北京郵電大學
BEIJING UNIVERSITY OF POSTS AND TELECOMMUNICATIONS

网络与交换技术国家重点实验室
State Key Laboratory of Networking and Switching Technology

Service Failure Diagnosis in Service Function Chain

Shilei Zhang, Ying Wang, Wenjing Li, Xuesong Qiu,

Content

Introduction-Background

Define

- Service Function Chain (SFC): defines an ordered set of abstract service functions and ordering constraints that must be applied to packets and/or frames and/or flows selected as a result of classification.
- Service Failure : Service availability failure and Service quality degradation failure.

Advantage

- Reduce the traffic of active detection
- Using network service header to locate service failure

Introduction-Challenges & Contributions

Heavy traffic of active detection

- Solution: Improved Depth First Traversal
- Target: reduce the traffic of active detection

Multi-types and multiple-layers failures

- Solution: Hierarchical and classification failure location algorithm
- Target: Accurately locate the source of the failure

Inspiring Scenario

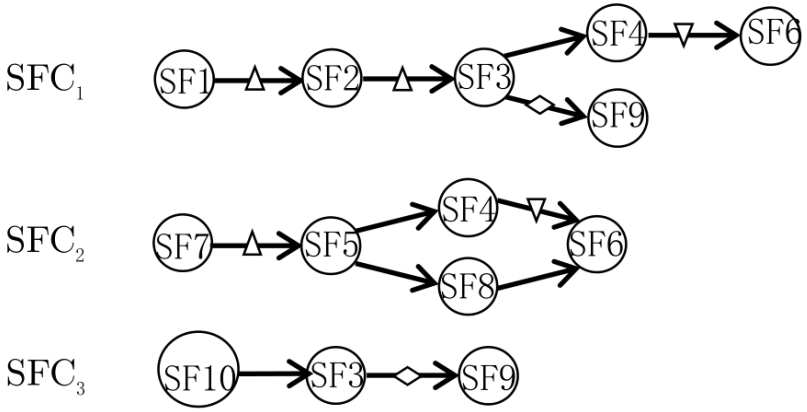


Fig. 1. Service Function Chains

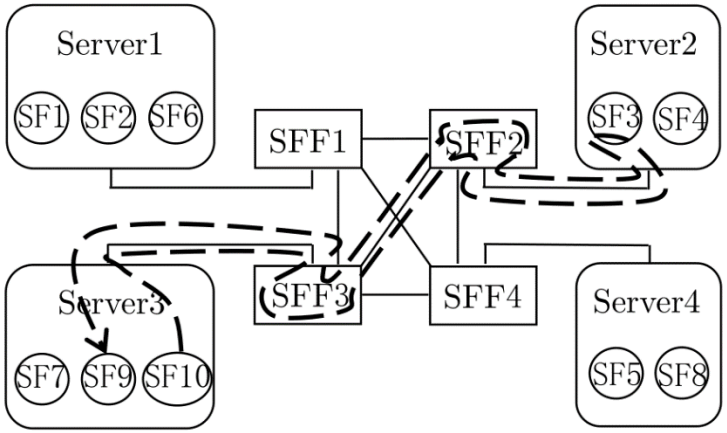


Fig. 2. Service Functions Deployment Graph

- **SFC 1 has the same starting point and different end points with a branching structure. SFC 2 has the same starting point and end point with a branching structure. SFC 3 is a line composed of service functions between the two endpoints.**
- **Figure 2 presents the deployment of service function chains in a network.**

Multi-layer Fault Diagnosis Method

- **Improved Depth First Traversal**
- **Hierarchical and classification failure location**

