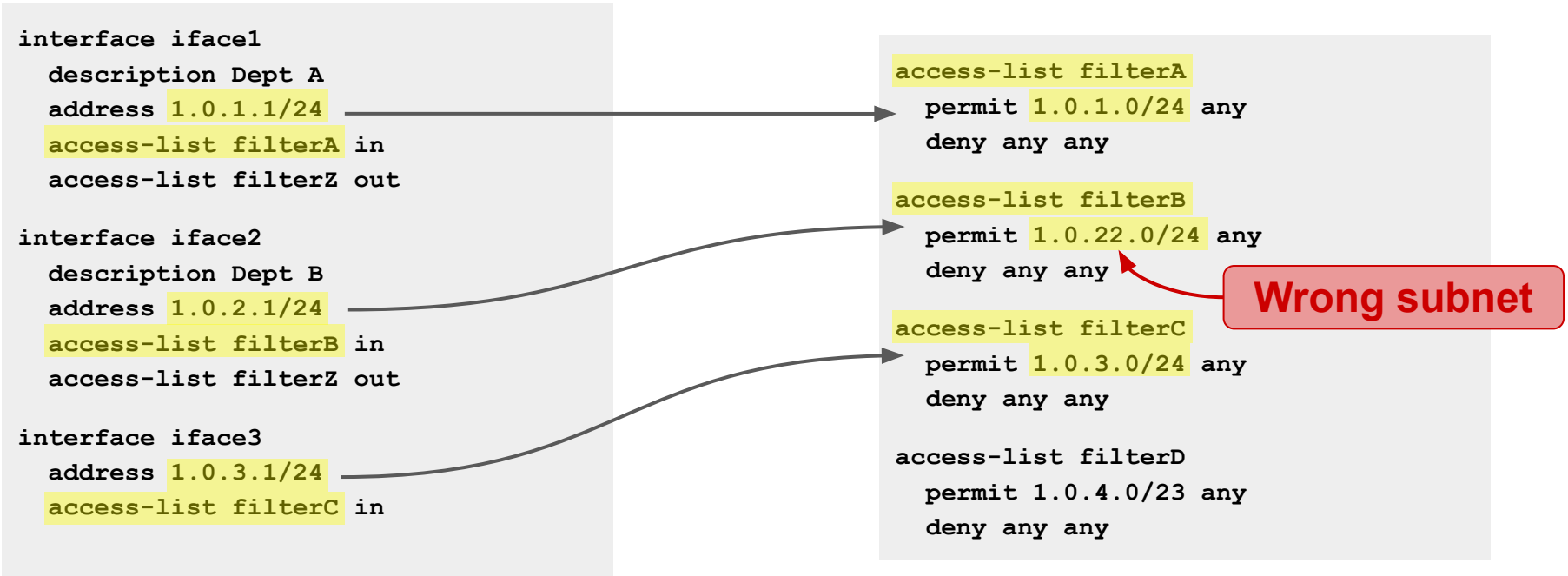
Wrong mask

Patterns also exist across different types of stanzas

prefix allowed by interface's inbound ACL == interface's prefix

```
interface iface1
  description Dept A
  address 1.0.1.1/24
  access-list filterA in
  access-list filterZ out

interface iface2
  description Dept B
  address 1.0.2.1/24
  access-list filterB in
  access-list filterZ out

interface iface3
  address 1.0.3.1/24
  access-list filterC in
```

```
access-list filterA
  permit 1.0.1.0/24 any
  deny any any

access-list filterB
  permit 1.0.22.0/24 any
  deny any any

access-list filterC
  permit 1.0.3.0/24 any
  deny any any

access-list filterD
  permit 1.0.4.0/23 any
  deny any any
```



Patterns also exist across different types of stanzas & non-operational attributes

```
interface iface1
  description Dept A
  address 1.0.1.1/24
  access-list filterA in
  access-list filterZ out

interface iface2
  description Dept B
  address 1.0.2.1/24
  access-list filterB in
  access-list filterZ out

interface iface3
  address 1.0.3.1/24
  access-list filterC in
```

```
access-list filterA
  permit 1.0.1.0/24 any
  deny any any

access-list filterB
  permit 1.0.22.0/24 any
  deny any any

access-list filterC
  permit 1.0.3.0/24 any
  deny any any

access-list filterD
  permit 1.0.4.0/23 any
  deny any any
```

