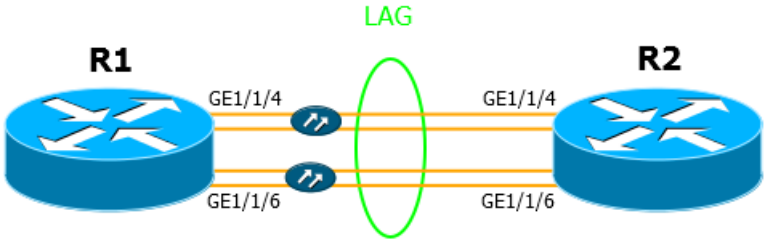
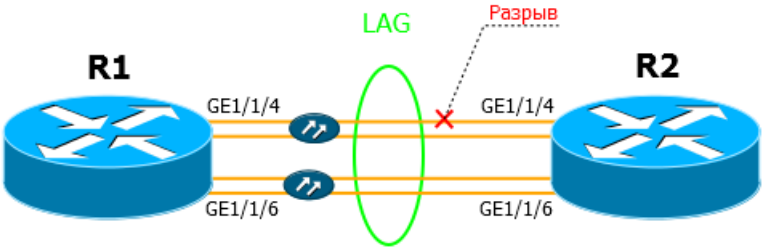
LAG и средства обнаружения проблем

Кому из нас не приходилось иметь дело с агрегацией каналов, настраивать LACP и медитировать на идеальную балансировку?

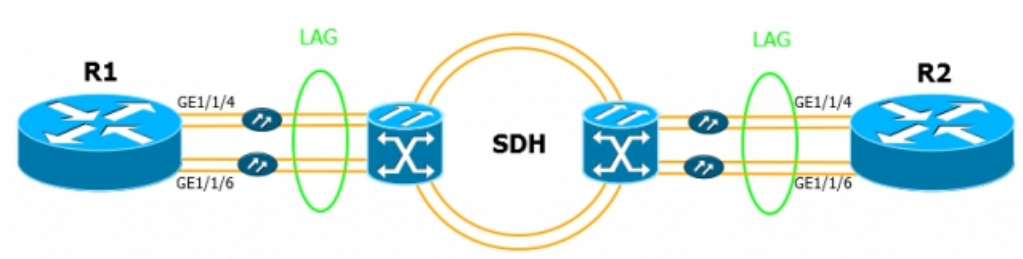


Всё здесь прекрасно, LAG обеспечивает резервирование – один линк падает, LACP удаляет его и работаем на оставшихся. Казалось бы. Но известная многим проблема в том, что LACP никак не отследит такую ситуацию:



Оптический порт перейдёт в состояние Down только если на его входе нет сигнала. То есть в вышеуказанном примере порт GE1/1/4 на R2 упадёт, а на R1 – нет. LACP на R2 отработает, а на R1 нет. Налицо потери трафика.

Для отслеживания таких проблем используются протоколы UDLD – UniDirectional Link Detection. К ним относятся BFD и Ethernet OAM. Оба они прекрасно справляются со своей задачей: если связь с удалённым устройством пропадает, UDLD вынуждает маршрутизатор перевести соответствующий порт в состояние Down. В случае, который я хочу рассмотреть , схема несколько более сложная:



То есть между двумя маршрутизаторами ещё сеть SDH. OSN принимает Ethernet-кадры, инкапсулирует их в SDH-фреймы и отправляет их в плавание. С другого конца другой OSN вылавливает их, декапсулирует обратно Ethernet и возвращает их назад в IP-сеть.

Инженер остановил свой выбор на Ethernet OAM для детектирования проблемы и… ошибся. Сам я прежде не сталкивался с данным протоколом, поэтому в лаборатории собрал тестовый стенд без SDH сети – всё работает. SDH не стал настраивать по той простой причине, что это транспорт – какая ему разница, что передавать? Резать и дропать он ничего не должен – что получил, то и отправил. У меня работает, у заказчика нет – начинаем углубляться в детали.

Для начала разберёмся, что такое OAM вообще и как применяется.

OAM – Operation, Administration and Maintenance

Существует их два вида:

EFM OAM – Ethernet in First Mile. Преследует следующие цели: обнаружение партнёра, мониторинг линка, уведомление об авариях, Remote Loopback. EFM OAM соответствует стандарту 802.3ah и ориентирован на отслеживание состояния простого линка.

