Internet Engineering Task Force (IETF)
Request for Comments: 9197
Category: Standards Track
ISSN: 2070-1721

F. Brockners, Ed.
Cisco
S. Bhandari, Ed.
Thoughtspot
T. Mizrahi, Ed.
Huawei

May 2022

Data Fields for In Situ Operations, Administration, and Maintenance (IOAM)

Поля данных для операций OAM «на месте»

Аннотация

IOAM[1] собирает операционные и телеметрические данные в пакет по мере прохождения им пути между парой точек сети. В этом документе обсуждаются поля данных для IOAM и связанные с этим типы. Поля данных (IOAM-Data-Fields) можно инкапсулировать в разные протоколы, такие как NSH[2], Segment Routing, Geneve[3], IPv6. IOAM можно использовать для дополнения механизмов на основе OAM, например, с ICMP или иными типами пакетов-зондов.

Статус документа

Документ относится к категории Internet Standards Track.

Документ является результатом работы IETF[4] и представляет согласованный взгляд сообщества IETF. Документ прошёл открытое обсуждение и был одобрен для публикации IESG[5]. Дополнительную информацию о стандартах Internet можно найти в разделе 2 в RFC 7841.

Информацию о текущем статусе документа, ошибках и способах обратной связи можно найти по ссылке https://www.rfc-editor.org/info/rfc9197.

Оглавление

[1] In situ Operations, Administration, and Maintenance - Операции, Администрирование и Поддержка на месте.
[2] Network Service Header - заголовок сетевого сервиса.
[3] Generic Network Virtualization Encapsulation - базовая инкапсуляция для виртуализации сетей.
[4] Internet Engineering Task Force - комиссия по решению инженерных задач Internet.
[5] Internet Engineering Steering Group - комиссия по инженерным разработкам Internet.

1. Введение

Этот документ определяет поля данных для IOAM. Данные OAM в IOAM записываются в пакеты по мере их прохождения через конкретный сетевой домен. Термин in situ (на месте) относится к тому, что сведения OAM добавляются к пакетам данных. А не передаются в специальных пакетах OAM. IOAM служит дополнением к таким механизмам как Ping или Traceroute. В терминах активных или пассивных типов OAM методы IOAM можно отнести к гибридным. Механизмы IOAM не требуют применения специальных пакетов и сведения добавляются в имеющиеся пакеты данных, поэтому метод нельзя считать пассивным. В классификации [RFC7799] IOAM можно считать гибридным типом I (Hybrid Type I). Механизмы IOAM можно применять там, где мехнизмы на основе, например, ICMP, не применимы или не дают нужных результатов, таких как подтверждение прохождения трафика по определённому пути, проверка соглашений об уровне обслуживания (Service Level Agreement или SLA), подробная статистика распределения путей трафика в сетях с применением нескольких путей или случаях, когда тестовый трафик может обрабатываться сетевыми устройствами не так, как трафик данных.

Термин in situ OAM исходно был обусловден применением связанных с OAM механизмов, добавляющих информацию в пакеты. В этом документе термин IOAM служит для определения технологии IOAM, включающей механизмы in situ, а также механизмы, способные вызывать создание дополнительных пакетов, предназначенных для OAM.

2. Соглашения

Ключевые слова **необходимо** (MUST), **недопустимо** (MUST NOT), **требуется** (REQUIRED), **нужно** (SHALL), **не следует** (SHALL NOT), **следует** (SHOULD), **не нужно** (SHOULD NOT), **рекомендуется** (RECOMMENDED), **не рекомендуется** (NOT RECOMMENDED), **возможно** (MAY), **необязательно** (OPTIONAL) в данном документе интерпретируются в соответствии с BCP 14 [RFC2119] [RFC8174] тогда и только тогда, когда они выделены шрифтом, как показано здесь.

Abbreviations and definitions used in this document:

E2E
Edge to Edge - сквозной.

Geneve
Generic Network Virtualization Encapsulation [RFC8926] - базовая инкапсуляция для сетевой виртуализации.

IOAM
In situ Operations, Administration, and Maintenance - выполняемые «на месте» действия OAM.

MTU
Maximum Transmission Unit - максимальный передаваемый блок.

NSH
Network Service Header [RFC8300] - заголовок сетевого сервиса.

OAM
Operations, Administration, and Maintenance - Операции, Администрирование и Поддержка.

PMTU
Path MTU - MTU на пути.

POT
Proof of Transit - доказательство (подтверждение) транзита.

Short format - короткий формат
Указывает IOAM-Data-Field размером 4 октета.

SID
Segment Identifier - идентификатор сегмента

SR
Segment Routing - посегментная маршрутизация.

VXLAN-GPE
Virtual eXtensible Local Area Network, Generic Protocol Extension [NVO3-VXLAN-GPE] - расширение базового протокола VXLAN.

Wide format - широкий формат
Указывает IOAM-Data-Field размером 8 октетов.

3. Область действия, применимость и допущения

IOAM предполагает набор ограничений, а также руководящие принципы и концепции, тесно связанные с определениями полей (IOAM-Data-Field) и описанные в данном разделе. Вопросы применения полей данных IOAM и связанных с ними концепций при развёртывании IOAM выходят за рамки документа и рассмотрены в [IPPM-IOAM-DEPLOYMENT].

Scope - область действия

Этот документ определяет поля данных IOAM и связанные с ними типы. IOAM-Data-Field можно инкапсулировать в разные протоколы, включая NSH, Segment Routing, Geneve, IPv6. Детали такой инкапсуляции выходят за рамки этого документа и предполагается, что для каждого типа инкапсуляции будет выпущен документ RFC, созданный в тесном контакте с рабочей группой, которая разрабатывает и поддерживает протокол инкапсуляции, а также с рабочей группой IETF по измерению производительности IP (IP Performance Measurement - IPPM).

Domain (or scope) of in situ OAM deployment - домен (область) развёртывания IOAM

IOAM ориентируется на домены с ограничениями, как указано в [RFC8799]. Таким доменом для IOAM может быть, например, кампус предприятия, использующий физические соединения между устройствами или наложенную сеть с виртуальными соединениями (туннелями) между устройствами. Домен с ограничениями, использующий IOAM может содержать один или несколько доменов IOAM со своими идентификаторами пространств имён. Границей IOAM-Domain является периметр или край (edge). Домены IOAM могут пересекаться внутри домена с ограничениями. Разработчики протоколов инкапсуляции для IOAM задают механизмы, обеспечивающие нахождение данных IOAM в пределах IOAM-Domain. Кроме того предполагается, что оператор домена принимает меры по предотвращению утечки данных IOAM через границу домена, например, путём фильтрации пакетов. Оператору **следует** учитывать потенциальное влияние IOAM на механизмы, такие как обработка ECMP (например, при распределении нагрузки по размерам пакетов может влиять рост размера из-за IOAM), PMTU (значения MTU на всех каналах в домене должны быть достаточными для передачи пакетов увеличенного из-за IOAM размера) и обработка пакетов ICMP (например для IPv6 желательна поддержка IOAM запросов и откликов ICMPv6 с трансляцией в расширения ICMPv6, чтобы можно было копировать IOAM-Data-Field из запросов в отклики).

IOAM control points - точки управления IOAM

Поля данных IOAM добавляются или удаляются из пользовательского трафика устройствами на краю домена. Устройства, формирующие IOAM-Domain, могут добавлять, обновлять или удалять поля данных IOAM. Крайвыми устройствами домена IOAM могут быть хосты или сетевые устройства.

Traffic sets that IOAM is applied to - наборы трафика для применения IOAM

IOAM может применяться для всего или части пользовательского трафика. Применение IOAM лишь для ограниченного подмножества трафика (например, на интерфейсе, по списку доступа или спецификации потоков для задания части трафика и т. п.) может быть полезно в случаях, когда издержки на обработку IOAM-Data-Field инкапсулирующими, промежуточными или декапсулирующими узлами велики с точки зрения производительности или эксплуатационных расходов. Таким образом, ограничение объёма трафика IOAM может давать преимущества в некоторых случаях.

Encapsulation independence - независимость от инкапсуляции

Определения IOAM-Data-Field независимы от протоколов, в которые инкапсулируются поля данных IOAM. IOAM-Data-Field можно инкапсулировать в несколько разных протоколов.

Layering - уровни

Если несколько протоколов инкапсуляции (например, при туннелировании) образуют стек, поля данных IOAM могут присутствовать на нескольких уровнях этого стека. Такое поведение соответствует модели ships-in-the-night (корабли в ночи), т. е. IOAM-Data-Field на одном уровне не зависят от полей данных IOAM на другом уровне. Уровни позволяют операторам задавать протокольный уровень, на котором выполняются измерения. На разных уровнях могут (но не обязаны) применяться одни механизмы инкапсуляции IOAM.

IOAM implementation - реализация IOAM

Определения полей данных IOAM учитывают специфику устройств с программной или аппаратной плоскостью данных.

4. Поля данных, типы и узлы IOAM

В этом разделе описана относящаяся к IOAM номенклатура и типы данных, такие как IOAM-Data-Field, IOAM-Type, IOAM-Namespace, а также разные типы узлов IOAM.

4.1. Поля данных и типы опций IOAM

Поле данных IOAM-Data-Field - это набор битов с заданным форматом и назначением, который сохраняется в неком месте пакета для целей IOAM.

Для разных вариантов применения IOAM поля данных IOAM делятся по категориям, которые называют типами опций - IOAM-Option-Type. Поддерживается общий реестр IOAM-Option-Type (7.1. Реестр IOAM Option-Type). В соответствии с IOAM-Option-Type определены разные IOAM-Data-Field. Этот документ задаёт 4 IOAM-Option-Type:

- Pre-allocated Trace Option-Type;
- Incremental Trace Option-Type;
- POT Option-Type;
- E2E Option-Type.

Новые значения IOAM-Option-Type могут быть выделены IANA, как указано в параграфе 7.1. Реестр IOAM Option-Type.

4.2. Домены и типы узлов IOAM

В разделе 3 уже отмечено, что развёртывание IOAM предполагается в домене с ограничениями [RFC8799]. В пакет добавляется 1 или несколько IOAM-Option-Type на входе в IOAM-Domain и они удаляются на выходе из домена. Внутри домена IOAM поля IOAM-Data-Field могут обновляться узлами, через которые они проходят. IOAM-Domain состоит их инкапсулирующих, декапсулирующих и транзитных узлов IOAM. Роль узла определяется в пространстве имён IOAM-Namespace, описанном ниже. Узел может иметь разные роли в разных IOAM-Namespace.

Устройство, добавляющее в пакет хотя бы один IOAM-Option-Type, называется инкапсулирующим узлом IOAM, а устройство, удаляющее IOAM-Option-Type, - декапсулирующим узлом IOAM. Узлы внутри домена, понимающие данные IOAM, считывающие, записывающие или обрабатывающие их, называются транзитными узлами IOAM. Узлы IOAM, добавляющие или удаляющие IOAM-Data-Field, могут также обновлять поля IOAM-Data-Field. Иными словами, узлы инкапсуляции и декапсуляции могут одновременно быть транзитными узлами IOAM. Отметим, что не каждый узел в домене IOAM обязан быть транзитным узлом IOAM. Например, пакеты могут проходить через межсетевые экраны (МСЭ) с поддержкой IOAM и в этом случае транзитными узлами IOAM будут эти МСЭ, а не все узлы.