Our contributions

1) Classification of patterns involving multiple types of stanzas and non-operational attributes

2) Methods to automatically mine such patterns

1) Classification of patterns involving multiple types of stanzas & non-operational attributes

- Uncovered through copious manual examination of configurations from nine university / research & education networks

- Patterns
  - Reference counts
  - Mutual references

  } *References between multiple types of stanzas*

  - Common keywords

  } *Non-operational attributes*

Reference counts example

- Dedicated VLAN for each pair of core routers
- VLAN allowed on a single interface

```
interface iface1
  description Bldg X
  switchport allowed vlans 10

interface iface2
  description Bldg Y
  switchport allowed vlans 10, 20

interface iface3
  description Bldg Z
  switchport allowed vlans 10, 20

interface iface4
  description Core 2
  switchport allowed vlans 10, 20, 30
```



```
vlan 10
  description Dept A
  ip address 1.0.10.5/24

vlan 20
  description Dept B
  ip address 1.0.20.5/24

vlan 30
  description Core 1 & 2
  ip address 1.0.30.5/24

router ospf
  no passive-interface vlan 30
```

Reference counts example

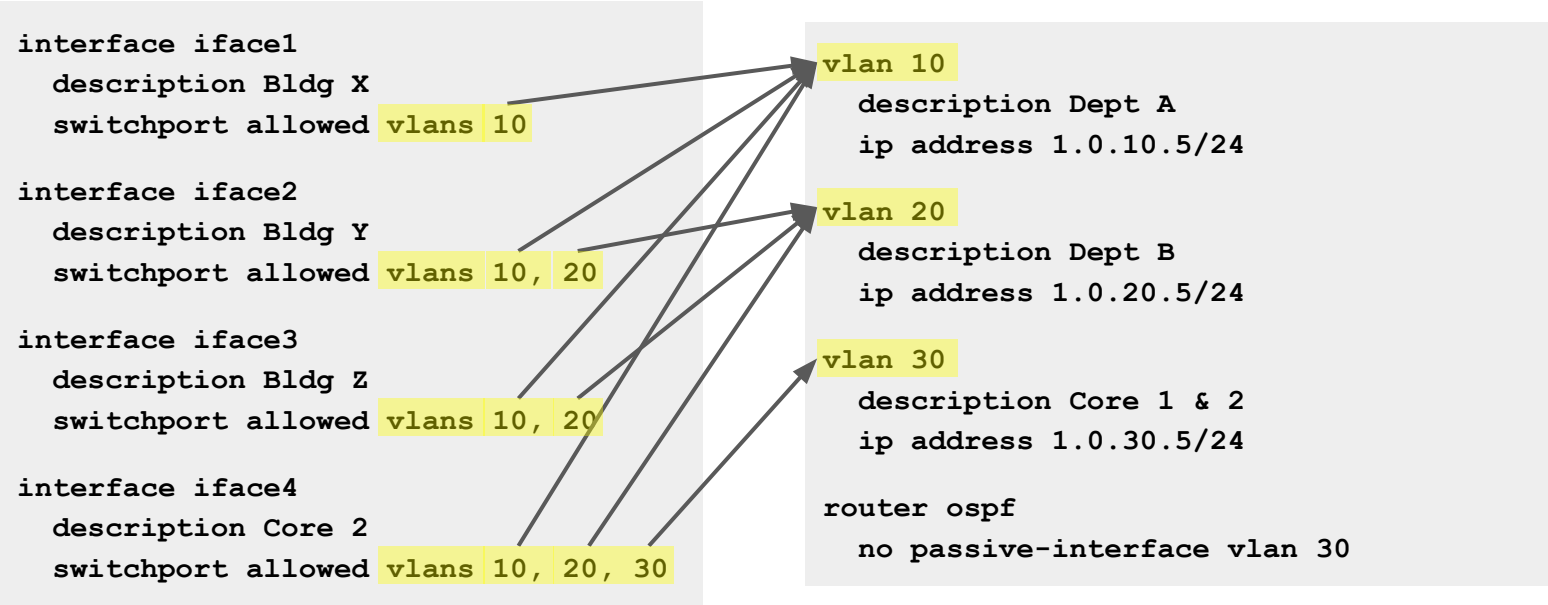
- Dedicated VLAN for each pair of core routers
- VLAN allowed on a single interface
- OSPF runs on this VLAN, but not other VLANs

```
interface iface1
  description Bldg X
  switchport allowed vlans 10

interface iface2
  description Bldg Y
  switchport allowed vlans 10, 20

interface iface3
  description Bldg Z
  switchport allowed vlans 10, 20

interface iface4
  description Core 2
  switchport allowed vlans 10, 20, 30
```

```
vlan 10
  description Dept A
  ip address 1.0.10.5/24

vlan 20
  description Dept B
  ip address 1.0.20.5/24

vlan 30
  description Core 1 & 2
  ip address 1.0.30.5/24

router ospf
  no passive-interface vlan 30
```