Ethernet CFM – Connectivity Fault Management – 802.3ag. Это механизм масштаба сети и призван обеспечивать End-to-End связность. Это достаточно мощный протокол и теоретически может использоваться для данных целей, но это всё равно, что поднимать OSPF между своим DLink и компьютером. Можно, но зачем?

Итак, поскольку был выбран EFM OAM, то копнём его поглубже.

EFM определяется на конкретном интерфейсе и может работать в двух режимах – Active и Passive. Только Active инициирует поиск соседа путём отправки OAMPDU в линк. Passive же слушает и отвечает на пришедшие запросы, но не может инициировать поиск соседа. Таким образом, хотя бы один из двух подключенных друг к другу интерфейсов должен иметь EFM в режиме Active. Вот типичная конфигурация:

```
interface GigabitEthernet1/1/4
eth-trunk 22
efm enable
efm trigger if-down
```

Последняя команда вынуждает устройство переводить интерфейс в состояние Down при наличии проблем. А вот так выглядит обмен любезностями и успешное установление сессии:

```
14 12:25:30.297366   HuaweiTe_43:4f:ef  Slow-Protocols  OAM   60 OAMPDU: Information
15 12:25:32.296809   HuaweiTe_43:4f:ef  Slow-Protocols  OAM   60 OAMPDU: Information
19 12:25:33.296875   HuaweiTe_43:4f:ef  Slow-Protocols  OAM   60 OAMPDU: Information
23 12:25:34.324253   HuaweiTe_43:4f:ef  Slow-Protocols  OAM   60 OAMPDU: Information
28 12:25:34.441878   HuaweiTe_43:4f:af  Slow-Protocols  LACP 124 Link Aggregation Control Protocolversion 1.
29 12:25:34.442671   HuaweiTe_43:4f:ef  Slow-Protocols  LACP 124 Link Aggregation Control Protocolversion 1.

⊞ Frame 15: 60 bytes on wire (480 bits), 60 bytes captured (480 bits)
⊞ Ethernet II, Src: HuaweiTe_43:4f:ef (28:6e:d4:43:4f:ef), Dst: Slow-Protocols (01:80:c2:00:00:02)
⊟ OAM Protocol
    Slow Protocols subtype: OAM (0x03)
  ⊞ Flags: 0x0050 (local: Discovery complete, remote: Discovery complete)
    OAMPDU code: Information (0x00)
  ⊞ Type: Local Information TLV (0x01)
  ⊞ Type: Remote Information TLV (0x02)
```

И вот тут, глядя на вид OAMPDU, я начинаю подозревать, что наличие SDH-сети в промежутке может играть фатальную роль. В качестве адреса отправителя стоит MAC-адрес Eth-trunk интерфейса, а вот в качестве получателя, какой-то специальный MAC-адрес, и EtherType – Slow Protocol.

Протоколы управления Ethernet – это часть стандарта 802.3. Они разделяются на два типа:

- Быстрые. Они должны отрабатывать моментально для предотвращения потери производительности. Как правило они реализуются аппаратно. Примером может служить механизм PAUSE – когда порт устройства получает трафика больше, чем может обработать, он отсылает PAUSE Frame противоположному узлу с просьбой понизить скорость отправки.
- Медленные – те самые Slow Protocols. У них не такие большие аппетиты на частоту отправки и задержки. Они реализуются программно. OAM и LACP относятся именно к этому виду. Для них используется специальный MAC-адрес: – 0180-c200-0002, и EtherType 8809.

Table 57A–3—Slow Protocols subtypes

Protocol Subtype value	Protocol name
0	Unused—Illegal value
1	Link Aggregation Control Protocol (LACP)
2	Link Aggregation—Marker Protocol
3	Operations, Administration, and Maintenance (OAM)
4	Reserved for future use
5	Reserved for future use
6	Reserved for future use
7	Reserved for future use
8	Reserved for future use
9	Reserved for future use
10	~~Reserved for future use~~ Organization Specific Slow Protcol (OSSP)
11–255	Unused—Illegal values

Выдержка из Annex 57A к стандарту 802.3:

This address is within the range reserved by ISO/IEC 15802-3 (MAC Bridges) for link-constrained protocols. As such, frames sent to this address will not be forwarded by conformant MAC Bridges; its use is restricted to a single link