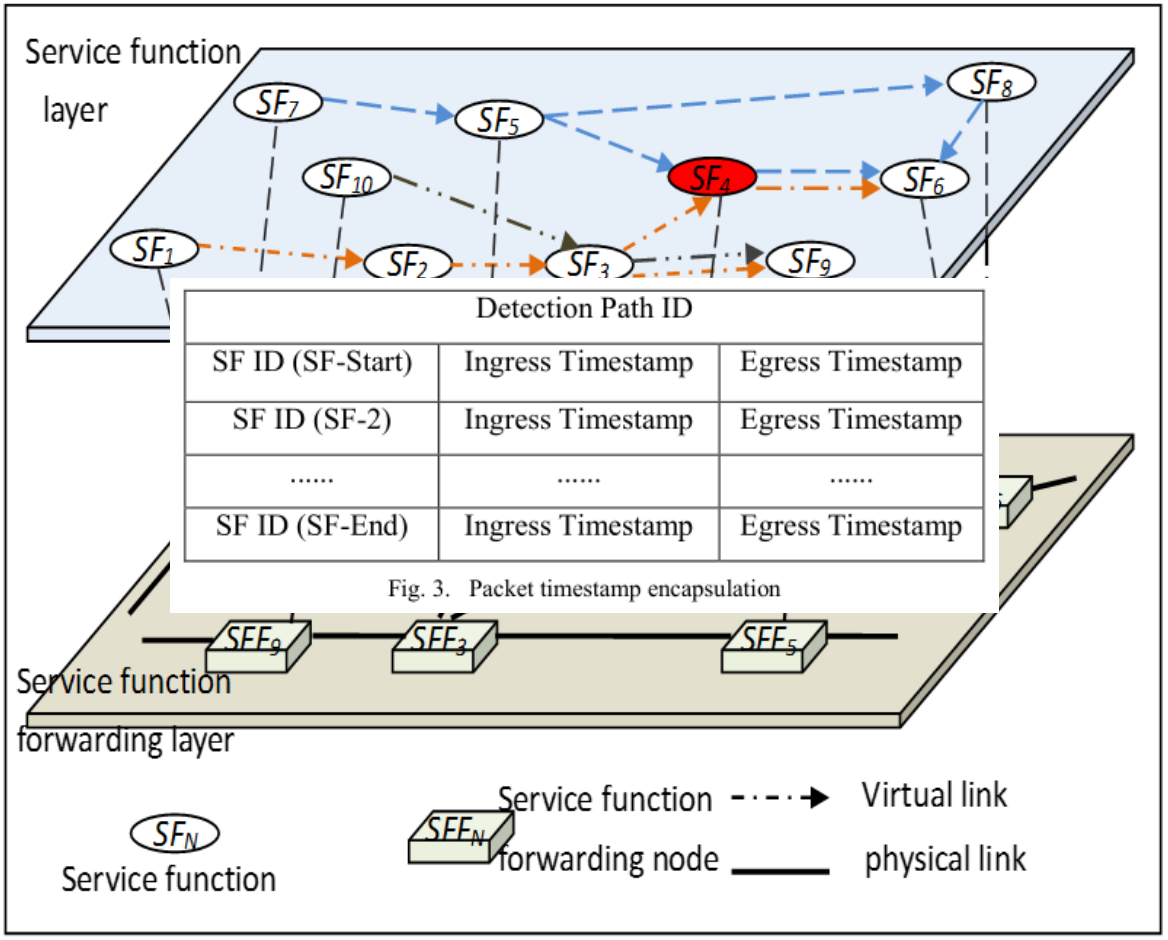
Improved Depth First Traversal

Algorithm 1 Improved Depth First Traversal (IDFT)

Input: v, **M**
Output: A Detection Path (DP)
Procedure:

1: DP ←null, add v to DP
2: select $\forall\ v' \in$ V, and **M**[v][v'].selected = **false**
3: **M**[v][v'].selected = **true**, add v' to DP
4: Remove the link between v and v' from E
5: v.outdegree--, v'.indegree--, S = **M**[v][v'].SFP
6: **for** v'' in V
7: **if M**[v'][v''].connected & !**M**[v'][v''].selected
8: **if** (S ∩ **M**[v'][v''].SFP != null)
9: **S** = S ∩ **M**[v'][v''].SFP
10: **break**
11: **end if**
12: **end for**
13: **if** $v'' \neq$ null
14: v=v', v'=v'', **go to** step 3.
15: **end if**

Hierarchical and classification failure location



Detection Path ID		
SF ID (SF-Start)	Ingress Timestamp	Egress Timestamp
SF ID (SF-2)	Ingress Timestamp	Egress Timestamp
.....	.....	.....
SF ID (SF-End)	Ingress Timestamp	Egress Timestamp

Fig. 3. Packet timestamp encapsulation

Hierarchical and classification failure location

Algorithm 2 service quality degradation localization

Input: DP_{Normal}, DP_{SQD}, NSH

Output: Set of service quality degradation S

Procedure:

init: SQS_{map} < key, value > =null , SQS = null

01 for(Path p : DP_{SQD})

02 for(Element e : p)

03 if(e is SF && e.time> $T_{sfnormal}$) S.add(e) ;

04 if(e is a virtual link && e.time> $T_{vlnormal}$) SQS.add(e) ;