Узел инкапсуляции IOAM внедряет хотя бы один IOAM-Option-Type (из списка в параграфе 7.1. Реестр IOAM Option-Type) в пакеты, для которых разрешено применять IOAM. Если IOAM применяется для части трафика, узел инкапсуляции отвечает за применение IOAM к указанному подмножеству трафика.

Транзитные узлы IOAM читают, записывают и/или обрабатывают хотя бы одно поле IOAM-Data-Field. При наличии в пакете одновременно типов Pre-allocated и Incremental каждый транзитный узел IOAM с учётом конфигурации и возможностей реализации IOAM может заполнять данные трассировки IOAM для типа Pre-allocated или Incremental (но не для обоих). Отметим, что отсутствие заполнения Trace Option-Type разрешено транзитным узлам IOAM. Транзитный узел **должен** игнорировать непонятные IOAM-Option-Type. Транзитному узлу **недопустимо** добавлять в пакет новые IOAM-Option-Type, **недопустимо** удалять IOAM-Option-Type из пакета и **недопустимо** менять IOAM-Data-Field для типа Edge-to-Edge Option-Type.

Узел декапсуляции IOAM удаляет из пакетов IOAM-Option-Type.

Роль узлов инкапсуляции, транзита и декапсуляции IOAM всегда выполняется в конкретном пространстве имён IOAM. Это означает, что узел IOAM, являющийся, например декапсулирующим для IOAM-Namespace A, но не для IOAM-Namespace B, будет удалять IOAM-Option-Type из пакета лишь для пространства A. Отметим, что это применимо и к IOAM-Option-Type, которые узел не понимает, например, IOAM-Option-Type, которые могут быть добавлены в будущем, сверх описанных выше 4 типов.

IOAM-Namespace позволяет специфические для пространства имён определения и интерпретацию IOAM-Data-Field. Идентификатор пространства имён может, например, указывать на физический интерфейс (чтобы понимать, какой из физических интерфейсов агрегированного канала использован для приёма или передачи пакета), а в другом случае - на логический (для туннеля). Пространства имён рассмотрены в следующем параграфе.

4.3. Пространства имён IOAM

IOAM-Namespace добавляет контекст к IOAM-Option-Type и связанным IOAM-Data-Field, которые интерпретируются, как указано в этом документе, независимо от пространства имён IOAM. IOAM-Namespace обеспечивает способ группировки узлов для поддержки разных подходов к развёртыванию IOAM (см. примеры ниже). Пространства имён позволяют также решать возможные проблемы из-за того, что IOAM-Data-Field (например, идентификаторы узлов IOAM) не уникальны в глобальном масштабе. Значимость IOAM-Data-Field всегда ограничена конкретным IOAM-Namespace. С учётом интерпретации IOAM-Data-Field как контекста конкретного пространства имён всегда требуется передавать поле Namespace-ID вместе с полями данных IOAM.

IOAM-Namespace указывается 16-битовым идентификатором пространства имён (Namespace-ID). Поле IOAM-Namespace включается во все IOAM-Option-Type, определённые в этом документе, и **должно** включаться во все будущие IOAM-Option-Type. Значения Namespace-ID поделены на два диапазона:

- операторские значения от 0x0001 до 0x7FFF;
- выделенные IANA значения от 0x8000 до 0xFFFF.

Диапазон выделяемых IANA значений предназначен для обеспечения совместимости с новой функциональностью в будущих расширениях IOAM, а операторские значения относятся к домену и назначаются оператором. Namespace-ID = 0x0000 является принятым по умолчанию (Default-Namespace-ID) и указывает, что с полями IOAM-Data-Field в пакете не связано конкретного пространства имён. Значение Default-Namespace-ID **должно** поддерживаться всеми узлами, реализующими IOAM. Примером использования Default-Namespace-ID является система, где не применяется конкретных пространств имён для некоторых или всех пакетов с IOAM-Data-Field.

Идентификаторы пространства имён позволяют устройствам с поддержкой IOAM определить:

- требуется ли устройству обрабатывать один или несколько IOAM-Option-Type; если Namespace-ID в пакете не соответствует ни одному из настроенных на узле Namespace-ID, где он работает, узлу **недопустимо** изменять содержимое полей IOAM-Data-Field;
- какие IOAM-Option-Type требуется обрабатывать/менять при наличии в пакете нескольких IOAM-Option-Type (это может быть в случае перекрытия IOAM-Domain или многоуровневого развёртывания IOAM);
- нужно ли удалять из пакета один или несколько IOAM-Option-Type, например, на границе домена.

Для IOAM-Namespace поддерживается несколько вариантов применения, указанных ниже.

- IOAM-Namespace могут использоваться оператором для идентификации разных IOAM-Domain. Устройства на краю IOAM-Domain могут фильтровать Namespace-ID для обеспечения изоляции доменов.
- IOAM-Namespace создают дополнительный контекст для полей IOAM-Data-Field и могут служить для обеспечения уникальности IOAM-Data-Field и подобающей интерпретации станциями управления сетью и контроллерами. Поле идентификатора узла (node_id, см. ниже) не обязано быть уникальным в развёртываний. Возможны ситуации, когда оператор хочет использовать разные идентификаторы узлов для разных уровней IOAM даже в одном устройстве или идентификаторы узлов могут быть не уникальными по организационным причинам, таким как слияние двух организаций. Namespace-ID может служить идентификатором контекста, чтобы комбинация node_id и Namespace-ID всегда была уникальной.
- IOAM-Namespace можно использовать для задания способа интерпретации IOAM-Data-Field. IOAM поддерживает 3 разных формата временных меток и Namespace-ID можно использовать для выбора нужного

формата. Поля IOAM-Data-Field (например, заполнение буферов), с которыми не связана единица измерения, интерпретируются в контексте IOAM-Namespace.

- IOAM-Namespace могут служить для идентификации различных наборов устройств (например, разных типов) в системе. Если оператор хочет внедрять разные поля IOAM-Data-Field по устройствам, эти устройства можно собрать в разные IOAM-Namespace. Это может быть связано с тем, что набор функций IOAM зависит от устройства или имеются причины оптимизировать пространство в заголовках пакетов. Возможна также связь с аппаратными или эксплуатационными ограничениями на размер данных трассировки, которые можно добавить и обработать, что препятствует сбору полной трассировки для пакета.
- Назначение различных IOAM Namespace-ID разным наборам узлов или частям сети и использование отдельного экземпляра IOAM-Option-Type для каждого Namespace-ID позволяет собрать и построить полную трассировку на основе частичных трассировок для каждого IOAM-Option-Type в каждом пакете потока. Например, оператор может разделить устройства домена по двум IOAM-Namespace так, что каждое пространство имён представляется одним или двумя IOAM-Option-Type в пакете. Каждый узел будет записывать данные только для IOAM-Namespace, к которому он относится, игнорируя IOAM-Option-Type с другим IOAM-Namespace. Для получения полного представления нужно сопоставить поля IOAM-Data-Field обоих IOAM-Namespace.

4.4. IOAM Trace Option-Type

В типичном развёртывании все узлы IOAM-Domain принимают участие в IOAM, выполняя функции транзитного, инкапсулирующего или декапсулирующего узла. Если не все узлы домена поддерживают функциональность IOAM, определённую в этом документе, данные трассировки IOAM (т. е. данные узлов, см. ниже) могут собираться лишь на узлах с поддержкой IOAM. Не поддерживающие определённые здесь функции IOAM узлы будут пересылать пакеты, не изменяя полей IOAM-Data-Field. Предполагается, что максимальное число пересылок и минимальное значение PMTU в IOAM-Domain известны. Индикатор переполнения (бит O) определён как один из способов обработки ситуаций с недооценкой значения PMTU, когда число интервалов пересылки с поддержкой IOAM превосходит возможности (место для данных) пакета.

Для оптимизации аппаратных и программных реализаций трассировка IOAM определена в двух вариантах. При развёртывании можно выбрать один изи этих вариантов или развернуть оба.

Pre-allocated Trace-Option - заранее определённая трассировка

Этот вариант трассировки определён как контейнер полей узлов данных (см. ниже) с выделенным заранее пространством для каждого узла, куда тот может поместить свои данные. Этот вариант полезен для реализаций, где эффективно однократное выделение пространства и поддерживается индексированный массив для заполнения данных в процессе транзита (например, к этому типу часто относятся программные узлы пересылки). Инкапсулирующий узел IOAM выделяет пространство для Pre-allocated Trace Option-Type в пакете и устанавливает соответствующие поля в IOAM-Option-Type. Декапсулирующий узел IOAM выделяет массив для хранения данных, полученных от каждого узла в домене, через который прошёл пакет. Транзитные узлы IOAM обновляют содержимое массива и, возможно, контрольные суммы во внешних заголовках. Указатель, являющийся частью данных трассировки IOAM, задаёт следующую позицию в массиве. Транзитный узел, обновляющий содержимое Pre-allocated Trace-Option, обновляет и указатель, определяющий место записи данных для следующего транзитного узла IOAM. Массив со списком узлов данных (см. ниже) в пакете заполняется итеративно по мере прохождения пакета через сеть, начиная с последней записи в массисе, т. е, сначала заполняется node data list [n], затем node data list [n-1] и т. д.

Incremental Trace-Option - инкрементная трассировка

Этот вариант трассировки определён как контейнер полей узлов данных, где каждый узел выделяет и выталкивает свои данные сразу после заголовка опции. Этот тип записи трассировки полезен для некоторых аппаратных реализаций, поскольку он исключает для транзитных элементов необходимость считывать полный массив в опции и позволяет использовать пакеты произвольного размера, разрешаемого MTU. Узел инкапсуляции IOAM выделяет пространство для Trace Option-Type, устанавливая на основе рабочего состояния и конфигурации поля Option-Type, определяющие поля IOAM-Data-Field для сбора сведений и размер списка данных узла. Транзитные узлы IOAM выталкивают свои данные в список с учётом протокольных ограничений на уровне инкапсуляции, а затем уменьшают значение размера, остающегося для заполнения последующими узлами, и корректируют размер (возможно и контрольную сумму) во внешних заголовках.

Узлы инкапсуляции и декапсуляции IOAM, поддерживающие трассировку, **должны** поддерживать оба Trace Option-Type, а транзитным узлам достаточно поддержки одного типа. При одновременном использовании в системе обоих вариантов опция Incremental Trace-Option **должна** размещаться до Pre-allocated Trace-Option. В системах с устройствами Incremental Trace-Option и Pre-allocated Trace-Option в пакетах могут присутствовать оба Option-Type. С учётом того, что оператору известно оборудование в конкретном IOAM-Domain, он может с помощью настройки выбрать вариант трассировки для домена.