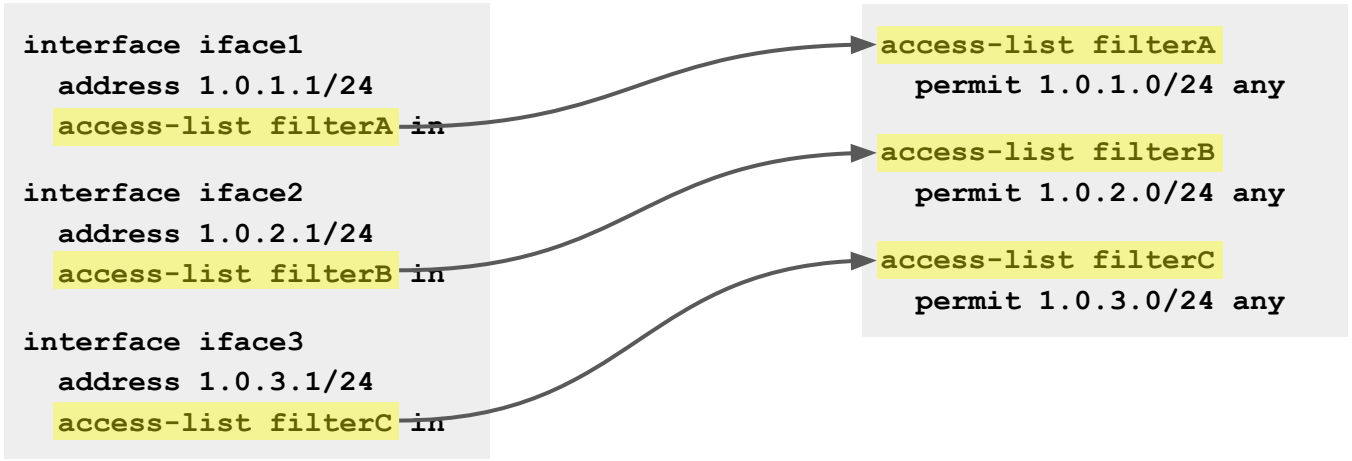
Reference counts example

- Dedicated VLAN for each pair of core routers
- VLAN allowed on a single interface
- OSPF runs on this VLAN, but not other VLANs
- Single reference to this VLAN, but many references to other VLANs



