Internet Engineering Task Force (IETF)
Request for Comments: 7408
Updates: 5812
Category: Standards Track
ISSN: 2070-1721

E. Haleplidis
University of Patras

November 2014

Forwarding and Control Element Separation (ForCES) Model Extension

Расширение модели ForCES

Аннотация

Этот документ расширяет модель ForCES[1], определённую в RFC 5812, и обновляет этот документ с целью поддержки комплексных типов для метаданных, необязательных принятых по умолчанию значений для типов данных и необязательных типов доступа для структур. Документ также решает вопросы, связанные с наследованием LFB[2] и добавляет две новых возможности - условие для событий eventBecomesEqualTo и свойства LFB. Внесённые этим документом изменения не меняют протокол и совместимы с прежними моделями LFB.

Статус документа

Документ является проектом стандарт Internet (Internet Standards Track).

Документ является результатом работы IETF[3] и представляет согласованный взгляд сообщества IETF. Документ прошёл открытое обсуждение и был одобрен для публикации IESG[4]. Не все одобренные IESG документы претендуют на статус Internet Standard (см. раздел 2 в RFC 5741).

Информацию о текущем статусе документа, ошибках и способах обратной связи можно найти по ссылке http://www.rfc-editor.org/info/rfc7408.

Оглавление

1. Введение

Модель ForCES [RFC5812] представляет формальный способ определения логических функциональных блоков (LFB) элементов пересылки (FE[5]) с использованием языка XML[6]. Документ [RFC5812] был опубликован за несколько лет до данного документа и опыт показал необходимость добавления в модель новых понятий и изменения имеющихся.

В частности, этот документ обновляет модель ForCES [RFC5812] с целью поддержки комплексных типов для метаданных (параграф 2.1), необязательных принятых по умолчанию значений для типов данных (параграф 2.2) и

[1]Forwarding and Control Element Separation - разделение элементов пересылки и управления.
[2]Logical Functional Block - логический функциональный блок.
[3]Internet Engineering Task Force - комиссия по решению инженерных задач Internet.
[4]Internet Engineering Steering Group - комиссия по инженерным разработкам Internet.
[5]Forwarding Element.
[6]eXtensible Markup Language.

необязательных типов доступа для структур (параграф 2.3). Решена также проблема с наследованием классов LFB (параграф 2.6). Дополнительно в документе введены две новых возможности - условие для событий eventBecomesEqualTo (параграф 2.4) и свойства LFB (параграф 2.5).

Эти расширения обновляют модель ForCES [RFC5812], но не требуют изменения протокола ForCES [RFC5810], поскольку они просто меняют определение схемы. Библиотеки XML, созданные с использованием прежней схемы, остаются действительными в новой схеме. Чтобы библиотеки XML, создаваемые новой схемой, были совместимы с имеющимися реализациями ForCES, в эти библиотеки XML **недопустимо** включать возможности, определённые в данном документе.

Расширения схемы включают теги <!-- Расширение RFC 7408 --> и <!-- /Расширение RFC 7408 -->, обозначающие начало и конец текста расширения, задаваемого этим документом применительно к схеме в исходной модели ForCES [RFC5812].

1.1. Уровни требований

Ключевые слова **необходимо** (MUST), **недопустимо** (MUST NOT), **требуется** (REQUIRED), **нужно** (SHALL), **не нужно** (SHALL NOT), **следует** (SHOULD), **не следует** (SHOULD NOT), **рекомендуется** (RECOMMENDED), **возможно** (MAY), **необязательно** (OPTIONAL) в данном документе должны интерпретироваться в соответствии с [RFC2119].

1.2. Термины

Этот документ использует термины, определённые в модели ForCES [RFC5812]. В частности, предполагается знакомство читателя с приведёнными ниже терминами.

FE Model - модель FE.

LFB (Logical Functional Block) Class - класс (или тип) LFB.

LFB Instance - экземпляр LFB.

LFB Model - модель LFB.

Element - элемент.

Attribute - атрибут.

LFB Metadata - метаданные LFB.

ForCES Component - компонента ForCES.

LFB Class Library - библиотека классов LFB.