В каждой записи данных узла содержатся сведения от конкретного транзитного узла IOAM, через который прошёл пакет. Узел декапсуляции IOAM удаляет IOAM-Option-Type и обрабатывает и/или экспортирует данные. Как и для всех IOAM-Data-Field, поля IOAM-Data-Field в IOAM Trace Option-Type определяются в контексте IOAM-Namespace.

При трассировке IOAM могут собираться указанные ниже типы сведений.

- Идентификация узла IOAM. Идентификатор узла IOAM может совпадать с идентификатором устройства, отдельной точки управления или подсистемы в устройстве.
- Идентификация устройства, принявшего пакет (входной интерфейс).
- Идентификация устройства, передавшего пакет (выходной интерфейс).
- Время суток, кода пакет был обработан узлом и задержка при транзите. Возможны и ожидаются разные определения времени обработки, но важно использовать одно определение на всех устройствах IOAM-Domain.

- Базовые данные, т. е. сведения в свободном формате, синтаксис и семантика которых определяются оператором для конкретного развёртывания. В определённом IOAM-Namespace все узлы IOAM интерпретируют базовые данные одним способом. Примеры таких данных IOAM включают местоположение (размещение узла в момент обработки пакета), степень заполнения буферов или кэша при обработке пакета или даже уровень заряда батарей.

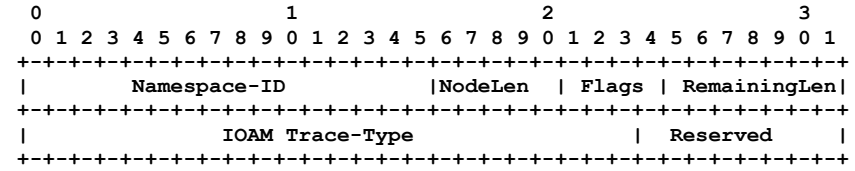
- Сведения о том, добавлялись ли данные трассировки IOAM на каждом узле пересылки (hop) или некоторые узлы пересылки не являются транзитными узлами IOAM.

Следует отметить, что семантика некоторых полей данных узла, заданных ниже, таких как глубина очереди и занятость буферов, зависит от реализации. Это предназначено для обеспечения работы узлов IOAM с разной архитектурой.

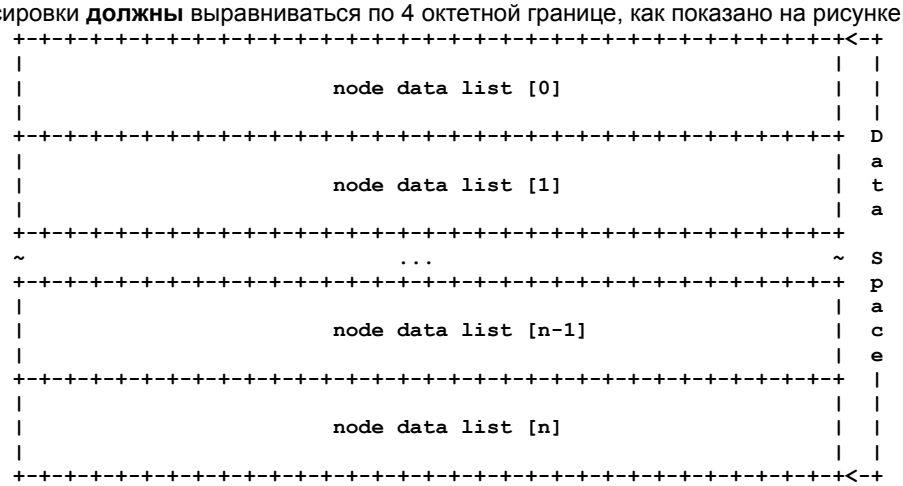
4.4.1. Выделенные заранее и инкрементные Trace Option-Type

Форматы IOAM Pre-allocated Trace-Option и IOAM Incremental Trace-Option похожи, а исключения указаны ниже. Оба варианта трассировки включают заголовок варианта трассировки с фиксированным размером и пространство переменного размера для хранения собранных данных (список данных узла - node data list). Транзитному узлу IOAM (т. е. узлу, не выполняющему инкапсуляцию или декапсуляцию) **недопустимо** менять какие-либо поля в заголовке фиксированного размера, за исключением полей Flags и RemainingLen. Т. е. транзитному узлу IOAM **недопустимо** менять поля Namespace-ID, NodeLen, IOAM Trace-Type, Reserved.

Формат заголовка фиксированного размера (Pre-allocated и Incremental) показан на рисунке.



Данные опции трассировки **должны** выравниваться по 4 октетной границе, как показано на рисунке.



Namespace-ID

16-битовый идентификатор IOAM-Namespace. Значение Namespace-ID = 0x0000 определено как Default-Namespace-ID (см. параграф 4.3) и **должно** быть известно всем узлам, реализующим IOAM. Для любого другого Namespace-ID, не соответствующего пространству имён, заданному для узла, этому узлу **недопустимо** менять содержимое полей IOAM-Data-Field.

NodeLen

5-битовое целое число без знака, задающее размер данных, добавляемых каждым узлом, в 4-октетных блоках без учёта поля Opaque State Snapshot.

Если бит 22 в IOAM Trace-Type не установлен, поле NodeLen задаёт фактический размер, добавляемый каждым узлом, а при установленном бите 22 фактический размер добавляемых узлом данных определяет сумма NodeLen и размера поля Opaque State Snapshot в 4-октетных блоках.

Например, если установлены 3 бита IOAM Trace-Type и ни для одного из них не задан широкий формат, поле NodeLen будет иметь значение 3. Если установлены 3 бита IOAM Trace-Type и для двух задан широкий формат, поле NodeLen будет иметь значение 5.

Узел инкапсуляции IOAM **должен** устанавливать NodeLen.

Узел, получаемый опцию трассировки IOAM Pre-allocated или Incremental Trace-Option, полагается на NodeLen.

Flags

4-битовое поле. Флаги выделяются IANA, как указано в параграфе 7.3. Этот документ задаёт 1 флаг.

Бит 0

Бит переполнения (Overflow или O-bit) является старшим в поле флагов. Если элемент сети, который считается добавляющим данные узла, обнаруживает нехватку места для записи этих данных, ему **недопустимо** добавлять в пакет какие-либо поля и он **должен** установить бит O = 1 в заголовке IOAM Trace-Option. Это полезно для транзитных узлов, чтобы исключить последующую обработку опции.

RemainingLen

7-битовое целое число без знака. Это поле задаёт размер пространства данных (в 4-октетных блоках), остающееся для записи, до того, как список данных узла будут сочтёт заполненным. Отправитель **должен** установить начальное значение поля RemainingLen. Отправитель **может** рассчитать значение RemainingLen, учитывая число байтов узлов данных, разрешённых PMTU, с учётом известности PMTU. Последующие узлы могут просто сравнивать RemainingLen и NodeLen вместе с Opaque State Snapshot (если нужно) для определения возможности добавить данные. Когда данные узла добавлены, узел **должен** уменьшить величину RemainingLen на размер добавленных данных. В Pre-allocated Trace-Option поле RemainingLen служит для определения смещения в пространстве данных для записи элемента данных узла. В частности, запись данных узла будет начинаться со

смещения RemainingLen - NodeLen - размер (Opaque State Snapshot) в 4-октетных блоках. Если RemainingLen в Pre-allocated Trace-Option превосходит размер опции, как указано в заголовке нижележащего уровня (который выходит за рамки этого документа), узлу **недопустимо** добавлять какие-либо поля.

IOAM Trace-Type

24-битовый идентификатор, указывающий, какие типы данных используются в списке данных узла.

IOAM Trace-Type является битовым полем. Ниже приведён список определённых в этом документе типов, которые описаны в параграфе 4.4.2. Порядок размещения полей данных в каждом элементе узла соответствует приведенному ниже порядку битов поля IOAM Trace-Type.

Бит 0

Старший бит, установка (1) которого указывает наличие Hop_Lim и node_id (короткий формат) в данных узла.

Бит 1

Значение 1 указывает наличие ingress_if_id и egress_if_id (короткий формат) в данных узла.

Бит 2

Значение 1 указывает наличие секунд временной метки в данных узла.

Бит 3

Значение 1 указывает наличие долей секунд временной метки в данных узла.

Бит 4

Значение 1 указывает наличие транзитной задержки в данных узла.

Бит 5

Значение 1 указывает наличие определяемых IOAM-Namespace сведений (короткий формат) в данных узла.

Бит 6

Значение 1 указывает наличие глубины очереди в данных узла.

Бит 7

Значение 1 указывает наличие Checksum Complement в данных узла.

Бит 8

Значение 1 указывает наличие Hop_Lim и node_id (широкий формат) в данных узла.

Бит 9

Значение 1 указывает наличие ingress_if_id и egress_if_id (широкий формат) в данных узла.

Бит 10

Значение 1 указывает наличие определяемых IOAM-Namespace сведений (широкий формат) в данных узла.

Бит 11

Значение 1 указывает наличие занятости буфера в данных узла.

Биты 12-21

Не определены и доступны для распределения в реестре IOAM Trace-Type (параграф 7.2). Каждый будущий узел данных, соответствующий одному из этих битов, **должен** иметь размер 4 октета. Узел инкапсуляции IOAM должен устанавливать для неопределённых битов значение 0. Если транзитный узел IOAM получает пакет со значением 1 в одном или нескольких резервных битах, он должен выполнить одно из действий:

1. добавить соответствующие данные узла, заполнив их резервным значением 0xFFFFFFFF, после полями данных узла для определённых выше битов IOAM Trace-Type, чтобы общий размер данных, добавленных этим узлом (в 4-октетных блоках) был равен NodeLen;
2. не добавлять никаких полей данных узла даже для указанных выше битов IOAM Trace-Type.

Бит 22

Установка (1) этого бита указывает наличие поля переменного размера Opaque State Snapshot.

Бит 23

Резервный бит, который **должен** сбрасываться при передаче и игнорироваться при получении. Этот бит зарезервирован для будущих расширений поля битов IOAM Trace-Type.

В параграфе 4.4.2 описаны типы IOAM-Data-Type и их форматы. Внутри IOAM-Domain возможные комбинации битов IOAM Trace-Type могут быть ограничены конфигурацией.

Reserved

8 резервных битов, которые узел инкапсуляции IOAM **должен** сбрасывать (0) до передачи, а транзитные узлы IOAM **должны** игнорировать.

Node data List [n]

Поле переменного размера со списком элементов узла данных, где содержимое каждого элемента определяется значением IOAM Trace-Type. Порядок полей данных в каждом элементу соответствует порядку битов в поле IOAM Trace-Type. Каждый узел **должен** помещать свой элемент данных перед полученными элементами данных, чтобы его элемент стал первым в списке. Последним элементом данных в этом списке является элемент данных первого узла с поддержкой IOAM на пути пакета. Такое заполнение списка данных узла обеспечивает совпадение порядка данных в списке с порядком в опциях трассировки Incremental и Pre-allocated. В Pre-allocated Trace-Option индекс в RemainingLen указывает смещение текущего активного узла для заполнения.