05 for(Element e : SQS)

06 mapping e into the Set Q

07 for(Element q : Q)

08 if(SQS_{map}.get(q)==null) SQS_{map}.put(q,1);

09 else SQS_{map}.put(q,SQS_{map}.get(q)+1) ;

10 sort(SQS_{map});

11 Iterator iter= SQS_{map}.entrySet().iterator()

12 while(iter.hasNext()&&DP_{SQD}!=null)

13 entry=iter.next(); Element e=entry.getKey();

14 if(e is SFF) find the set SFQ that are connected with e

15 if(∀sf ∈ SFQ, DP_{Normal}.*contains*(sf)) continue;

16 else S.add(e) ;

DP_{SQD}.remove(paths that contain the sf);

17 if(e is a link between SFFs)

find the virtual link set VLQ which are related to e

18 if(∀vl ∈ VLQ, DP_{Normal}.*contains*(vl)) continue;

19 else S.add(e) ;

DP_{SQD}.remove(paths that contain the vl) ;

20 return the set S ;

Hierarchical and classification failure location

Algorithm 3 service availability failure localization

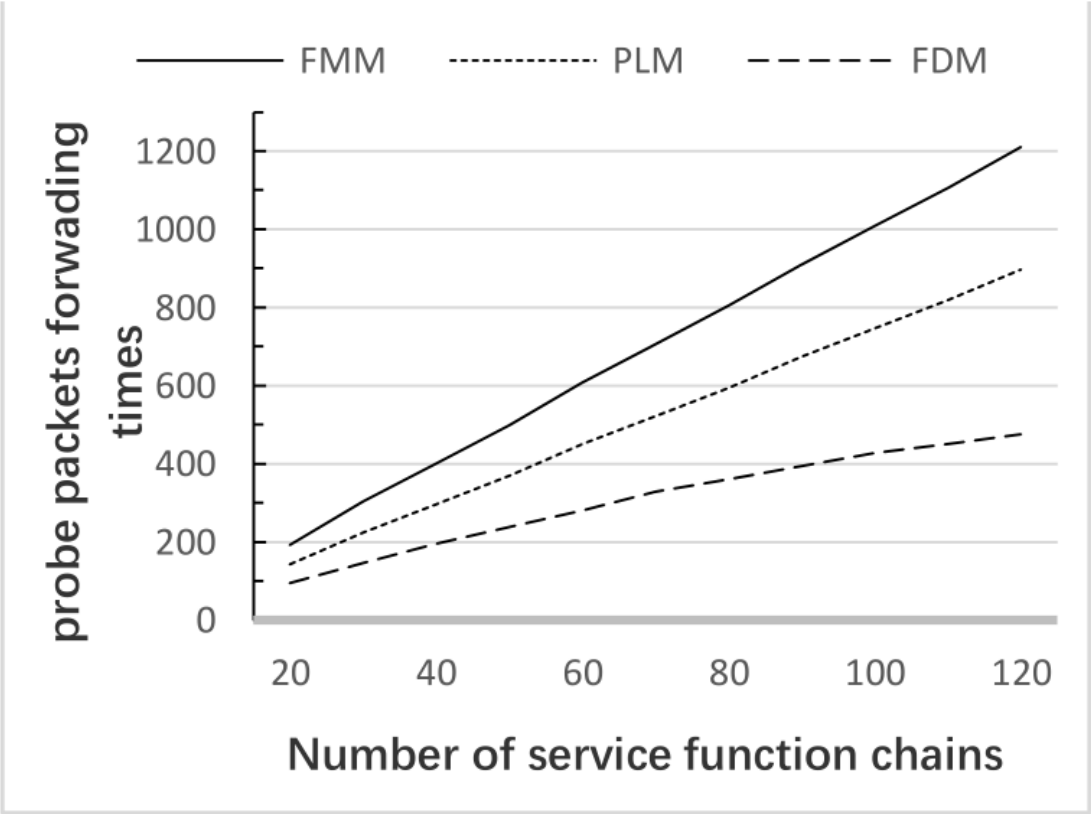
Input: DP_{Normal}, DP_{SAF}, DP_{SQD}

Output: Set of service availability S

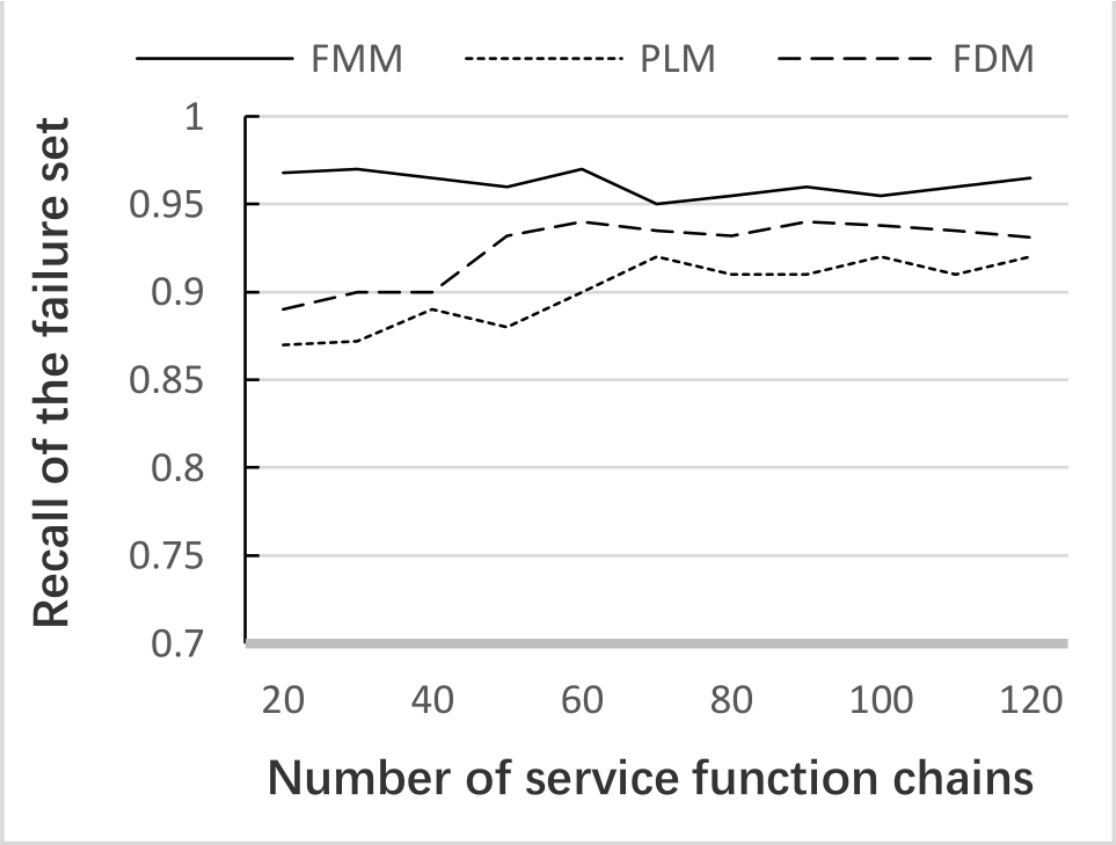
Init: $SAS_{map} < key, value > = null$, $SAS = null$

```
1  for(Path p : DP_SAF)
2    for(Element e : p)
3      if(!DP_Normal.contains(e) && !DP_SQD.contains(e))