2. Расширения модели ForCES

2.1. Комплексные типы для метаданных

Параграф 4.6 модели ForCES [RFC5812] разрешает применять для метаданных лишь неделимые (atomic) типы. На рисунке 1 показан фрагмент схемы XML, где разрешены лишь typeRef и atomic для определений метаданных.

```
<xsd:complexType name="metadataDefsType">
  <xsd:sequence>
    <xsd:element name="metadataDef" maxOccurs="unbounded">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="name" type="xsd:NMTOKEN"/>
          <xsd:element ref="synopsis"/>
          <xsd:element name="metadataID" type="xsd:integer"/>
          <xsd:element ref="description" minOccurs="0"/>
          <xsd:choice>
            <xsd:element name="typeRef" type="typeRefNMTOKEN"/>
            <xsd:element name="atomic" type="atomicType"/>
          </xsd:choice>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

Рисунок 1. Изначальное определение metadataDefsType в схеме.

Однако были случаи использования комплексных метаданных в пути передачи данных, например, два простых случая описаны в версии 1.1.0 (и последующих) спецификации коммутатора OpenFlow [OpenFlowSpec1.1].

1. Метаданные Action Set представляют собой массив описаний действий, которые передаются через конвейер вместе с пакетами данных.
2. Когда пакет приходит от контроллера, он может сопровождаться списком действий (как метаданными), выполняемых до передачи пакета в конвейер обработки. Список действий также является массивом.

Расширение, показанное на рисунке 2, разрешает для метаданных комплексные типы, в частности, массивы и структуры. Требуются объявления ключей для проверки пригодности ключей содержимого в массивах и componentID в структурах.

```
<xsd:complexType name="metadataDefsType">
  <xsd:sequence>
    <xsd:element name="metadataDef" maxOccurs="unbounded">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="name" type="xsd:NMTOKEN"/>
          <xsd:element ref="synopsis"/>
          <xsd:element name="metadataID" type="xsd:integer"/>
          <xsd:element ref="description" minOccurs="0"/>
          <xsd:choice>
            <xsd:element name="typeRef" type="typeRefNMTOKEN"/>
            <xsd:element name="atomic" type="atomicType"/>
            <!-- Расширение RFC 7408 -->
            <xsd:element name="array" type="arrayType">
              <xsd:key name="contentKeyID1">
                <xsd:selector xpath="lfb:contentKey"/>
                <xsd:field xpath="@contentKeyID"/>
              </xsd:key>
            </xsd:element>
            <xsd:element name="struct" type="structType">
              <xsd:key name="structComponentID1">
                <xsd:selector xpath="lfb:component"/>
                <xsd:field xpath="@componentID"/>
              </xsd:key>
            </xsd:element>
            <!-- /Расширение RFC 7408 -->
          </xsd:choice>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

Рисунок 2. Новое определение metadataDefsType в схеме.

2.2. Необязательные значения по умолчанию для типов данных

В исходной схеме используемые по умолчанию значения могли определяться лишь для типов данных, определённых в компонентах LFB, но не в структурах или массивах. Поэтому принятые по умолчанию значения для многократно используемых элементов (например, счётчики с начальным значением 0) приходилось постоянно переопределять. Кроме того, комплексные типы данных не могли определяться с принятыми по умолчанию значениями (например, нельзя было определить в структуре счётчик с начальным значением 0).

Это расширение позволяет определять принятые по умолчанию значения для типов данных. Эти типы могут указываться как простые компоненты или включаться в составные типы, такие как структуры. Простым примером является структура, где одной из компонент является счётчик с начальным значением 0. Для решения задачи нужно сначала определить счётчик как тип данных с нужным значением по умолчанию, а затем указать этот тип в структуре. Принятые по умолчанию значения **должны** следовать приведённым ниже формальным ограничениям.

1. Принятые по умолчанию значения **должны** игнорироваться, если тип данных не является неделимым (atomic) или базовым.

2. Когда тип данных X с принятым по умолчанию значением A указывается в типе данных Y с принятым по умолчанию значением B, это будет переопределять принятое по умолчанию значение внутри типа Y (т. е. в нем по умолчанию будет применяться значение B).

3. Принятые по умолчанию значения компонент LFB переопределяют любые заданные по умолчанию значения из определений dataTypeDef.