4.4.2. Поля данных узла IOAM и связанные форматы

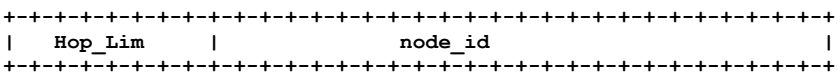
Все поля IOAM-Data-Field **должны** выравниваться по 4 октетам. Если узел, в котором предполагается обновление IOAM-Data-Field, не способен заполнить значение, указанное IOAM Trace-Type, в это поле должно быть помещено значение 0xFFFFFFFF (4-октетное поле) или 0xFFFFFFFFFFFFFFFF (8-октетное поле), указывающее, что значение не заполнено (за исключением явно указанных полей).

Некоторые IOAM-Data-Field, определённые ниже, такие как определяемые IOAM-Namespace поля, могут иметь коротких и длинный (широкий) формат, которые не исключают друг друга и в развёртывании могут применяться оба сразу. Например, ingress_if_id_(short format) может быть идентификатором физического интерфейса, а ingress_if_id_(wide format) - идентификатором логического субинтерфейса на нем.

Поля данных и связанные с ними типы для каждого IOAM-Data-Field заданы ниже. Определения IOAM-Data-Field сосредоточены на синтаксисе полей данных и не избегают задания семантики. Поэтому для полей данных не заданы единицы измерения для таких полей, как заполнение буфера или глубина очереди. При таком подходе узлы могут предоставлять сведения в их исходном формате и не потребуется преобразовывать единицы измерения или формат. Предполагается, что системы с дополнительной обработкой сведений IOAM, например, системы управления сетью, способны преобразовывать единицы измерения в рамках обработки IOAM-Data-Field. Сочетание конкретного поля данных и Namespace-ID обеспечивает контекст для корректной интерпретации представленных данных.

4.4.2.1. Hop_Lim и node_id в коротком формате

Hop_Lim и короткое поле node_id занимают 4 октета, как показано на рисунке.



Hop_Lim

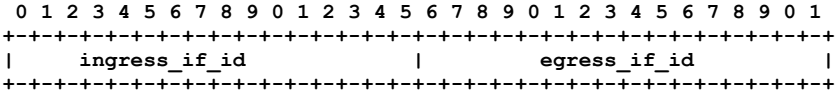
1-октетное целое число без знака. В поле помещается значение Hop Limit в пакете на выходе из узла, записавшего эти данные. Данные Hop Limit служат для определения местоположения узла на пути связи. Значение копируется из нижележащего уровня, например, TTL в заголовке IPv4 или Hop Limit в заголовке IPv6 пакета, готового к передаче. Семантика Hop_Lim зависит от протокола нижележащего уровня, в который инкапсулируется IOAM, поэтому в данном документе не рассматривается. В это поле **должно** помещаться значение 0xff, если нижележащий уровень не имеет поля, эквивалентного TTL или Hop Limit.

node_id

3-октетное целое число без знака. Поле однозначно указывает узел в IOAM-Namespace и связанном IOAM-Domain. Процедура выделения, управления и отображения node_id выходит за рамки документа. Аспекты node_id, связанные с развёртыванием, рассматриваются в [IPPM-IOAM-DEPLOYMENT].

4.4.2.2. ingress_if_id и egress_if_id в коротком формате

Поля ingress_if_id и egress_if_id образуют 4-октетный блок, как показано ниже.



ingress_if_id

2-октетное целое число без знака, указывающее идентификатор интерфейса, через который принят пакет.

egress_if_id

2-октетное целое число без знака, указывающее идентификатор интерфейса, через который пересылается пакет.

Отметим, что в результате использования в IOAM своих IOAM-Namespace для IOAM-Data-Field поля данных, такие как идентификаторы интерфейсов, можно гибко применять для представления системных ресурсов, связанных со входными или выходными пакетами, например, ingress_if_id может представлять физический, логический или виртуальный интерфейс и даже очередь.

4.4.2.3. Секунды временной метки

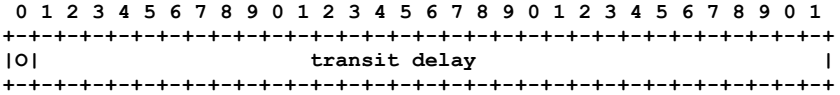
Поле timestamp seconds является 4-октетным целым числом без знака и содержит метку абсолютного времени в секундах для момент получения пакета узлом. Для этого поля могут применяться 3 формата на основе протокола точного времени (Precision Time Protocol или PTP) (см., например, [RFC8877]), NTP [RFC5905] или POSIX [POSIX], описанных в разделе 5. Во всех трёх случаях поле содержит 32 старших бита временной метки в формате, заданном в разделе 5. Если узел не способен указать время, он помещает в это поле значение 0xFFFFFFFF. Отметим, что это значение указывает корректное время в течение 1 секунды примерно один раз за 136 лет. Анализатор сопоставляет несколько пакетов или сравнивает значение метки со своим временем суток для обнаружения ошибки.

4.4.2.4. Дробная часть временной метки

Поле timestamp fraction является 4-октетным целым числом без знака и содержит дробную часть числа секунд с начала эпохи NTP [RFC8877] для момента получения пакета узлом. Для этого поля могут применяться 3 формата на основе протокола точного времени (Precision Time Protocol или PTP) (см., например, [RFC8877]), NTP [RFC5905] или POSIX [POSIX], описанных в разделе 5. Во всех трёх случаях поле содержит 32 младших бита временной метки в формате, заданном в разделе 5. Если узел не способен указать время, он помещает в это поле значение 0xFFFFFFFF. Отметим, что это значение указывает корректное время вформате NTP примерно в течение 233 пикосекунд каждой секунды. При использовании формата NTP анализатор сопоставляет несколько пакетов или сравнивает значение метки со своим временем суток для обнаружения ошибки.

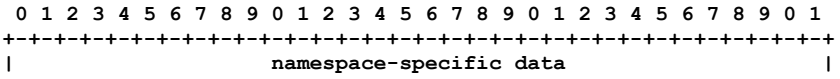
4.4.2.5. Задержка при передаче

Поле transit delay является 4-октетным целым числом без знака из диапазона 0 - 2^{31}-1 и указывает число наносекунд, проведённых пакетом в транзитном режиме, которое может показывать задержку в очередях на узле. Если транзитная задержка больше 2^{31}-1 нсек, устанавливается бит O для индикации переполнения значением поле 0x80000000. Когда поле является частью данных, но узел не способен указать значение, он **должен** установить 0xFFFFFFFF.



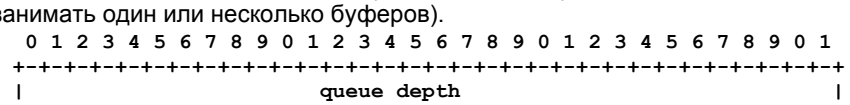
4.4.2.6. Данные, связанные с пространством имён

Поле namespace-specific data является 4-октетным целым числом без знака, которое может использоваться узлом для добавления связанных с IOAM-Namespace данных. Семантика значений поля определяется в контексте IOAM-Namespace.



4.4.2.7. Глубина очереди

Поле queue depth является 4-октетным целым числом без знака и указывает текущий размер выходной очереди на интерфейсе, через который пересылается пакет. Глубина очереди указывается число используемых ею буферов памяти (пакет может занимать один или несколько буферов).



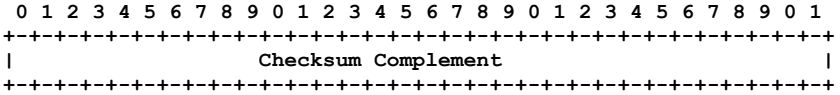
4.4.2.8. Дополнение контрольной суммы

Поле Checksum Complement является 4-октетным целым числом без знака, содержащим значение дополнения контрольной суммы. Это значение полезно при транспортировке IOAM с инкапсуляцией, использующей транспорт UDP, такой как VXLAN-GPE или Geneve. Без Checksum Complement узел IOAM, добавляющий данные, обновляет контрольную сумму UDP в соответствии с рекомендациями протокола инкапсуляции, а при наличии Checksum Complement узле инкапсуляции или транзитный узел IOAM, добавляющий данные, **должен** следовать одному из приведённых вариантов для поддержки корректности UDP Checksum.

1. Повторный расчёт поля UDP Checksum.
2. Использование Checksum Complement для нейтрального к контрольной сумме обновления данных UDP (payload). Полю Checksum Complement назначается значение, которое учитывает добавленные текущим узлом данные, чтобы значение UDP Checksum оставалось корректным.

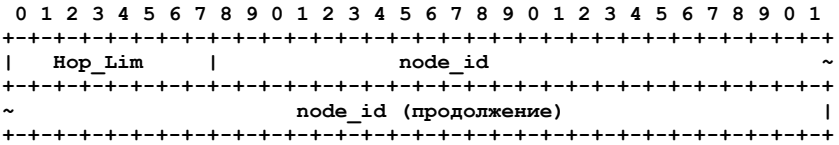
Узлы декапсуляции IOAM **должны** пересчитывать поле UDP Checksum, поскольку они не знают, какие из предшествующих узлов изменили поле UDP Checksum или Checksum Complement.

Поля Checksum Complement аналогичным способом применяются в [RFC7820] и [RFC7821].



4.4.2.9. Hop_Lim и node_id в широком формате

Поля Hop_Lim node_id в широком формате занимают 8 октетов, как показано ниже.



Hop_Lim

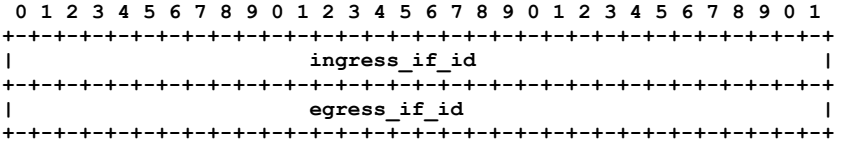
1-октетное целое число без знака (см. параграф 4.4.2.1).

node_id

7-октетное целое число без знака, указывающее идентификатор узла в IOAM-Namespace и связанном IOAM-Domain. Процедура выделения, управления и отображения node_id выходит за рамки документа.

4.4.2.10. ingress_if_id и egress_if_id в широком формате

Поля ingress_if_id и egress_if_id в широком формате занимают 8 октетов, как показано ниже.



ingress_if_id

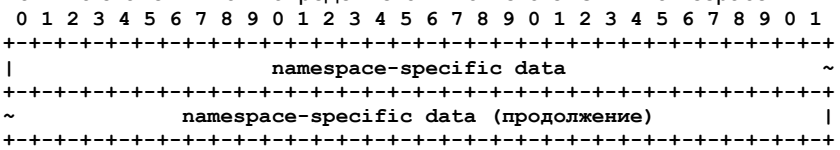
4-октетное целое число без знака, указывающее идентификатор входного интерфейса, принявшего поток.

egress_if_id

4-октетное целое число без знака, указывающее идентификатор выходного интерфейса, переславшего поток.