4        SAS.add(e);
5  for(Element e : SAS)
6    if(e is SF)   find the SFF which is connected with e
7      if(SAS_map.get(e) == null)  SAS_map.put(e,1);
8      else  SAS_map.put(e,SAS_map.get(e)+1);
9      if(SAS_map.get(SFF) == null)  SAS_map.put(SFF,1);
10     else  SAS_map.put(SFF,SAS_map.get(SFF)+1);
11   if(e is virtual link) mapping e into the Set Q
12     for(Element q : Q)
13       if(SAS_map.get(q) == null)  SAS_map.put(q,1);
14       else  SAS_map.put(q,SAS_map.get(q)+1);
15 sort(SAS_map);
16 Iterator iter=SAS_map.entrySet().iterator()
17 while(iter.hasNext() && DP_SAF != null )
18   entry=iter.next(); Element e=entry.getKey();
19   if(e is SF) S.add(e);
20   if(e is SFF) find the set SFQ which are connected with e
21     if(∀sf ∈ SFQ, DP_Normal.contains(sf) || DP_SQD.contains(sf)) continue;
22     else  S.add(e), S.remove(∀sf ∈ SFQ);
23     DP_SAF.remove(paths which contain the sf in SFQ);
24   if(e is a link between SFFs) find the virtual link set VLQ which are related to e
25     if(∀vl ∈ VLQ, DP_Normal.contains(vl) || DP_SQD.contains(vl)) continue;
26     else  S.add(e); DP_SAF.remove(paths which contain the virtual link in VLQ);
27 return the set S;
```

