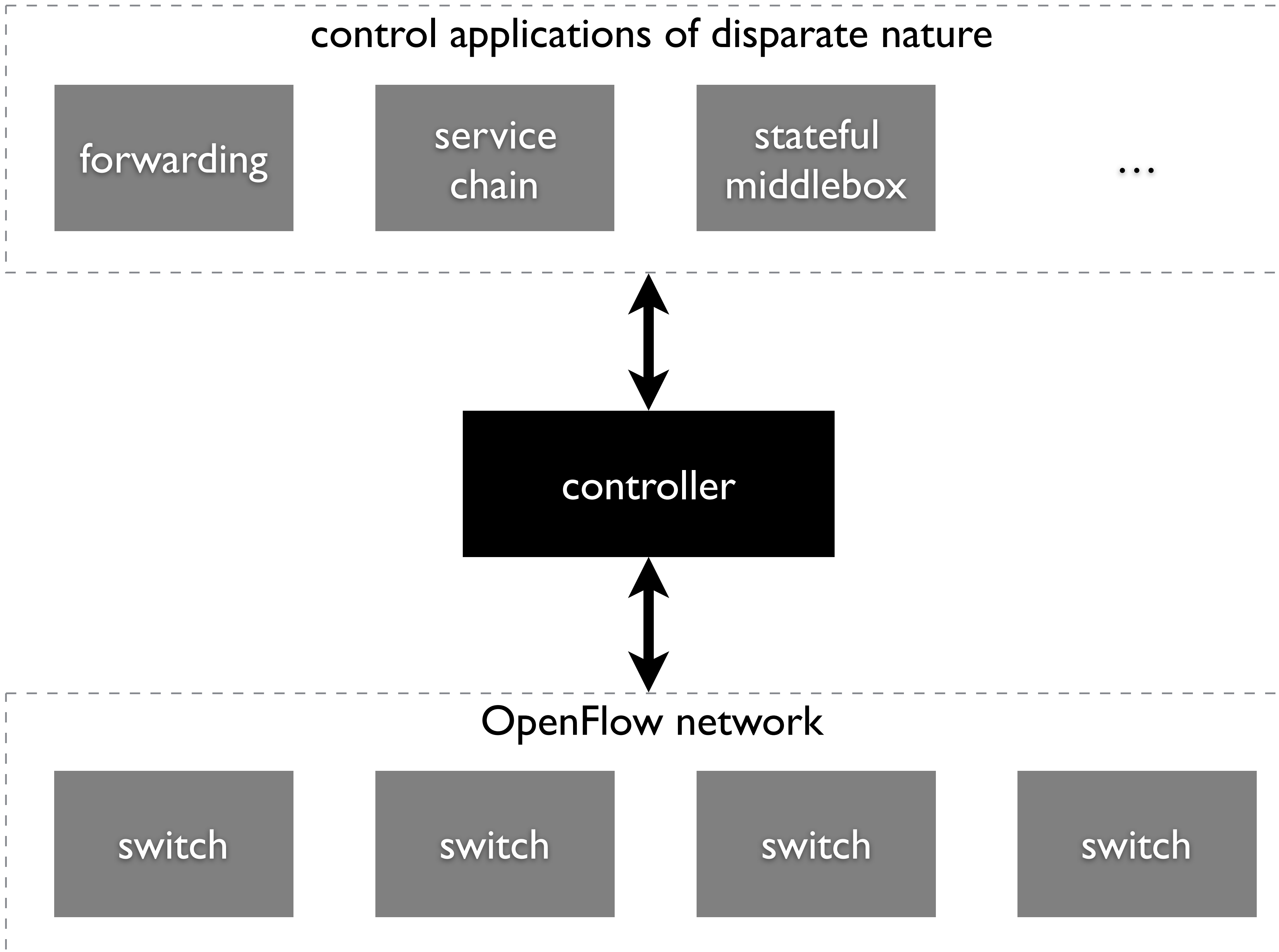
SDN abstraction and security: a database perspective

Anduo Wang* Jason Croft† Xueyuan Mei†
Matthew Caesar† Brighten Godfrey†

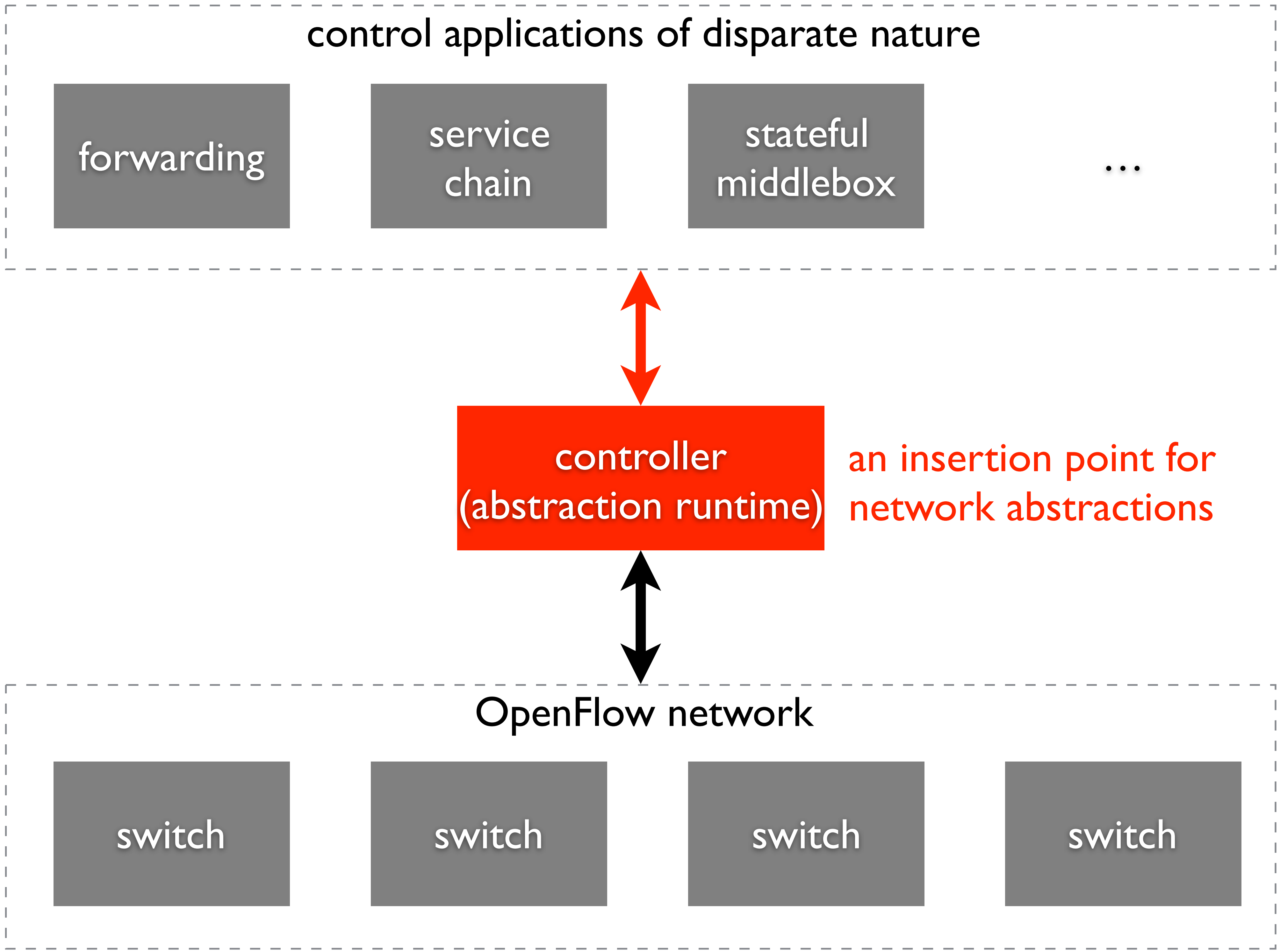
*Temple University †University of Illinois Urbana-Champaign