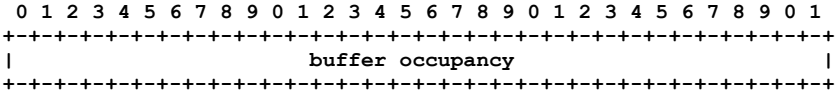
4.4.2.11. Зависящие от Namespace данные в широком формате

Поле namespace-specific занимает 8 октетов, которые могут использоваться узлом для добавления связанных с IOAM-Namespace данных. Семантика значений поля определяется в контексте IOAM-Namespace.

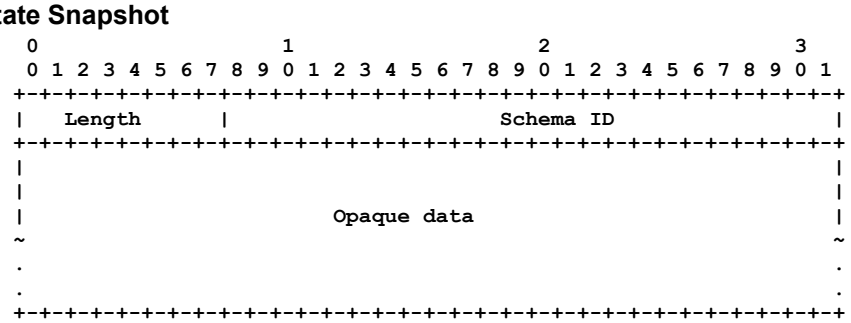


4.4.2.12. Buffer Occupancy

Поле buffer occupancy является 4-октетным целым числом без знака и указывает текущее состояние занятости общего пула буферов, используемого набором очередей, в зависимых от реализации единицах измерения. Значение интерпретируется в контексте IOAM-Namespace и/или идентификатора узла, если он применяется. Авторы отмечают, что в некоторых случаях требуется согласованность единиц на всем пути через сеть, поэтому реализациям рекомендуется применять стандартные единицы, такое как байты.



4.4.2.13. Opaque State Snapshot



Поле Opaque State Snapshot имеет переменный размер и следует приведённым выше правилам для IOAM-Data-Field. Это поле позволяет узлам сети сохранять произвольное состояние на узле данных без предопределённой схемы, а фактическая схема задаётся в контексте IOAM-Namespace и должна быть передана анализатору с помощью отдельного механизма (out-of-band), который данный документ не задаёт. 24-битовое поле Schema ID в контексте IOAM-Namespace указывает конкретную используемую схему и настраивается для сетевого элемента оператором.

Length

1-октетное целое число без знака - размер поля Opaque data, следующего за Schema ID, в 4-октетных блоках.

Schema ID

3- октетное целое число без знака, указывающее схему для поля Opaque data.

Opaque data

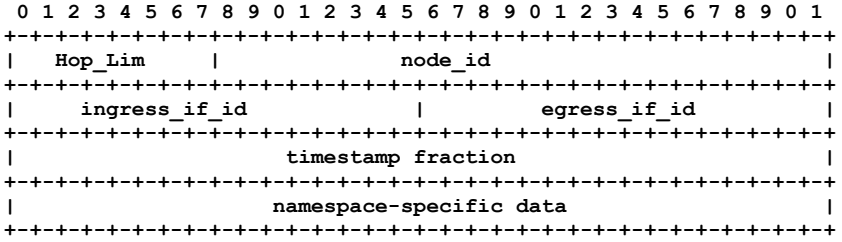
Поле переменного размера, интерпретируемое в соответствии со схемой, указанной в поле Schema ID.

Когда поле является частью данных, но узел не способен указать значение, он должен установить Length = 0 и Schema ID = 0xFFFFFFFF, что говорит об отсутствии схемы.

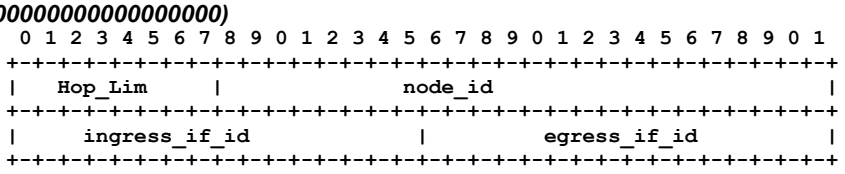
4.4.3. Примеры данных узла IOAM

The format used for the entries in a packet's "node data list" array can vary from packet to packet and deployment to deployment. Some deployments might only be interested in recording the node identifiers, whereas others might be interested in recording node identifiers and timestamps. This section provides example entries of the "node data list" array.

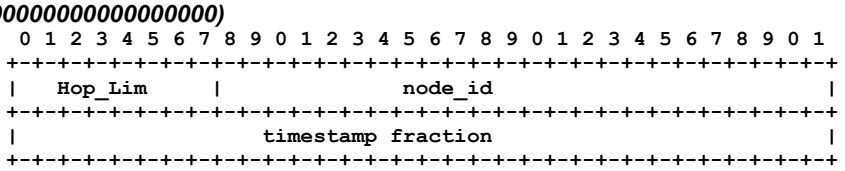
0xD40000 (0b11010100000000000000000)



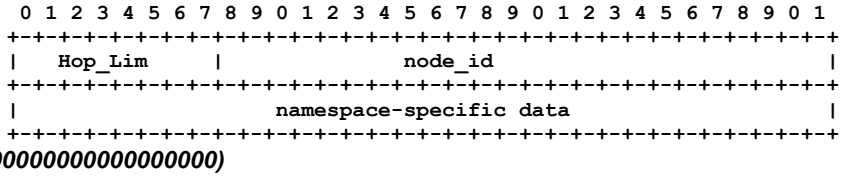
0xC00000 (0b11000000000000000000000)



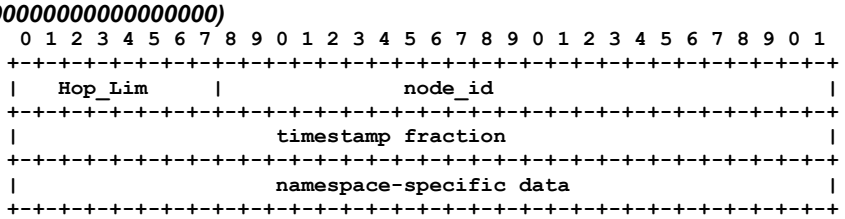
0x900000 (0b10010000000000000000000)



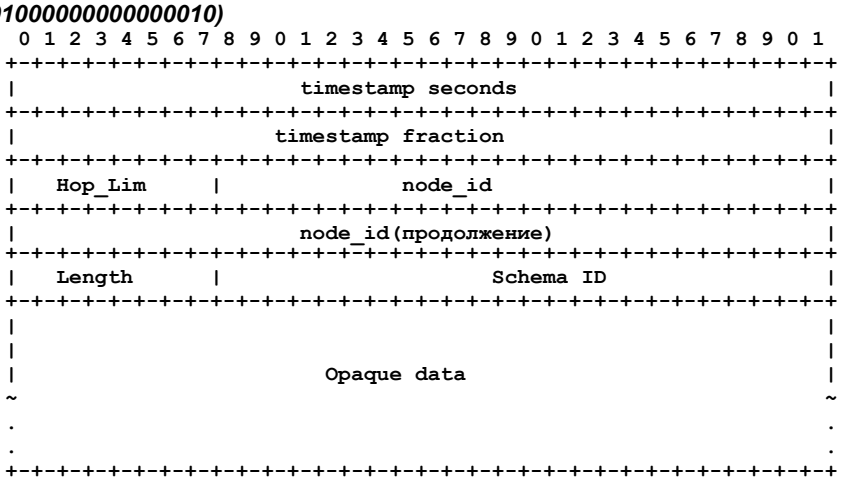
0x840000 (0b10000100000000000000000)



0x940000 (0b10010100000000000000000)



0x308002 (0b00110000100000000000010)



4.5. Подтверждение Transit Option-Type

IOAM Proof of Transit Option-Type (POT) служит для поддержки проверки вариантов использования пути или цепочки функций [RFC7665], т. е. подтверждения того, что трафик прошёл по определённому пути. Хотя детали обработки данных IOAM для POT инкапсулирующими, транзитными и декапсулирующими узлами IOAM выходят за рамки документа, подходы POT разделяют необходимость однозначной идентификации пакетов, а также интерактивной работы с набором сведений, обрабатываемых от узла к узлу. Поэтому в пакет добавляются 2 элемента данных как IOAM-Data-Field.

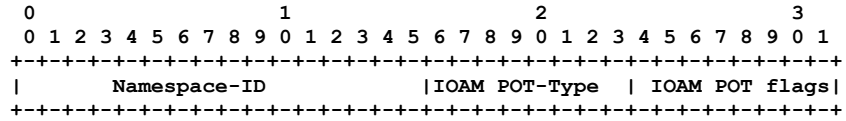
PktID

Уникальный идентификатор пакета.

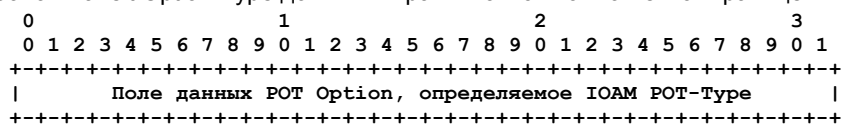
Cumulative

Информация, обрабатываемая от узла к узлу и обновляемая каждым узлом по алгоритму проверки.

IOAM Proof of Transit Option-Type состоит из заголовка IOAM Proof of Transit Option фиксированного размера и полей данных IOAM Proof of Transit Option. Формат заголовка показан ниже.



Поля данных IOAM Proof of Transit Option-Type должны выравниваться по 4-октетной границе.



Namespace-ID

16-битовый идентификатор IOAM-Namespace. Значение 0x0000 определено как Default-Namespace-ID (параграф 4.3) и **должно** быть известно всем узлам, реализующим IOAM. Для любого Namespace-ID, не соответствующего пространству имён на узле, этому узлу **недопустимо** менять содержимое полей IOAM-Data-Field.

IOAM POT-Type

8-битовый идентификатор конкретного варианта POT, задающего данные POT для включения. Этот документ определяет IOAM POT-Type 0

0: данные POT являются 16-октетным полем для переноса данных, связанных с процедурами POT.

Если узел получает непонятное значение IOAM POT-Type, ему **недопустимо** менять, добавлять или удалять содержимое полей IOAM-Data-Field.