4. Принятые по умолчанию значения типов данных, указанных в возможностях или метаданных, **должны** игнорироваться.

```
<xsd:complexType name="dataTypeDefsType">
  <xsd:sequence>
    <xsd:element name="dataTypeDef" maxOccurs="unbounded">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="name" type="xsd:NMTOKEN"/>
          <xsd:element name="derivedFrom" type="xsd:NMTOKEN"
             minOccurs="0"/>
          <xsd:element ref="synopsis"/>
          <xsd:element ref="description" minOccurs="0"/>
          <xsd:group ref="typeDeclarationGroup"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

Рисунок 3. Выдержка из исходного определения dataTypeDefsType в схеме.

Это расширение просто добавляет определение dataTypeDefsType в схему XML, показанную на рисунке 3, чтобы позволить принятые по умолчанию значения для dataTypeDefsType. Новое определение показано на рисунке 4.

Примеры использования принятых по умолчанию значений показаны на рисунке 5.

```
<dataTypeDef>
  <name>ZeroCounter</name>
  <synopsis>Счётчик с начальным значением 0</synopsis>
  <typeRef>uint32</typeRef>
  <defaultValue>0</defaultValue>
</dataTypeDef>
<dataTypeDef>
  <name>CounterValues</name>
  <synopsis>Пример значения по умолчанию в структуре</synopsis>
  <struct>
    <component componentID="1">
      <name>GoodPacketCounter</name>
      <synopsis>Счётчик «хороших» пакетов</synopsis>
      <typeRef>ZeroCounter</typeRef>
    </component>
    <component componentID="2">
      <name>BadPacketCounter</name>
      <synopsis>Счётчик «плохих» пакетов</synopsis>
      <typeRef>ZeroCounter</typeRef>
    </component>
  </struct>
</dataTypeDef>
```

Рисунок 5. Пример необязательных значений для использования по умолчанию.

2.3. Необязательны типы доступа для структур

В исходной схеме тип доступа мог быть определён лишь для компонент LFB, но не для компонент массива или структуры. Это означало невозможность тонкой настройки правил доступа к компонентам структур. Простым примером может служить структура с доступом «чтение и запись», одной из компонент которой является счётчик, для которого имеет смысл доступ «чтение и сброс» или «только чтение».

Это расширение позволяет указать тип доступа к компонентам структуры при определении типа данных или компоненты LFB.

При определении необязательного типа доступа к компонентам структуры этот тип **должен** переопределять тип доступа, заданный в определении структуры. Например, если для структуры задан доступ на чтение и запись, но она включает счётчик с доступом лишь для чтения, для этого счётчика **должен** быть задан режим доступа read-only.

В соответствии с [RFC5812] для компонент возможностей всегда доступно только чтение. Если задан тип доступа для компоненты в возможности, этот тип **должен** игнорироваться. Точно так же **должен** игнорироваться тип доступа для структур в метаданных.

```
<xsd:complexType name="structType">
  <xsd:sequence>
    <xsd:element name="derivedFrom" type="typeRefNMTOKEN"
      minOccurs="0"/>
    <xsd:element name="component" maxOccurs="unbounded">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="name" type="xsd:NMTOKEN"/>
          <xsd:element ref="synopsis"/>
          <xsd:element ref="description" minOccurs="0"/>
          <xsd:element name="optional" minOccurs="0"/>
          <xsd:group ref="typeDeclarationGroup"/>
        </xsd:sequence>
        <xsd:attribute name="componentID" type="xsd:unsignedInt"
         use="required"/>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

Рисунок 6. Исходное определение структуры в схеме.

Данное расширение меняет определение структуры в схеме XML, показанное на рисунке 6, расширенным определением, приведённым на рисунке 7.

Пример использования необязательного типа доступа в структуре показан на рисунке 8.

```
<component componentID="1" access="read-write">
   <name>PacketFlows</name>
   <synopsis>Поток пакетов, сопоставление и счётчик</synopsis>
   <struct>
    <component componentID="1">
      <name>FlowMatch</name>
      <synopsis>Сопоставление с потоком</synopsis>
      <typeRef>MatchType</typeRef>
    </component>
    <component componentID="2" access="read-only">
      <name>MatchCounter</name>
      <synopsis>Пакеты, соответствующие сопоставлению с потоком</synopsis>
      <typeRef>ZeroCounter</typeRef>
    </component>
  </struct>
</component>
```

Рисунок 8. Пример необязательного типа доступа в структуре.

```
            </xsd:simpleType>
          </xsd:attribute>
          <!-- /Расширение RFC 7408 -->
          <xsd:attribute name="componentID" type="xsd:unsignedInt"
            use="required"/>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
```

Рисунок 7. Новое определение структуры в схеме.