June 17, 2016 SoSSDN

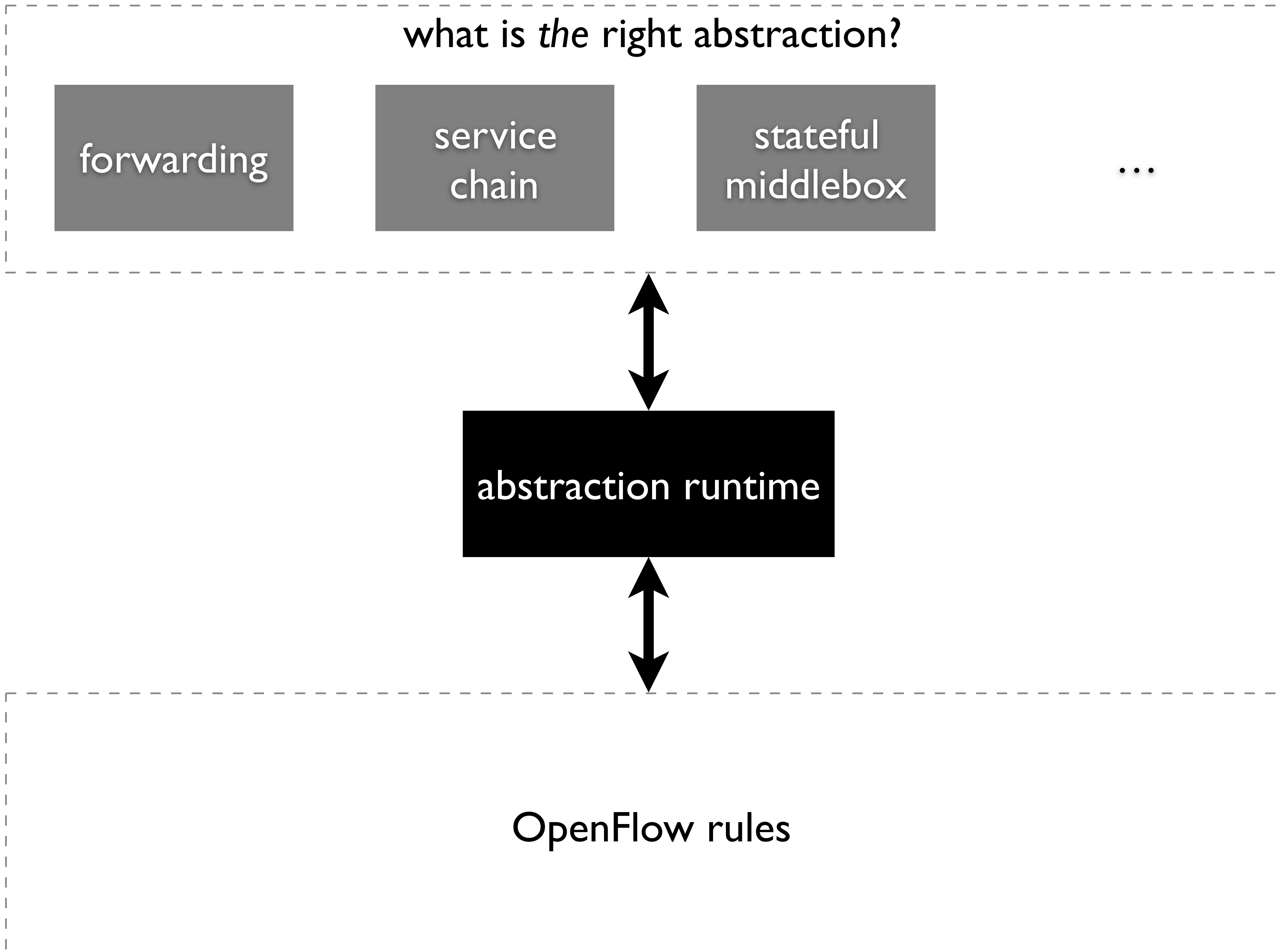
software-defined network



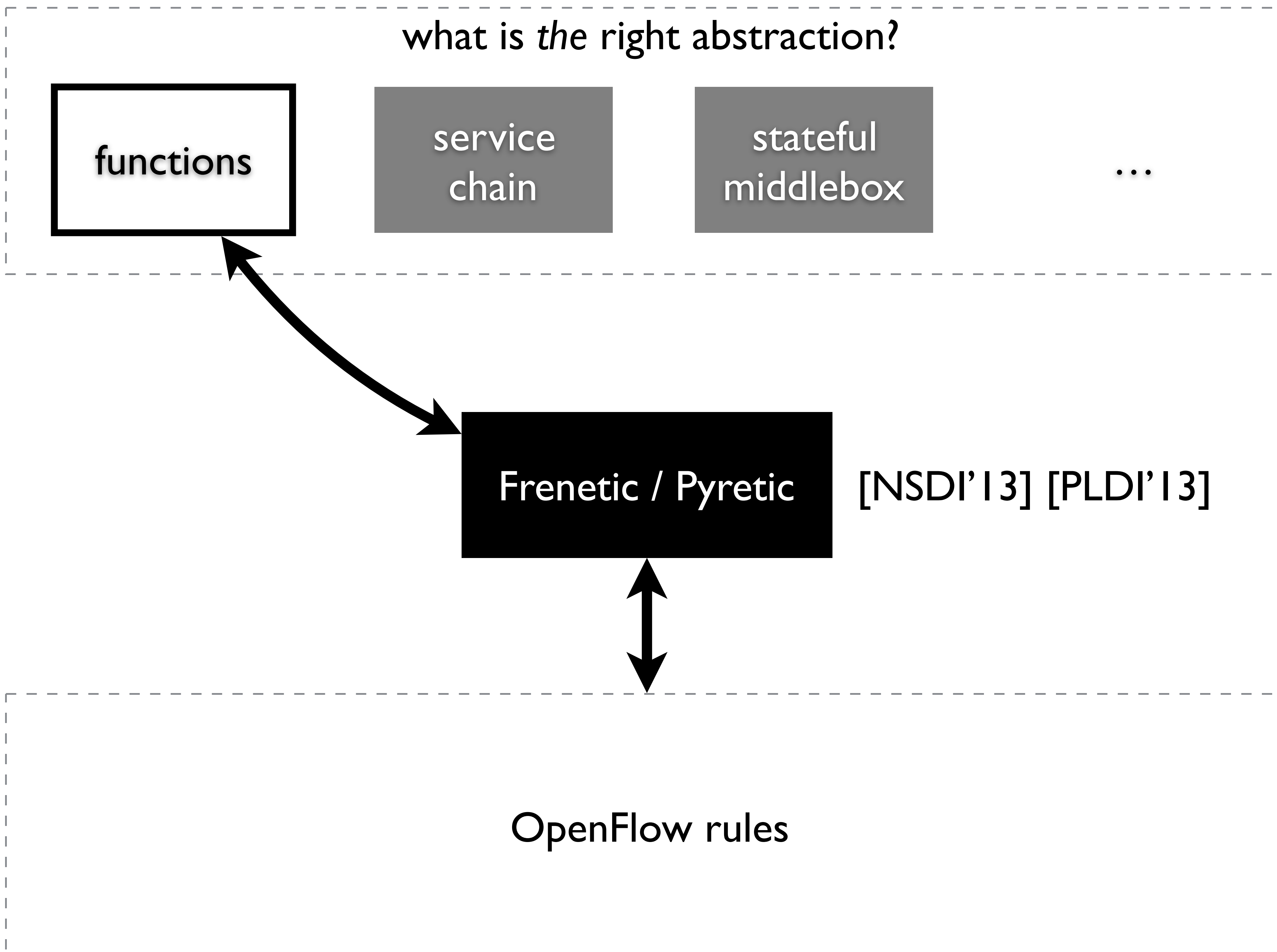
software-defined network



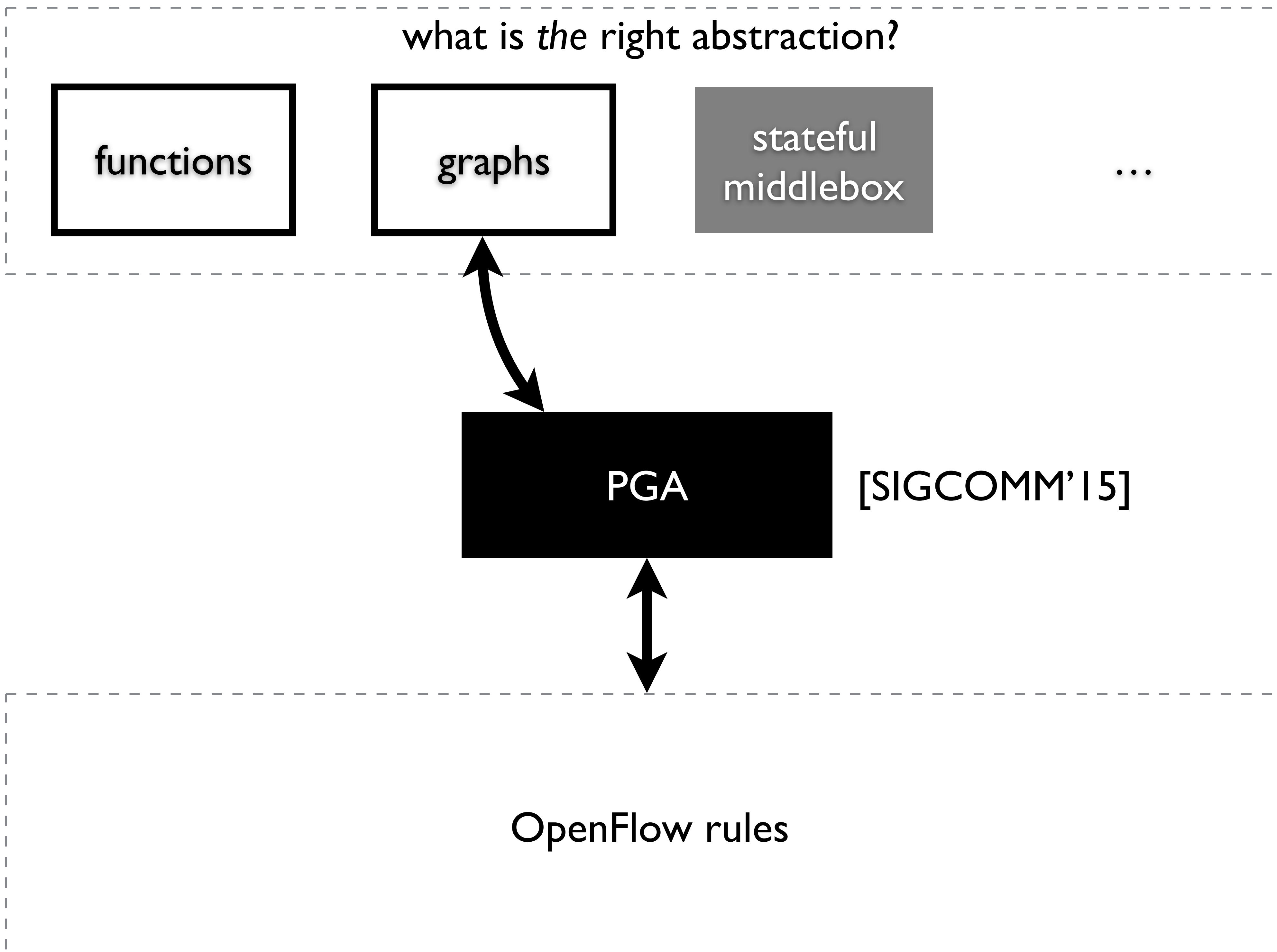
abstractions



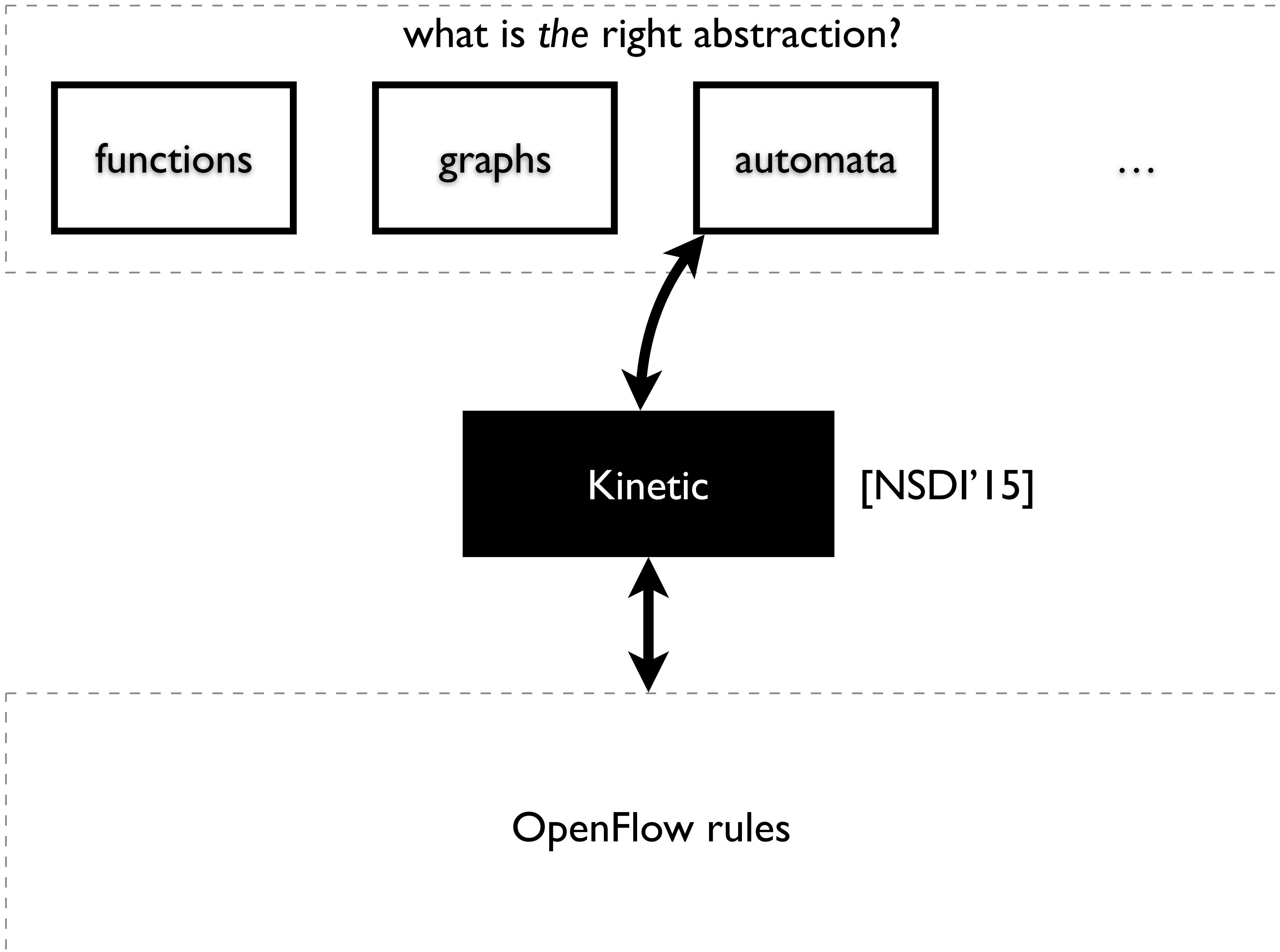
abstractions



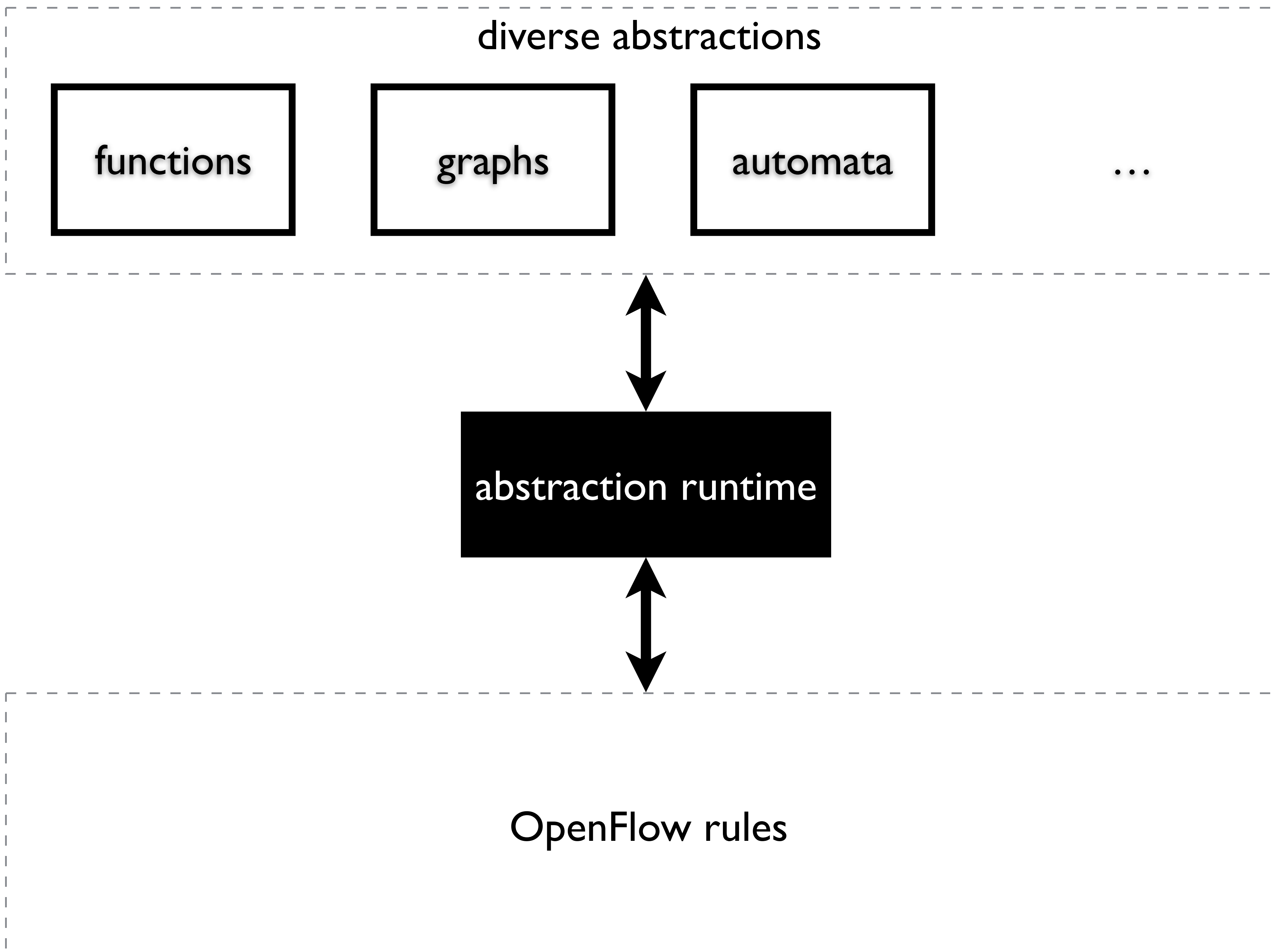
abstractions



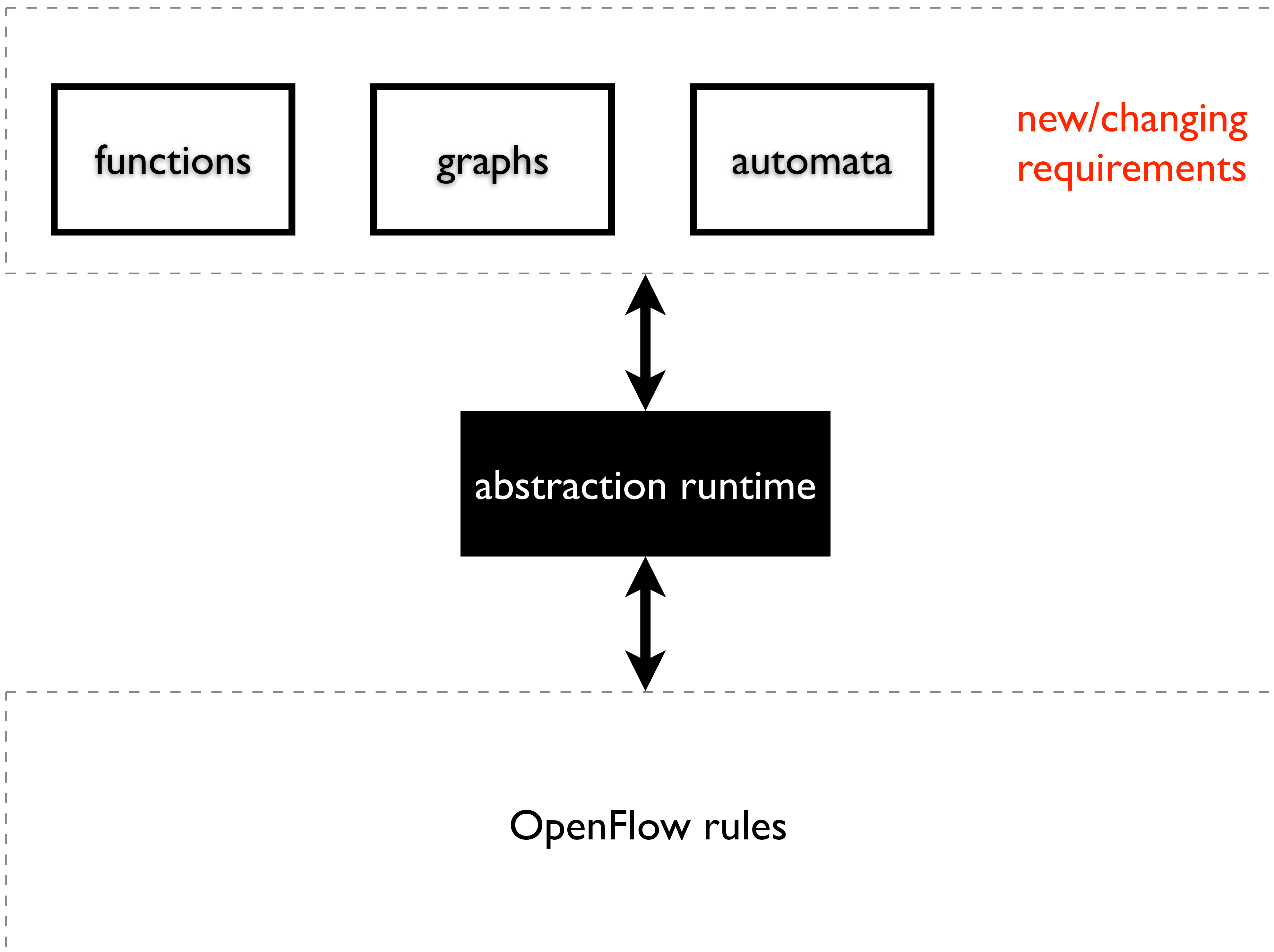
abstractions



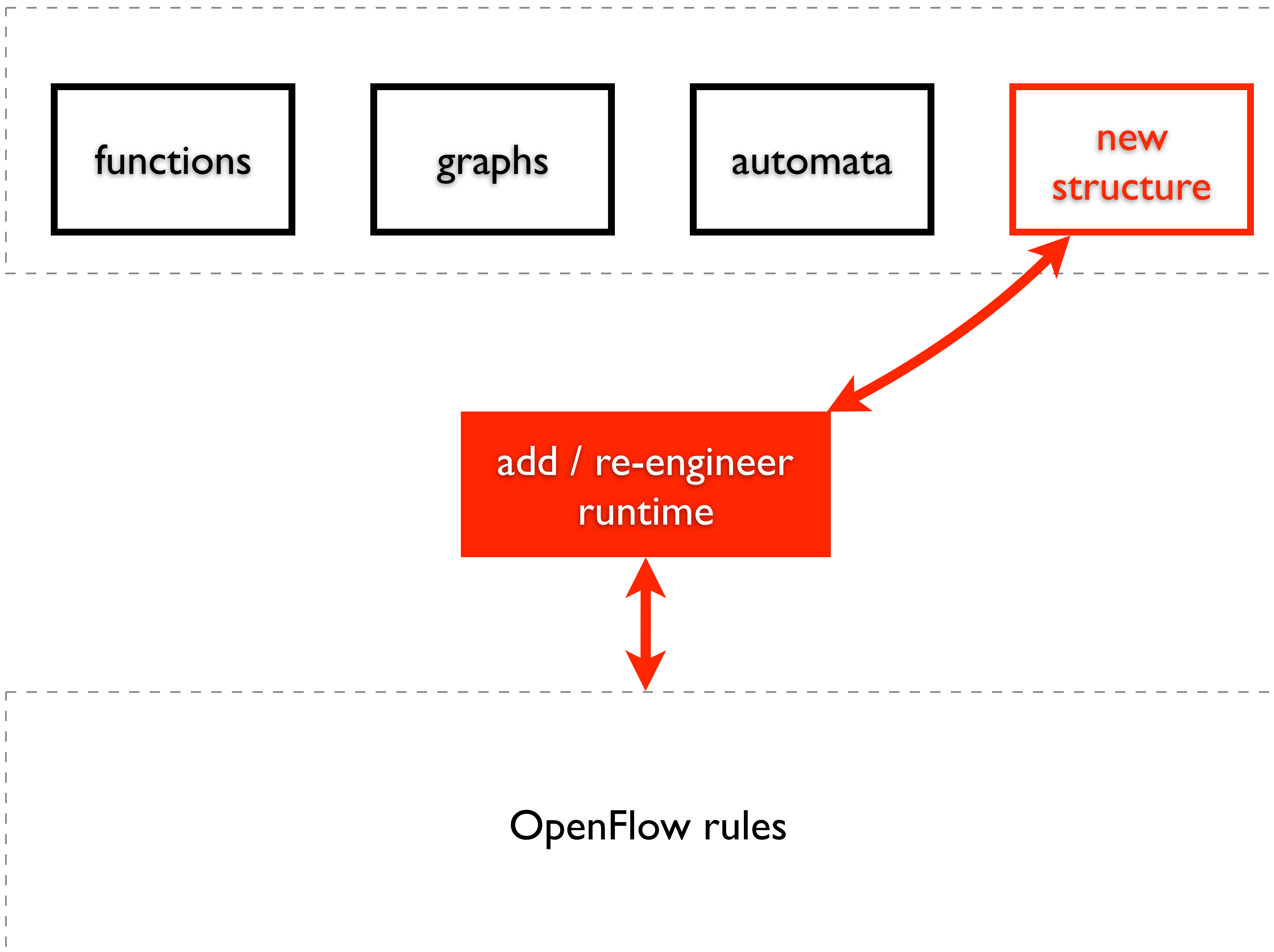
abstractions



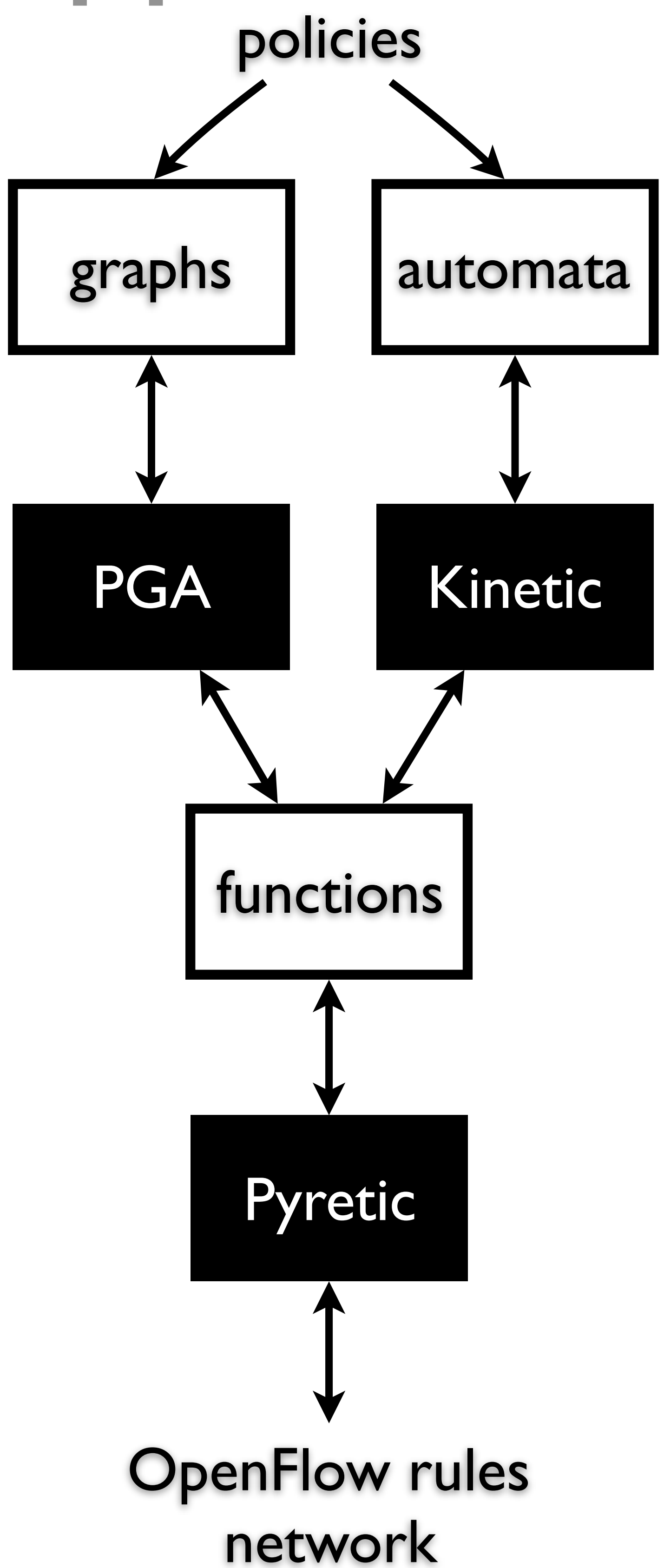
but network keeps evolving



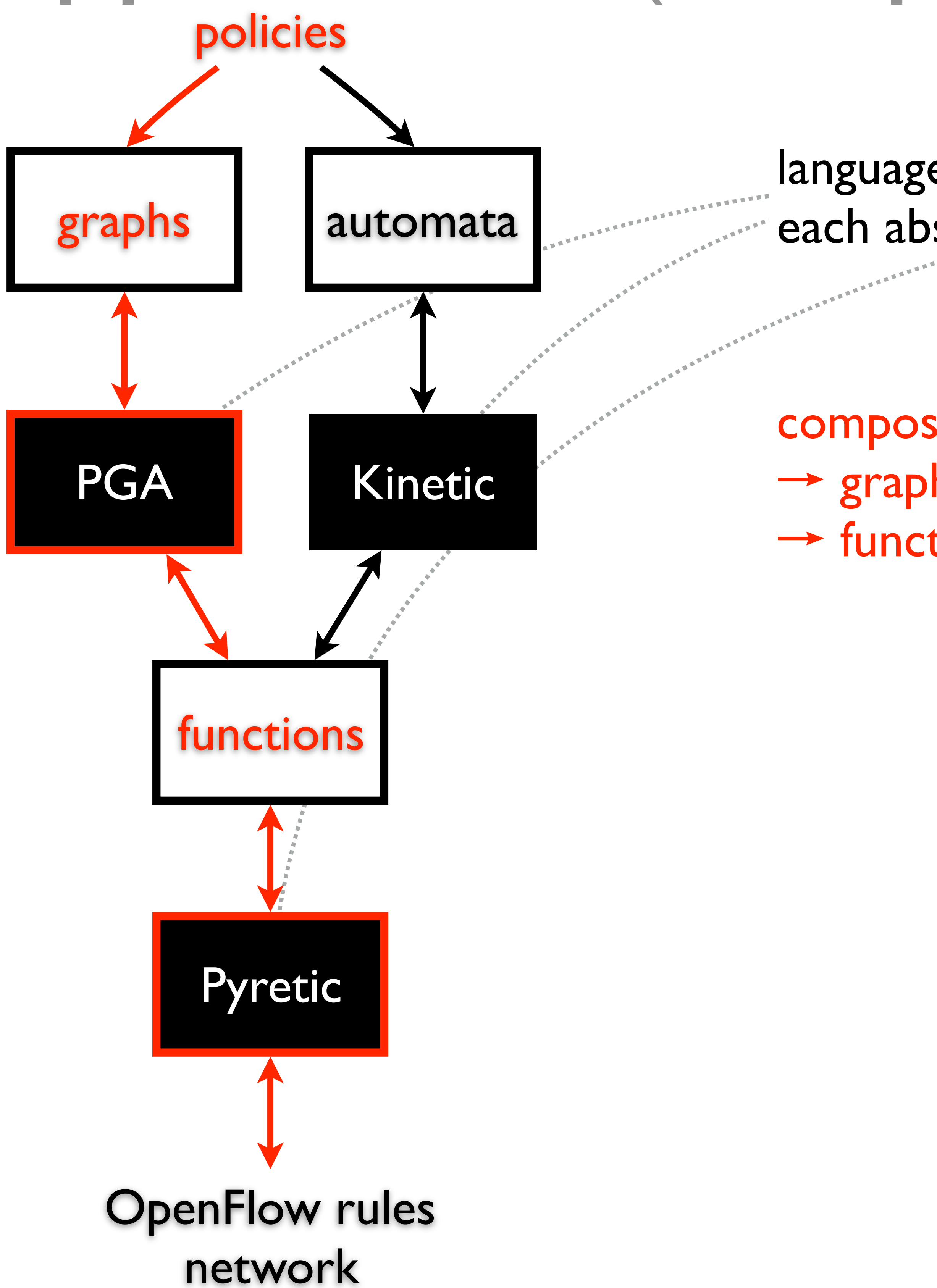
but network keeps evolving



and applications (components) interact



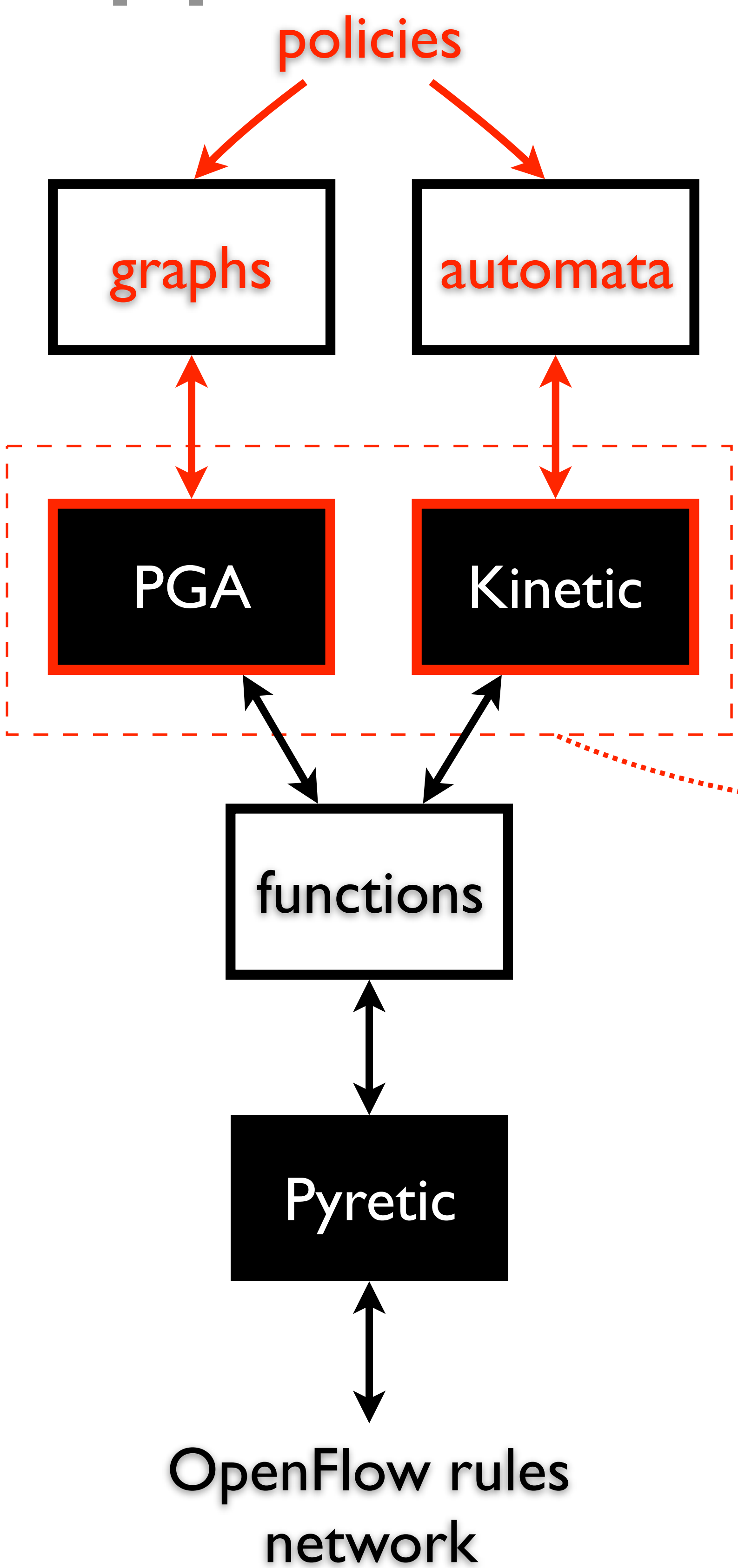
and applications (components) interact



language-level orchestration restricted to each abstraction

composing (+) policy
→ graph $+_{PGA}$ graph
→ function $+_{Pyretic}$ function

and applications (components) interact

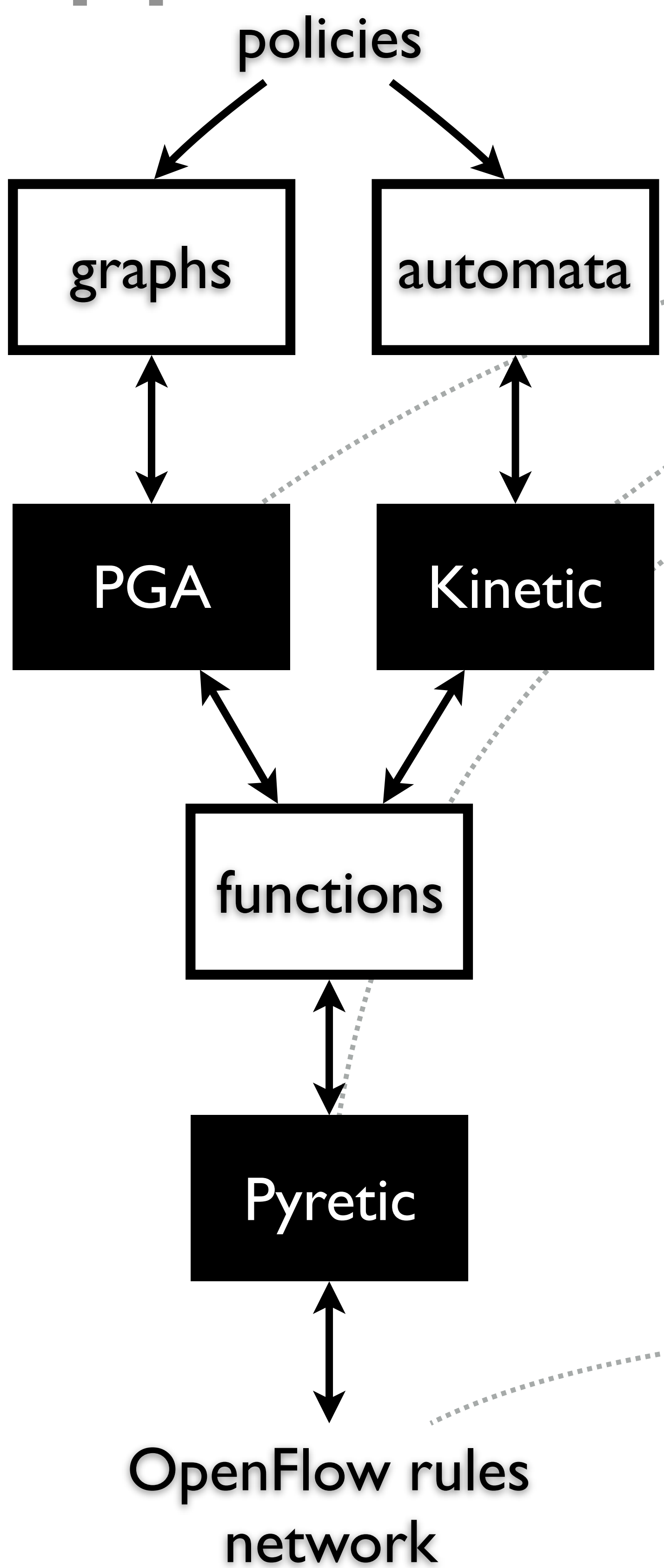


language-level orchestration restricted to each abstraction

composing (+) policy
→ graph $+_?$ automata

how to integrate the runtime?
hard-wire internals?

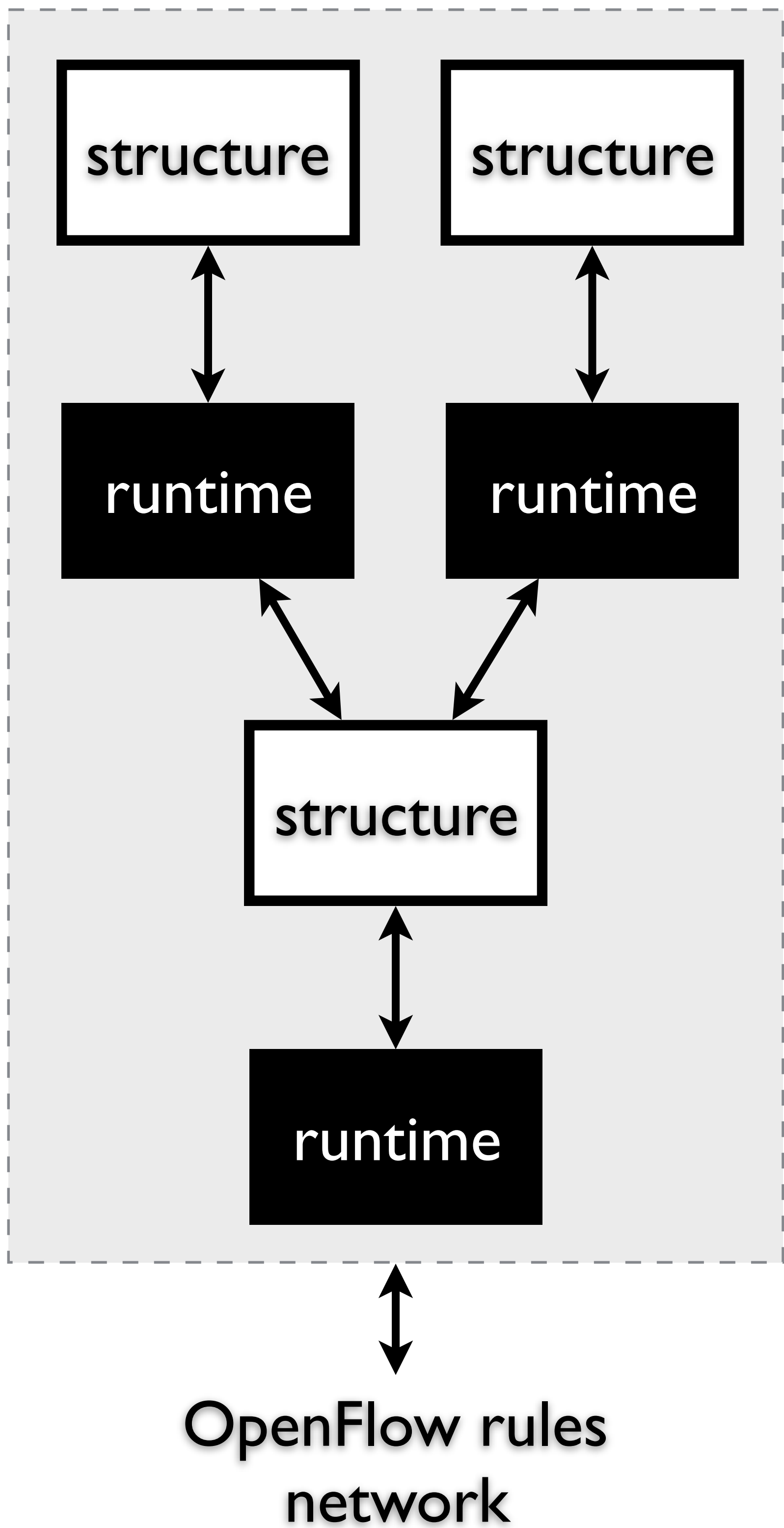
and applications (components) interact



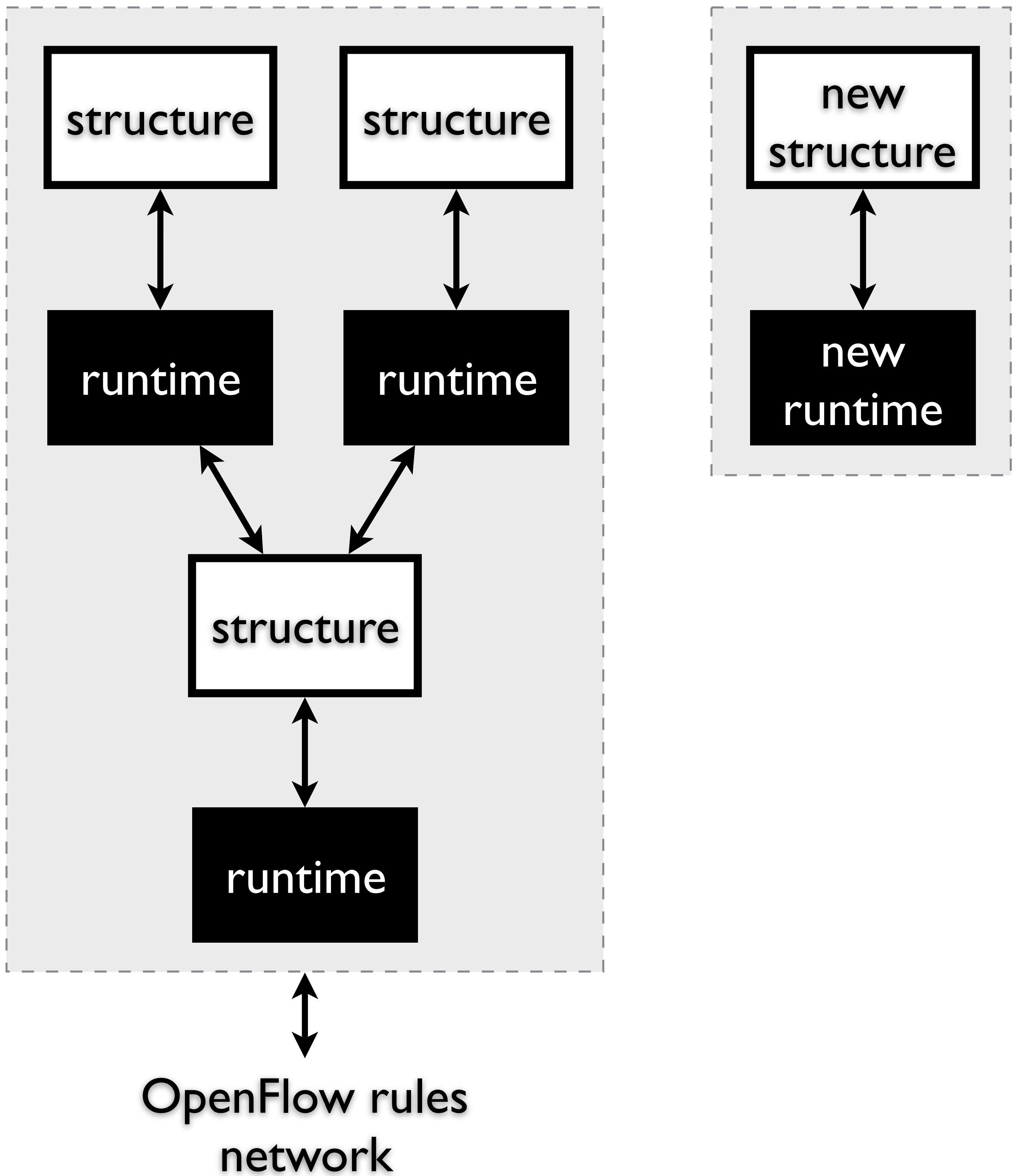
language-level orchestration restricted to each abstraction

abstraction-agonistic coordination often low-level
Co-visor [NSDI'15] statesman [SIGCOMM'14]

current state of abstraction research

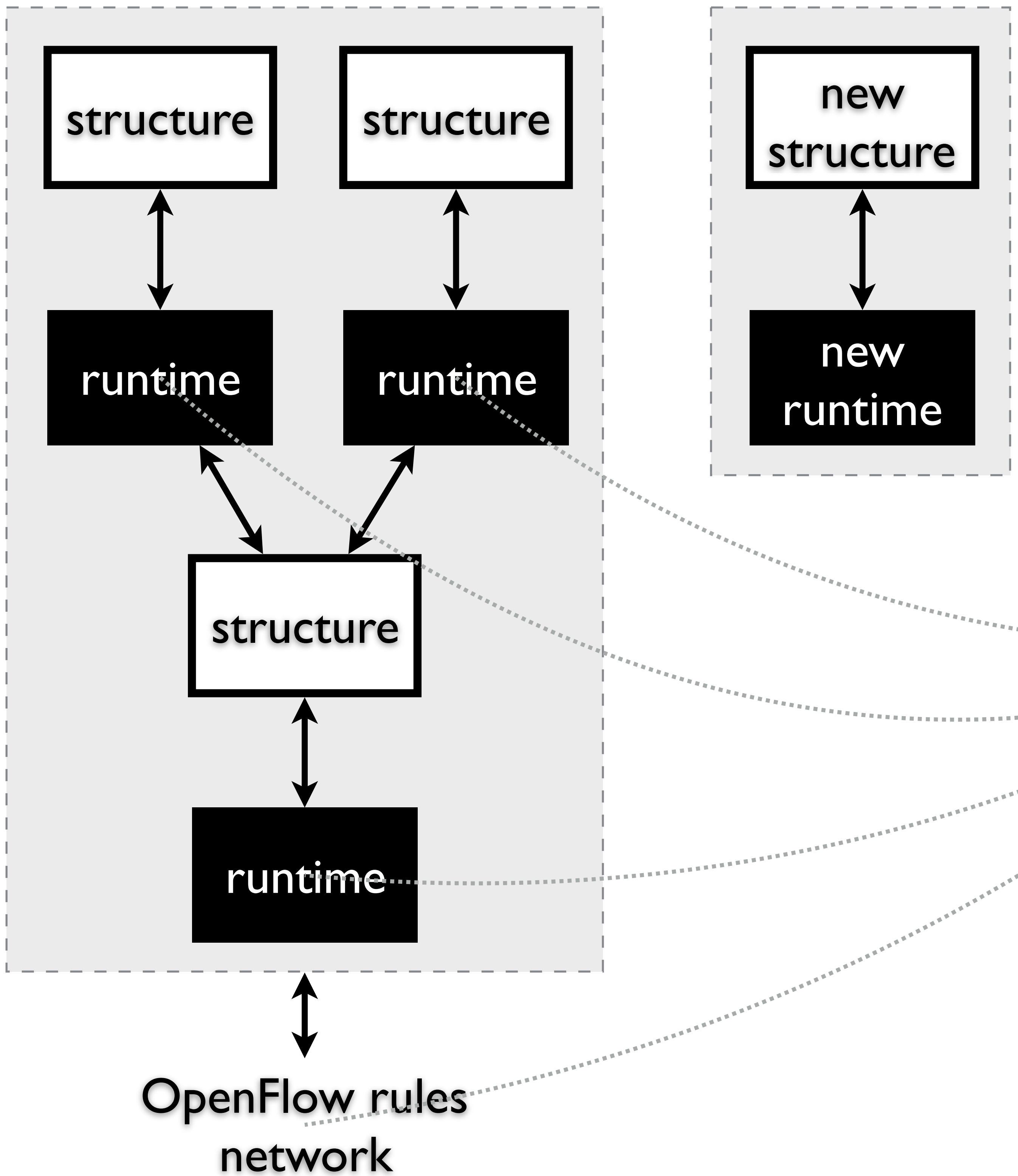


current state of abstraction research



enlarging body of abstractions

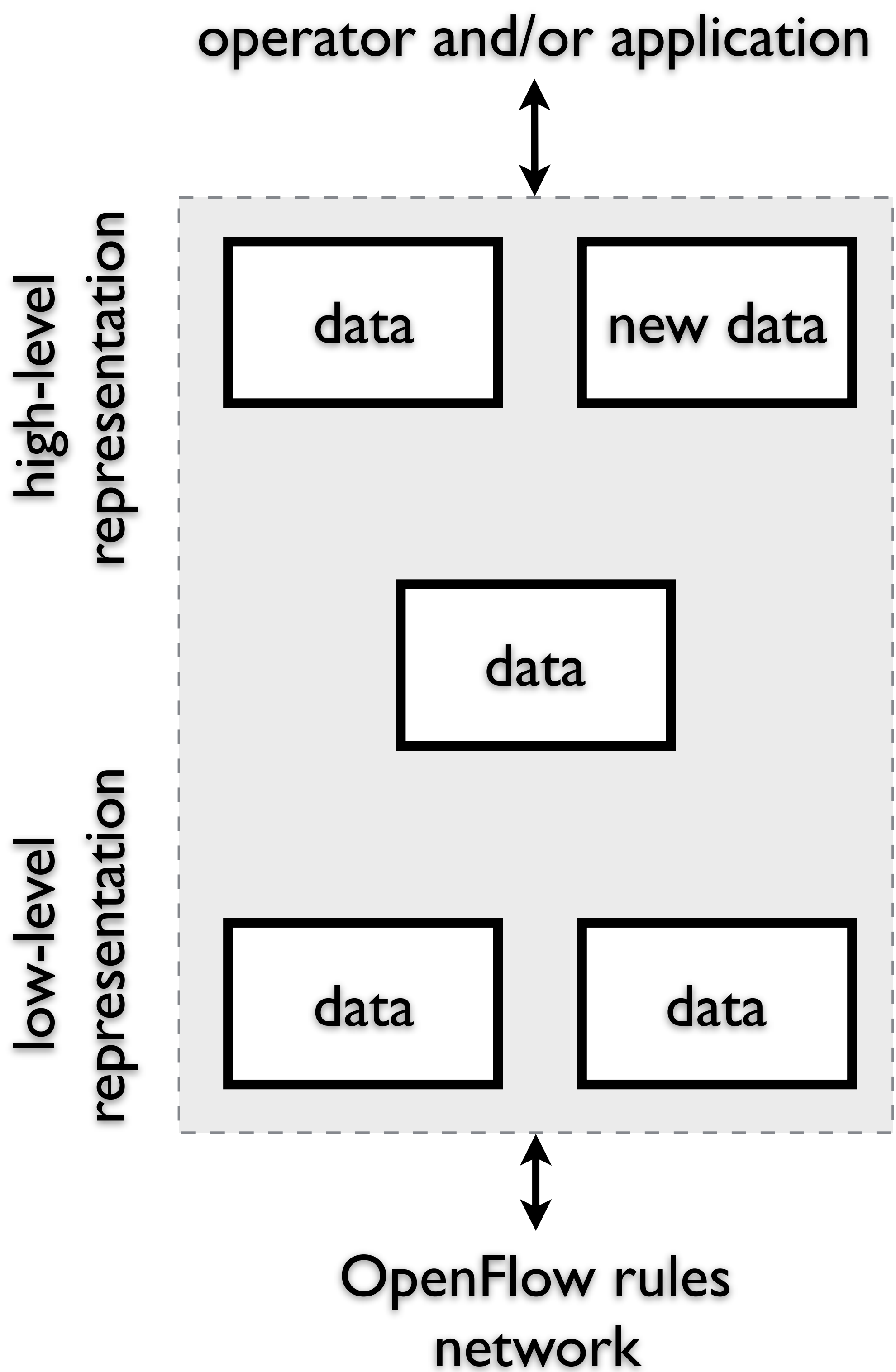
current state of abstraction research



enlarging body of abstractions

fragmented orchestration

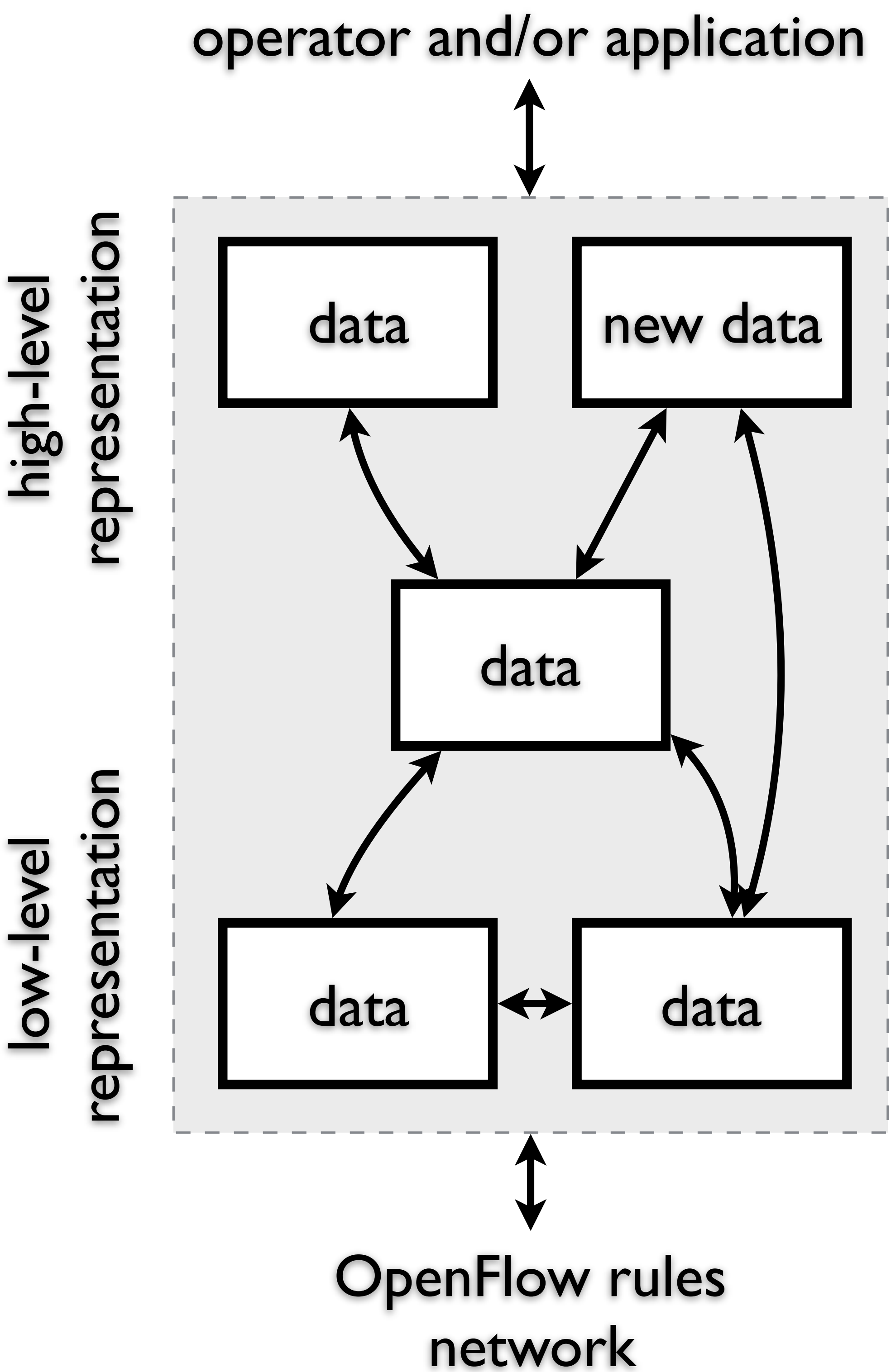
our perspective



SDN control revolves around data representation

- discard specialized, pre-compiled, fixed structures
- adopt a *plain data representation*