IOAM POT flags

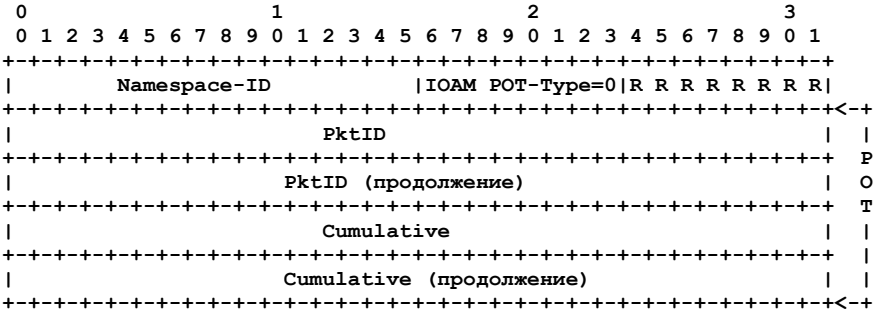
8 битов. Этот документ не задаёт флагов и биты 0-7 доступны для распределения (см. параграф 7.5). Невыделенные биты **должны** сбрасываться (0) при передаче и игнорироваться при получении.

POT Option data

Поле переменного размера, тип которого определяет IOAM POT-Type.

4.5.1. POT типа 0

Формат IOAM Proof of Transit Option с IOAM POT-Type 0 показан на рисунке.



Namespace-ID

16-битовый идентификатор IOAM-Namespace (см. параграф 4.3).

IOAM POT-Type

8-битовый идентификатор конкретного варианта POT, задающего данные POT для включения (см. параграф 4.5). В данном случае устанавливается IOAM POT-Type = 0.

Bit 0-7

Не определены (см. параграф 4.5).

PktID

64-битовый идентификатор пакета.

Cumulative

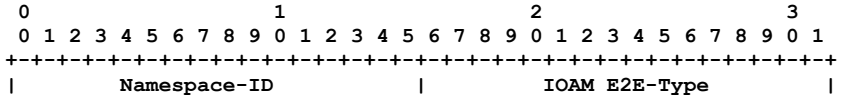
64-битовое значение Cumulative, обновляемое на конкретных узлах путём обработки по значению поля PktID и параметров настройки.

Примечание. Допустимы большие или меньшие значения PktID и Cumulative, которые могут потребоваться в некоторых развёртываниях, например, при ограничении пространства протоколом инкапсуляции. В будущих версиях документа могут быть заданы другие размеры данных POT.

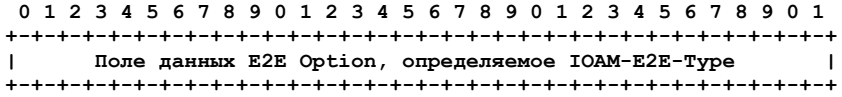
4.6. IOAM Edge-to-Edge Option-Type

IOAM Edge-to-Edge Option-Type содержит данные, добавляемые узлом инкапсуляции и интерпретируемые узлом декапсуляции IOAM. Транзитные узлы IOAM **могут** обрабатывать данные, но им **недопустимо** менять их.

IOAM Edge-to-Edge Option-Type содержит заголовок фиксированного размера IOAM Edge-to-Edge Option-Type и поля данных IOAM Edge-to-Edge Option-Type. Формат заголовка показан на рисунке.



Поля данных IOAM Edge-to-Edge Option-Type **должны** выравниваться по 4-октетной границе.



Namespace-ID

16-битовый идентификатор IOAM-Namespace. Значение 0x0000 определено как Default-Namespace-ID (параграф 4.3) и **должно** быть известно всем узлам, реализующим IOAM. Для любого Namespace-ID, не соответствующего пространству имён на узле, этому узлу **недопустимо** менять содержимое полей IOAM-Data-Field.

IOAM-E2E-Type

16-битовый идентификатор, задающий тип данных, используемых в E2E Option. Это битовое поле и порядок полей данных E2E Option следует показанному ниже порядку битов.

Бит 0

Старший бит, установка (1) которого указывает наличие 64-битового порядкового номера, добавленного к конкретной группе пакетов, используемой для обнаружения потерь, переупорядочения или дублирования пакетов этой группы. Группировка пакетов зависит от развёртывания и определяется инкапсулирующим узлом IOAM, например, путём классификации по n полям (n-tuple) пакетов. При установке этого бита бит 1 (для 32-битового порядкового номера) **должен** быть сброшен (0).

Бит 1

Установка этого бита указывает наличие 32-битового порядкового номера, добавленного к конкретной группе пакетов, используемой для обнаружения потерь, переупорядочения или дублирования пакетов этой группы. Группировка пакетов зависит от развёртывания и определяется инкапсулирующим узлом IOAM, например, путём классификации по n полям (n-tuple) пакетов. При установке этого бита бит 0 (для 64-битового порядкового номера) **должен** быть сброшен (0).

Бит 2

Установка этого бита указывает наличие секунд временной метки, представляющей момент входа пакета в домен IOAM. На узле IOAM время взятия метки может зависеть от реализации. Возможна фиксация времени, когда 1) пакет был получен узлом, 2) передан узлом или 3) инкапсулирован в туннель (при использовании туннельной инкапсуляции. Каждая реализация документирует момент взятия метки E2E для включения в пакет. Это 4-октетное поле метки может иметь формат PTP (см., например, [RFC8877]), NTP [RFC5905] или POSIX [POSIX], как описано в разделе 5. Во всех трёх случаях поле содержит 32 старших бита временной метки в формате, заданном в разделе 5. Если узел не способен указать время, он помещает в это поле значение 0xFFFFFFFF. Отметим, что это значение указывает корректное время в течение 1 секунды примерно один раз за 136 лет. Анализатор сопоставляет несколько пакетов или сравнивает значение метки со своим временем суток для обнаружения ошибки.

Бит 3

Установка этого бита указывает наличие дробной части числа секунд, представляющей момент входа пакета в домен IOAM. time at which the packet entered the IOAM-Domain. Это 4-октетное поле метки может иметь формат PTP (см., например, [RFC8877]), NTP [RFC5905] или POSIX [POSIX], как описано в разделе 5. Во всех трёх случаях поле содержит 32 старших бита временной метки в формате, заданном в разделе 5. Если узел не способен указать время, он помещает в это поле значение 0xFFFFFFFF. Если узел не способен указать время, он помещает в это поле значение 0xFFFFFFFF. Отметим, что это значение указывает корректное время вформате NTP примерно в течение 233 пикосекунд каждой секунды. При использовании формата NTP анализатор сопоставляет несколько пакетов или сравнивает значение метки со своим временем суток для обнаружения ошибки.

Биты 4-15

Не определены. Инкапсулирующий узел IOAM **должен** установить 0 при передаче и игнорировать при получении.

E2E Option data

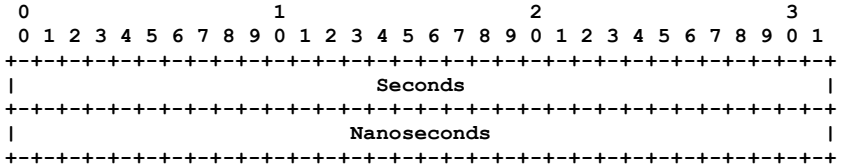
Поле переменного размера, тип которого определяется значением IOAM E2E-Type.

5. Форматы временных меток

Поля IOAM-Data-Field включают временные метки, использующие один из трёх возможных форматов. Предполагается, что плоскость управления способна определить применяемый формат.

5.1. Усечённый формат PTP

Протокол точного времени PTP использует 80-битовые метки времени, из которых ссечённый формат использует 64 старших бита. Усечённый формат PTP задан в параграфе 4.3 [RFC8877] и подробно описан ниже для полноты.



Seconds

Целое число секунд с начала эпохи PTP, заданное 32-битовым значением без знака.

Nanoseconds

Дробная часть числа секунд с начала эпохи PTP, выраженная в наносекундах 32-битовым значением от 0 до 10^9-1 с дискретностью 1 нсек.

Эпоха

PTP (см. например, [RFC8877]).

Переход через максимум

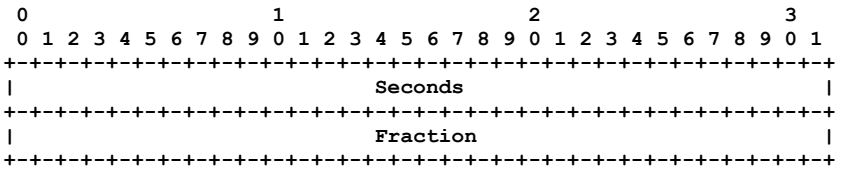
Это поле заполняется каждые 2^{32} секунды, что составляет примерно 136 лет. Следующий максимум будет в 2106 г.

Аспекты синхронизации

Предполагается, что использующие этот протокол узлы синхронизированы между собой. Узлы **могут** синхронизироваться с глобальным источником точного времени. Отметим, что при использовании синхронизации PTP временная метка **может** выводиться из синхронизированных по протоколу PTP часов, что позволяет измерять время относительно часов PTP Grandmaster.

5.2. 64-битовые метки NTP

Протокол сетевого времени (Network Time Protocol или NTP) [RFC5905] задает 64-битовые временные метки. Эта спецификация использует метки NTP в соответствии с параграфом 4.2.1 of [RFC8877], описанный здесь для полноты.



Seconds

Целое число секунд с начала эпохи NTP, заданное 32-битовым значением без знака.

Fraction

Дробная часть числа секунд с начала эпохи NTP, выраженная 32-битовым значением с дискретностью 2^{-32} сек, что соответствует приблизительно 233 пикосекундам.

Эпоха

NTP, см. [RFC5905].

Переход через максимум

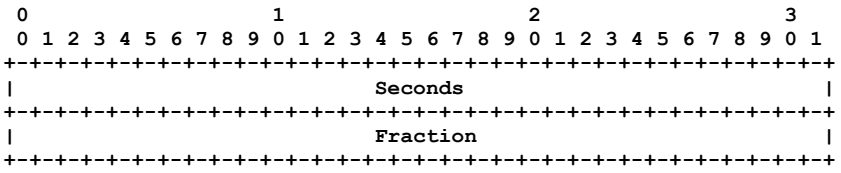
Это поле заполняется каждые 2^{32} секунды, что составляет примерно 136 лет. Следующий максимум будет в 2036 г.

Аспекты синхронизации

Использующие этот протокол узлы обычно синхронизированы с UTC по протоколу NTP [RFC5905]. Таким образом, временная метка **может** выводиться из синхронизированных по протоколу NTP часов, что позволяет измерять время относительно часов сервера NTP.

5.3. Метки POSIX

Эти метки основаны на формате времени POSIX [POSIX]. Спецификация используемого формата приведена ниже.



Seconds

Целое число секунд с начала эпохи POSIX, заданное 32-битовым значением без знака.

Microseconds

Дробная часть числа секунд с начала эпохи POSIX, выраженная в микросекундах 32-битовым значением от 0 до 10^6-1 с дискретностью 1 мсек.

Эпоха

POSIX, см. [POSIX], Приложение A.4.16.

Переход через максимум

Это поле заполняется каждые 2^{32} секунды, что составляет примерно 136 лет. Следующий максимум будет в 2106 г.