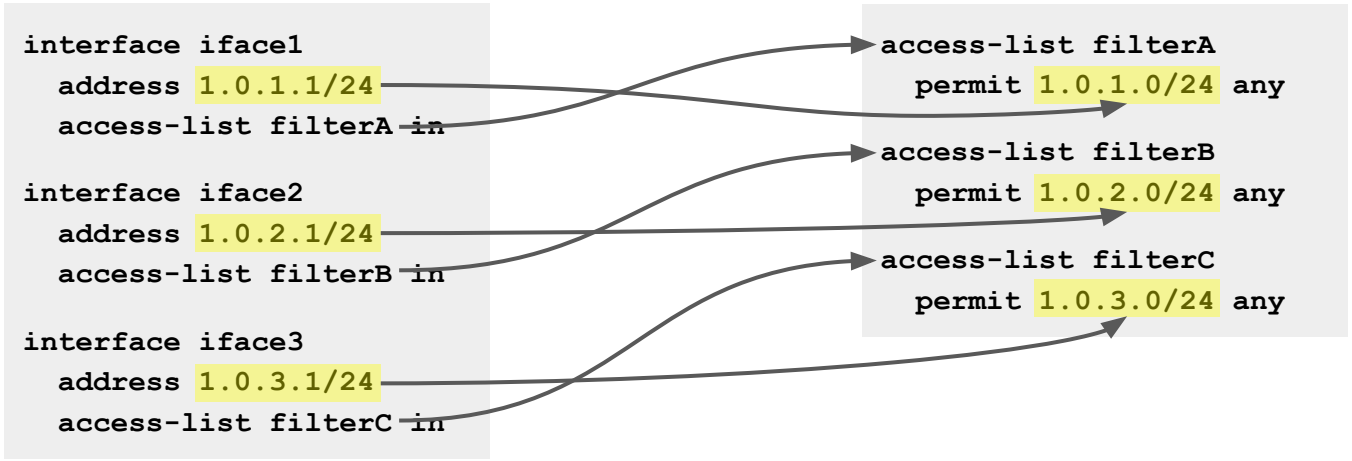
Mutual references example

- Each interface refers to an ACL



Mutual references example

- Each interface refers to an ACL
- Interface's subnet is contained in the referenced ACL



Non-operational attributes

- Ignored by router; relevant to human operators
  - Interface/VLAN descriptions

```
interface iface1
  description Bldg X Management
  …
```

  - ACL names and remarks

```
access-list ManagementAccess
  remark Allow monitoring servers
  …
```

- Keywords are often meaningful

Common keywords example

- Specific ACL contains remarks with “management”

```
vlan 10
  description Bldg X management
  access-list filterP in

vlan 20
  description Dept A labs

vlan 30
  description Bldg Y management
  access-list filterP in
```

```
access-list filterP
  remark Permit management servers
  permit 1.0.99.0/24 any
  deny any any

access-list filterQ
  permit 1.0.0.0/8 any
  deny any any
```

Common keywords example

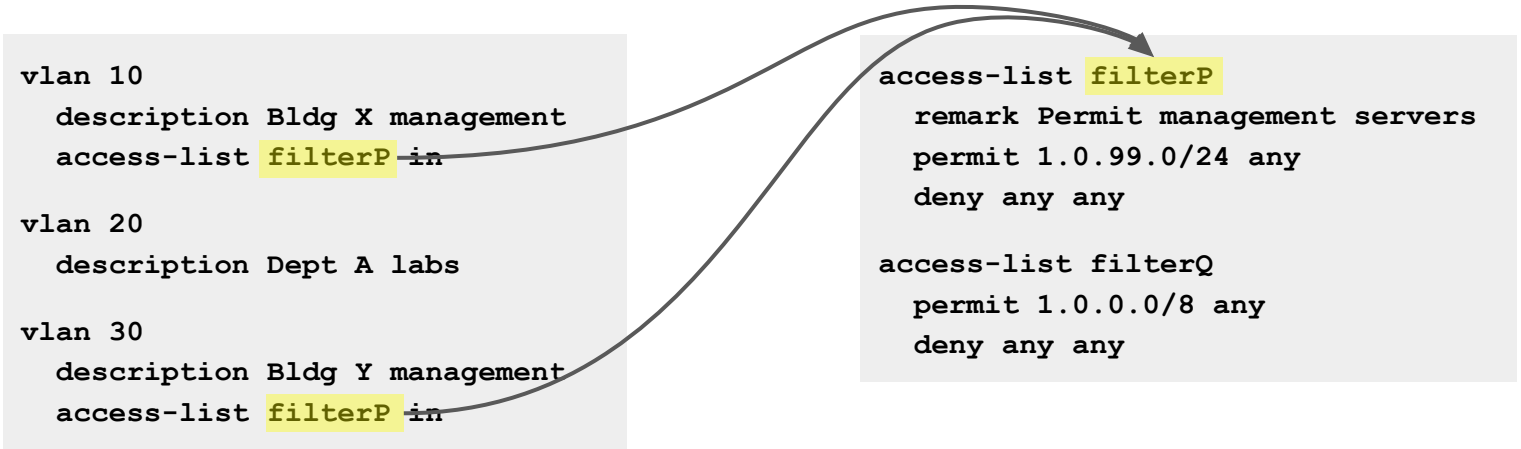
- Specific ACL contains remarks with “management”
- Descriptions of some VLANs contain “management”

```
vlan 10
  description Bldg X management
  access-list filterP in

vlan 20
  description Dept A labs

vlan 30
  description Bldg Y management
  access-list filterP in
```

```
access-list filterP
  remark Permit management servers
  permit 1.0.99.0/24 any
  deny any any

access-list filterQ
  permit 1.0.0.0/8 any
  deny any any
```