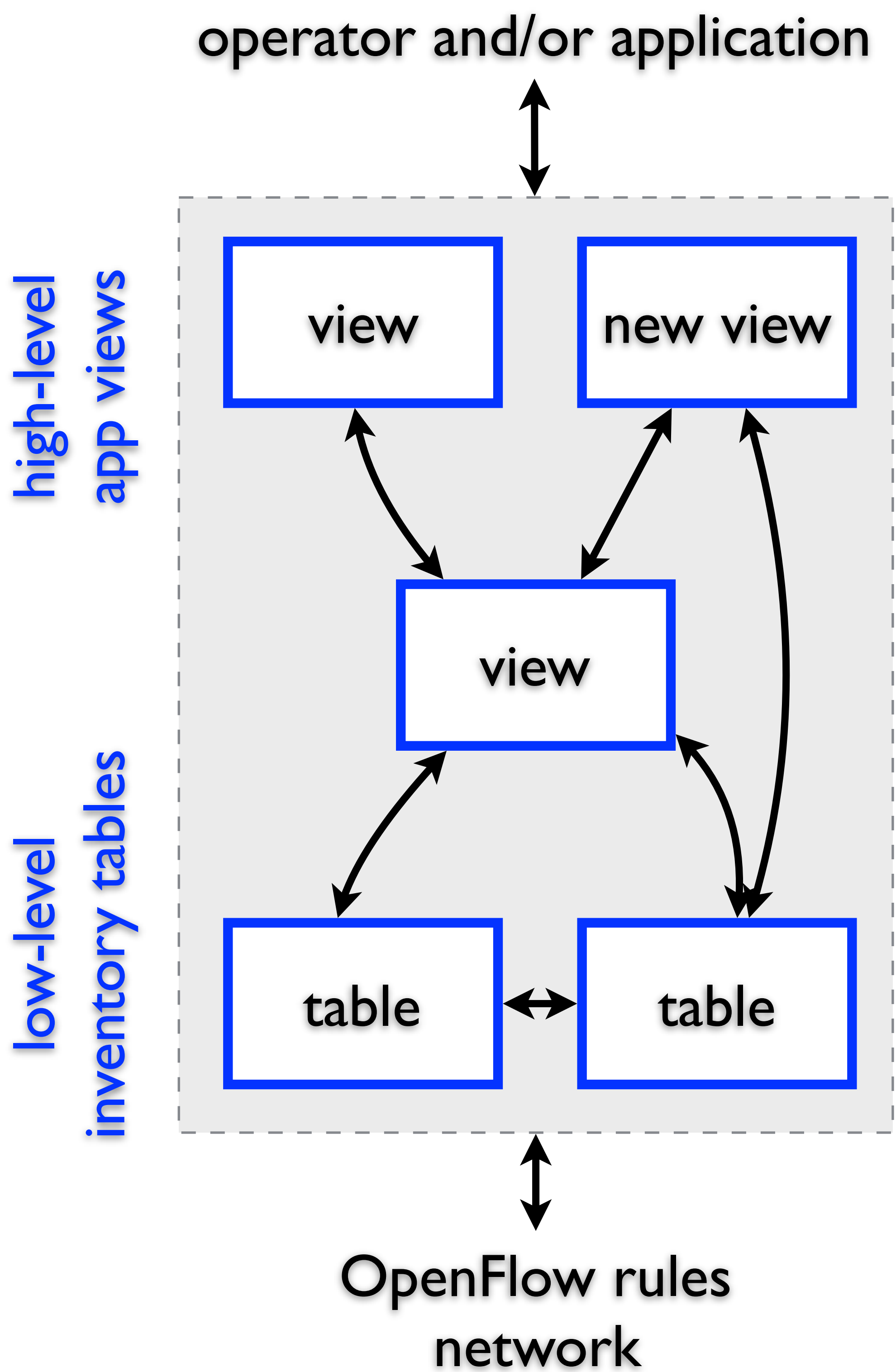
our perspective



SDN control revolves around data representation

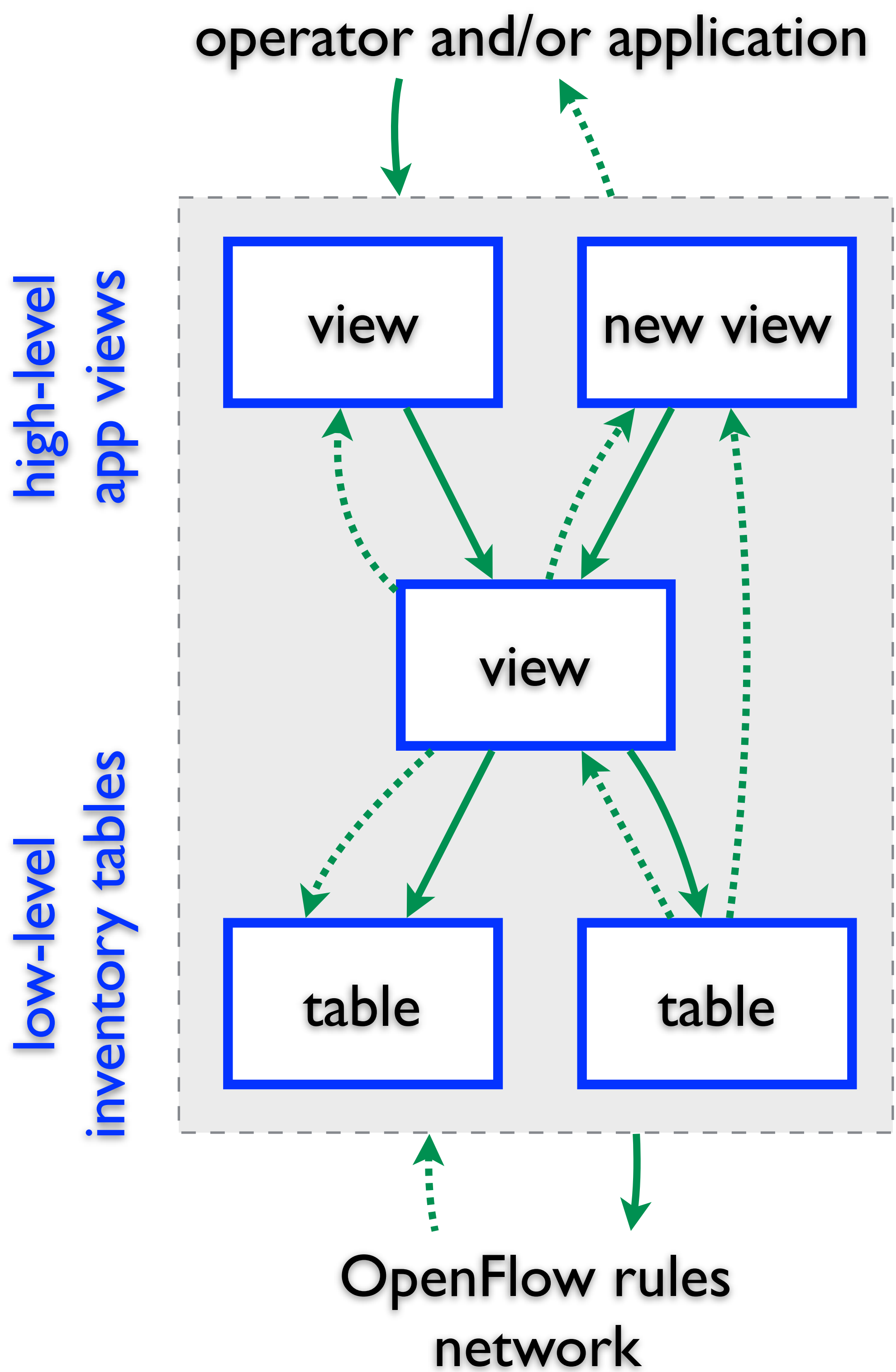
- discard specialized, pre-compiled, fixed structures
- adopt a *plain data representation*
- use a *universal data language*

a database-defined network



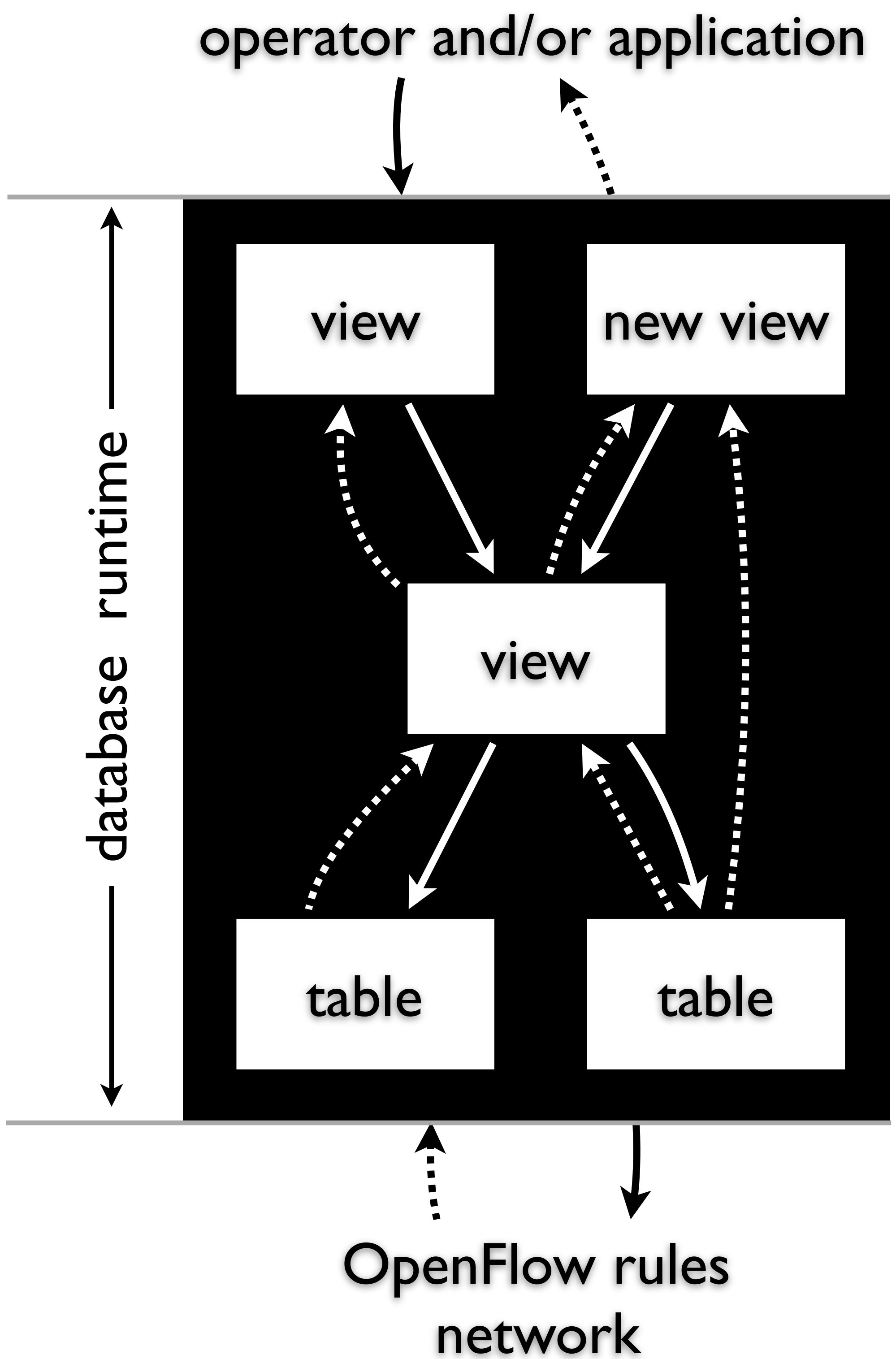
- relation — the plain data representation
  - table — stored relation
  - view — virtual relation

a database-defined network



- relation — the plain data representation
  - table — stored relation
  - view — virtual relation
- SQL — the universal data language
  - query, update, trigger, rule

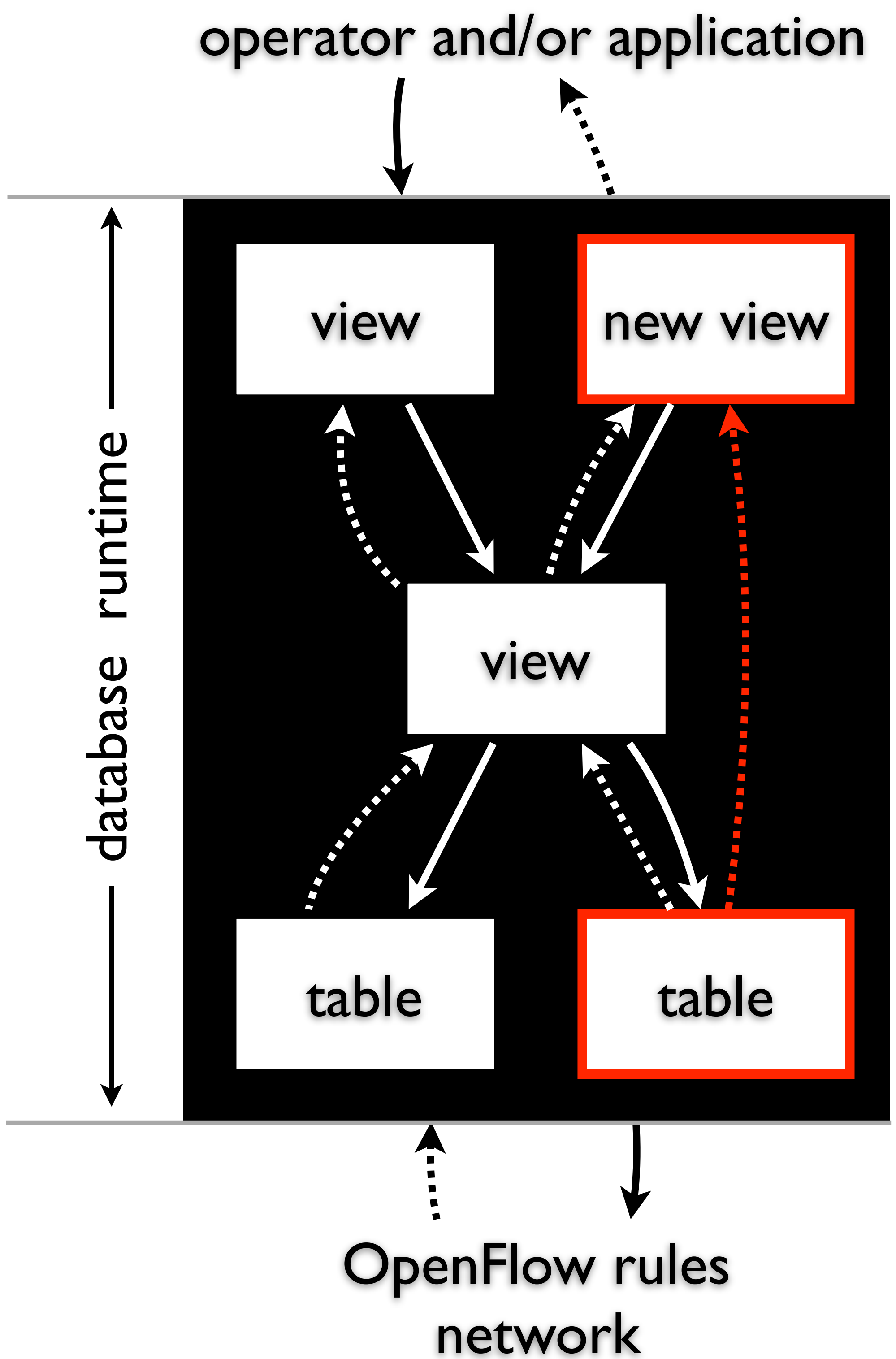
Ravel: a realization with SQL database



attractive features

- ad-hoc programmable abstraction via views
- orchestration across abstractions via view mechanism
- orchestration across applications via data mediation
- network control via SQL

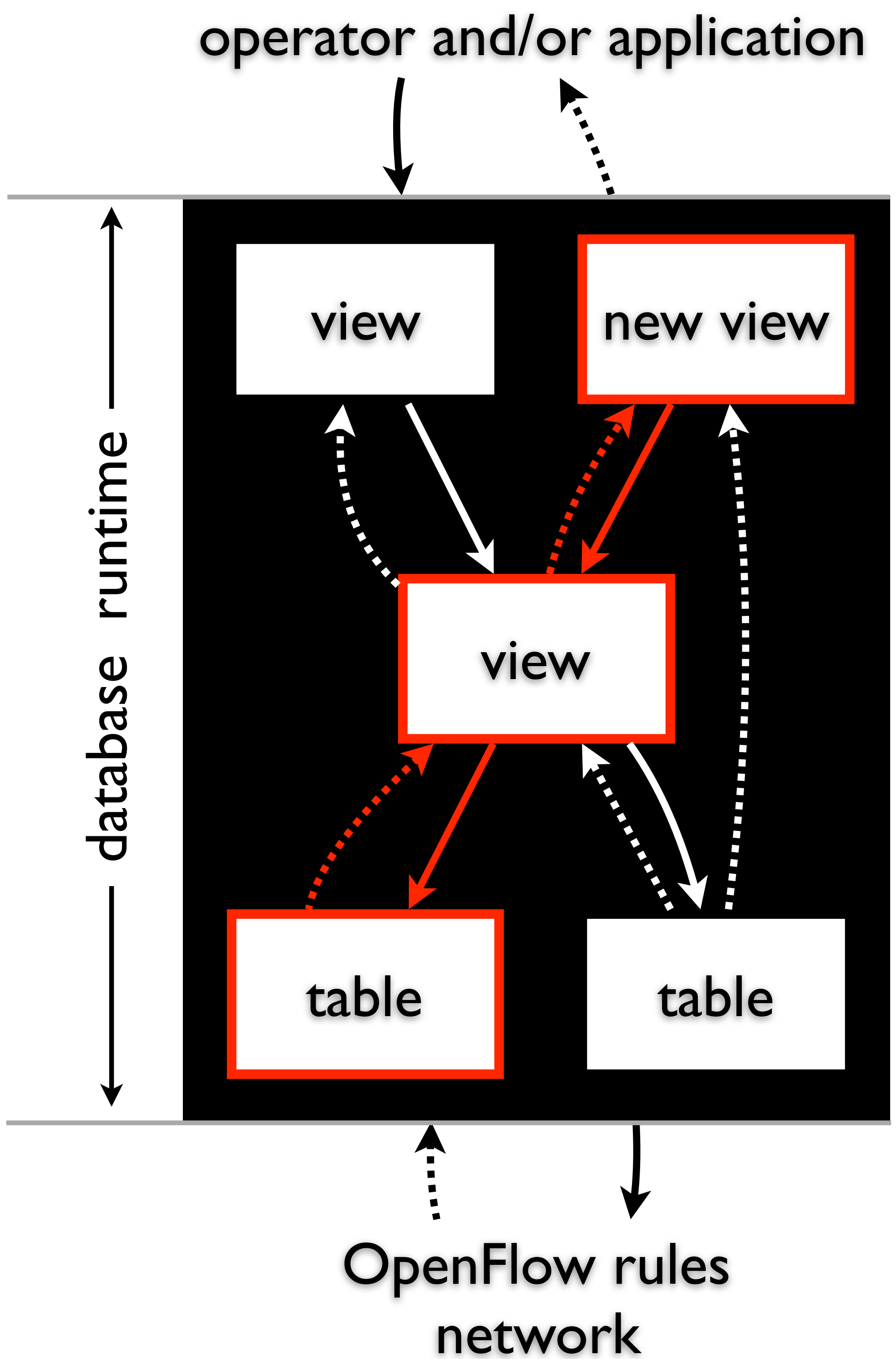
Ravel: a realization with SQL database



attractive features

- ad-hoc programmable abstraction via views
- orchestration across abstractions via view mechanism
- orchestration across applications via data mediation
- network control via SQL

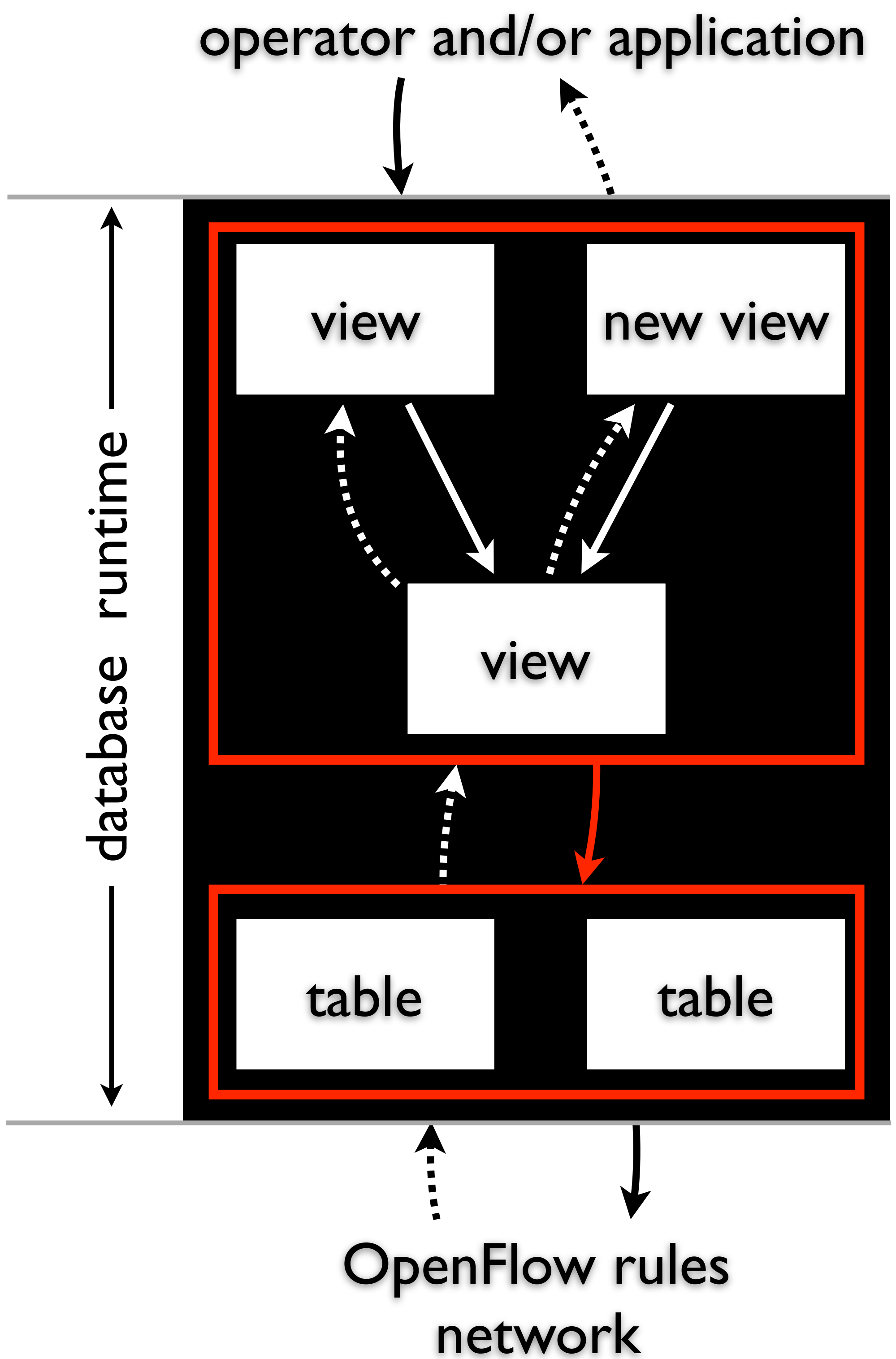
Ravel: a realization with SQL database



attractive features

- ad-hoc programmable abstraction via views
- orchestration across abstractions via view mechanism
- orchestration across applications via data mediation
- network control via SQL

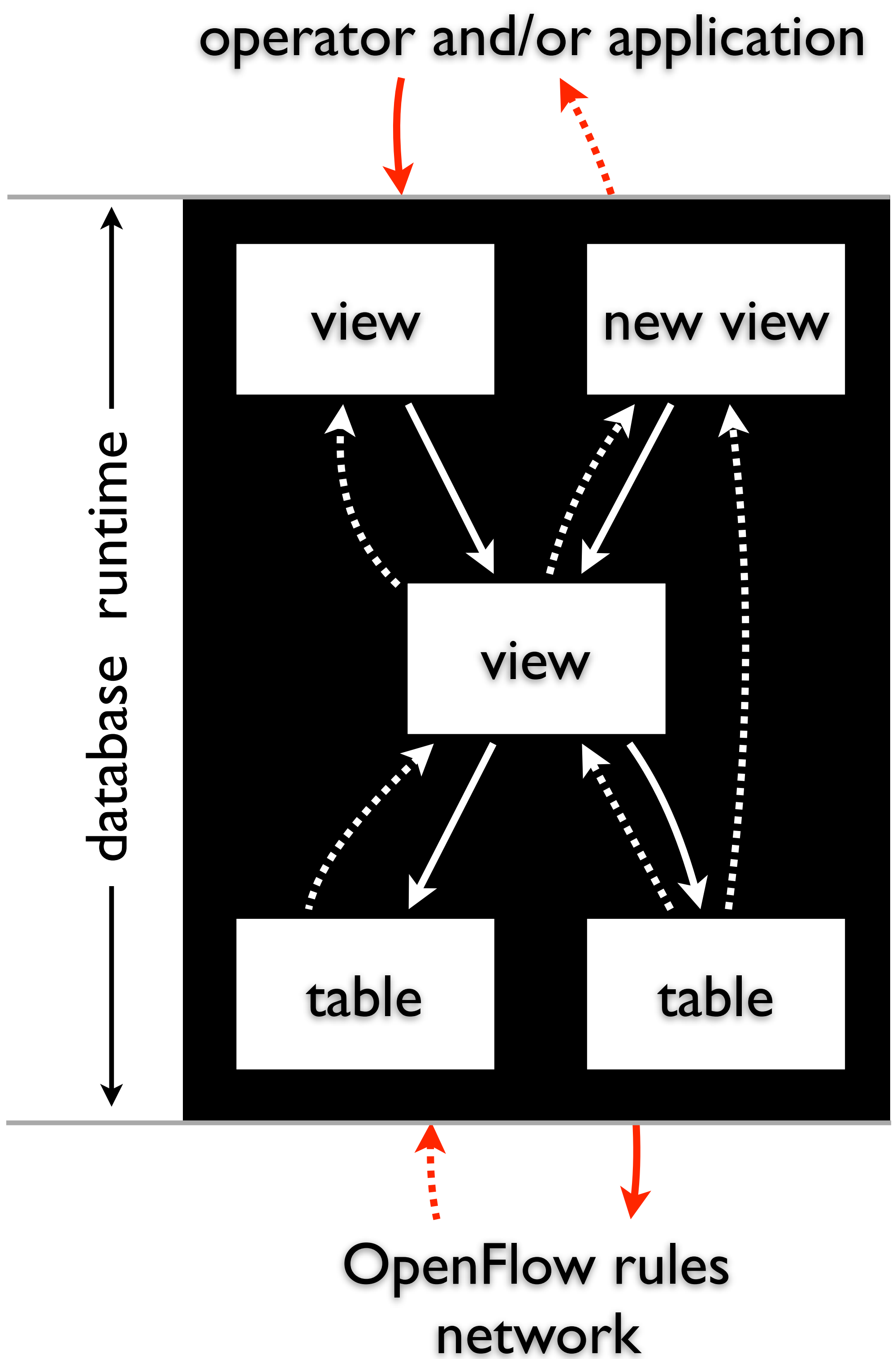
Ravel: a realization with SQL database



attractive features

- ad-hoc programmable abstraction via views
- orchestration across abstractions via view mechanism
- orchestration across applications via data mediation
- network control via SQL