Аспекты синхронизации

Предполагается, что использующие этот формат узлы работают на основе операционной системы Linux и применяют время POSIX. В некоторых случаях узлы могут синхронизироваться с UTC с помощью механизма, выходящего за рамки этого документа, например, NTP [RFC5905]. Таким образом, временная метка **может** выводиться из синхронизированных по протоколу NTP часов, что позволяет измерять время относительно часов сервера NTP.

6. Экспорт данных IOAM

Узлы IOAM собирают сведения о пакетах, проходящих через домен с поддержкой IOAM. Инкапсулирующие и транзитные узлы IOAM могут извлекать информацию из пакетов, обрабатывать её и экспортировать, например, с помощью IP Flow Information Export (IPFIX). Механизмы и форматы данных для экспорта данных IOAM выходят за рамки документа.

Способ экспорта необработанных данных IOAM с помощью IPFIX описан в [IPPM-IOAM-RAWEXPORT].

7. Взаимодействие с IANA

Агентство IANA определило группу реестров In Situ OAM (IOAM), ключающую:

IOAM Option-Type;
IOAM Trace-Type;
IOAM Trace-Flags;
IOAM POT-Type;
IOAM POT-Flags;
IOAM E2E-Type;
IOAM Namespace-ID.

Эти реестры более подробно описаны ниже.

7.1. Реестр IOAM Option-Type

Этот реестр определяет 128 кодов полей IOAM Option-Type для идентификации типов IOAM-Option-Type, как описано в разделе 4. Этот документ определяет 4 кодо:

0: IOAM Pre-allocated Trace Option-Type;
1: IOAM Incremental Trace Option-Type;
2: IOAM POT Option-Type;
3: IOAM E2E Option-Type.

Коды 4 - 127 доступны для распределения по процедуре IETF Review, как указано в [RFC8126].

При регистрации новых кодов **должен** использоваться приведённый ниже шаблон.

Name: имя регистрируемого Option-Type;
Code point: желаемое значение кода;
Description: краткое описание регистрируемого Option-Type
Reference: ссылка на документ с определением нового Option-Type.

При оценке новых регистраций **должно** проверяться наличие в новом IOAM-Option-Type поля IOAM-Namespace и его размещение в качестве первого поля заголовка для нового Option-Type.

7.2. Реестр IOAM Trace-Type

Этот реестр определяет назначение для каждого бита поля IOAM Trace-Type для Pre-allocated Trace Option-Type и Incremental Trace Option-Type, определённых в параграфе 4.4. Биты 0 - 11 определены в параграфе 4.4.1.

Бит 0: hop_Lim и node_id в коротком формате;
Бит 1: ingress_if_id и egress_if_id в коротком формате;
Бит 2: секунды timestamp;
Бит 3: доли секунд;
Бит 4: транзитная задержка;
Бит 5: зависящие от пространства имён данные в коротком формате;
Бит 6: глубина очереди;
Бит 7: дополнение контрольной суммы;
Бит 8: hop_Lim и node_id в широком формате;
Бит 9: ingress_if_id и egress_if_id в широком формате;
Бит 10: зависящие от пространства имён данные в широком формате;
Бит 11: занятость буфера
Бит 22: Opaque State Snapshot с переменным размером;
Бит 23: резерв
Биты 12-21 доступны для распределения по процедуре IETF Review, как указано в [RFC8126].

При регистрации новых кодов **должен** использоваться приведённый ниже шаблон.

Bit: желаемый бит для выделения в 24-битовом поле IOAM Trace Option-Type для Pre-allocated Trace Option-Type и Incremental Trace Option-Type;
Description: краткое описание регистрируемого бита;
Reference: ссылка на документ с определением нового бита.

7.3. Реестр IOAM Trace-Flags

Этот реестр определяет биты флагов для Pre-allocated Trace-Option и Incremental Trace-Option, определённых в параграфе 4.4.1.

Бит 0: Переполнение (бит O)
Биты 1-3 доступны для распределения по процедуре IETF Review, как указано в [RFC8126].

При регистрации новых кодов **должен** использоваться приведённый ниже шаблон.

Bit: желаемый бит для выделения в поле флагов Pre-allocated Trace Option-Type и Incremental Trace Option-Type;
Description: краткое описание регистрируемого бита;
Reference: ссылка на документ с определением нового бита.

7.4. Реестр IOAM POT-Type

Этот реестр определяет 256 кодов IOAM POT-Type для IOAM Proof of Transit Option (параграф 4.5). Данный документ выделяет лишь код 0.

0: 16 октетов данных POT.

Коды 1-255 доступны для распределения по процедуре IETF Review, как указано в [RFC8126].

При регистрации новых кодов **должен** использоваться приведённый ниже шаблон.

Name: Имя регистрируемого POT-Type;
Code point: желаемое значение для регистрируемого кода;
Description: краткое описание регистрируемого POT-Type;
Reference: ссылка на документ с определением нового POT-Type.

7.5. Реестр IOAM POT-Flags

Этот реестр определяет биты флагов для типа IOAM POT Option-Type, определённого в параграфе 4.5.

Биты 0-7 доступны для распределения по процедуре IETF Review, как указано в [RFC8126].

При регистрации новых кодов **должен** использоваться приведённый ниже шаблон.

Bit: желаемый бит для выделения в 8-битовом поле флагов IOAM POT Option-Type;
Description: краткое описание регистрируемого бита;
Reference: ссылка на документ с определением нового бита.

7.6. Реестр IOAM E2E-Type

Этот реестр определяет биты 16-битового поля IOAM E2E-Type для IOAM E2E Option (параграф 4.6). Данный документ выделяет биты 0-3.

Бит 0: 64-битовый порядковый номер;
Бит 1: 32-битовый порядковый номер;
Бит 2: секунды timestamp;
Бит 3: доли секунды timestamp.
Биты 4-15 доступны для распределения по процедуре IETF Review, как указано в [RFC8126].

При регистрации новых кодов **должен** использоваться приведённый ниже шаблон.

Bit: желаемый бит для выделения в 16-битовом поле флагов IOAM E2E-Type;
Description: краткое описание регистрируемого бита;
Reference: ссылка на документ с определением нового бита.

7.7. Реестр IOAM Namespace-ID

Агентство IANA организовало реестр IOAM Namespace-ID, содержащий 16-битовые значения и соответствующий шаблону для запросов, приведённому ниже. Определение для 0x0000 приведено в этом документе, значения 0x0001 - 0x7FFF зарезервированы для приватного использования (управляются операторами), как указано в параграфе 4.3 данного документа. Значения 0x8000 - 0xFFFF будут выделяться по процедуре Expert Review в соответствии с [RFC8126].

При получении нового запроса назначенные эксперты выполняют указанные ниже действия.

- Проверка полноты запроса, т. е. соответствия шаблону. Неполные запросы возвращаются заявителю.
- Проверка дублирования имеющихся или ожидающих пространств имён и конфликтов с ними. При наличии дублирования или конфликта запрос возвращается заявителю для исправления.
- Запрос отзывов соответствующих рабочих групп и сообществ для обеспечения должного рассмотрения запроса и предварительного согласия с ним. Эксперт будет запрашивать отклик по меньшей мере от рабочей группы IPPM путём отправки запроса в почтовую конференцию ippm@ietf.org. На ожидание откликов отводятся 3 недели. При получении откликов от рабочих групп и сообществ м предварительным согласием эксперт будет одобрять запрос и просить IANA выделить новое значение Namespace-ID. Если эксперт не получит одобрения в откликах, он будет просить заявителя связаться с соответствующими рабочими группами и сообщеятвами для дальнейшего рассмотрения и уточнения запроса.

Предполагается, что выделение пространств имён будет сопровождаться публикацией RFC. Для выделения пространства до одобрения публикации RFC назначенный эксперт может утвердить выделение, как только ему станет ясно, что RFC будет опубликован.

0x0000: принятое по умолчанию пространство имён (известно всем узлам IOAM);
0x0001 - 0x7FFF: резерв для приватного использования;
0x8000 - 0xFFFF: не распределены.

При регистрации новых кодов **должен** использоваться приведённый ниже шаблон.

Name: имя регистрируемого Namespace-ID;
Code point: желаемое значение кода для регистрируемого Namespace-ID;
Description: краткое описание регистрируемого Namespace-ID;
Reference: ссылка на документ с определением регистрируемого Namespace-ID;
Status of the registration: регистрация может быть постоянной (permanent) или временной (provisional). Регистрации Namespace-ID, успешно прошедшие рецензирование экспертов, получают статус provisional. После публикации RFC для нового Namespace-ID статус меняется на permanent.

8. Вопросы управления и развёртывания

Этот документ определяет структуру и применение полей IOAM-Data-Field, но не задаёт инкапсуляцию этих полей в те или иные протоколы. Вопросы развёртывания и управления IOAM должны рассматриваться в контексте протокола инкапсуляции IOAM-Data-Field, поэтому не обсуждаются в документе. Развёртыванию IOAM посвящён документ [IPPM-IOAM-DEPLOYMENT], где описана структура развёртывания и приведены практические примеры.

9. Вопросы безопасности

Как обсуждалось в [RFC7276], успешная атака на протокол OAM в целом и IOAM, в частности, может помешать обнаружению отказов и аномалий, а также создать иллюзию их отсутствия. В частности, такие угрозы относятся к нарушению целостности данных IOAM путём изменения параметров IOAM в пути или внедрения пакетов со злонамеренными опциями IOAM. Такой атаке могут быть подвергнуты все узлы на пути доставки пакетов IOAM.

Тип Proof of Transit Option-Type (параграф 4.5) служит для проверки пути пакетов данных, т. е. подтверждения того, что пакеты проходят через определённый набор узлов.

Если атакующий имеет доступ к некоторым узлам сети и может менять программы на этих узлах, он сможет воспользоваться полями IOAM-Data-Field для целей, отличных от указанных в этом документе, например, для сокрытия несанкционированных каналов между узлами сети.

Несмотря на то, что параметры IOAM не должны раскрывать (содержать) пользовательских данных, ими можно воспользоваться для сетевой разведки с целью сбора сведений о путях, производительности, состоянии очередей и буферов и т. п. Если данные IOAM уходят за пределы домена IOAM, это может позволить сбор сведений о IOAM-Domain, например, для оценки эффективности предпринятой атаки в плане переполнения очередей и буферов.

IOAM может служить для организации или усиления атак на службы (Denial-of-Service или DoS). Например, атакующий может добавлять заголовок IOAM к пакетам с целью истощения ресурсов на сетевых устройствах, участвующих в IOAM, или элементов, принимающих, собирающих или анализирующих данные IOAM. Другим примером является атака

с размером пакетов, где злоумышленник помещает связанные с IOAM-Option-Type заголовки в пакеты данных, что ведёт к фрагментированию или отбрасыванию пакетов по причине превышения MTU. В случае применения POT атакующий может повреждать поля данных POT в пакетах для отказа при проверке даже в случае корректного пути.