Common keywords example

- Specific ACL contains remarks with “management”
- Descriptions of some VLANs contain “management”
- ACL is applied to all of these VLAN interfaces



Our contributions

1) Classification of patterns involving multiple types of stanzas and non-operational attributes

2) Methods to automatically mine such patterns

Two methods to automatically mine patterns

Contrast Set Learning

Identify meaningful differences in attributes between separate groups of stanzas

Link Prediction

Identify stanzas with many common attributes

Contrast Set Learning

Router **attributes** ⇨

Stanza

Interface					

Contrast Set Learning

Router **attributes** 

Stanza					**Group**
Interface					Applied ACL

Contrast Set Learning

Router **attributes** 

Stanza	Used for contrast sets				Group
	Allowed VLANs		Keywords		
Interface	10	20	management	dept	Applied ACL

Challenge: determining which attributes to include

Contrast Set Learning

Router **attributes** 

Stanza	Used for contrast sets				Group
	Allowed VLANs		Keywords		
Interfaces	10	20	management	dept	Applied ACL
iface1	✓		✓	✓	filterA

Contrast Set Learning

Router **attributes** 

Stanza	Used for contrast sets				Group
	Allowed VLANs		Keywords		
Interfaces	10	20	management	dept	Applied ACL
iface1	✓		✓	✓	filterA
iface2	✓		✓	✓	filterA
iface3	✓		✓		filterA
iface4		✓		✓	filterB

Contrast Set Learning

Stanza	Used for contrast sets				Group
	Allowed VLANs		Keywords		
Interfaces	10	20	management	dept	Applied ACL
iface1	✓		✓	✓	filterA
iface2	✓		✓	✓	filterA
iface3	✓		✓		filterA
iface4		✓		✓	filterB

Contrast Set Learning

Stanza	Used for contrast sets				Group
	Allowed VLANs		Keywords		
Interfaces	10	20	management	dept	Applied ACL
iface1	✓		✓	✓	filterA
iface2	✓		✓	✓	filterA
iface3	✓		✓		filterA
iface4		✓		✓	filterB

Contrast Set Learning

Stanza	Used for contrast sets				Group
	Allowed VLANs		Keywords		
Interfaces	10	20	management	dept	Applied ACL
iface1	✓		✓	✓	filterA
iface2	✓		✓	✓	filterA
iface3	✓		✓		filterA
iface4		✓		✓	filterB

Rule: *Allowed VLAN **10** & Keyword **management** → Applied ACL **filterA***

Challenge: determining rule size

Contrast Set Learning

	Stanza	Used for contrast sets				Group
		Allowed VLANs		Keywords		
	Interfaces	10	20	management	dept	Applied ACL
TRUE +	iface1	✓		✓	✓	filterA
TRUE +	iface2	✓		✓	✓	filterA
TRUE +	iface3	✓		✓		filterA
TRUE -	iface4		✓		✓	filterB