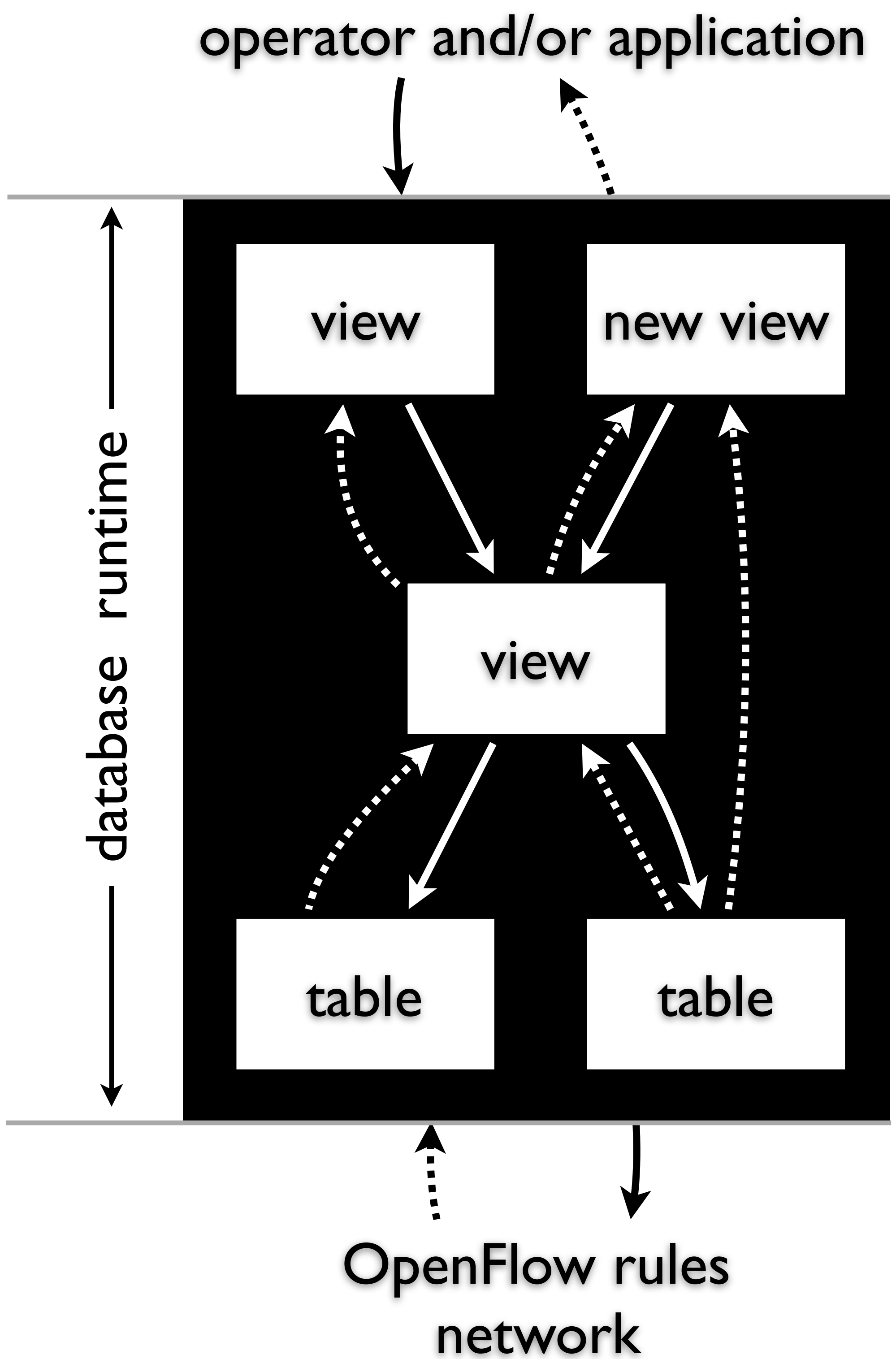
Ravel: a realization with SQL database



attractive features

- ad-hoc programmable abstraction via views
- orchestration across abstractions via view mechanism
- orchestration across applications via data mediation
- network control via SQL

Ravel: a realization with SQL database



attractive features

- abstraction
- orchestration
- SQL

abstraction: network tables

reachability matrix

fid	src	dst	vol	…
1	h_1	h_4	5	
2	h_2	h_3	9	

…

topology

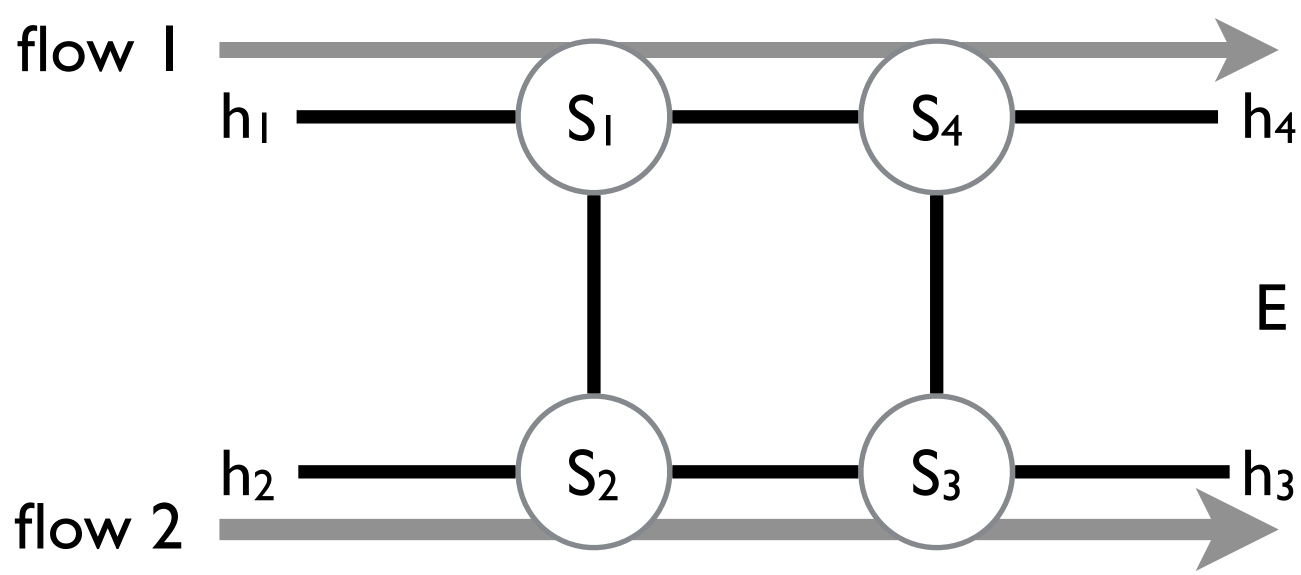
sid	nid
S_1	S_2
S_1	S_3
S_1	h_1

…

configuration

fid	sid	nid
1	S_1	S_4
1	S_4	h_4

…



abstraction: application view

firewall view: monitoring unsafe flows violating acl policy

```
CREATE VIEW acl_violation AS (
   SELECT fid
   FROM rm
   WHERE FW = 1  AND
     (src, dst) NOT IN
     (SELECT end1, end2 FROM acl
                         WHERE allow = 1)
);
```

```
CREATE TABLE acl (
  end1 integer, end2 integer, allow integer
);
```

firewall control: repairing violation

```
CREATE RULE acl_repair AS
   ON DELETE TO acl_violation
   DO INSTEAD
         DELETE FROM rm WHERE fid = OLD.fid;
```

abstraction: application view

firewall view: monitoring unsafe flows violating acl policy

```
CREATE VIEW acl_violation AS (
   SELECT fid
   FROM rm
   WHERE FW = 1  AND
     (src, dst) NOT IN
     (SELECT end1, end2 FROM acl
                        WHERE allow = 1)
);
```

```
CREATE TABLE acl (
  end1 integer, end2 integer, allow integer
);
```

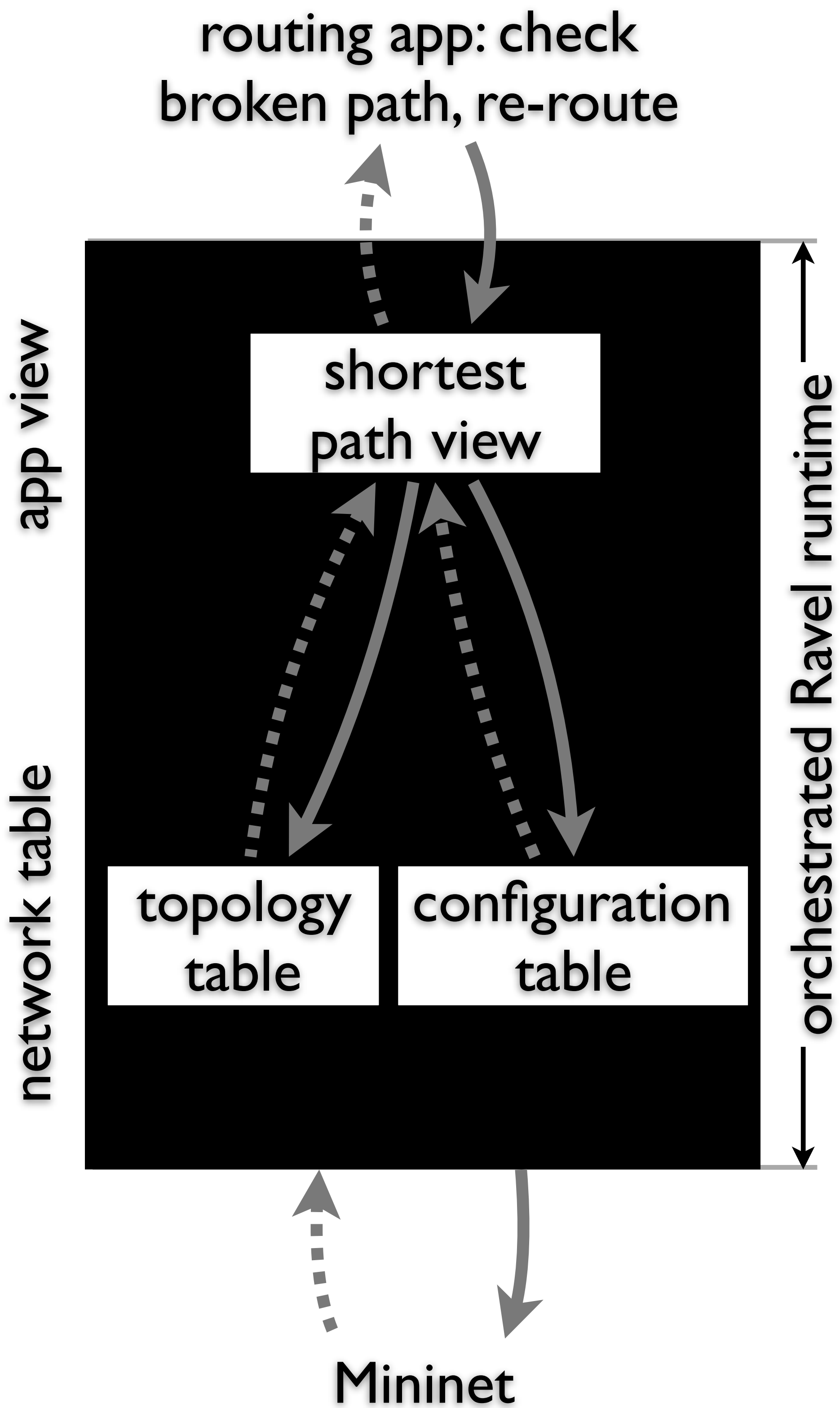
firewall control: repairing violation

```
CREATE RULE acl_repair AS
   ON DELETE TO acl_violation
   DO INSTEAD
         DELETE FROM rm WHERE fid = OLD.fid;
```

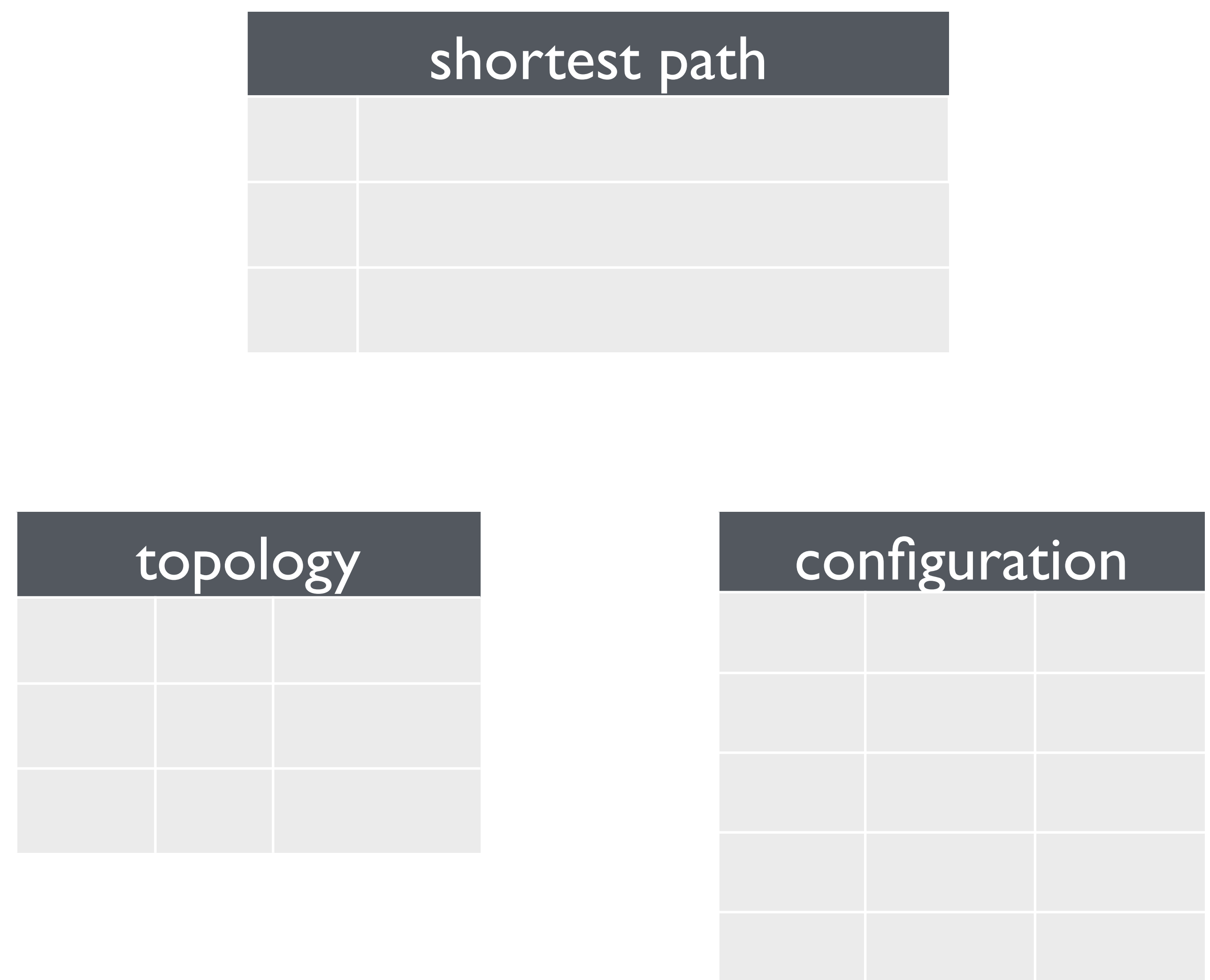
many more

- routing, stateful firewall, service chain policy between subdomains …

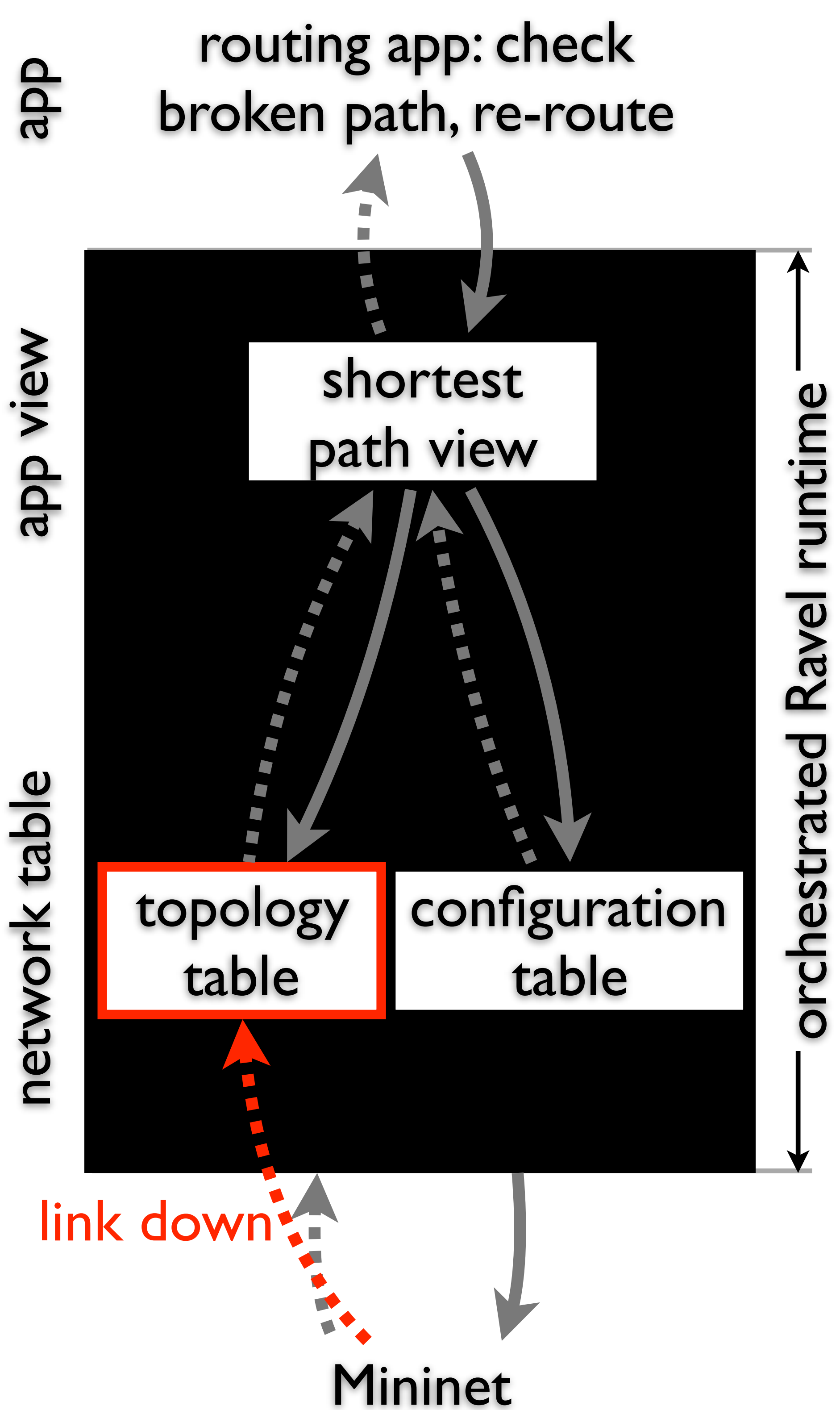
orchestration across representations



SQL rule:
upon broken path, re-route



orchestration across representations



SQL rule:
upon broken path, re-route

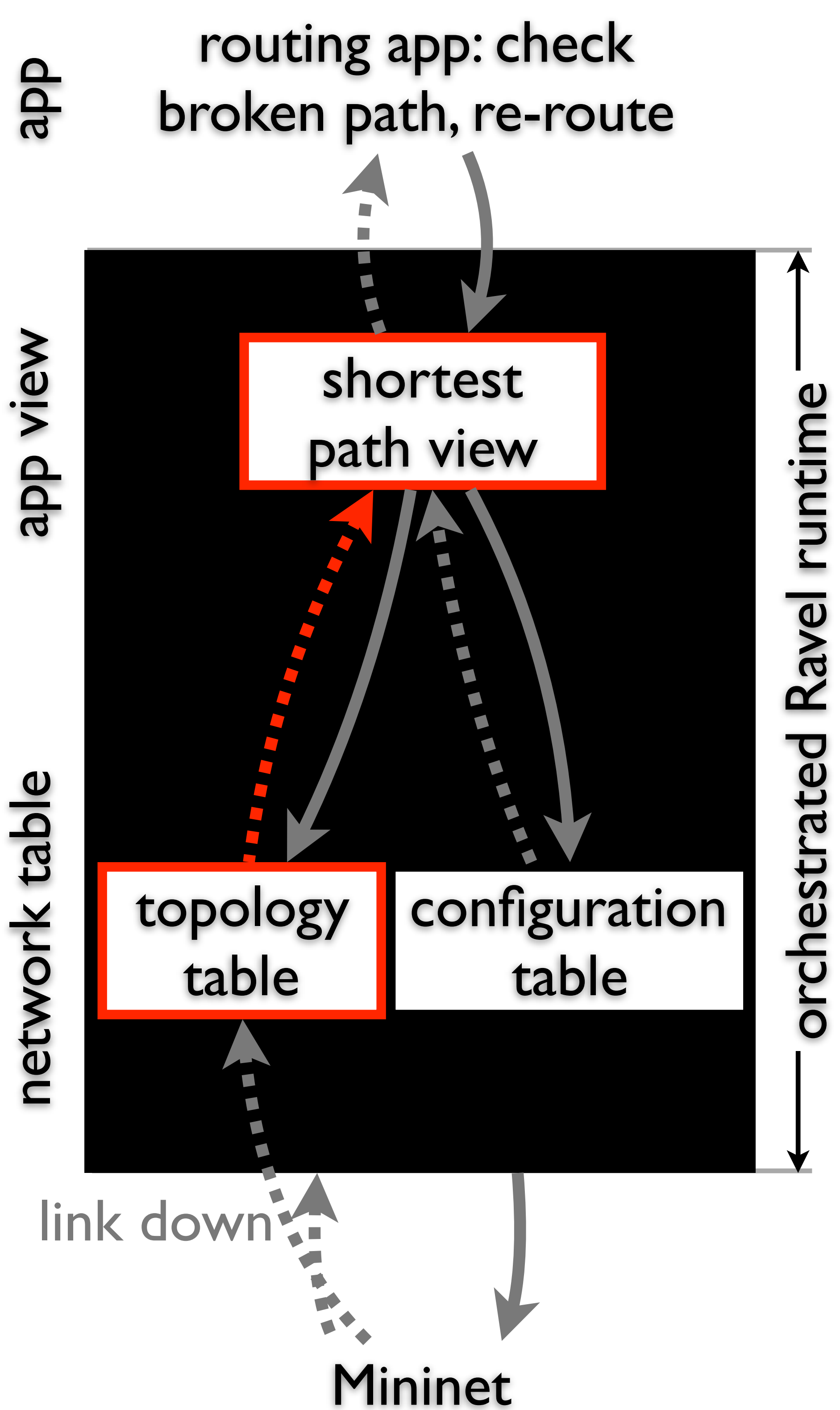
shortest path	

	topology		
	sid	nid	active
-	172	39	1
+	172	39	0

Mininet link (172,39) down

configuration		

orchestration across representations



SQL rule:
upon broken path, re-route

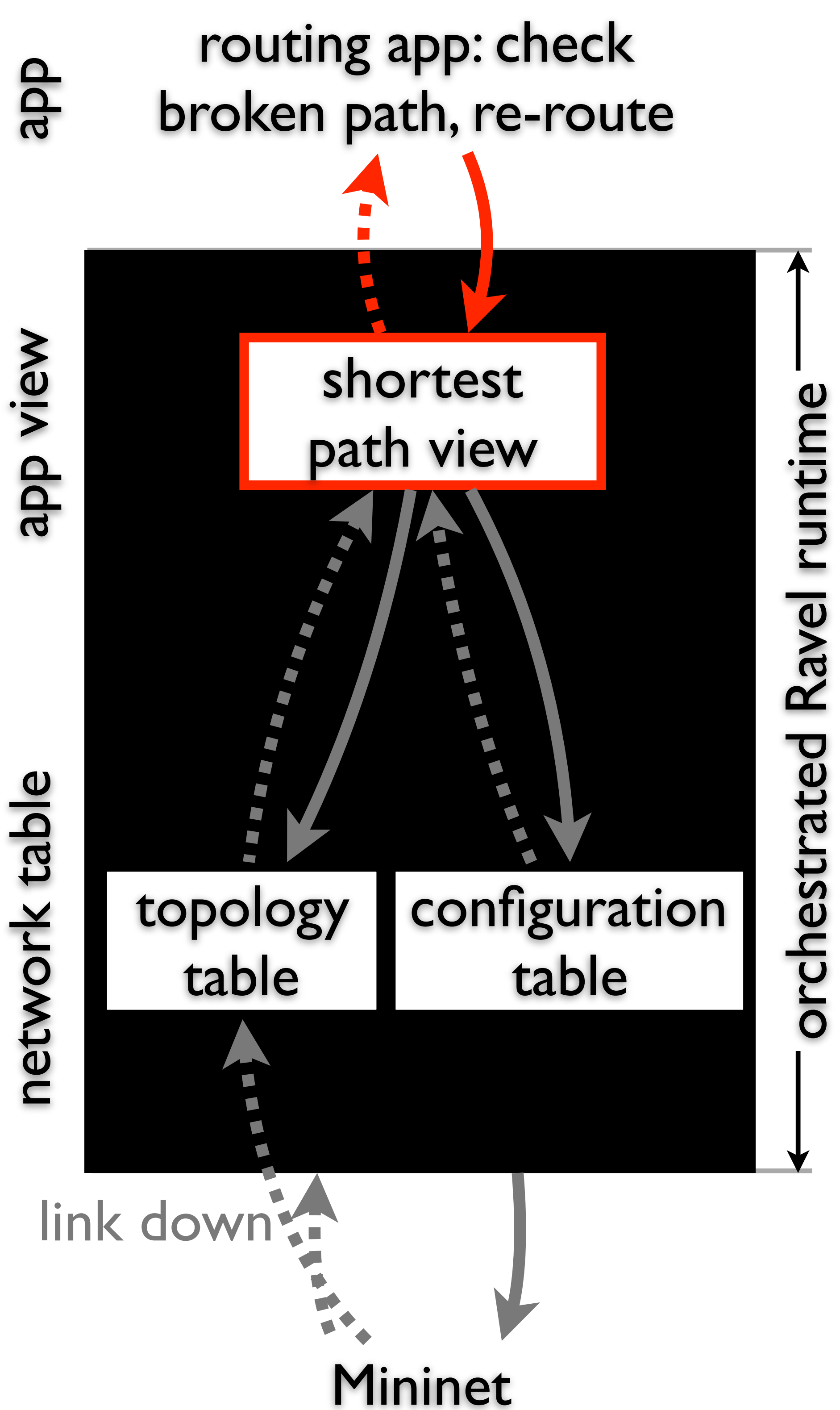
	shortest path	
	…	path
-		{…,172,39,156,…}

	topology		
	sid	nid	active
-	172	39	1
+	172	39	0

configuration		

Mininet link (172,39) down

orchestration across representations



SQL rule:
upon broken path, re-route

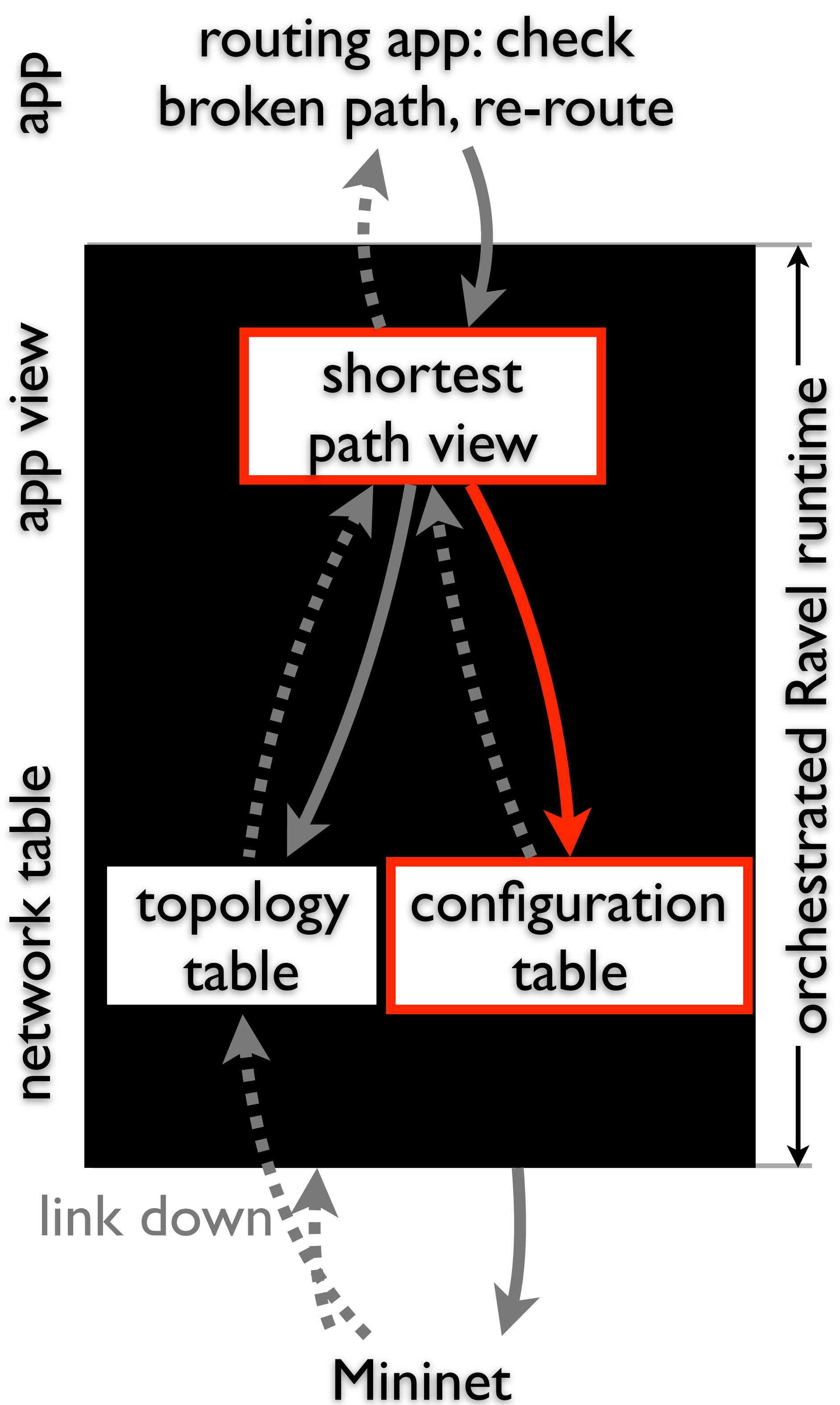
	shortest path	
	...	path
-		{...,172,39,156,...}
+		{...,172,38,148,...}