2.4. Новое условие для событий eventBecomesEqualTo

Это расширение добавляет одно условие для событий в схему модели - eventBecomesEqualTo. Это условие отличается от eventGreaterThan и eventLessThan, поскольку оно включается при точном совпадении с порогом eventBecomesEqualTo. Такое условие полезно, в частности, для случаев, когда нужно отслеживать одно или множество состояний LFB или FE. Например, в CEHA[1] [RFC7121] для ведущего CE может быть полезно знать о новом резервном CE, чтобы соединиться с ним и синхронизировать состояние FE. Ведущий элемент CE может разрешать опросы для нужной информации, но получение таких событий будет ускорять процесс, а кроме того, событие может быть полезно в других случаях для эффективного мониторинга состояний.

Событие следует инициировать лишь при совпадении целевой компоненты с заданным условием порогом. **Недопустимо** продолжение генерации событий, пока значение целевой компоненты сохраняется.

Определение eventBecomesEqualTo добавлено в схему, как показано на рисунке 9.

```
<xsd:element name="eventBecomesEqualTo"
  substitutionGroup="eventCondition"/>
```

Рисунок 9. Определение eventBecomesEqualTo в схеме.

Для CE могут быть полезны уведомления в случаях, когда состояние изменилось после события eventBecomesEqualTo, например, CE может быть нужно знать о потере связи с резервным CE. Такое событие может генерироваться путём определения второго события в той же компоненте (eventChanged) или просто путём повторного использования eventBecomesEqualTo с применением свойств событий (в частности, eventHysteresis). Для этого к определению eventHysteresis из параграфа 4.8.5.2 [RFC5812] здесь добавляется гистерезис V:

- для условия eventBecomesEqualTo после предшествующего уведомления **должно** генерироваться новое уведомление eventBecomesEqualTo только один при изменении значения на +/- V.

Например, при V=1 будет создаваться одно уведомление для CE при изменении значения по меньшей мере на 1.

Разработчикам CE следует предусматривать использование счётчика или фильтра для предотвращения избыточных сообщений, например в случаях быстрых изменений.

2.5. Свойства LFB

Предыдущая модель задавала свойства для компонент LFB. Опыт показал, что было бы полезно, по меньшей мере для отладки, иметь статистику по каждому экземпляру LFB для мониторинга отправленных и принятых сообщений и ошибок в коммуникациях между CE и FE. Эти свойства открыты лишь для чтения.

```
<xsd:attribute name="componentID" type="xsd:unsignedInt"
  use="required">
```

Рисунок 10. Исходное определение componentID.

Для предотвращения неоднозначности в протокольной семантике путей этот документ задаёт, что компонента LFB с componentID 0 **должна** указывать на свойства LFB и **недопустимо** применять идентификатор 0 в любом определении компонент. Этот запрет совместим с прежней версией, поскольку ни одно из известных определений LFB не применяло LFB componentID 0. Любые команды, начинающиеся с LFB componentID 0, указывают на свойства LFB. На рисунках 10 и 11 показано изменение в схеме XML, запрещающее использовать LFB componentID 0.

Приведённые ниже определения типов данных будет служить свойствами экземпляров LFB.

[1]Control Element High Availability - высокая доступность элемента управления.

```
<!-- Расширение с ограничением для componentID -->
<xsd:attribute name="componentID" use="required">
  <xsd:simpleType>
    <xsd:restriction base="xsd:unsignedInt">
      <xsd:minExclusive value="0"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:attribute>
<!-- Конец расширения -->
```