Rule: *Allowed VLAN 10* & *Keyword management* → *Applied ACL filterA*

Perfect predictor!
(Precision: 1.0 Recall: 1.0)

Contrast Set Learning

	Stanza	Used for contrast sets				Group
		Allowed VLANs		Keywords		
	Interfaces	10	20	management	dept	Applied ACL
TRUE +	iface1	✓		✓	✓	filterA
TRUE +	iface2	✓		✓	✓	filterA
FALSE +	iface3	✓		✓		filterB
TRUE -	iface4		✓		✓	filterB

Rule: *Allowed VLAN 10* & *Keyword management* → *Applied ACL filterA*

False positives are bugs
(Precision: 0.66 Recall: 1.0)

Two methods to automatically mine patterns

Contrast Set Learning

Identify meaningful differences in attributes between separate groups of stanzas

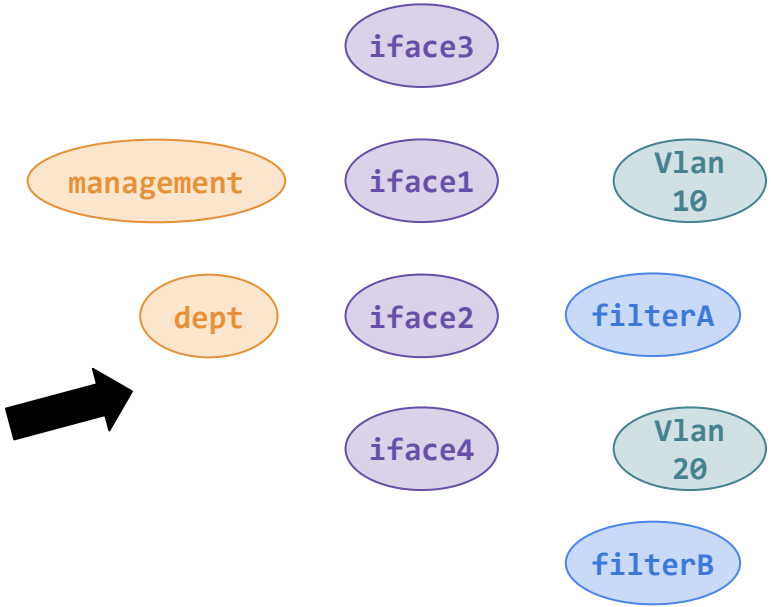
Link Prediction

Identify stanzas with many common attributes

Graph Creation

- Represent configuration as a directed graph
  - Nodes = attributes

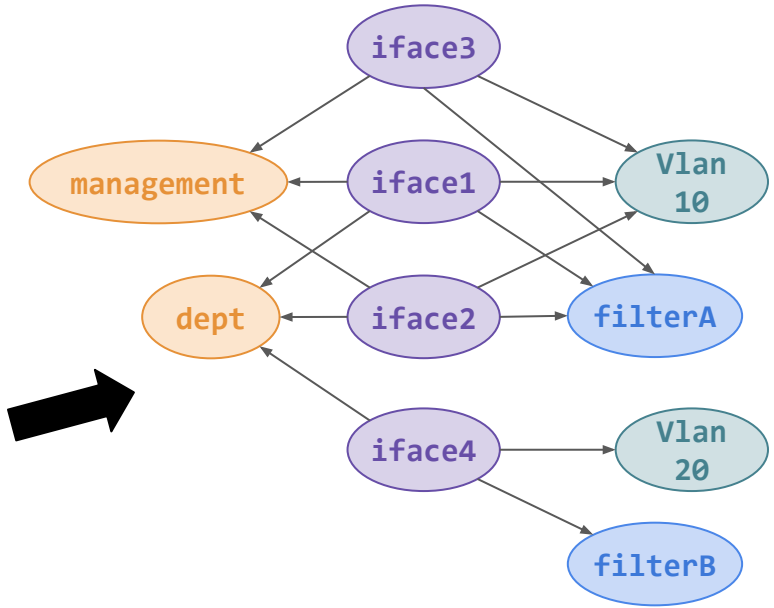
Interface	Allowed VLANs		Keywords		Applied ACL
	10	20	management	dept	
iface1	✓		✓	✓	filterA
iface2	✓		✓	✓	filterA
iface3	✓		✓		filterA
iface4		✓		✓	filterB