	topology		
	sid	nid	active
-	172	39	1
+	172	39	0

configuration		

Mininet link (172,39) down

orchestration across representations



SQL rule:
upon broken path, re-route

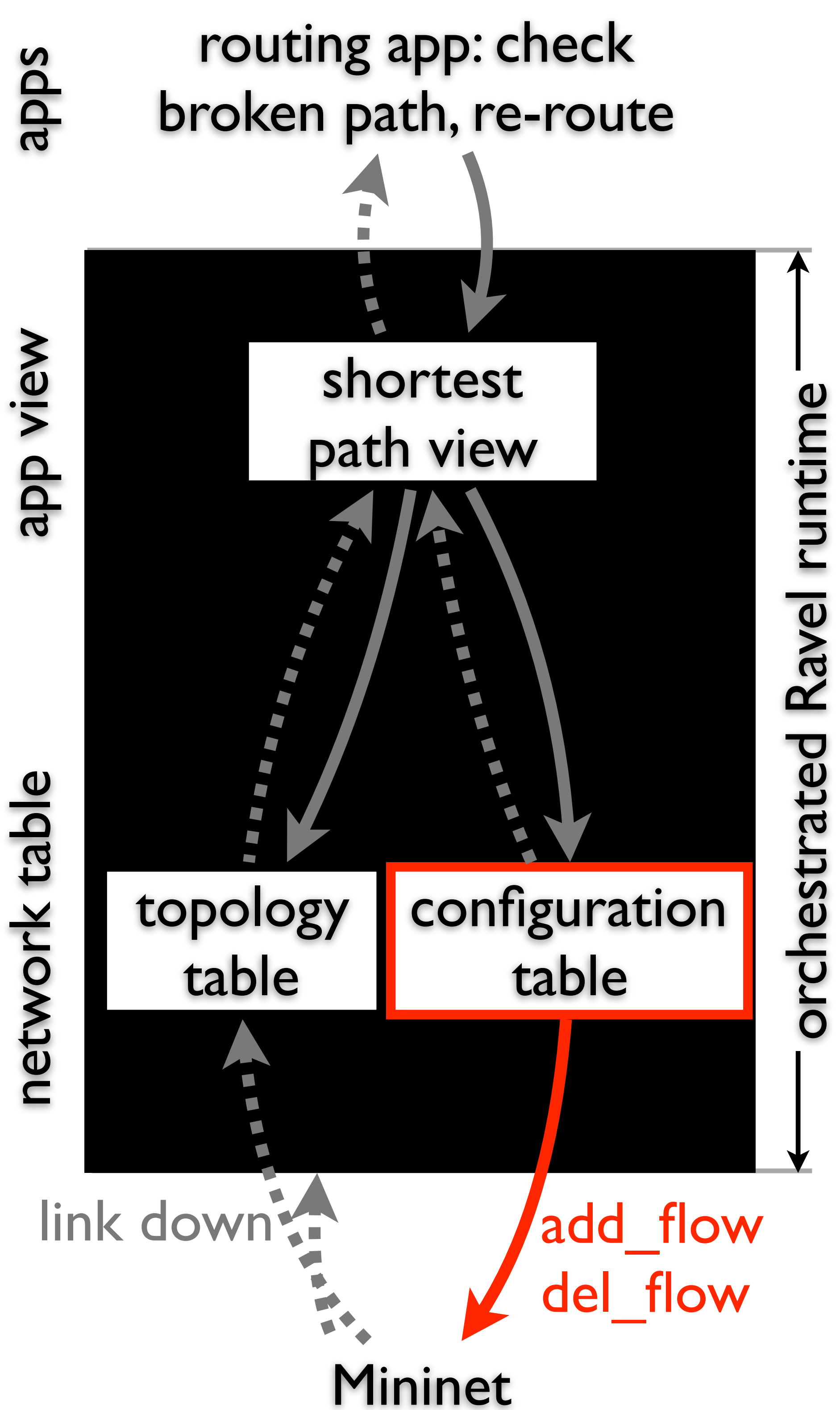
	shortest path	
	...	path
-		{...,172,39,156,...}
+		{...,172,38,148,...}

	topology		
	sid	nid	active
-	172	39	1
+	172	39	0

	configuration		
	fid	sid	nid
-	...	172	39
-	...	39	156
+	...	172	38
+	...	38	148

Mininet link (172,39) down

orchestration across representations



SQL rule:
upon broken path, re-route

shortest path

	…	path
-		{…,172,39,156,…}
+		{…,172,38,148,…}

topology

	sid	nid	active
-	172	39	1
+	172	39	0

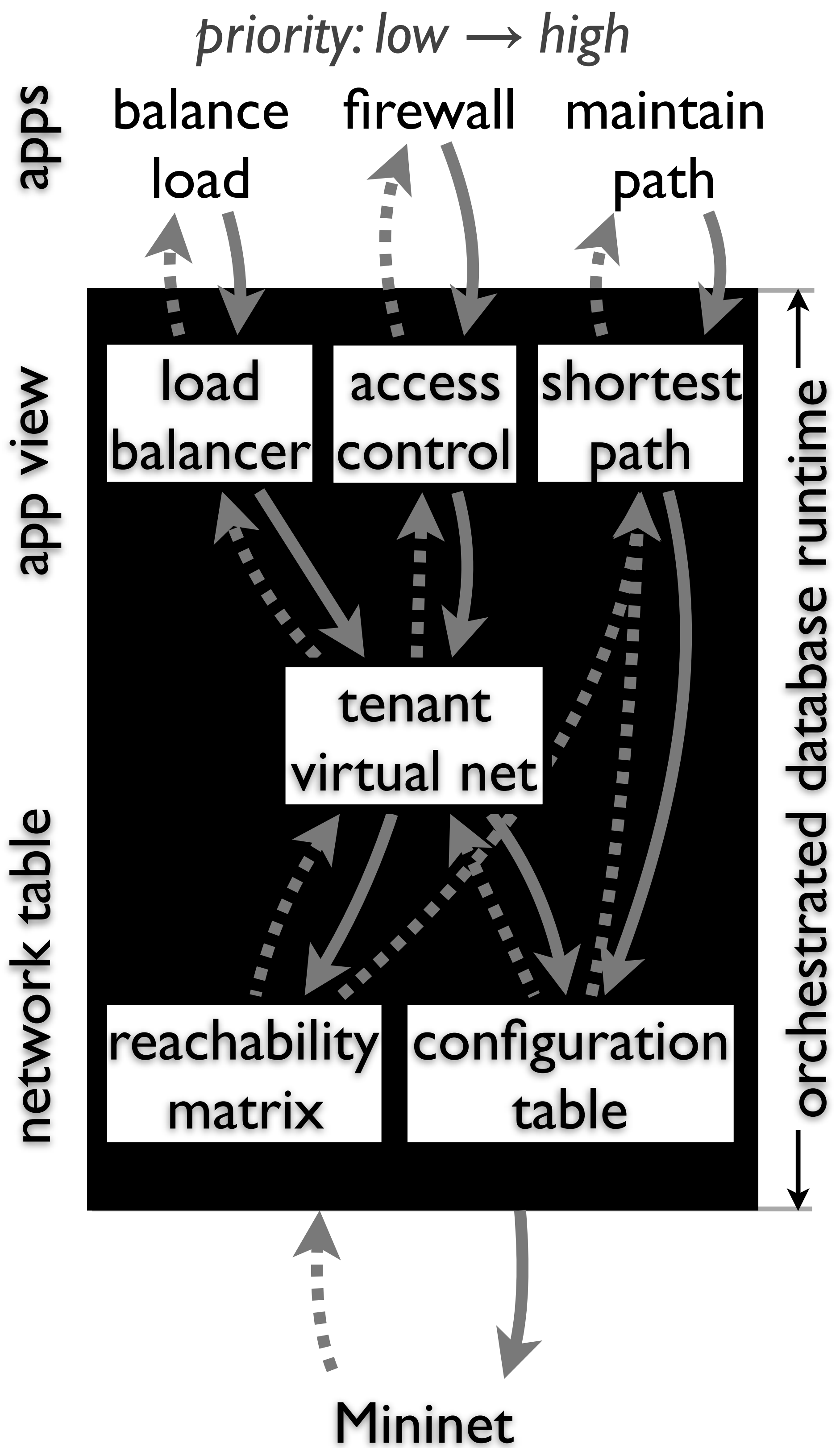
Mininet link (172,39) down

configuration

	fid	sid	nid
-	…	172	39
-	…	39	156
+	…	172	38
+	…	38	148

orchestrated updates:
re-route via (172, 38)