Дело в том, что EFM OAM относится к группе протоколов, скажем так, "одного линка". Их данные не могут покинуть один простой линк. Как только противоположное устройство принимает такой фрейм, оно его обрабатывает нужным образом и уничтожает, не передавая никуда дальше. То есть, когда с обратной стороны у нас стоит маршрутизатор/коммутатор с настроенным EFM на интерфейсе, он, приняв OAMPDU, проверяет его и отправляет ответ на R1, а старый кадр уничтожает. R1 получает OAMPDU от R2 и сессия установлена в лучшем виде.

```
[R2]dis efm ses al
Interface                    EFM State        Loopback Timeout
-------------------------------------------------------------------------
GigabitEthernet1/1/4         detect                  --
GigabitEthernet1/1/6         detect                  --
```

В нашем же случае OSN, получив такой фрейм, просто отбрасывает его, потому что никакого EFM на нём не запущено.

```
[R1]dis efm session all
Interface              EFM State              Loopback Timeout
-------------------------------------------------------------------------
GigabitEthernet1/1/4   discovery                      --
GigabitEthernet1/1/6   discovery                  --
```

Для очистки совести я собрал полностью аналогичную схему и увидел, что R1, как активный член партии отсылает данные, но ничего не получает в ответ. А на R2 совсем всё тихо.

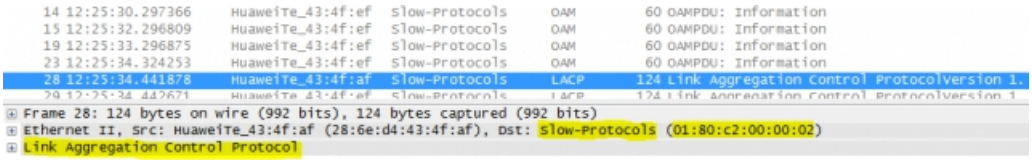
```
debugging efm interface GigabitEthernet 1/1/4 all
Info: Operation succeeded.
May 14 2013 09:18:26.880.1+03:00 R1 EFM/7/OAMPDU:Slot=1;
EFM GigabitEthernet1/1/4 Send Packet
Flags:00 08
Code:00
01 10 01 00 00 00 0F 00 80 00 E0 FC 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00

May 14 2013 09:18:27.880.1+03:00 R1 EFM/7/OAMPDU:Slot=1;
EFM GigabitEthernet1/1/4 Send Packet
Flags:00 08
Code:00
01 10 01 00 00 00 0F 00 80 00 E0 FC 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00

May 14 2013 09:18:28.880.1+03:00 R1 EFM/7/OAMPDU:Slot=1;
EFM GigabitEthernet1/1/4 Send Packet
Flags:00 08
Code:00
01 10 01 00 00 00 0F 00 80 00 E0 FC 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00
```

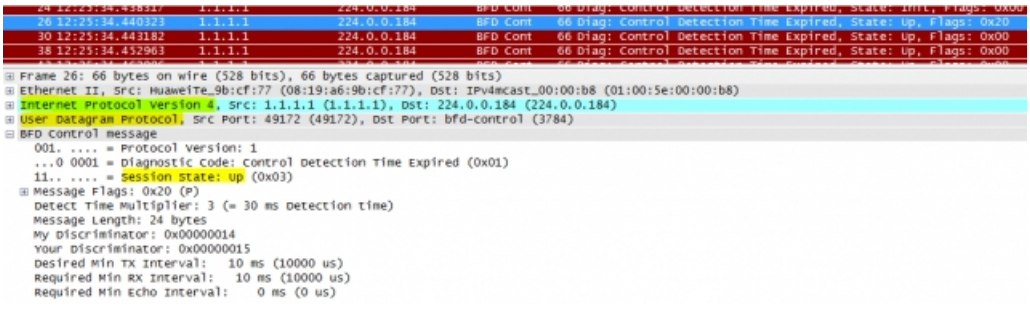
Замечу также, что LACP в таком виде тоже отрабатывать не будет – его кадры тоже не будут доходить до удалённой стороны по той же причине.

```
14 12:25:30.297366   HuaweiTe_43:4f:ef  Slow-Protocols   OAM    60 OAMPDU: Information
15 12:25:32.296809   HuaweiTe_43:4f:ef  Slow-Protocols   OAM    60 OAMPDU: Information
19 12:25:33.296875   HuaweiTe_43:4f:ef  Slow-Protocols   OAM    60 OAMPDU: Information
23 12:25:34.324253   HuaweiTe_43:4f:ef  Slow-Protocols   OAM    60 OAMPDU: Information
28 12:25:34.441878   HuaweiTe_43:4f:af  Slow-Protocols   LACP  124 Link Aggregation Control ProtocolVersion 1.
29 12:25:34.442671   HuaweiTe_43:4f:ef  Slow-Protocols   LACP  124 Link Aggregation Control ProtocolVersion 1.
⊞ Frame 28: 124 bytes on wire (992 bits), 124 bytes captured (992 bits)
⊞ Ethernet II, Src: HuaweiTe_43:4f:af (28:6e:d4:43:4f:af), Dst: Slow-Protocols (01:80:c2:00:00:02)
⊞ Link Aggregation Control Protocol
```