Graph Creation

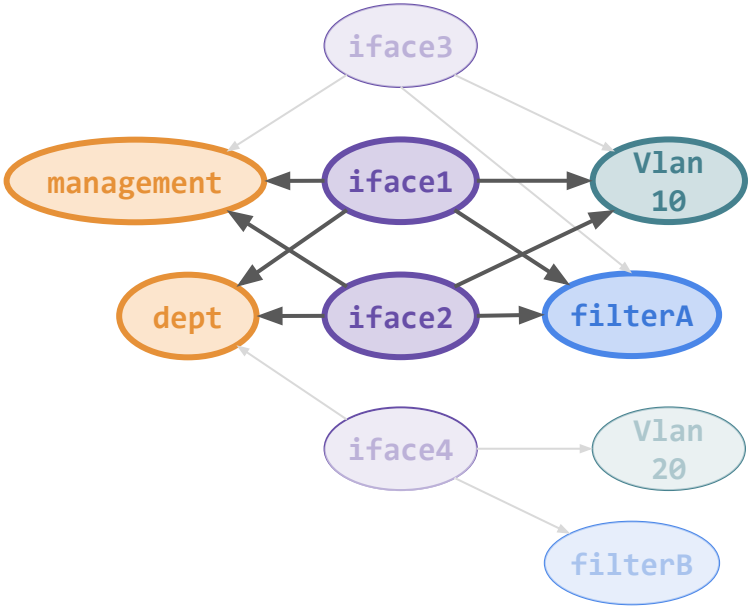
- Represent configuration as a directed graph
  - Vertices = attributes
  - Edges = references to attributes

Interface	Allowed VLANs		Keywords		Applied ACL
	10	20	management	dept	
iface1	✓		✓	✓	filterA
iface2	✓		✓	✓	filterA
iface3	✓		✓		filterA
iface4		✓		✓	filterB



Link Prediction

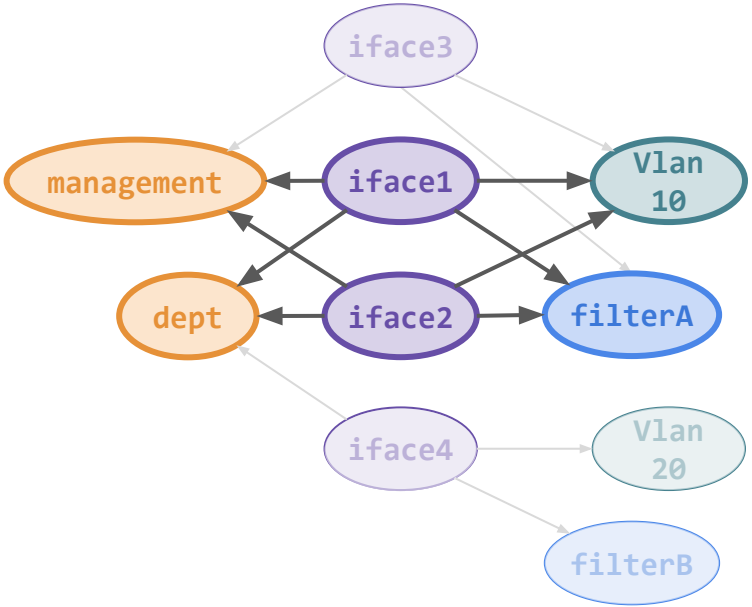
- For pairs of vertices:
  - Identify common neighbors



Identifying neighbours

- For pairs of vertices:
  - Identify common neighbors
  - Compute fraction of neighbors in common