orchestration across applications



load balancer	

access control		

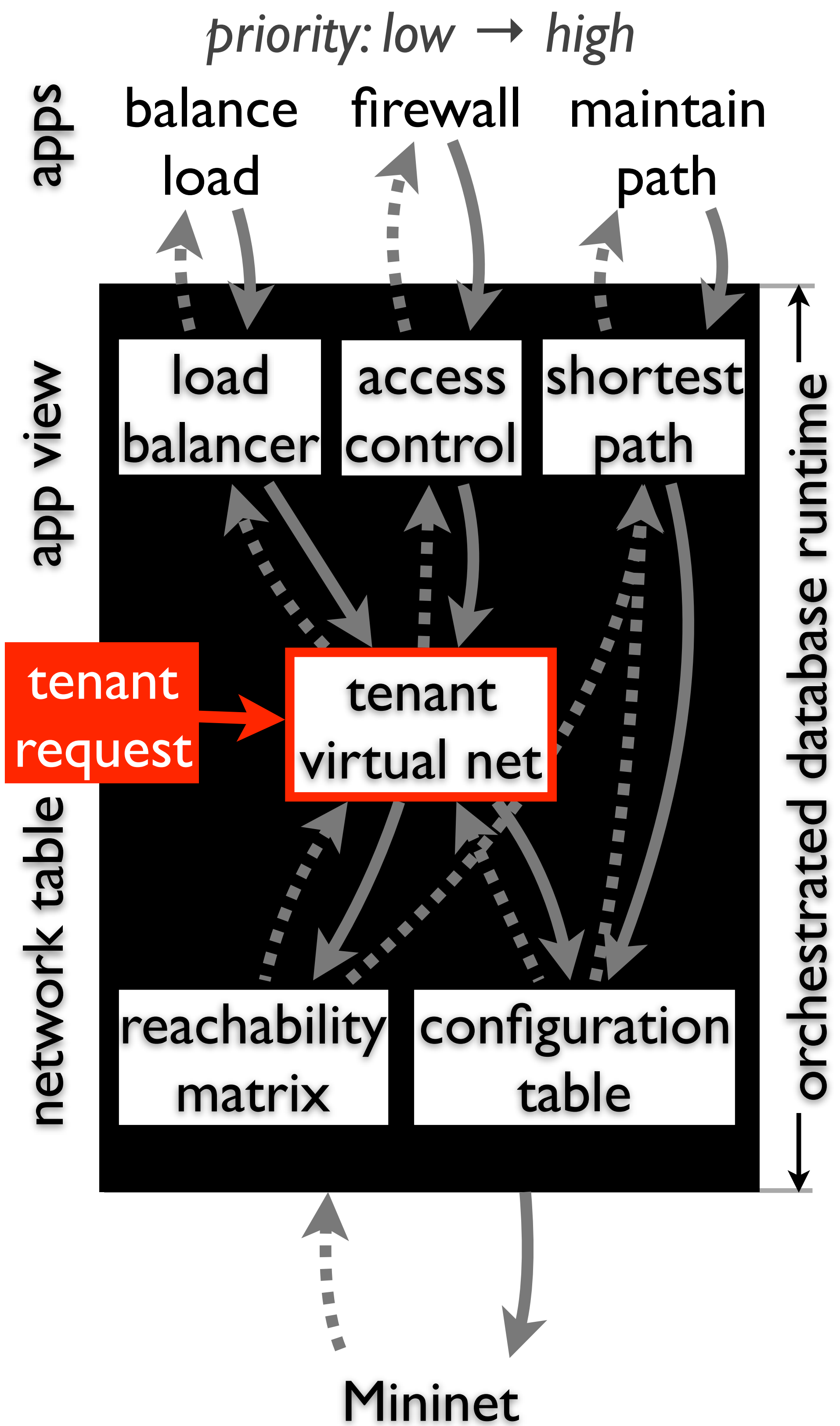
shortest path	

tenant virtual net		

reachability matrix		

configuration		

orchestration across applications



load balancer	

access control		

shortest path	

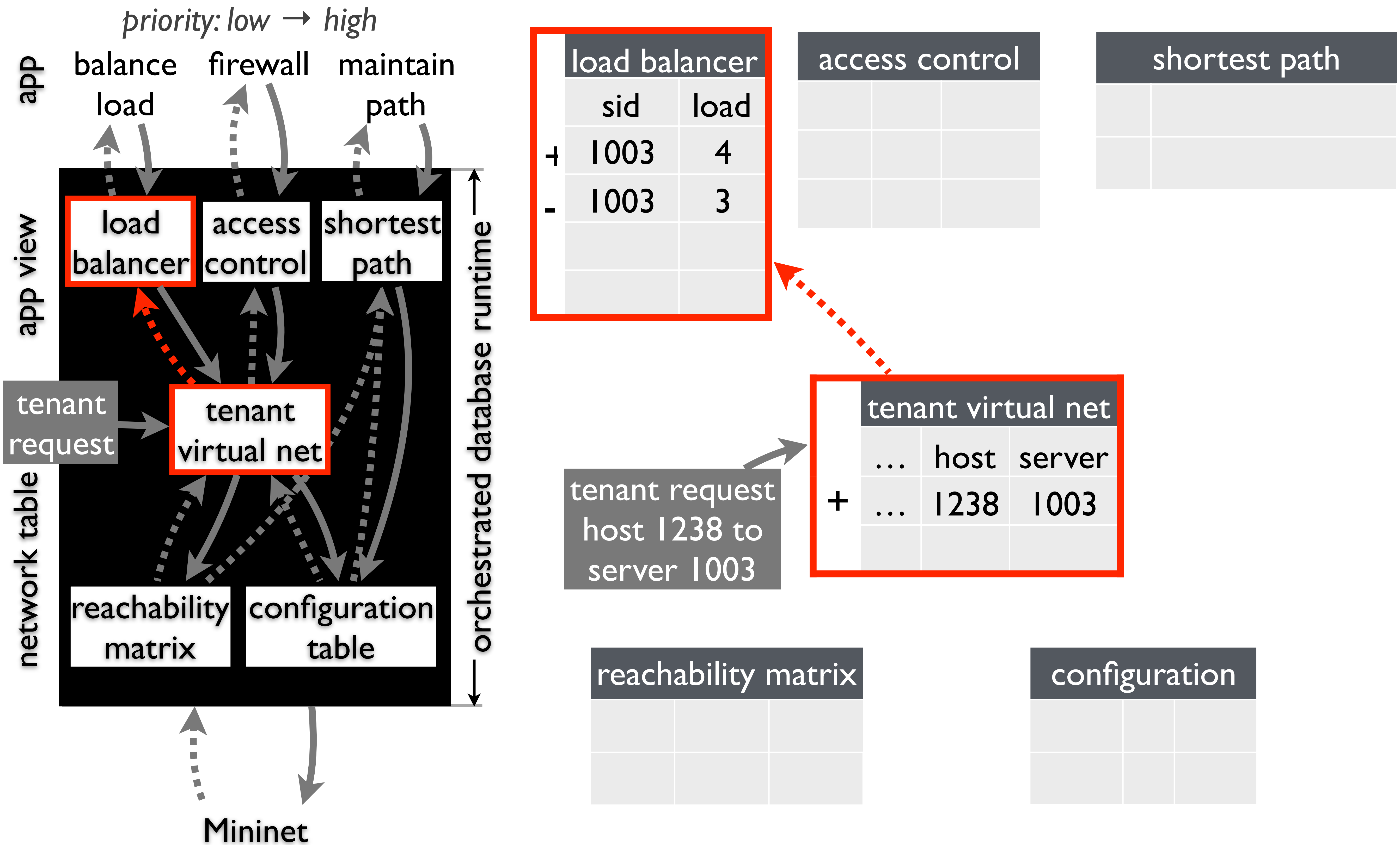
tenant request host 1238 to server 1003

	tenant virtual net		
	…	host	server
+	…	1238	1003

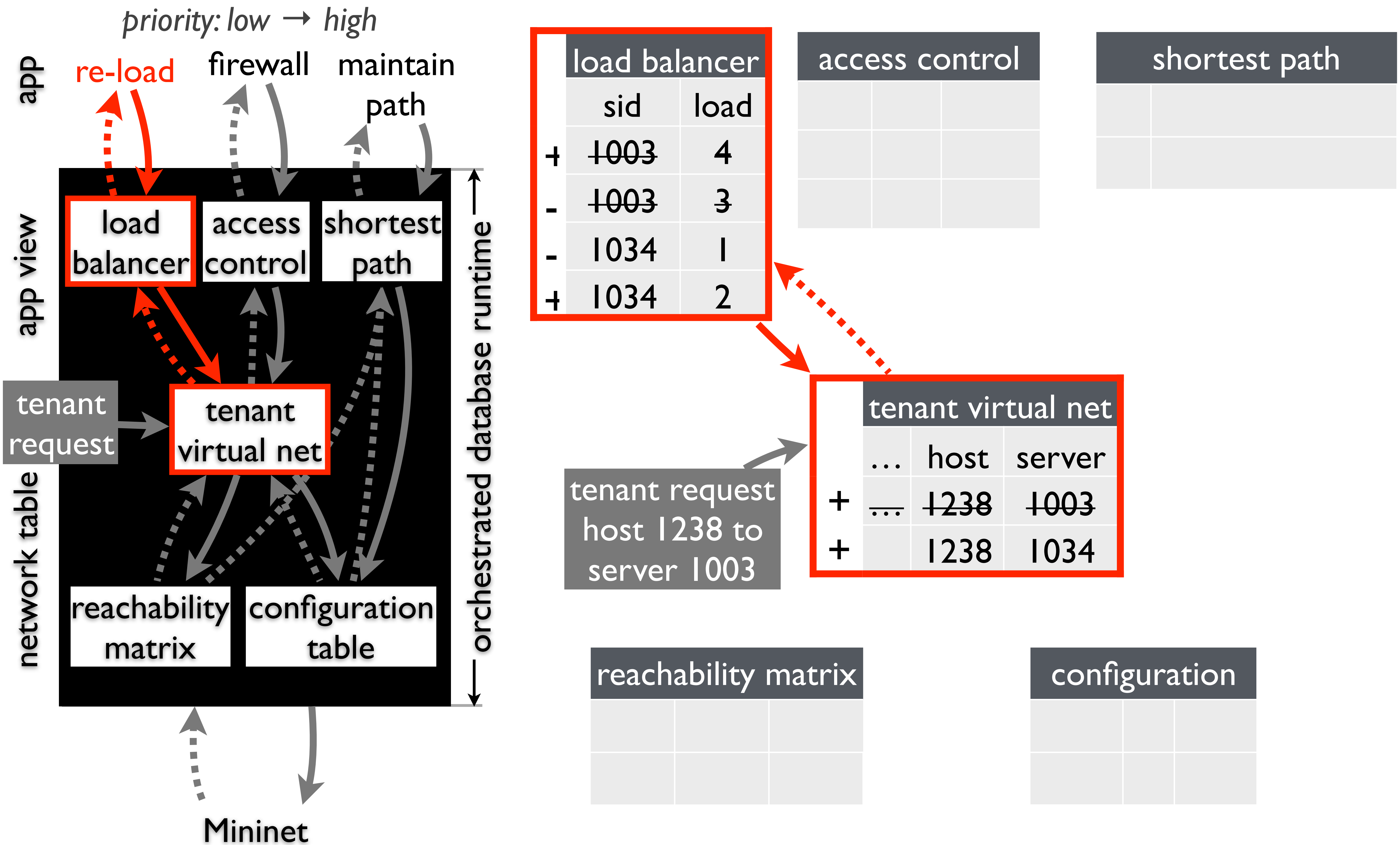
reachability matrix		

configuration		

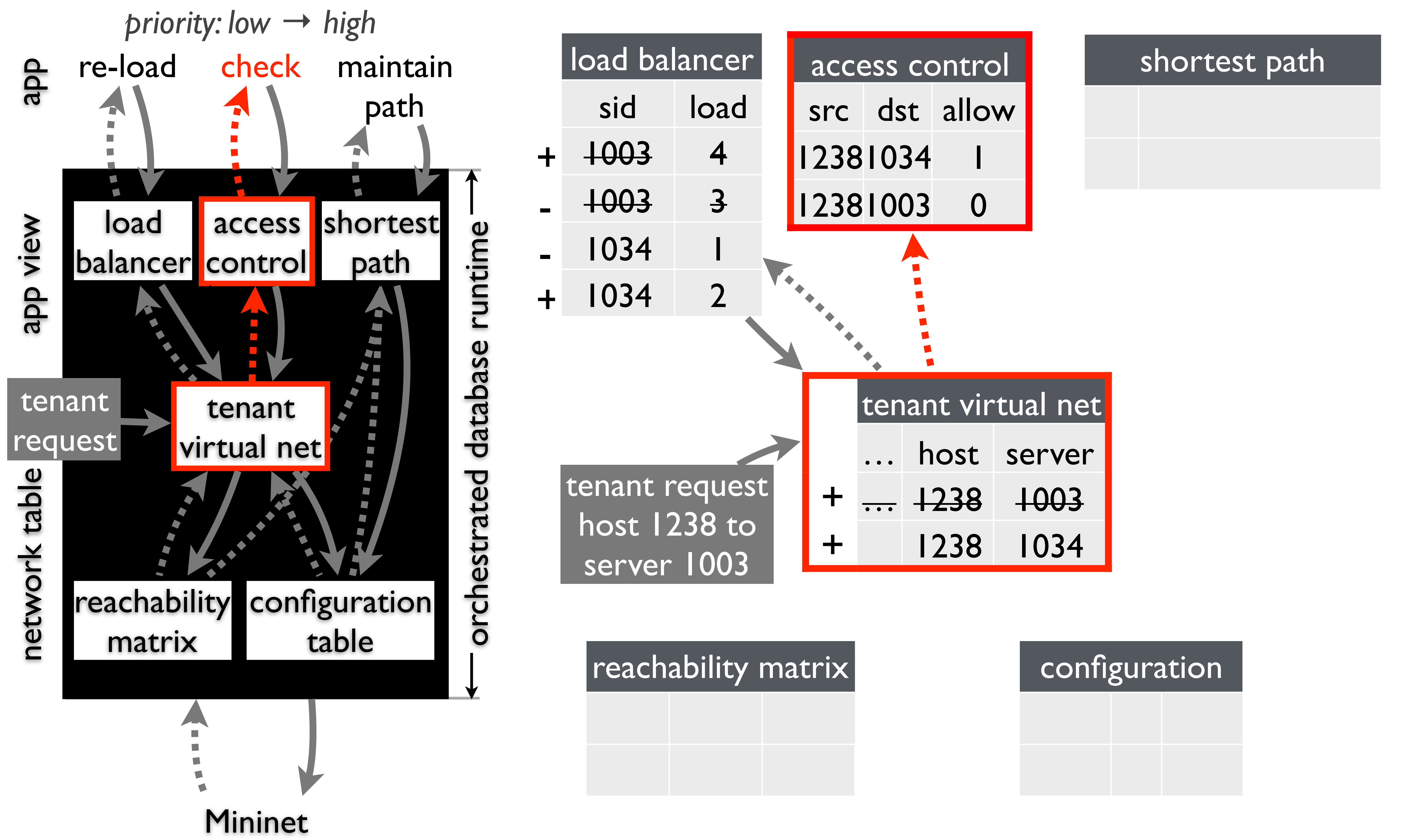
orchestration across applications



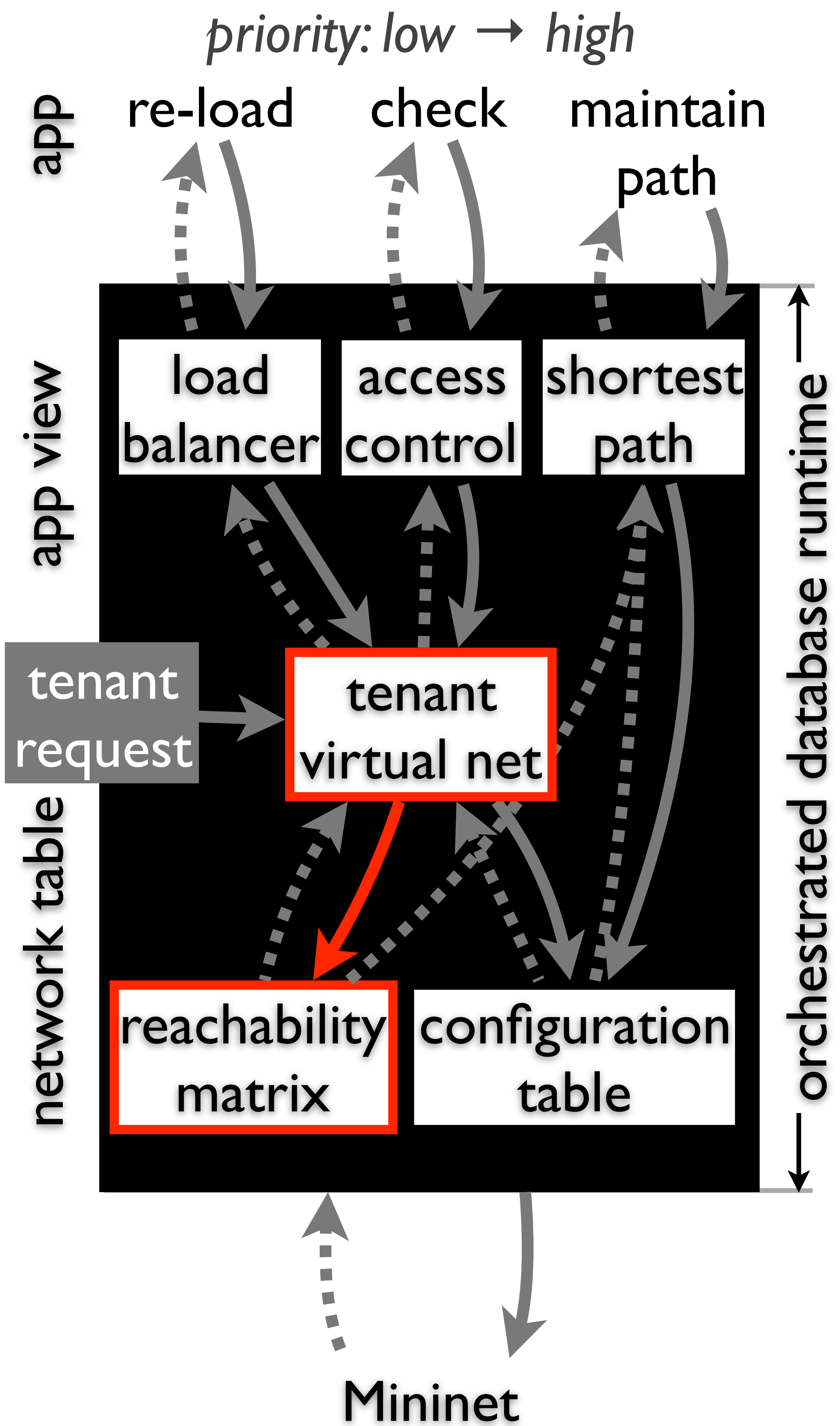
orchestration across applications



orchestration across applications



orchestration across applications



	load balancer	
	sid	load
+	~~1003~~	4
-	~~1003~~	~~3~~
-	1034	1
+	1034	2

access control		
src	dst	allow
1238	1034	1
1238	1003	0

shortest path	

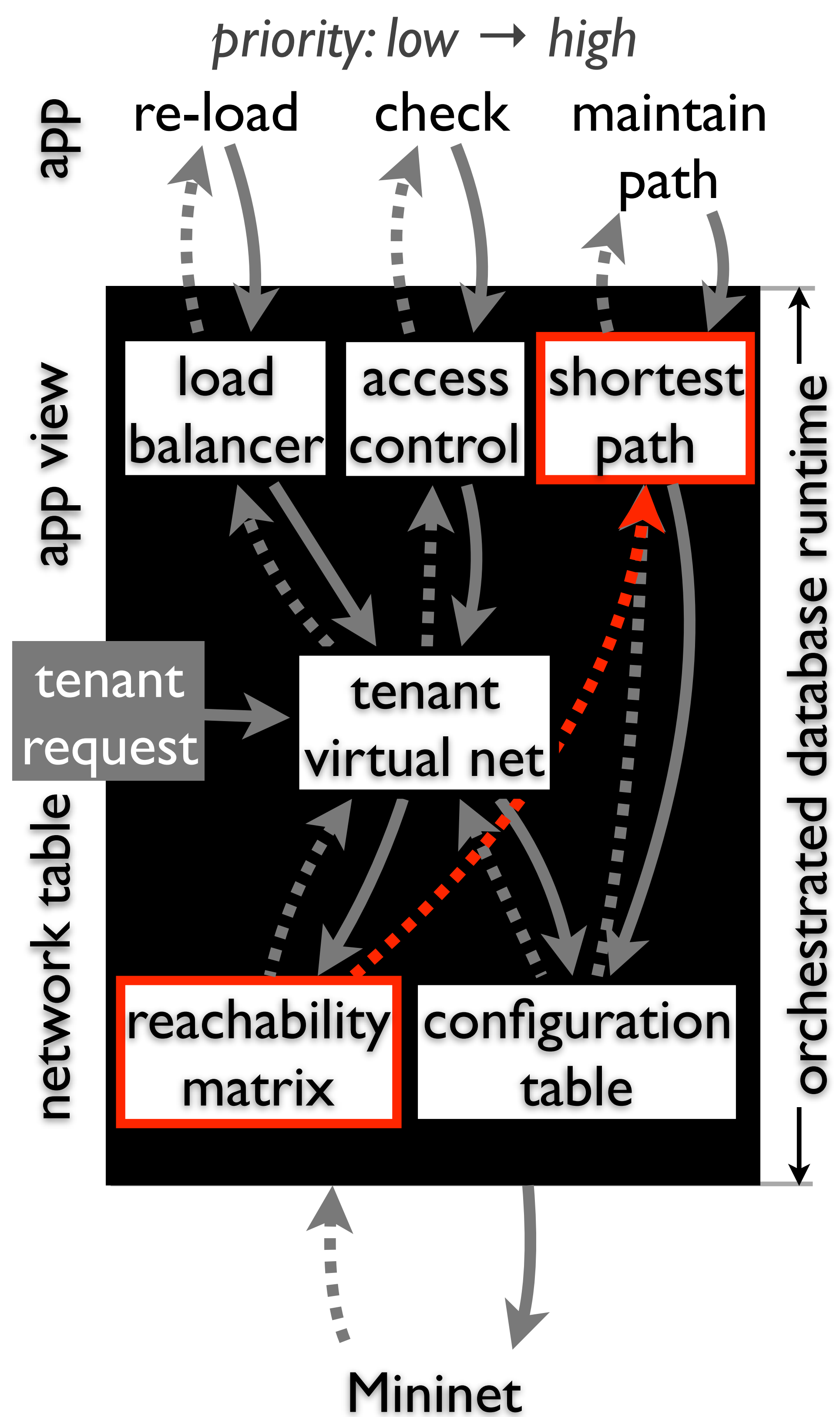
tenant request host 1238 to server 1003

	tenant virtual net		
	…	host	server
+	~~…~~	~~1238~~	~~1003~~
+		1238	1034

	reachability matrix		
	fid	sid	nid
+	…	1238	1034

configuration		

orchestration across applications



	load balancer	
	sid	load
+	~~1003~~	4
-	~~1003~~	~~3~~
-	1034	1
+	1034	2

access control		
src	dst	allow
1238	1034	1
1238	1003	0

	shortest path	
	…	path
+	…	{1238,…,1034}

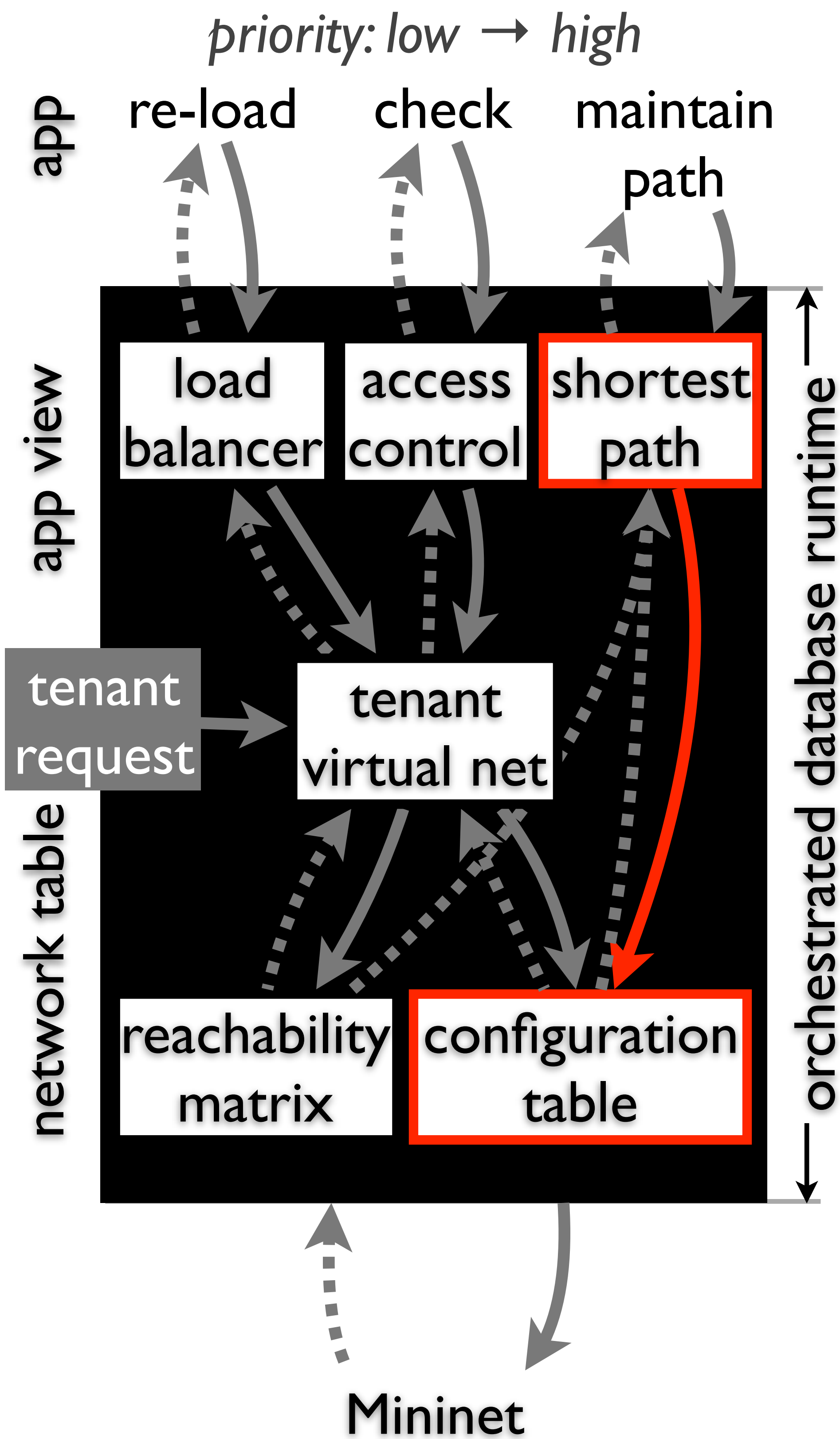
tenant request host 1238 to server 1003

	tenant virtual net		
	…	host	server
~~+~~	~~…~~	~~1238~~	~~1003~~
+		1238	1034

	reachability matrix		
	fid	sid	nid
+	…	1238	1034

configuration		

orchestration across applications



	load balancer	
	sid	load
+	~~1003~~	4
-	~~1003~~	~~3~~
-	1034	1
+	1034	2

access control		
src	dst	allow
1238	1034	1
1238	1003	0

	shortest path	
	…	path
+	…	{1238,…,1034}

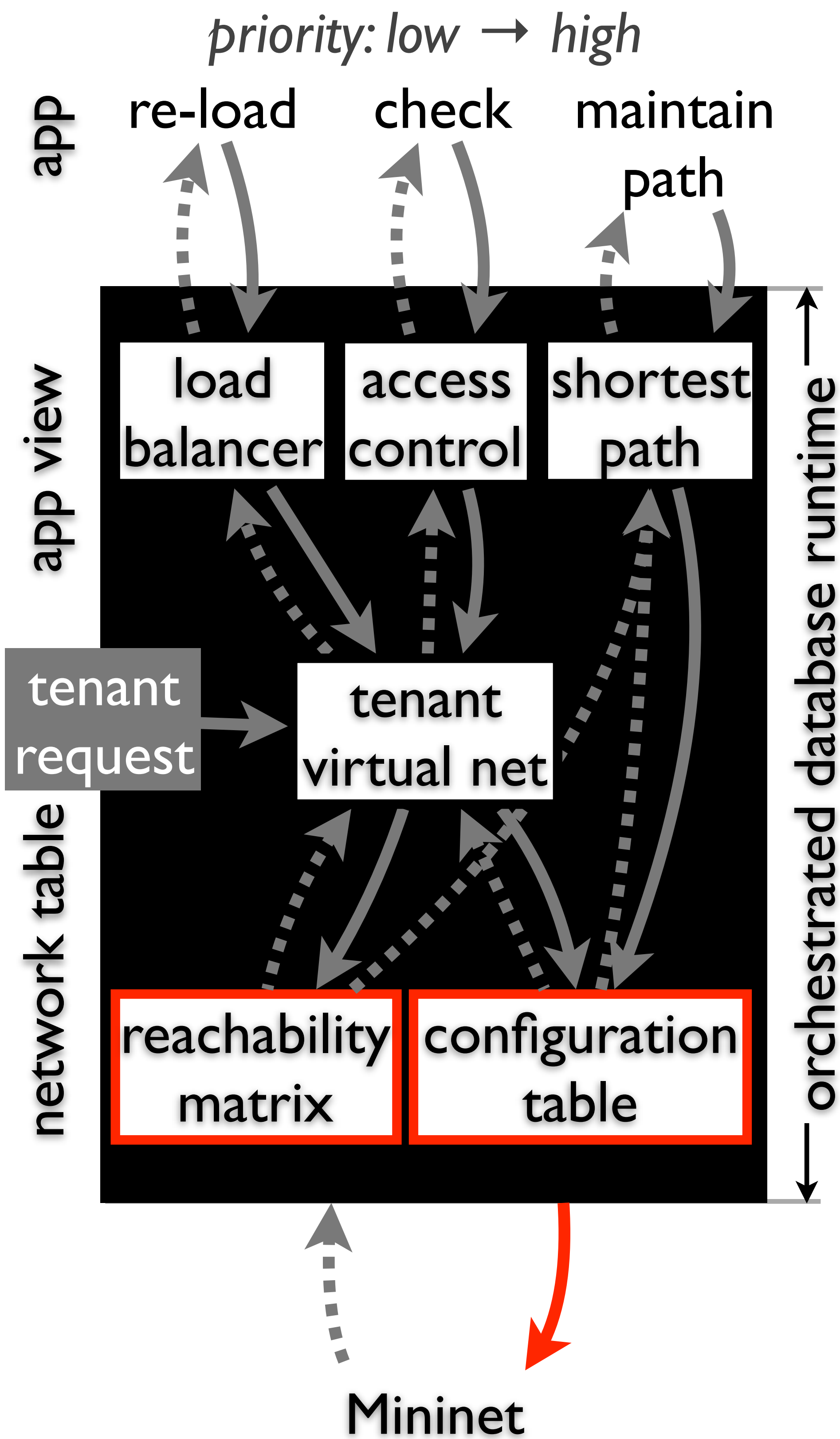
tenant request host 1238 to server 1003

	tenant virtual net		
	…	host	server
+	~~…~~	~~1238~~	~~1003~~
+		1238	1034

	reachability matrix		
	fid	sid	nid
+	…	1238	1034

	configuration		
	fid	sid	nid
+	…	…	1034

orchestration across applications



	load balancer	
	sid	load
+	~~1003~~	4
-	~~1003~~	~~3~~
-	1034	1
+	1034	2

access control		
src	dst	allow
1238	1034	1
1238	1003	0

	shortest path	
	…	path
+	…	{1238,…,1034}

	tenant virtual net		
	…	host	server
+	~~…~~	~~1238~~	~~1003~~
+		1238	1034

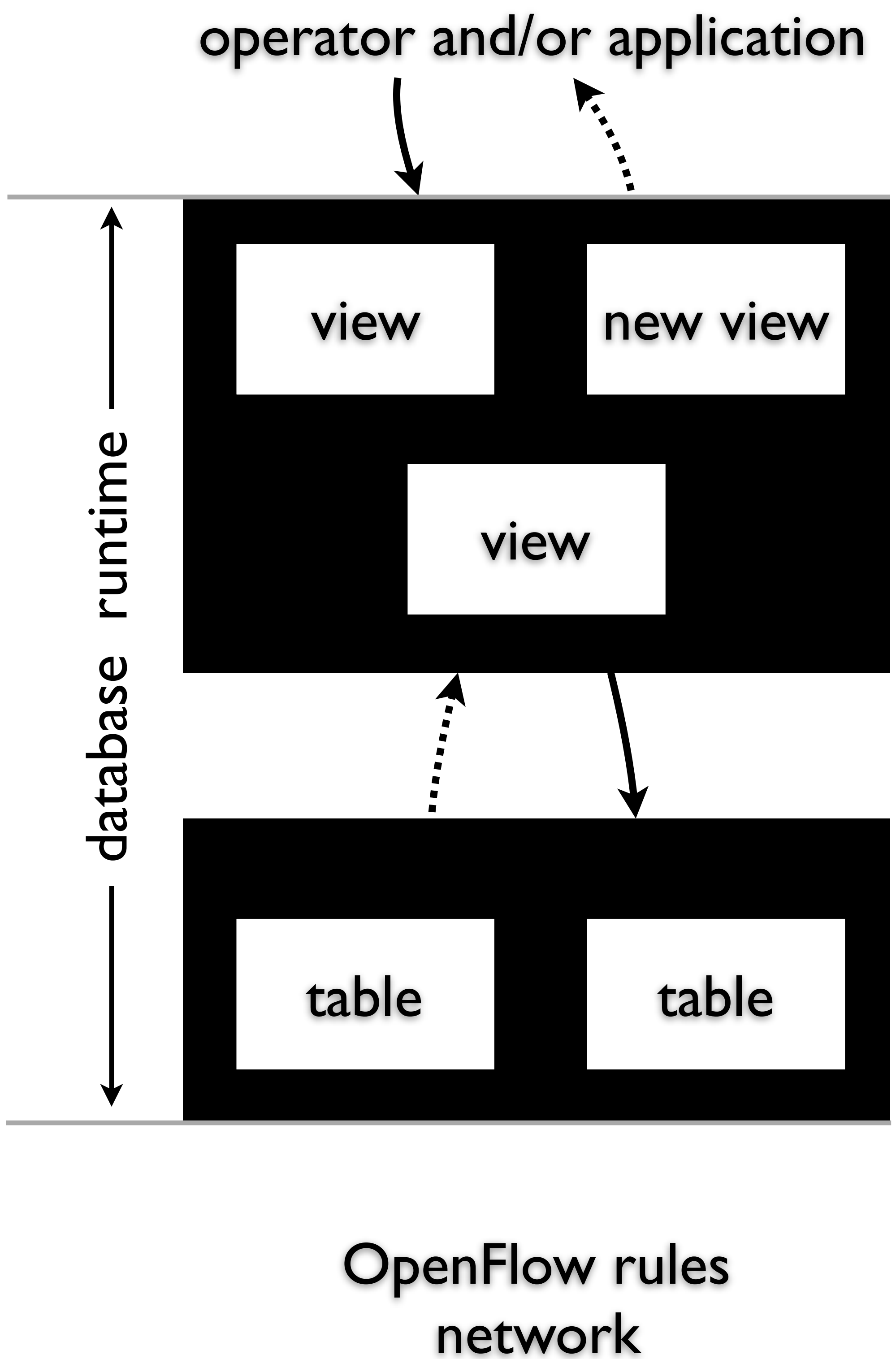
tenant request host 1238 to server 1003

	reachability matrix		
	fid	sid	nid
+	…	1238	1034

	configuration		
	fid	sid	nid
+	…	…	1034

orchestrated updates: install alternative route that is load-balanced and safe

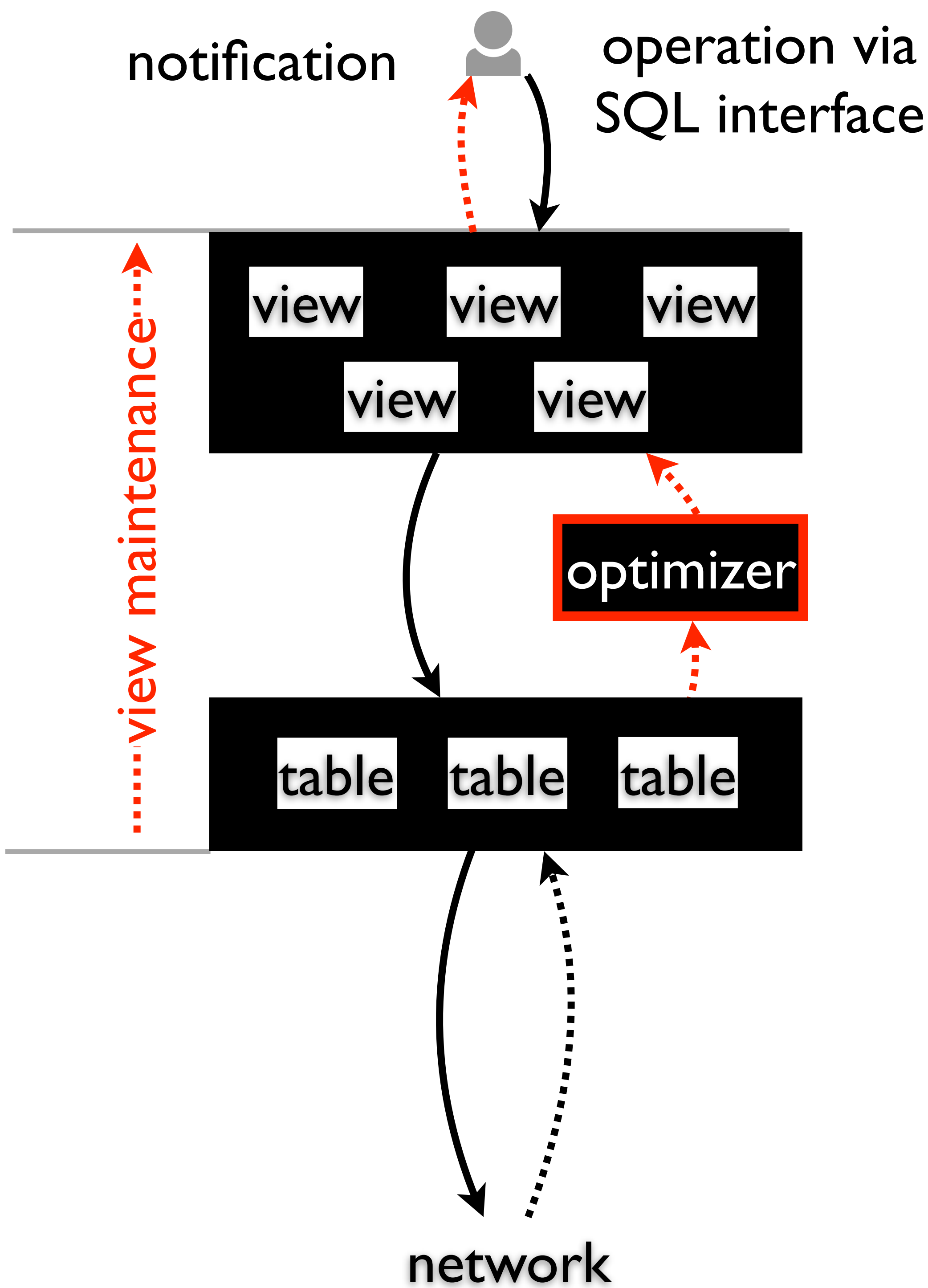
achieving *Ravel* advantages



attractive features

- ad-hoc programmable abstraction via views
- orchestration across abstractions via view mechanism
- orchestration across applications via data mediation
- network control via SQL

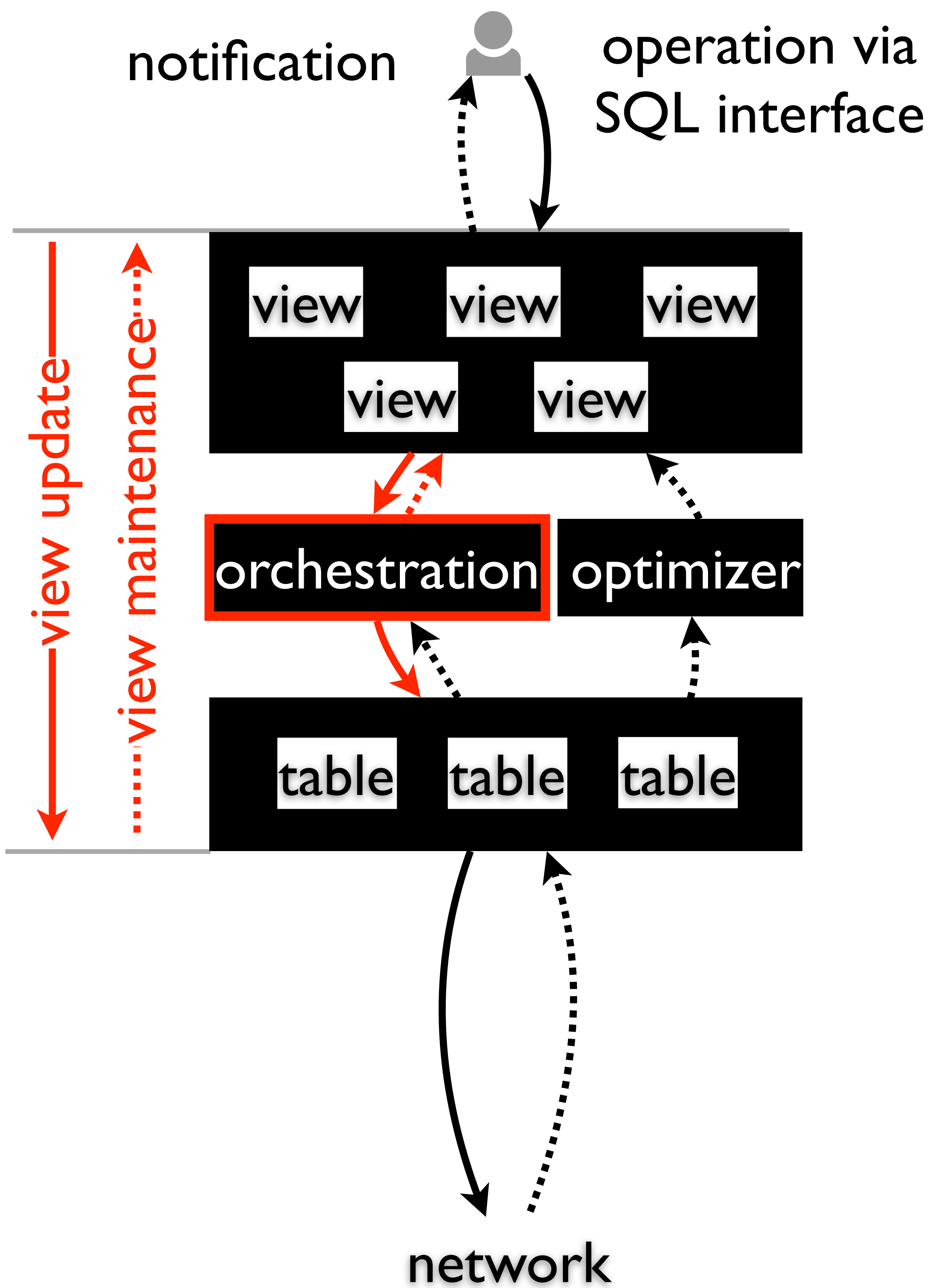
runtime



ad-hoc programmable abstraction via views

- challenge: inefficient user view
- solution: optimizer
  - materialize user view with fast maintenance algorithm
  - one order of magnitude faster access with small maintenance overhead — 0.01~10ms

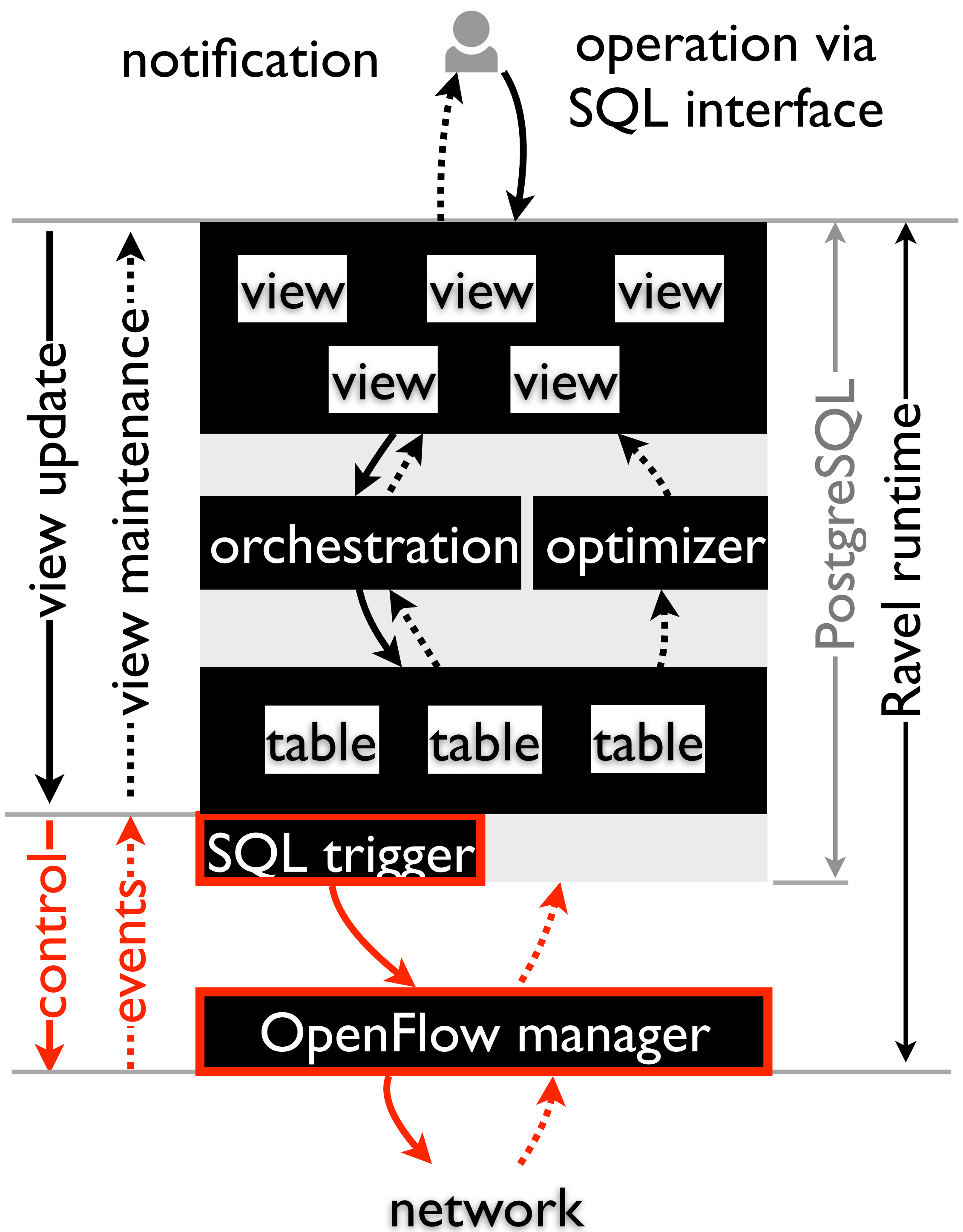
runtime



orchestration across applications

- challenge: database lacking inter-view support
- solution: mediation protocol
  - translate app priority into view updates that dynamically merge into a coherent data plane

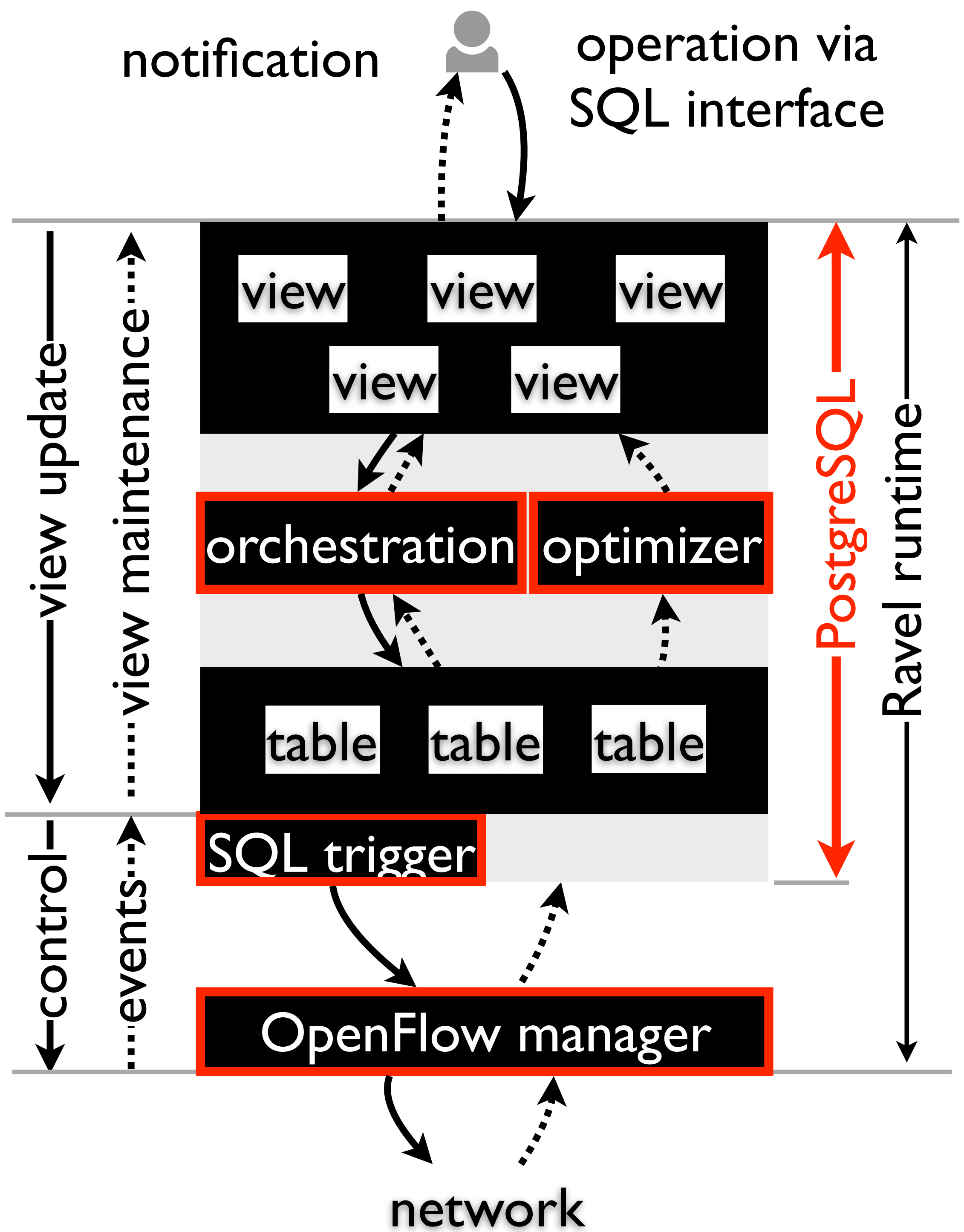
runtime



SDN control via SQL

- challenge: database lacks connection to network data plane
- solution: SQL trigger + OF manager

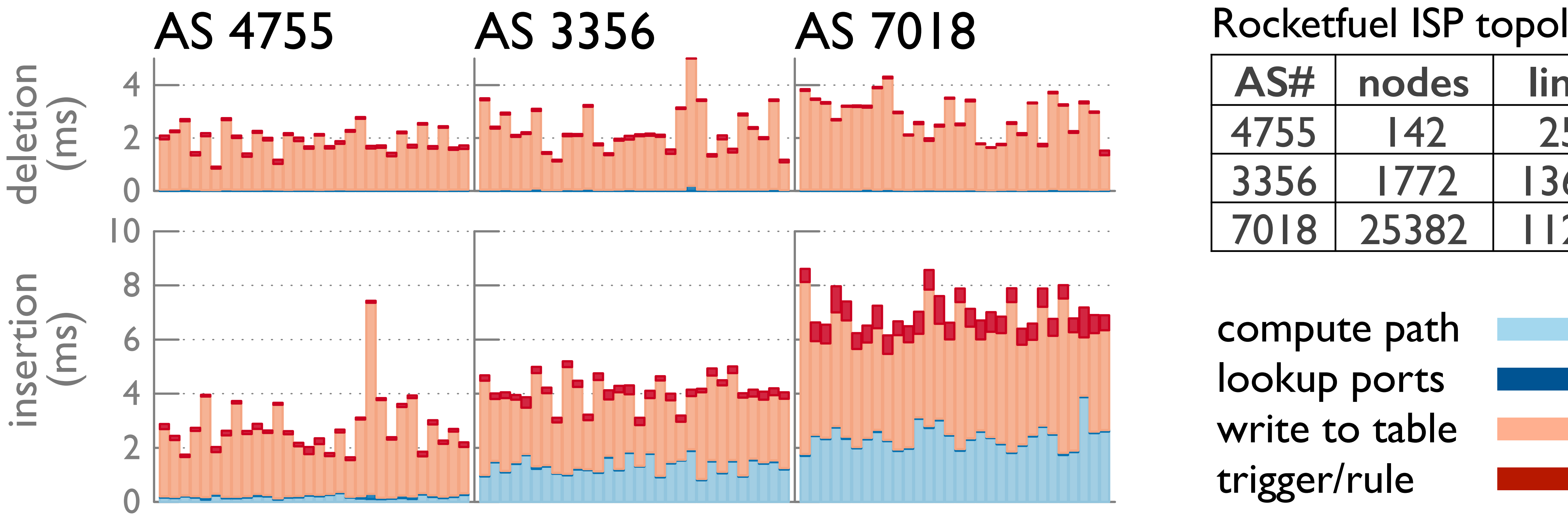
runtime



a high-performance runtime

- PostgreSQL
- orchestration
- optimizer
- SQL trigger and OF manager

evaluation

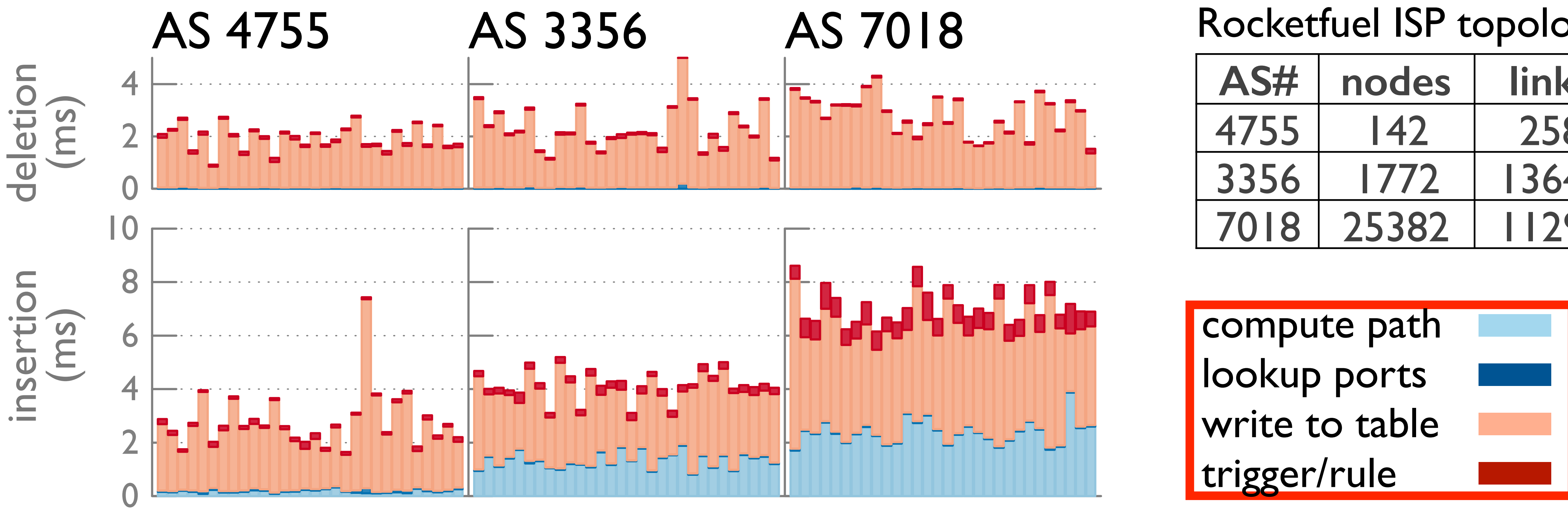


profile end to end delay (normalized per-rule, 30 rounds) for route insertion and deletion