Спасение

Единственный здесь выход – это использование BFD – Bidirectional Forwarding Detection. В то время, как EFM свои данные инкапсулирует напрямую в Ethernet, BFD использует IP и UDP. То есть для него никакой трудности не представляют промежуточные L2-устройства.



BFD отправляет свои пакеты на мультикастовый адрес 224.0.0.184 и, если противоположное устройство настроено соответствующим образом, оно высылает ответные пакеты – BFD-сессия поднимается. Вот пример конфигурации, которая принуждает устройство погасить интерфейс в случае проблем:

R1
bfd TEST bind peer-ip default-ip interface GigabitEthernet1/1/4
discriminator local 10
discriminator remote 11
process-interface-status
commit

R2
bfd TEST bind peer-ip default-ip interface GigabitEthernet1/1/4
discriminator local 11
discriminator remote 10
process-interface-status
commit

Первой строкой задаём привязку BFD к физическому интерфейсу. Далее определяем дискриминаторы – обязательный параметр BFD, позволяющий различать сессии. И командой **process-interface-status** указываем, что устройство должно гасить интерфейс, если BFD обнаружил проблему.

```
[R1]dis bfd session all
--------------------------------------------------------------------------------
Local     Remote     PeerIpAddr    State     Type        InterfaceName
--------------------------------------------------------------------------------
11          10        224.0.0.184    Up     S_IP_IF     GigabitEthernet1/1/4
21          20        224.0.0.184    Up     S_IP_IF     GigabitEthernet1/1/6
--------------------------------------------------------------------------------
          Total UP/DOWN Session Number : 2/0
```

В случае проблемы на линке BFD это сразу замечает:

May 14 2013 09:45:45+03:00 R2 %%01BFD/4/STACHG_TODWN(l)[23]:Slot=1;BFD session changed to Down. (SlotNumber=1, Discriminator=20, Diagnostic=DetectDown, Applications=IFNET, ProcessPST=False, BindInterfaceName=GigabitEthernet1/1/6, InterfacePhysicalState=Up, InterfaceProtocolState=Down)

Статус интерфейса принимает вид:

```
[R2]dis int br
*down: administratively down
^down: standby
(l): loopback
(s): spoofing
(b): BFD down
(e): ETHOAM down
(d): Dampening Suppressed
InUti/OutUti: input utility/output utility
Interface                PHY       Protocol     InUti     OutUti      inErrors    outErrors
Aux0/0/1                 down        down         0%         0%            0           0
Eth-Trunk22               up          up        0.05%      0.05%          0           0
GigabitEthernet1/1/4      up          up        0.05%      0.05%          0           0
GigabitEthernet1/1/6      up          up(b)       0%         0%           0           0

[R2]dis int gig1/1/6
GigabitEthernet1/1/6 current state : UP
Line protocol current state : UP(BFD status down)
```

Разумеется, BFD также детектирует обрыв одного оптического кабеля. Правда смысл LACP в такой ситуации, так или иначе, теряется. Ещё одно из преимуществ BFD (скорость реакции) - это уровень миллисекунд.

Вообще говоря, многие устройства поддерживают команды, которые позволяют "прокидывать" данные Slow Protocls, но это от лукавого.

В общем решение предоставлено, инженер счастлив, а я узнал для себя что-то новое.