`iface1` **vs.** `iface2`**: 100%**

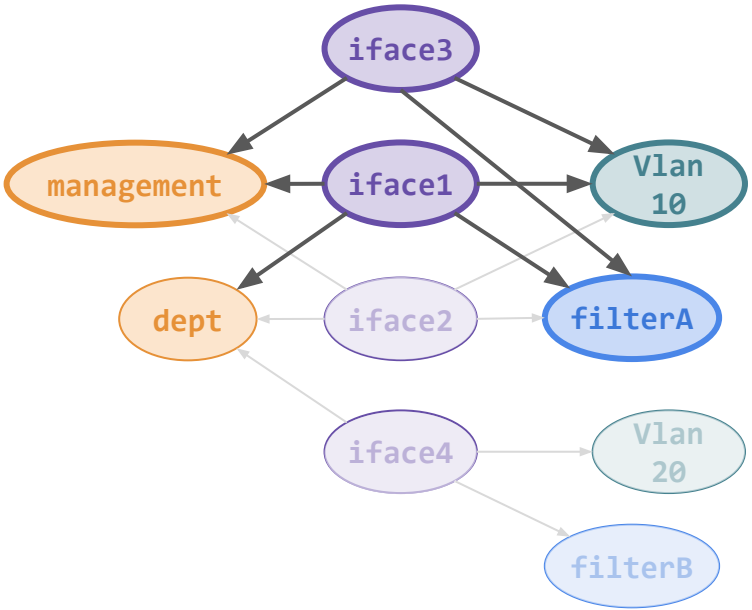


Comparing Nodes

- For pairs of vertices:
  - Identify common neighbors
  - Compute fraction of neighbors in common

    `iface1` **vs.** `iface2`**: 100%**

    `iface1` **vs.** `iface3`**: 75%**



Predicting Links

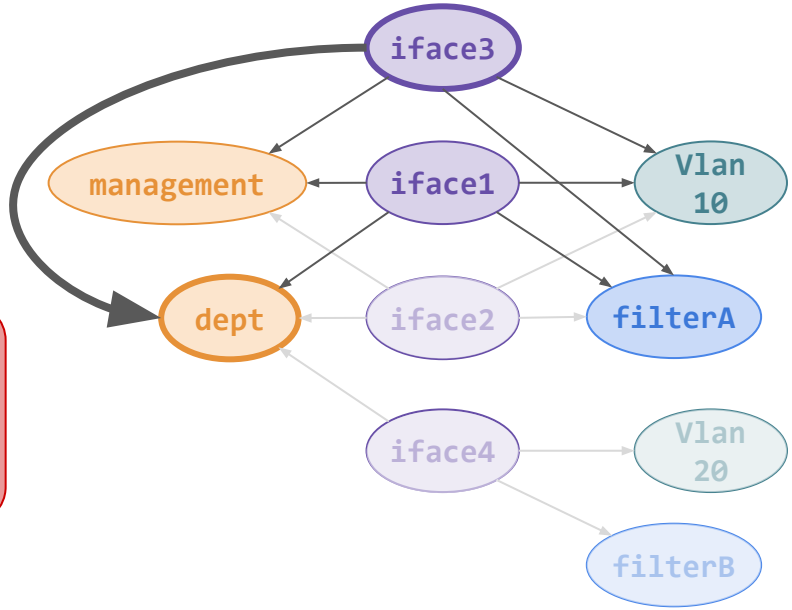
- For pairs of vertices:
  - Identify common neighbors
  - Compute fraction of neighbors in common

    `iface1` **vs.** `iface2`**: 100%**

    `iface1` **vs.** `iface3`**: 75%**
  - If similarity > threshold

    suggest additional neighbors

Challenges:
1) Choosing similarity threshold
2) Selecting which neighbors to add



Two methods to automatically mine patterns

Contrast Set Learning

Focuses on a **small set of attributes that differentiate** router stanzas

Link Prediction

Focuses on **broad similarity** between router stanzas

Conclusion

1) Classification of patterns involving multiple types of stanzas and non-operational attributes

2) Methods to automatically mine such patterns

Future Work

- Combine useful elements of contrast set learning and link prediction into a single system
- Work with operators to validate automatically mined patterns and potential errors