Rocketfuel ISP topology

AS#	nodes	links
4755	142	258
3356	1772	13640
7018	25382	11292

evaluation

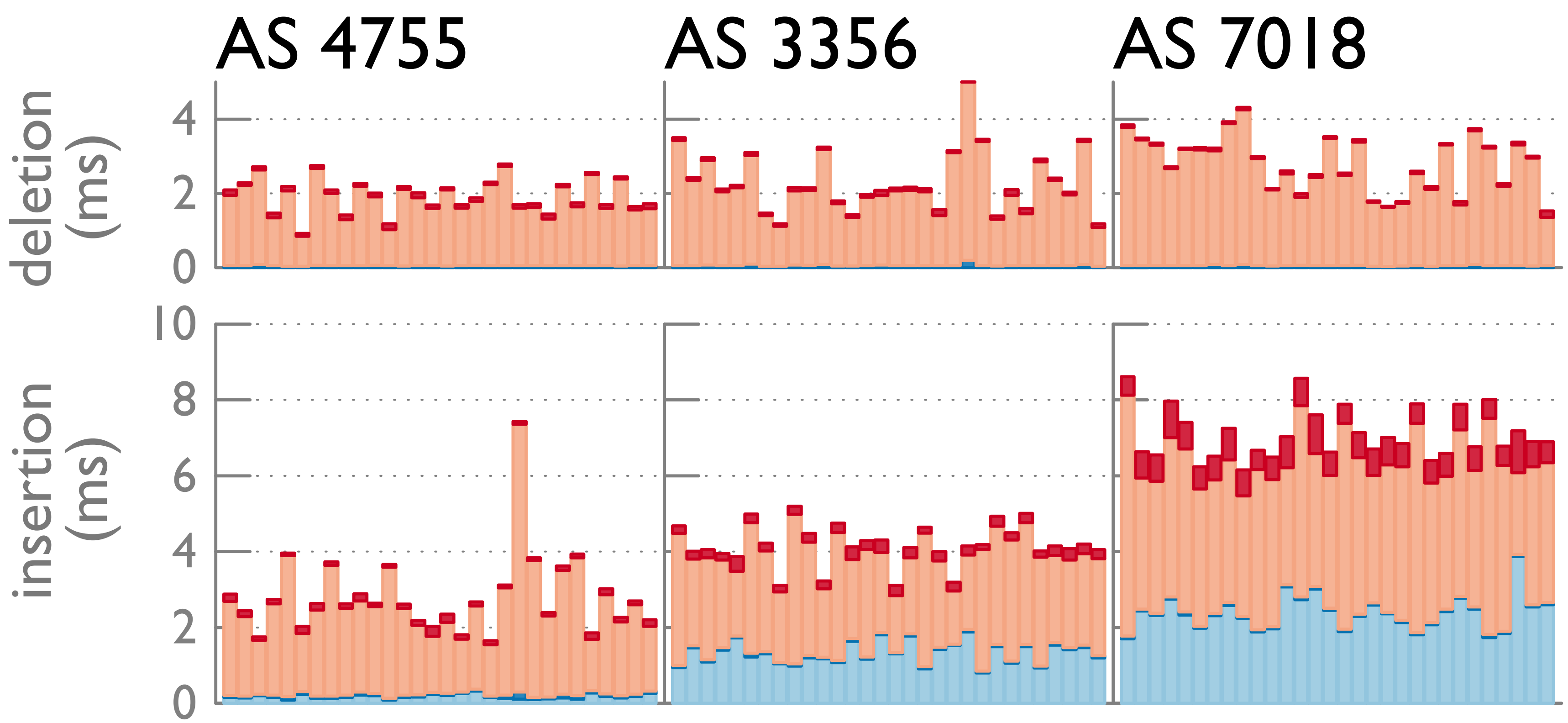


profile end to end delay (normalized per-rule, 30 rounds) for route insertion and deletion

Rocketfuel ISP topology

AS#	nodes	links
4755	142	258
3356	1772	13640
7018	25382	11292

evaluation

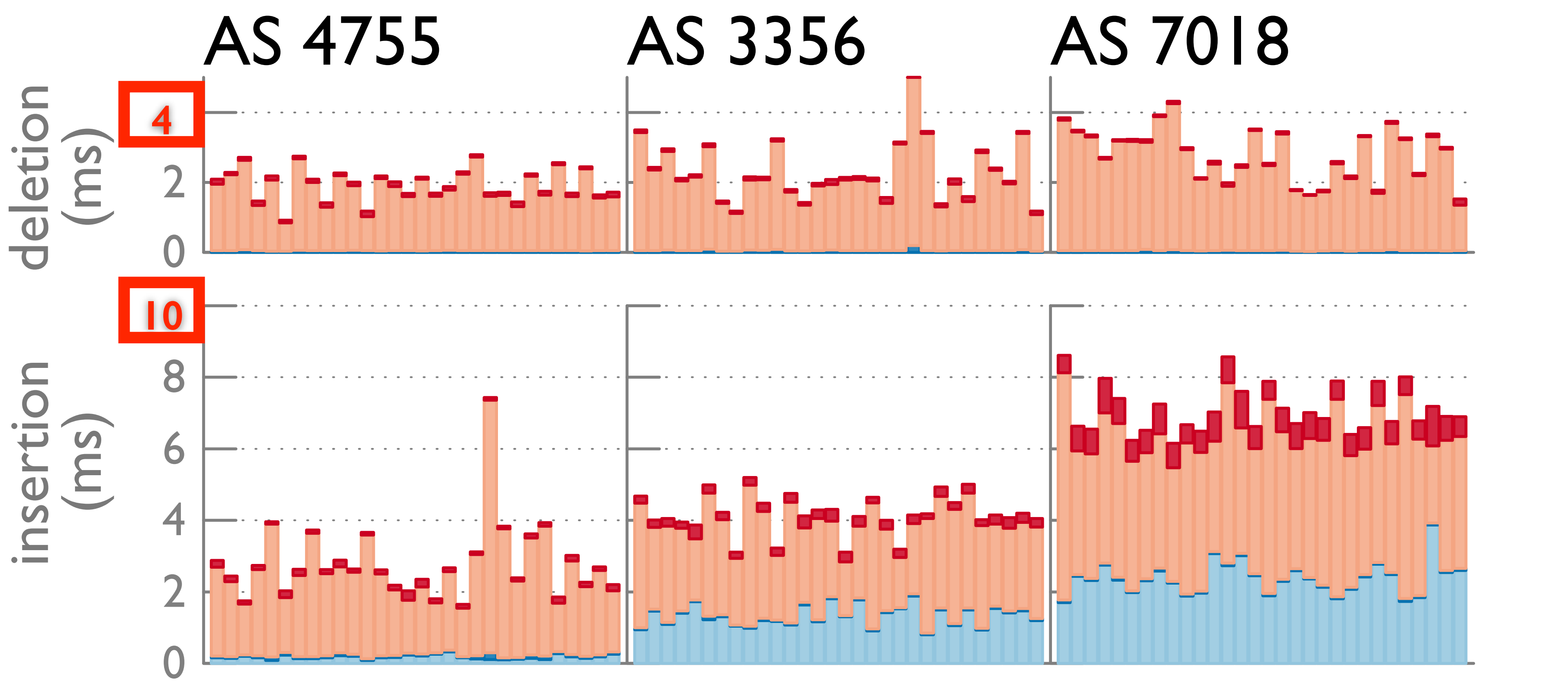


profile end to end delay (normalized per-rule, 30 rounds) for route insertion and deletion

Rocketfuel ISP topology

AS#	nodes	links
4755	142	258
3356	1772	13640
7018	25382	11292

evaluation

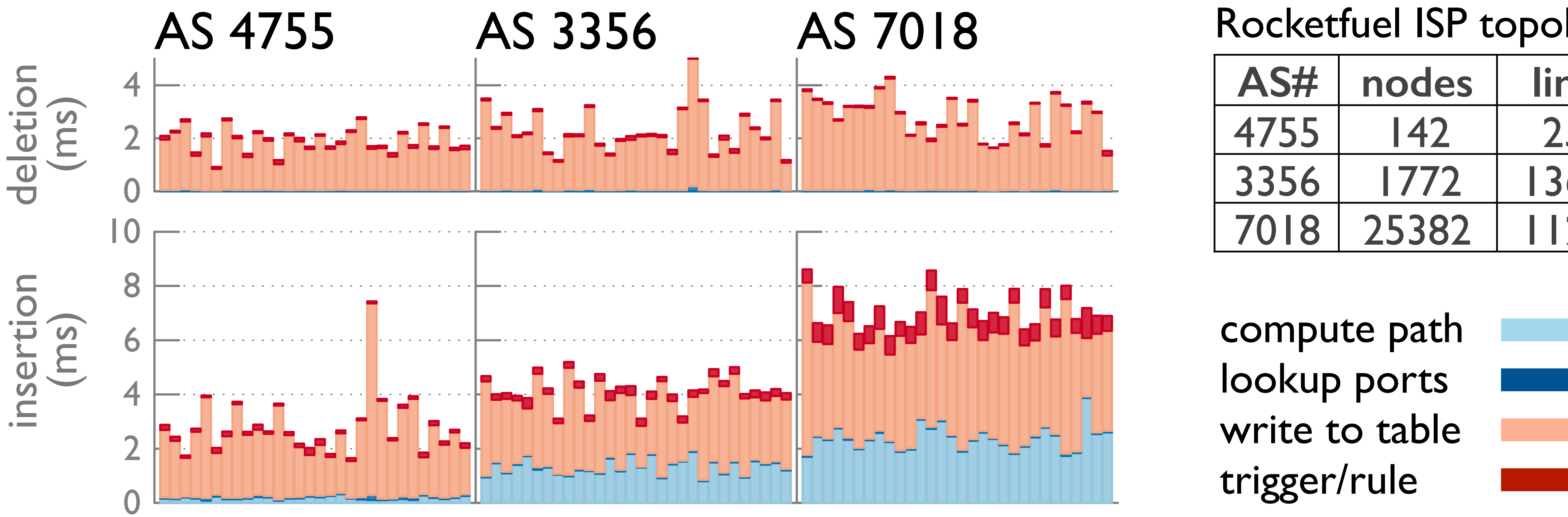


Rocketfuel ISP topology

AS#	nodes	links
4755	142	258
3356	1772	13640
7018	25382	11292

profile end to end delay (normalized per-rule, 30 rounds) for route insertion and deletion

evaluation

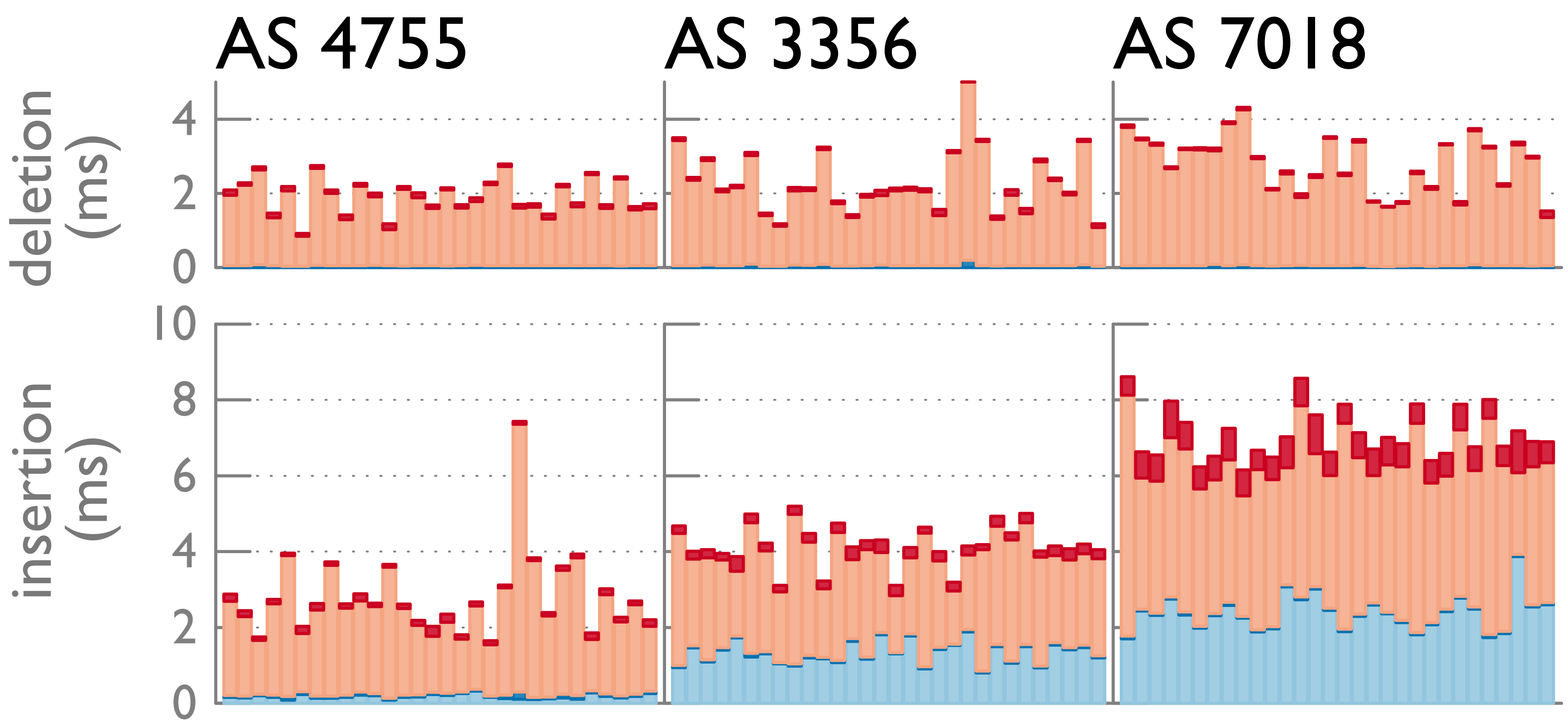


profile end to end delay (normalized per-rule, 30 rounds)
for route insertion and deletion

Rocketfuel ISP topology

AS#	nodes	links
4755	142	258
3356	1772	13640
7018	25382	11292

evaluation



Rocketfuel ISP topology

AS#	nodes	links
4755	142	258
3356	1772	13640
7018	25382	11292

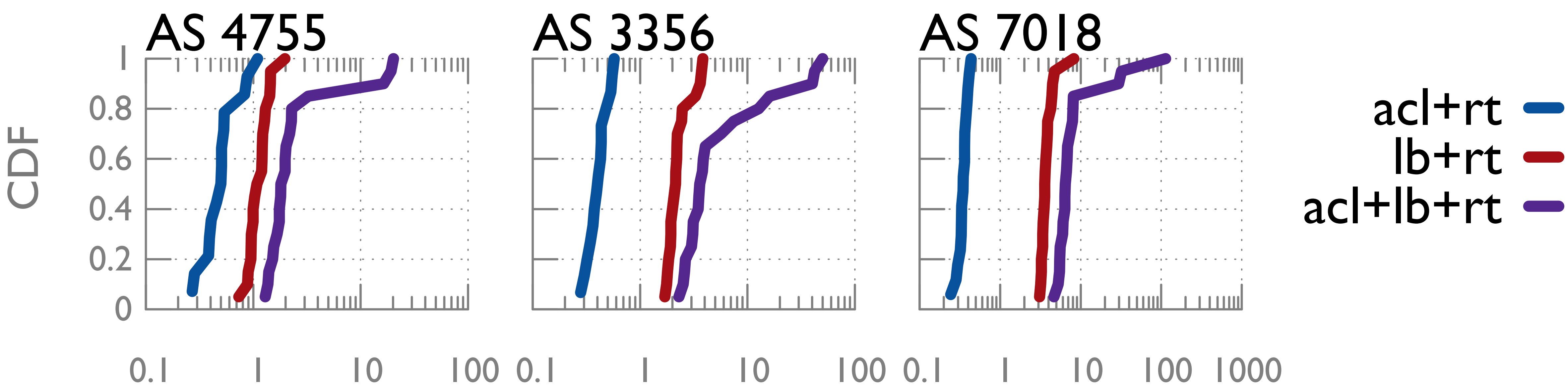
compute path
lookup ports
write to table
trigger/rule

profile end to end delay (normalized per-rule, 30 rounds)
for route insertion and deletion

similar profile on fat-tree topology (fewer nodes, more links)

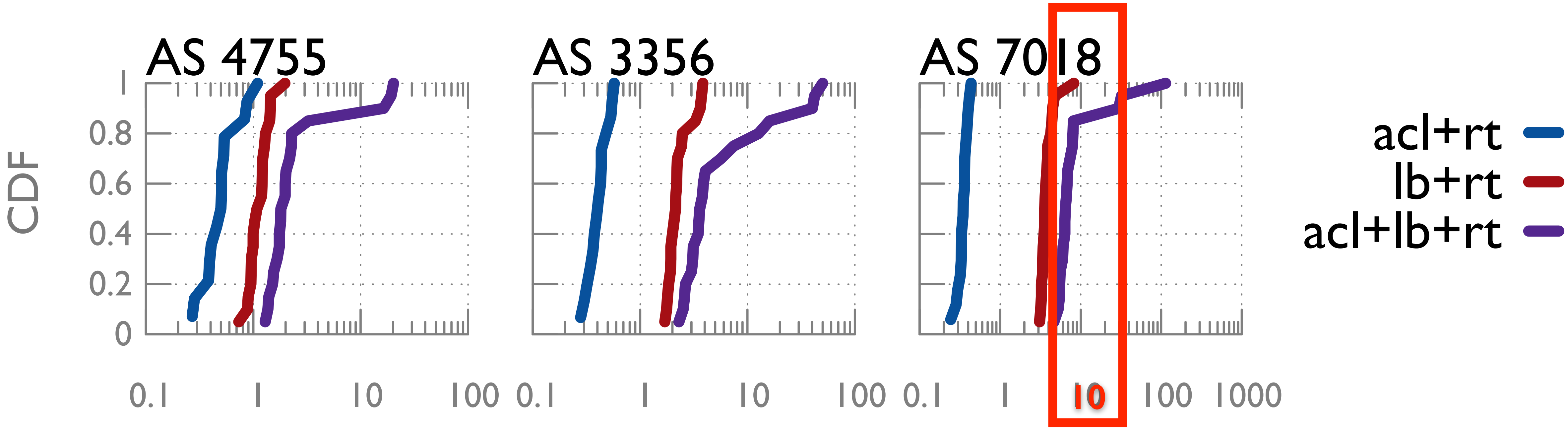
- total delay < 30ms for fat-tree with 5120 switches and 196608 links

evaluation



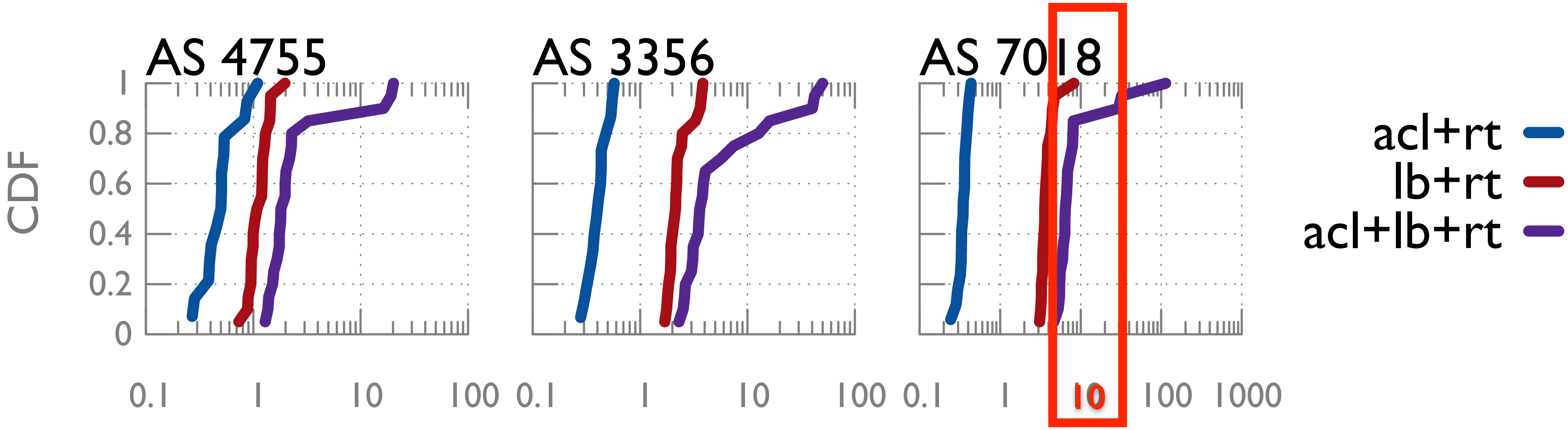
orchestration delay (ms) normalized per-rule for 3 scenarios:
access control and routing (acl+rt), load balancing and routing (lb+rt), access control, load balancing, and routing (acl+lb+rt)

evaluation



orchestration delay (ms) normalized per-rule for 3 scenarios:
access control and routing (acl+rt), load balancing and routing (lb+rt), access control, load balancing, and routing (acl+lb+rt)

evaluation



orchestration delay (ms) normalized per-rule for 3 scenarios:
access control and routing (acl+rt), load balancing and routing (lb+rt), access control, load balancing, and routing (acl+lb+rt)

orchestration also scales gracefully on fat-tree

- < 30ms for fat-tree with 5120 switches and 196608 links demo

demo

towards a secure Ravel

improper modification of data

- unauthorized modification
- one-directional information flow

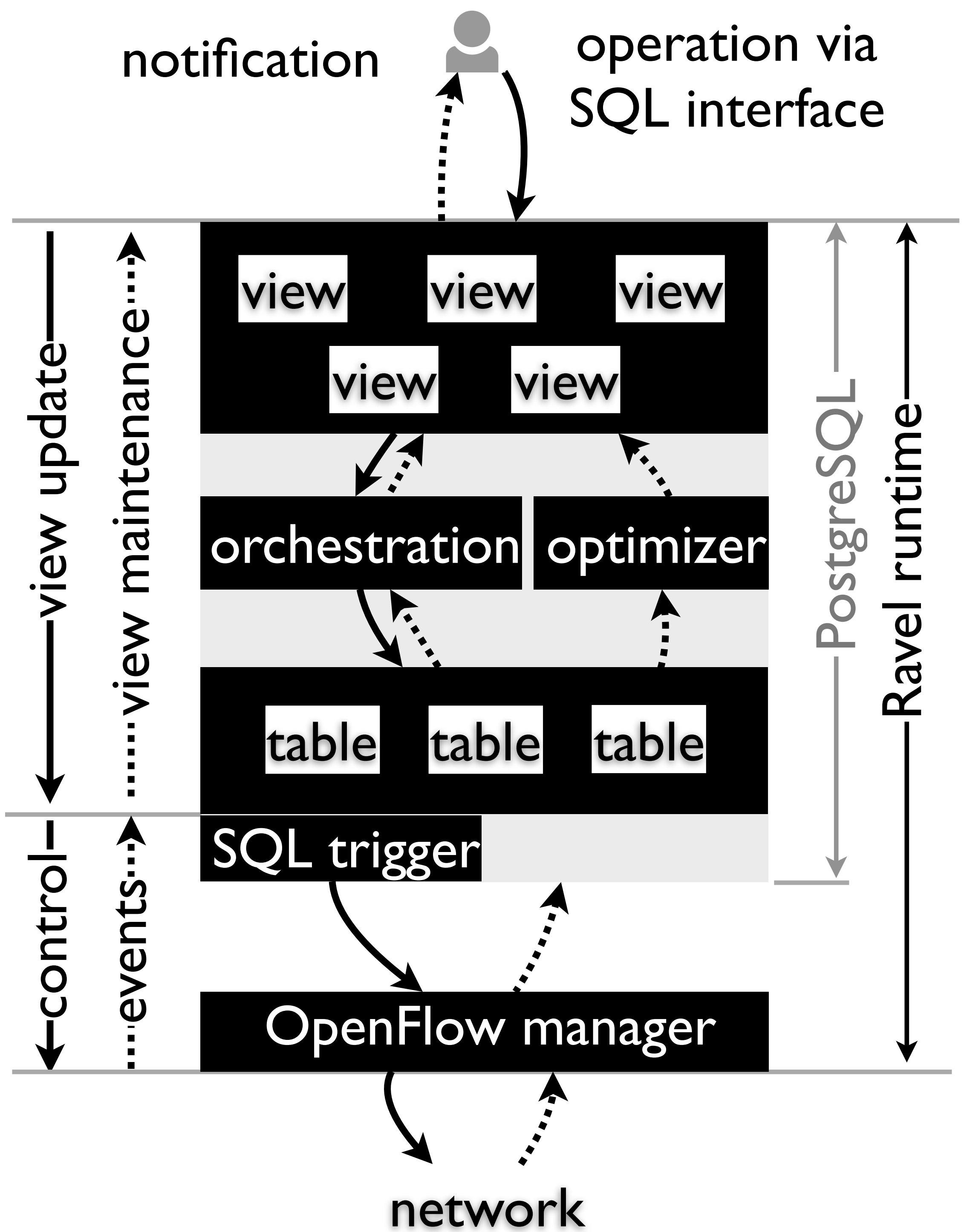
towards a secure Ravel

expectation of data quality

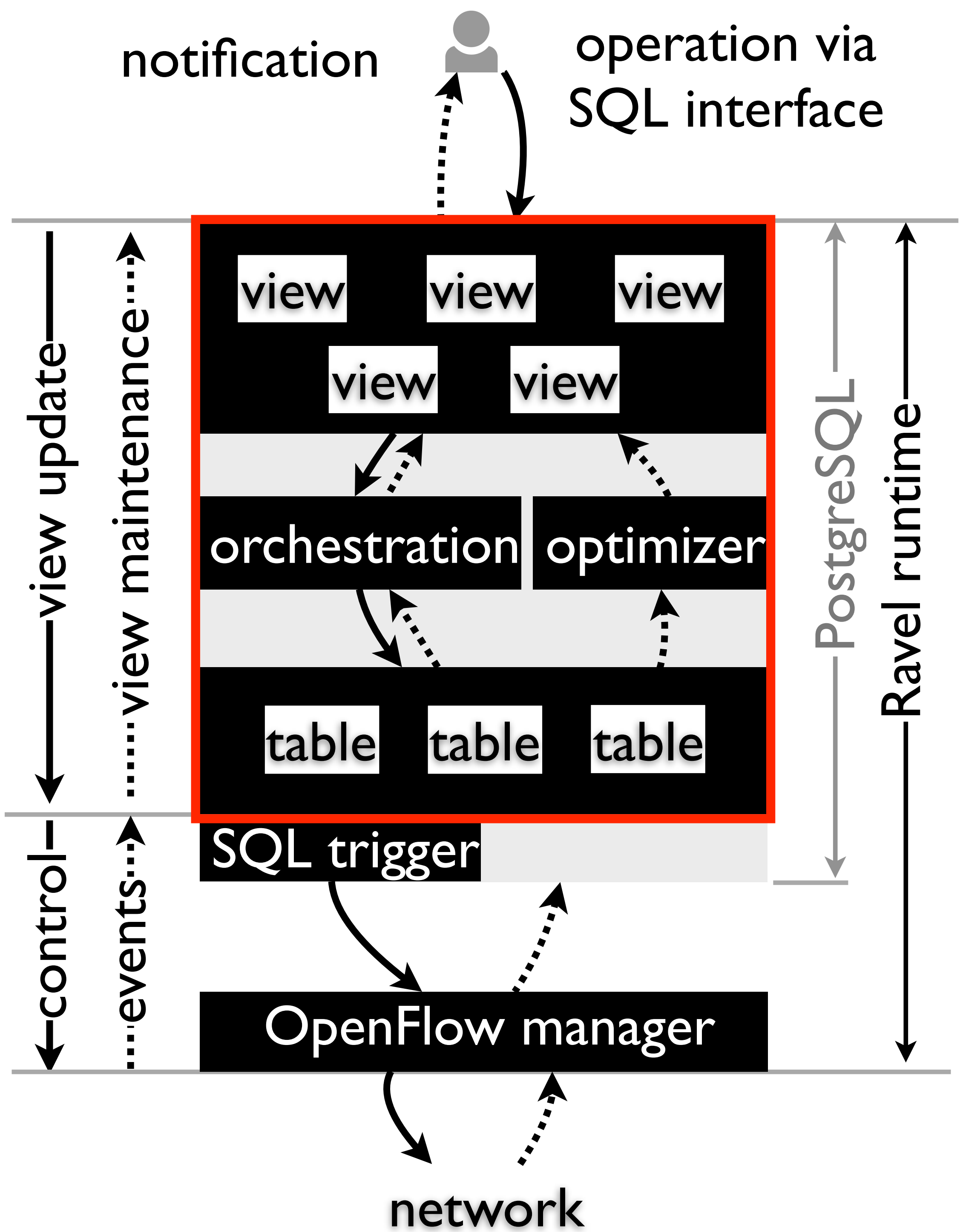
improper medication of data

- unauthorized modification — access control (ACL)
- one-directional information flow

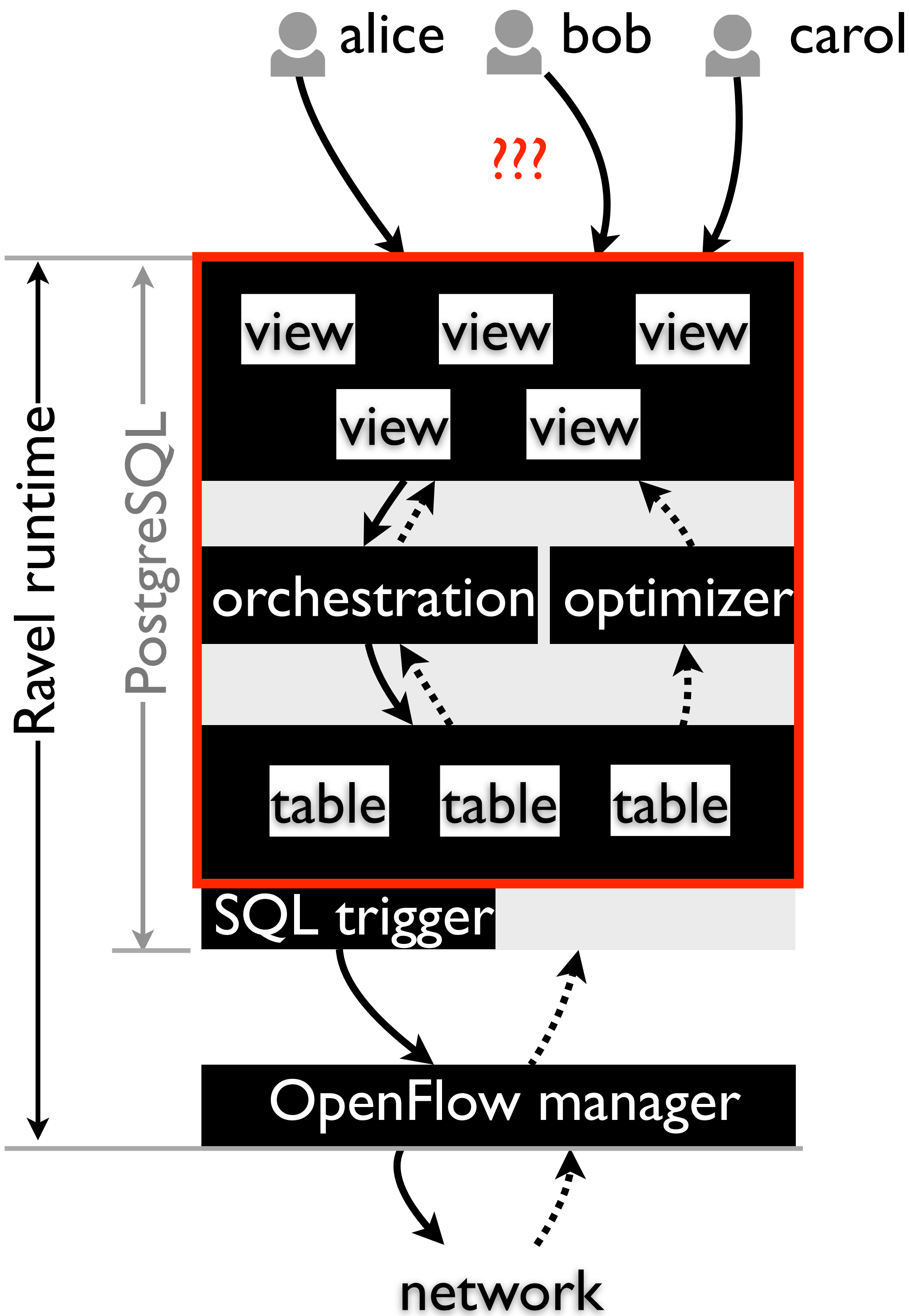
ACL in Ravel


notification
operation via
SQL interface
view
view
view
view
view
orchestration
optimizer
table
table
table
SQL trigger
OpenFlow manager
network
view update
view maintenance
control
events
PostgreSQL
Ravel runtime