Опции IOAM могут включать временные метки, поэтому при использовании сетевыми устройствами протоколов синхронизации на эти протоколы возможны атаки [RFC7384] способные нарушить целостность меток времени.

В плоскости управления атаки могут выполняться путём нарушения настроек узлов IOAM, позволяющего организовать другие атаки. Настройку IOAM следует разрешать лишь уполномоченным процессам или пользователям.

Протоколам IETF требуются функции, обеспечивающие безопасность. Хотя поля IOAM-Data-Field сами не представляют протокола, они добавляют протокол, который служит для инкапсуляции IOAM-Data-Field. Любая спецификация, задающая передачу полей IOAM-Data-Field протоколом инкапсуляции, **должна** обеспечивать механизм криптографической защиты целостности IOAM-Data-Field. Такая защита может быть обеспечена механизмом протокола инкапсуляции или с помощью механизмов, указанных в [IPPM-IOAM-DATA-INTEGRITY].

Данный документ не задаёт конкретной инкапсуляции IOAM и следует отметить, что с некоторыми типами инкапсуляции IOAM могут быть связаны проблемы безопасности. Предполагается, что спецификации, задающие инкапсуляцию IOAM, будут учитывать аспекты защиты, связанные с инкапсуляцией.

Важно отметить, что развёртывание IOAM предполагается в доменах с ограничениями, что сужает фронт возможных атак. Административный домен с ограничениями позволяет его оператору средства для выбора, отслеживания и контроля доступа ко всем сетевым устройствам, делая их доверенными для оператора. Для ограничения области действия упомянутых угроз конкретным доменом с ограничениями от операторов ожидается применение правил, предотвращающих утечку трафика IOAM за пределы IOAM-Domain и раскрытие данных IOAM вне домена.

Этот документ не определяет содержимого настраиваемых полей IOAM-Data-Field, таких как Opaque State Snapshot или зависящие от пространства имён данные. С такими полями могут быть связаны вопросы безопасности, которые должны рассматриваться в документах, определяющих содержимое и формат полей.

Системы IOAM, где применяются оба типа опций IOAM Trace Option-Type (Pre-allocated Trace Option-Type и Incremental Trace Option-Type) могут страдать от неполной видимости, если информацию, собираемую через два Trace Option-Type, должным образом не сопоставлять и не агрегировать. Если транзитные узлы IOAM используют IOAM-Data-Field в пакетах для дальнейших действий или анализа, узлы с поддержкой лишь одного IOAM Trace Option-Type в системе IOAM с двумя Trace Option-Type будут иметь ограниченную видимость, что может приводить к неверным выводам или действиям.

Вопросы безопасности систем, развёртывающих IOAM, как и любых иных систем, должны рассматриваться для каждого варианта развёртывания на основе анализа угроз для конкретной системы, что может потребовать специальных решений для защиты, выходящих за рамки этого документа. В частности, при развёртывании IOAM, не ограниченном одной ЛВС и соединяющем несколько сайтов (например, через наложенную сеть) каналы между сайтами должны быть защищены (например, с помощью IPsec) для предотвращения угроз извне.

Вопросы развёртывания IOAM, включая подходы к снижению рассмотренных выше угроз и возможных атак, выходят за рамки этого документа. Развёртывание IOAM обсуждается в [IPPM-IOAM-DEPLOYMENT].

10. Литература

10.1. Нормативные документы

[POSIX] IEEE, "IEEE/Open Group 1003.1-2017 - IEEE Standard for Information Technology--Portable Operating System Interface (POSIX(TM)) Base Specifications, Issue 7", IEEE Std 1003.1-2017, January 2018, <https://standards.ieee.org/ieee/1003.1/7101/>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <https://www.rfc-editor.org/info/rfc5905>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

10.2. Дополнительная литература

[IPPM-IOAM-DATA-INTEGRITY] Brockners, F., Bhandari, S., Mizrahi, T., and J. Iurman, "Integrity of In-situ OAM Data Fields", Work in Progress, Internet-Draft, draft-ietf-ippm-ioam-data-integrity-01, 2 March 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-ippm-ioam-data-integrity-01>.

[IPPM-IOAM-DEPLOYMENT] Brockners, F., Bhandari, S., Bernier, D., and T. Mizrahi, "In-situ OAM Deployment", Work in Progress, Internet-Draft, draft-ietf-ippm-ioam-deployment-01, 11 April 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-ippm-ioam-deployment-01>.

[IPPM-IOAM-RAWEXPORT] Spiegel, M., Brockners, F., Bhandari, S., and R. Sivakolundu, "In-situ OAM raw data export with IPFIX", Work in Progress, Internet-Draft, draft-spiegel-ippm-ioam-rawexport-06, 21 February 2022, <https://datatracker.ietf.org/doc/html/draft-spiegel-ippm-ioam-rawexport-06>.

[IPV6-RECORD-ROUTE] Kitamura, H., "Record Route for IPv6 (RR6) Hop-by-Hop Option Extension", Work in Progress, Internet-Draft, draft-kitamura-ipv6-record-route-00, 17 November 2000, <https://datatracker.ietf.org/doc/html/draft-kitamura-ipv6-record-route-00>.

[NVO3-VXLAN-GPE] Maino, F., Ed., Kreeger, L., Ed., and U. Elzur, Ed., "Generic Protocol Extension for VXLAN (VXLAN-GPE)", Work in Progress, Internet-Draft, draft-ietf-nvo3-vxlan-gpe-12, 22 September 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-nvo3-vxlan-gpe-12>.

[RFC7276] Mizrahi, T., Sprecher, N., Bellagamba, E., and Y. Weingarten, "An Overview of Operations, Administration, and Maintenance (OAM) Tools", RFC 7276, DOI 10.17487/RFC7276, June 2014, <https://www.rfc-editor.org/info/rfc7276>.

[RFC7384] Mizrahi, T., "Security Requirements of Time Protocols in Packet Switched Networks", RFC 7384, DOI 10.17487/RFC7384, October 2014, <https://www.rfc-editor.org/info/rfc7384>.

[RFC7665] Halpern, J., Ed. and C. Pignataro, Ed., "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015, <https://www.rfc-editor.org/info/rfc7665>.

[RFC7799] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <https://www.rfc-editor.org/info/rfc7799>.

[RFC7820] Mizrahi, T., "UDP Checksum Complement in the One-Way Active Measurement Protocol (OWAMP) and Two-Way Active Measurement Protocol (TWAMP)", RFC 7820, DOI 10.17487/RFC7820, March 2016, <https://www.rfc-editor.org/info/rfc7820>.

[RFC7821] Mizrahi, T., "UDP Checksum Complement in the Network Time Protocol (NTP)", RFC 7821, DOI 10.17487/RFC7821, March 2016, <https://www.rfc-editor.org/info/rfc7821>.

[RFC8300] Quinn, P., Ed., Elzur, U., Ed., and C. Pignataro, Ed., "Network Service Header (NSH)", RFC 8300, DOI 10.17487/RFC8300, January 2018, <https://www.rfc-editor.org/info/rfc8300>.

[RFC8799] Carpenter, B. and B. Liu, "Limited Domains and Internet Protocols", RFC 8799, DOI 10.17487/RFC8799, July 2020, <https://www.rfc-editor.org/info/rfc8799>.

[RFC8877] Mizrahi, T., Fabini, J., and A. Morton, "Guidelines for Defining Packet Timestamps", RFC 8877, DOI 10.17487/RFC8877, September 2020, <https://www.rfc-editor.org/info/rfc8877>.

[RFC8926] Gross, J., Ed., Ganga, I., Ed., and T. Sridhar, Ed., "Geneve: Generic Network Virtualization Encapsulation", RFC 8926, DOI 10.17487/RFC8926, November 2020, <https://www.rfc-editor.org/info/rfc8926>.

Благодарности

Авторы благодарны Éric Vyncke, Nalini Elkins, Srihari Raghavan, Ranganathan T S, Karthik Babu Harichandra Babu, Akshaya Nadahalli, LJ Wobker, Erik Nordmark, Vengada Prasad Govindan, Andrew Yourtchenko, Aviv Kfir, Tianran Zhou, Zhenbin (Robin) и Greg Mirsky за комментарии и советы.

Этот документ использует несколько концепций, описанных в [IPV6-RECORD-ROUTE]. Авторы хотели бы отметить работу Hiroshi Kitamura и других создателей этого документа.

Авторы выражают признательность за полезные советы и проницательные комментарии Joe Clarke, Al Morton, Tom Herbert, Carlos J. Bernardos, Haoyu Song, Mickey Spiegel, Roman Danyliw, Benjamin Kaduk, Murray S. Kucherawy, Ian Swett, Martin Duke, Francesca Palombini, Lars Eggert, Alvaro Retana, Erik Kline, Robert Wilton, Zaheduzzaman Sarker, Dan Romascanu, Barak Gafni.

Участники работы

Этот документ является результатом коллективной работы. Текст и содержимое подготовлены редакторами и указанными ниже соавторами.

Carlos Pignataro
Cisco Systems, Inc.
Research Triangle Park
7200-11 Kit Creek Road
NC 27709
United States of America
Email: cpignata@cisco.com

Mickey Spiegel
Barefoot Networks, an Intel company
101 Innovation Drive
San Jose, CA 95134-1941
United States of America
Email: mickey.spiegel@intel.com

Barak Gafni
Nvidia
Suite 100
350 Oakmead Parkway
Sunnyvale, CA 94085
United States of America
Email: gbarak@nvidia.com

Jennifer Lemon
Broadcom
270 Innovation Drive
San Jose, CA 95134
United States of America
Email: jennifer.lemon@broadcom.com

Hannes Gredler
RtBrick Inc.
Email: hannes@rtbrick.com

John Leddy
United States of America
Email: john@leddy.net

Stephen Youell
JP Morgan Chase
25 Bank Street
London
E14 5JP
United Kingdom
Email: stephen.youell@jpmorgan.com

David Mozes
Email: mosesster@gmail.com

Petr Lapukhov
Facebook
1 Hacker Way
Menlo Park, CA 94025
United States of America

Email: petr@fb.com

Remy Chang
Barefoot Networks, an Intel company
101 Innovation Drive
San Jose, CA 95134-1941
United States of America
Email: remy.chang@intel.com

Daniel Bernier
Bell Canada
Canada
Email: daniel.bernier@bell.ca

Адреса авторов

Frank Brockners (editor)
Cisco Systems, Inc.
3rd Floor
Nordhein-Westfalen
Hansaallee 249
40549 Duesseldorf
Germany
Email: fbrockne@cisco.com

Shwetha Bhandari (editor)
Thoughtspot
3rd Floor
Indiqube Orion
Garden Layout
HSR Layout
24th Main Rd
Bangalore 560 102
Karnataka
India
Email: shwetha.bhandari@thoughtspot.com

Tal Mizrahi (editor)
Huawei
8-2 Matam
Haifa 3190501
Israel
Email: tal.mizrahi.phd@gmail.com

Перевод на русский язык

Николай Малых

nmalykh@protokols.ru