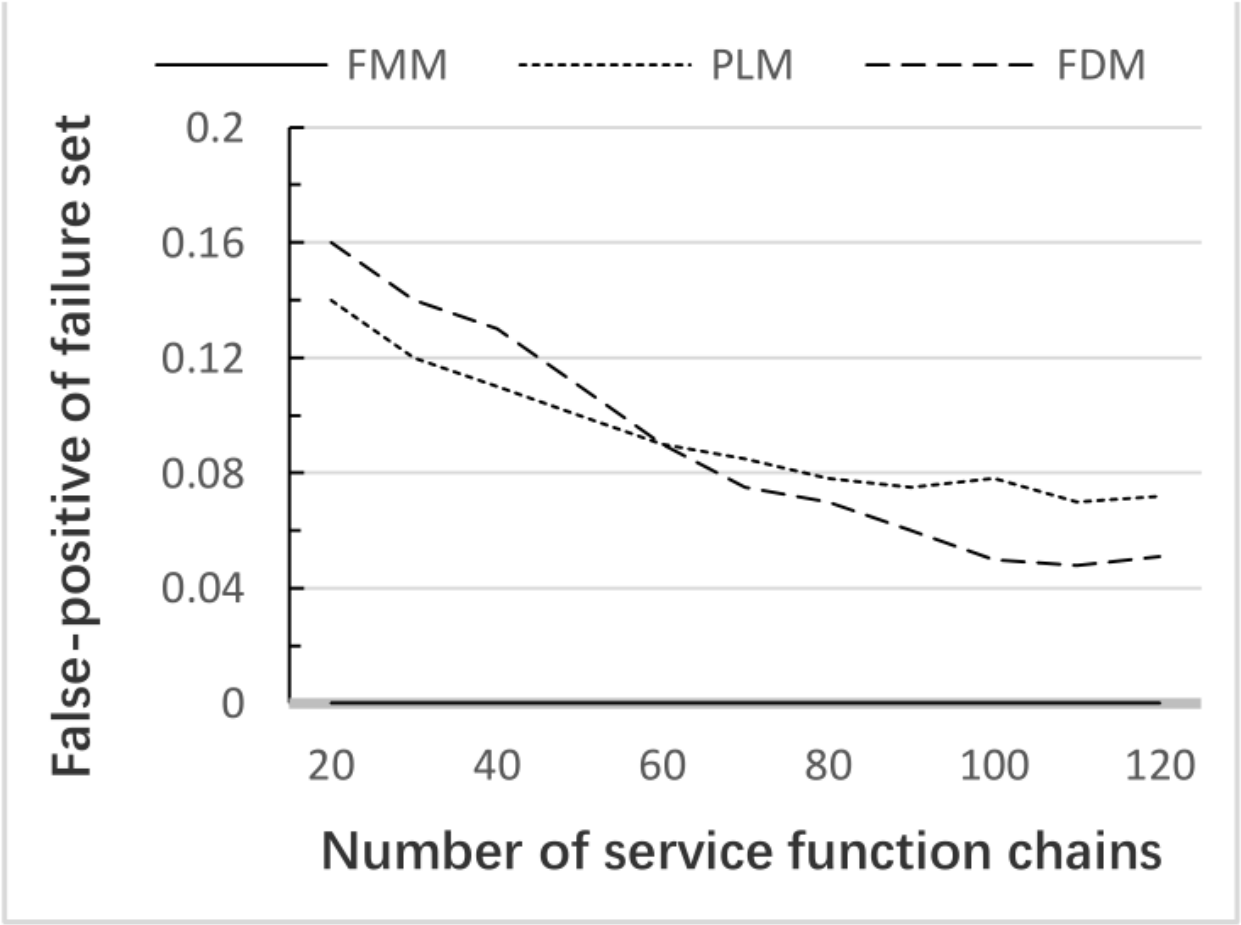
Simulation and Result Analysis(1)



Simulation and Result Analysis(1)



Simulation and Result Analysis(2)



Conclusions

The results of the experiments show that:

- Our improved depth first traversal method can significantly reduce the active detection cost.
- Our failure localization algorithm improves the recall and false positive by removing nodes with less probability of failure.

Q&A

best.lei@bupt.edu.cn