ACL in Ravel


notification
operation via
SQL interface
view
view
view
view
view
orchestration
optimizer
table
table
table
SQL trigger
OpenFlow manager
network
view update
view maintenance
control
events
PostgreSQL
Ravel runtime

ACL in Ravel



alice
bob
carol
???
view
view
view
view
view
orchestration
optimizer
table
table
table
SQL trigger
OpenFlow manager
network
Ravel runtime
PostgreSQL

example scenario

a SDN network and multiple tenants

- admin can see/modify all resources, see/modify the network
- tenants can only see the resources they pay
- tenants can only manage their portions of network under contract

SLA (service level agreement)

tenant	switches	rate limit	connectivity
alice	{1,2,3,4}	20	{alice}
bob	{51,52,53,…}	50	{bob, alice}
carol	{100,101,…}	10	{carol, alice}

explicit access control list (ACL)

<principal, subject, operation>

ACL on topology

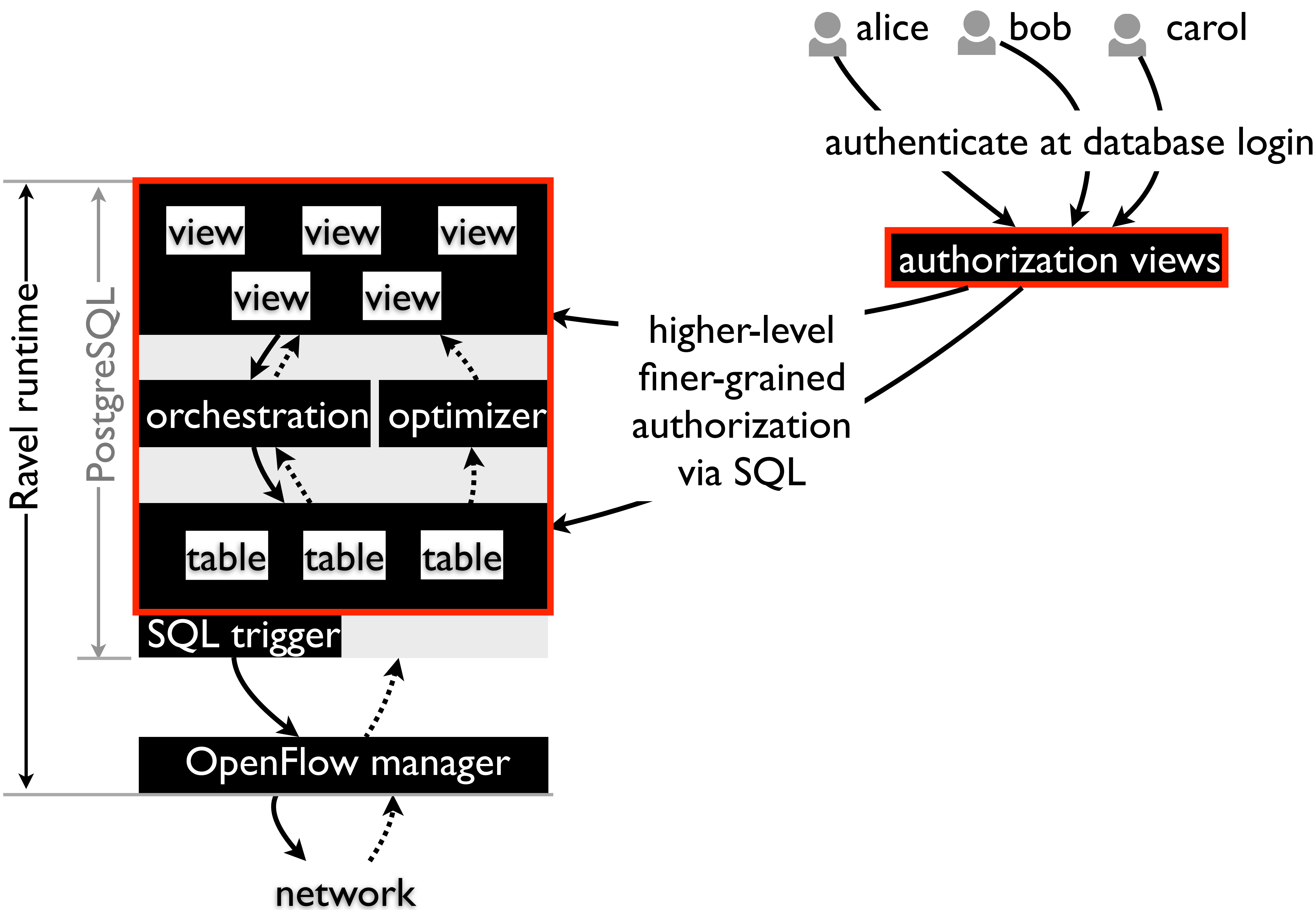
users	switches	privilege
alice	1	read
alice	2	read
alice	…	read
bob	…	read
carol	…	read
admin	…	read,write
…	…	…

ACL on configuration

users	flows (source, destination, rate)	privilege
alice	(1,2,<20)	read,write
alice	(2,3,<20)	read,write
alice	…	…
bob	…	…
…	…	…

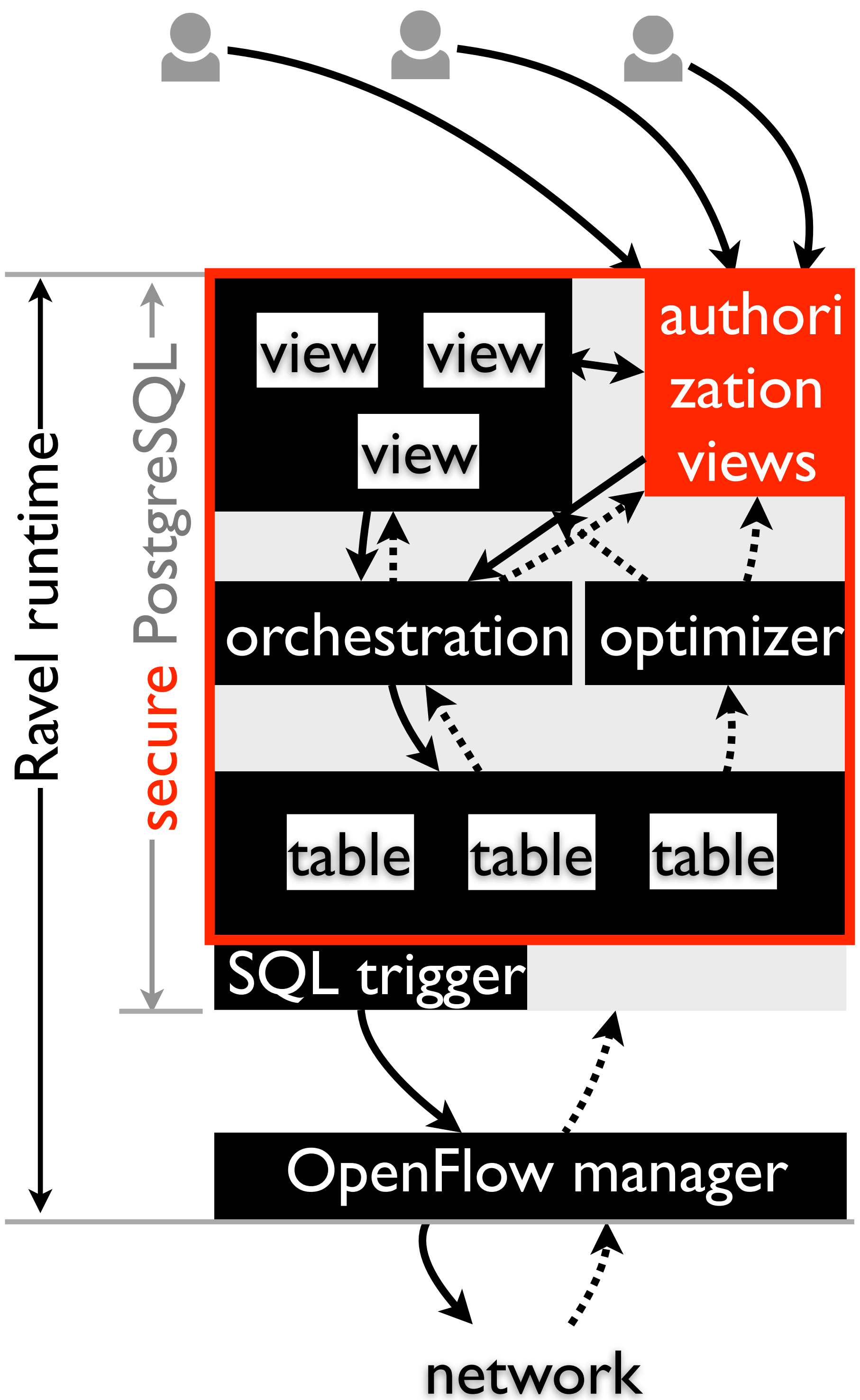
- very low-level
- update ACL as tenant contract evolves

ACL in Ravel



alice
bob
carol
authenticate at database login
authorization views
higher-level
finer-grained
authorization
via SQL
Ravel runtime
PostgreSQL
view
view
view
view
view
orchestration
optimizer
table
table
table
SQL trigger
OpenFlow manager
network

ACL in Ravel



Ravel runtime
secure PostgreSQL
view
view
view
authorization views
orchestration
optimizer
table
table
table
SQL trigger
OpenFlow manager
network

authorization views: a strawman solution

associate each table with an ACL

- <principal, allowed operation>

create a separate view

- if only a portion of a table is granted to a principal
- benefit: dynamic, content-based

authorization views: a strawman solution

```
-- admin policy
GRANT SELECT, UPDATE, INSERT, DELETE ON topology TO admin;
GRANT SELECT, UPDATE, INSERT, DELETE ON configuration TO admin;
```

```
-- alice policy
CREATE OR REPLACE VIEW topology_alice AS (
       SELECT sid, nid FROM topology
       WHERE (topology.sid = 1 OR topology.sid = 2 OR …);

CREATE OR REPLACE VIEW configuration_alice AS (
       SELECT fid, sid, nid FROM configuration
       WHERE ((topology.sid = 1 AND topology.nid = 2) OR
              (topology.sid = 1 AND topology.nid = 2) OR …) AND
              rate < 20);

GRANT SELECT ON topology_alice TO alice;
GRANT SELECT, INSERT, DELETE, UPDATE ON configuration_alice TO alice;
```

```
-- bob policy, carol policy ...
```

limitations

many tenants

- for each tenant, create a separate view?

dynamic tenant membership

- add/remove views?

SLAs evolving

- update tenant views?

more examples:

- tenants can only access the resources the pay
- raise tenant rate limit to100

finer-grained, higher-level ACL

capture the intent rather than extent

dynamic, context-based

SQL query over data
in p *and other parts* of
the network database

a network table of arity n
p(_,_,…,_)

→

access control view of n+1 arity
p_acl (principal, _,_,…,_)

finer-grained, higher-level ACL

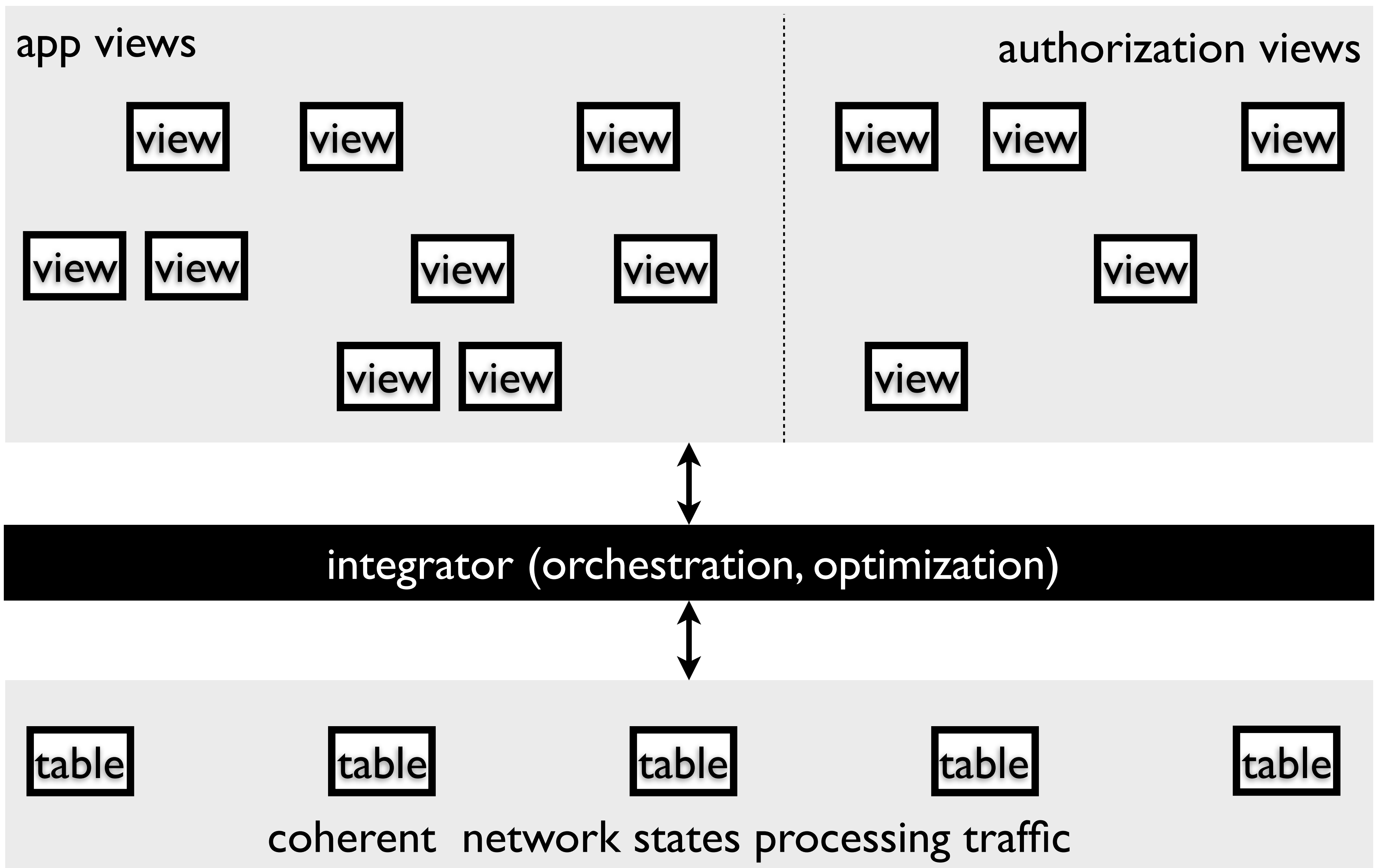
- a tenant can only access the leased network topology
- admin can access the whole topology

```
CREATE VIEW topology_acl AS (
      -- admin policy
      (SELECT 'admin' as principal,
       sid, nid
       FROM topology)
      UNION

      -- tenant policy
      (SELECT tenant as principal,
             sid, nid
       FROM topology, SLA
       WHERE topology.sid IN SLA.switches
             AND topology.nid IN SLA.switches));
```

```
CREATE VIEW topology_public AS (
       SELECT sid, nid FROM topology_acl
       WHERE principal = 'current_user')

GRANT SELECT ON topology_public TO public;
```

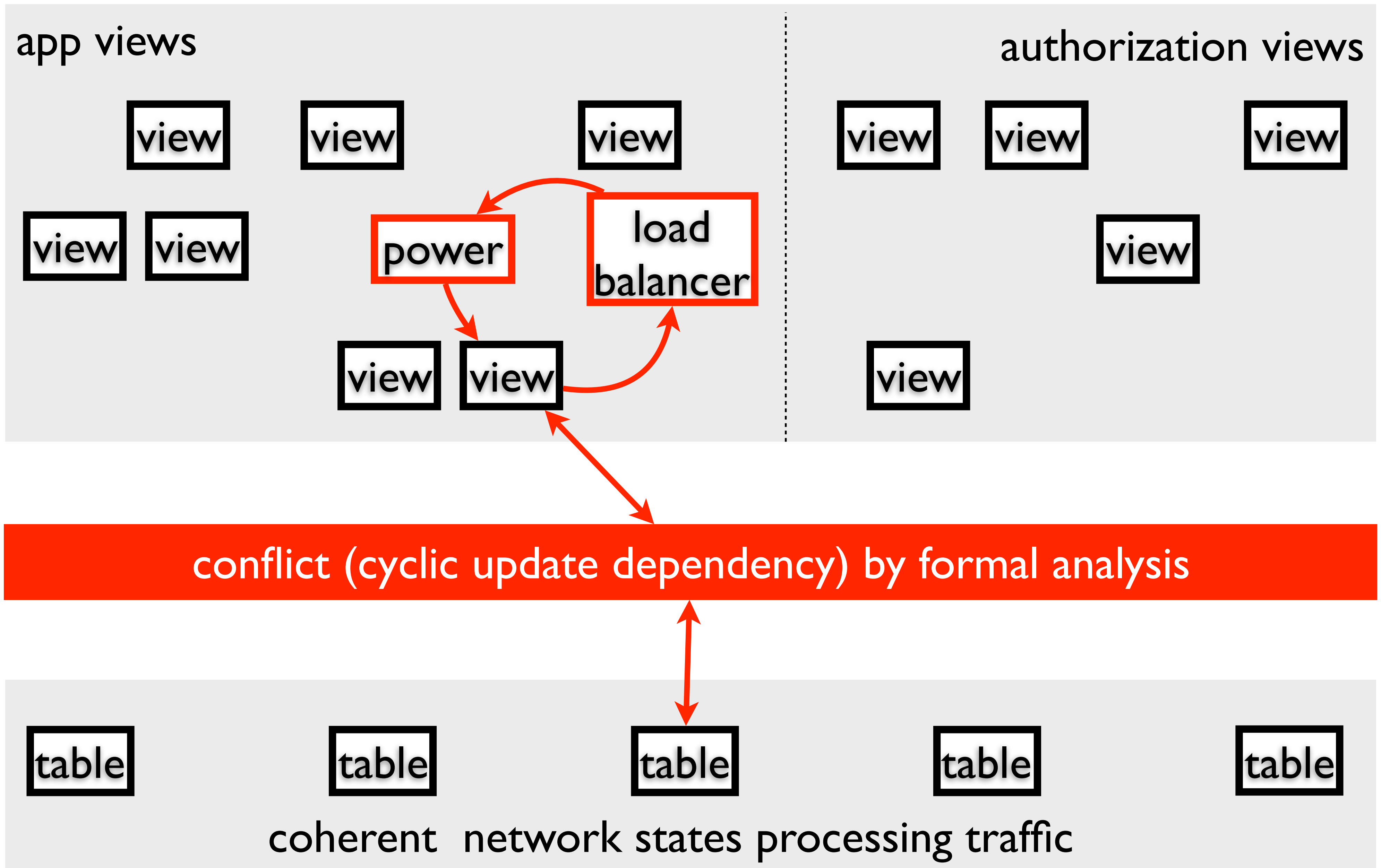
looking forward

data integration as a networking service



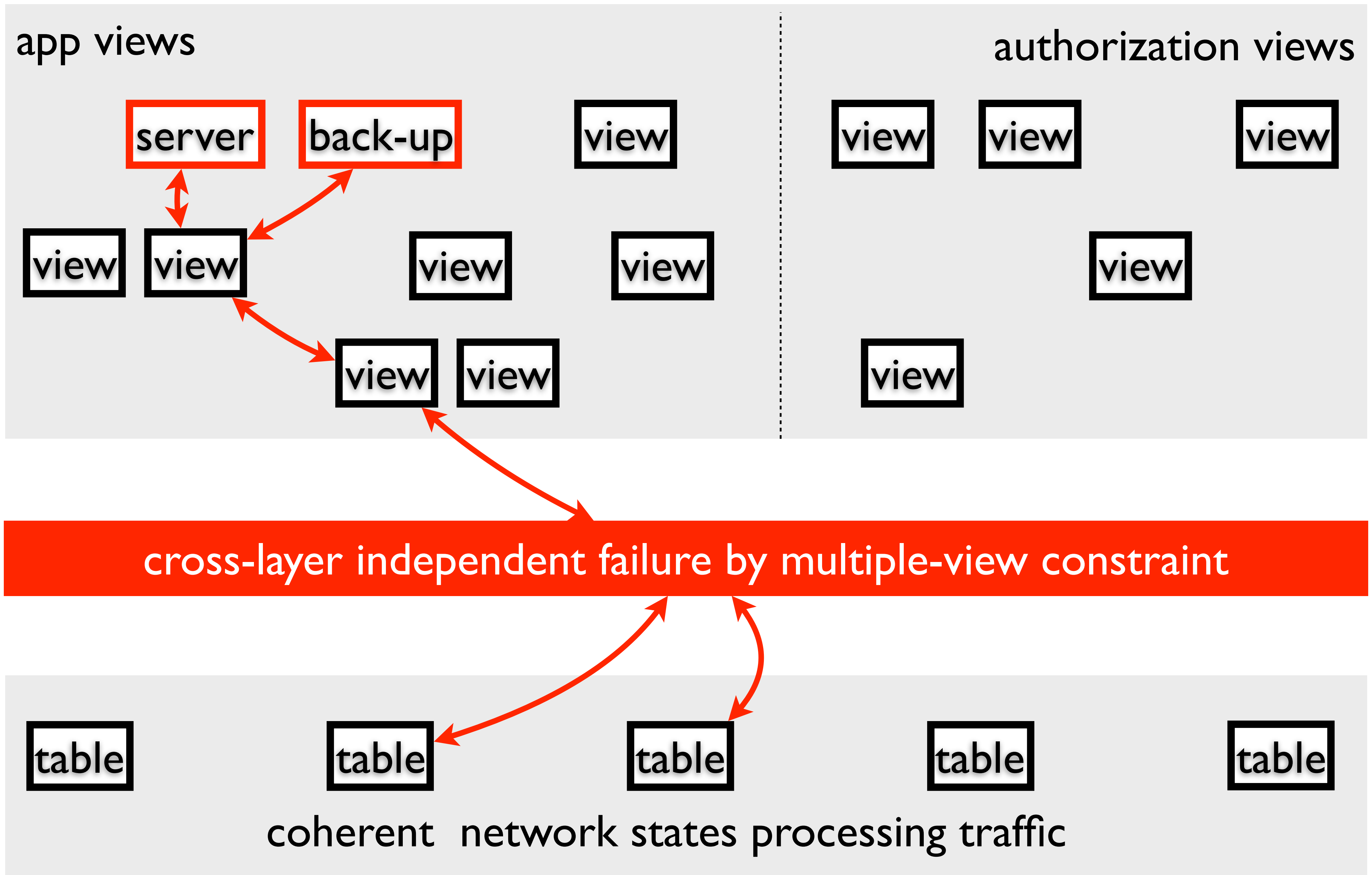
looking forward

data integration as a networking service



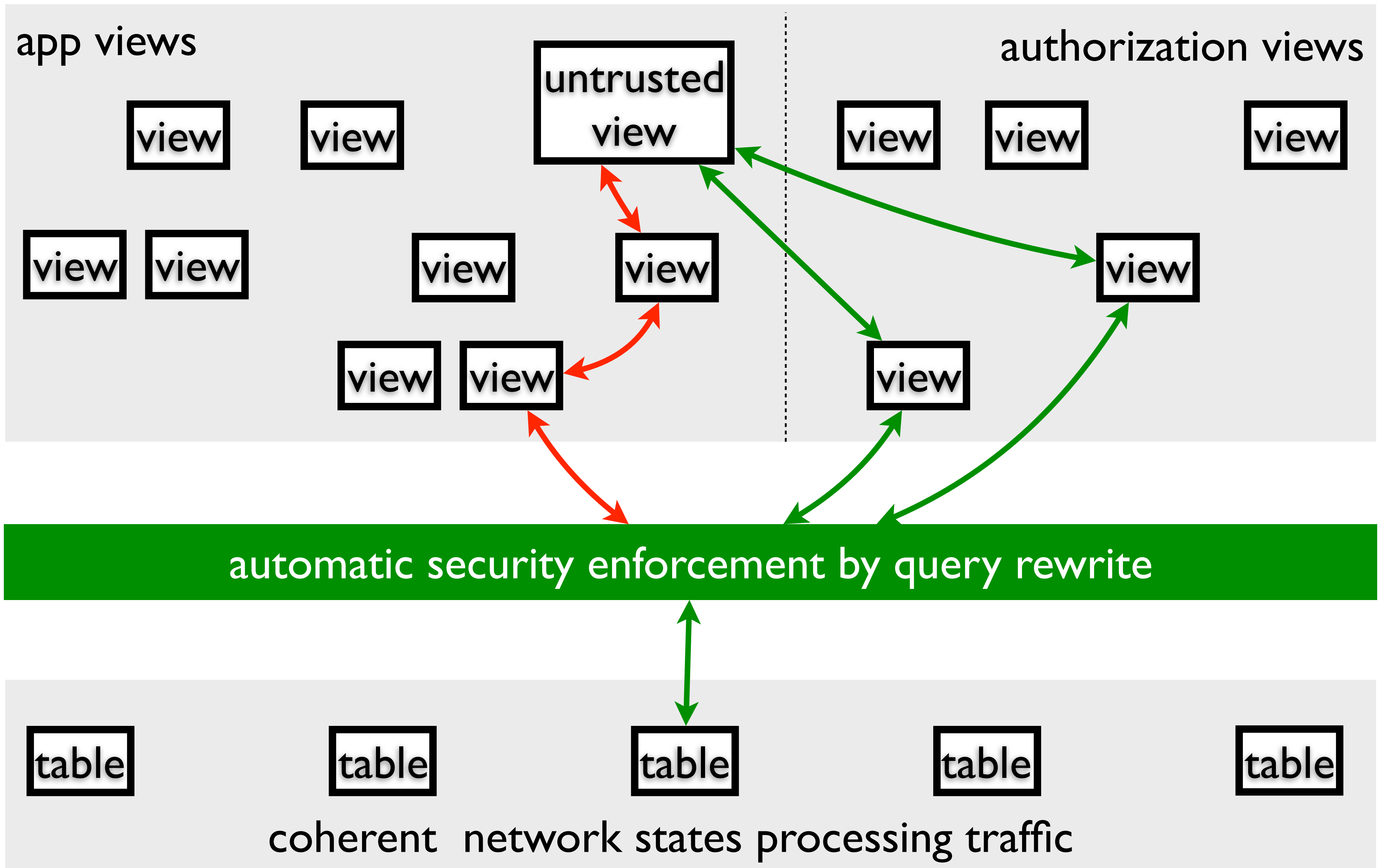
looking forward

data integration as a networking service

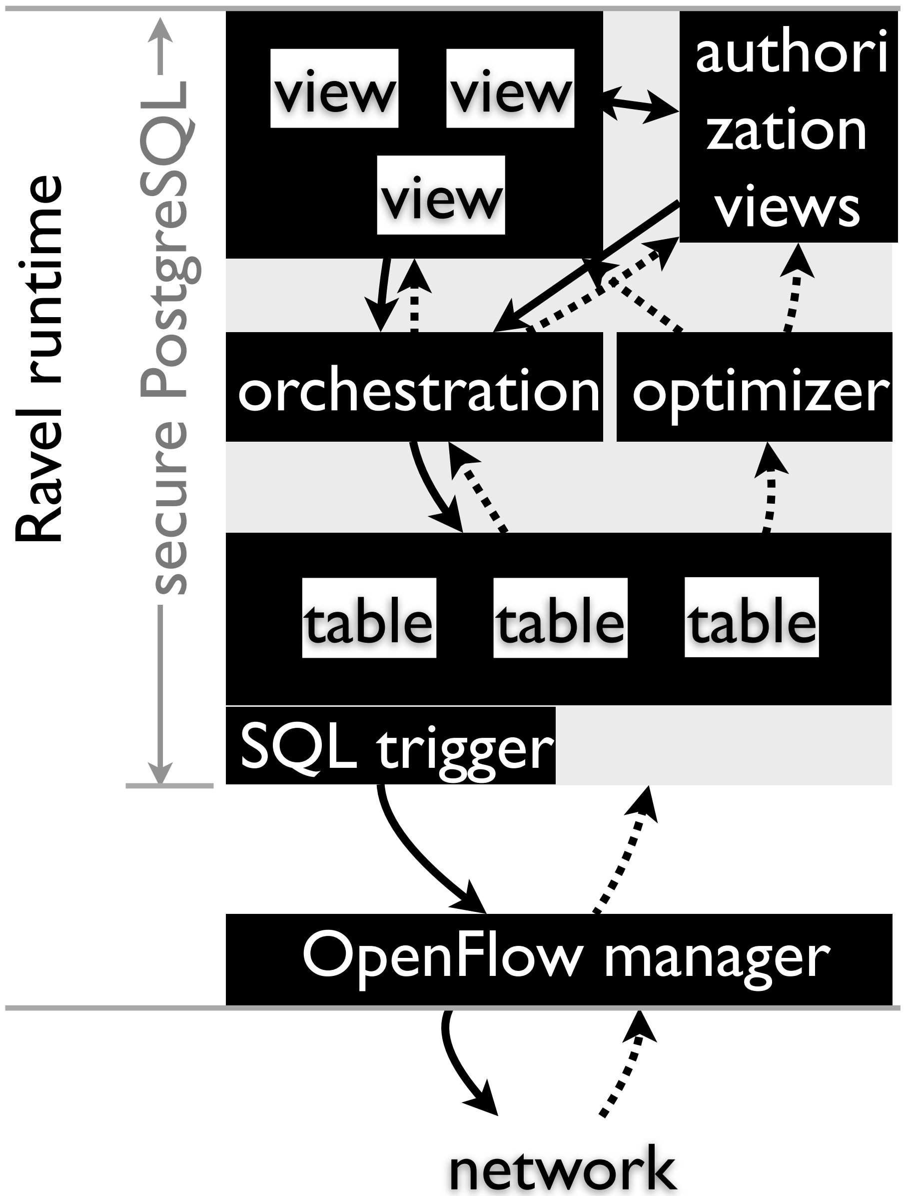


looking forward

data integration as a networking service



conclusion



this talk: via SQL

- orchestratable abstraction
- finer-grained access control

looking forward

- data integration as a networking service

playtime

download *Ravel*

ravel-net.org/download

start playing: tutorials, add your own app

ravel-net.org

explore more

github.com/ravel-net