Рисунок 11. Новое определение для запрета использования componentID = 0.

```
<dataTypeDef>
   <name>LFBProperties</name>
   <synopsis>Определение свойств LFB</synopsis>
   <struct>
      <component componentID="1">
         <name>PacketsSentToCE</name>
         <synopsis>Пакеты, переданные CE</synopsis>
         <typeRef>uint32</typeRef>
      </component>
      <component componentID="2">
         <name>SentErrorPacketsToCE</name>
         <synopsis>Пакеты с ошибками, переданные CE</synopsis>
         <typeRef>uint32</typeRef>
      </component>
      <component componentID="3">
         <name>BytesSentToCE</name>
         <synopsis>Число байтов, переданных CE</synopsis>
         <typeRef>uint32</typeRef>
      </component>
      <component componentID="4">
         <name>SentErrorBytesToCE</name>
         <synopsis>Число байтов пакетов с ошибками, переданных CE</synopsis>
         <typeRef>uint32</typeRef>
      </component>
      <component componentID="5">
         <name>PacketsReceivedFromCE</name>
         <synopsis>Пакеты, принятые от CE</synopsis>
         <typeRef>uint32</typeRef>
      </component>
      <component componentID="6">
         <name>ReceivedErrorPacketsFromCE</name>
         <synopsis>Пакеты с ошибками, принятые от CE</synopsis>
         <typeRef>uint32</typeRef>
      </component>
      <component componentID="7">
         <name>BytesReceivedFromCE</name>
         <synopsis>Число байтов, принятых от CE</synopsis>
         <typeRef>uint32</typeRef>
      </component>
      <component componentID="8">
         <name>ReceivedErrorBytesFromCE</name>
         <synopsis>Число байтов в пакетах с ошибками, принятых от  CE</synopsis>
         <typeRef>uint32</typeRef>
      </component>
   </struct>
</dataTypeDef>
```

Рисунок 12. Свойства экземпляров LFB.

2.6. Наследование классов LFB

Модель ForCES [RFC5812] разрешает наследование классов LFB, однако схема XML определяет лишь класс LFB от которого наследуется данный класс LFB. Недавние реализации выявили проблему, когда возникает неоднозначность при наличии разных версий родительского класса LFB. Данный документ дополняет derivedFrom в определении класса LFB путём включения необязательного атрибута версии в случаях использования поля derivedFrom.

Решение о необязательности атрибута версии было обусловлено желанием обеспечить совместимость со схемой XML, определённой в [RFC5812]. Однако при отсутствии версии derivedFrom будет всегда задавать первую версию родительского класса LFB, которая обычно имеет значение 1.0.

```
<xsd:element name="derivedFrom" minOccurs="0"/>
```

Рисунок 13. Исходное определение для наследования класса LFB.

Данное расширение меняет определение derivedFrom в схеме XML (рисунок 13) на расширенное (рисунок 14).

```
<xsd:element name="derivedFrom" minOccurs="0">
  <xsd:complexType>
    <xsd:simpleContent>
      <xsd:extension base="xsd:NMTOKEN">
        <xsd:attribute name="version"
          type="versionType" use="optional"/>
      </xsd:extension>
    </xsd:simpleContent>
  </xsd:complexType>
</xsd:element>
```

Рисунок 14. Новое определение для наследования класса LFB.

Пример использования новой версии атрибута показан на рисунке 15.

```
<derivedFrom version="1.0">EtherPHYCop</derivedFrom>
```

Рисунок 15. Пример использования нового наследования класса LFB.

2.7. Улучшенная проверка пригодности XML

Как указано выше, это не расширение, а улучшение схемы для обеспечения дополнительных правил проверки. Это включает добавление объявлений новых ключей для обеспечения уникальности в соответствии с определением модели ForCES [RFC5812]. Такие проверки работают только в одном и том же файле XML.

Документ добавляет перечисленные ниже правила проверки пригодности, которых не было в исходной схеме [RFC5812].

1. Каждый идентификатор metadataID должен быть уникальным.
2. Идентификаторы LFBClassID должны быть уникальными.
3. Идентификаторы componentID, capabilityID и Event baseID должны быть уникальны в рамках LFB.
4. Идентификаторы eventID должны быть уникальны в рамках LFB.
5. Специальные значения в типах данных atomic должны быть уникальны в рамках типа данных atomic.

3. Схема расширения для документов библиотеки классов LFB

В этом разделе приведена новая схема XML. Номер пространства имён обновлён с 1.0 на 1.1.

Описанные в документе расширения совместимы с прежней версией в терминах операций протокола ForCES. В терминах XML все определения, действительные в старом пространстве имён, сохраняют силу и в новом. Следует отметить, что любые дополнительные средства обработки определений XML, написанные в любом из пространств, должны быть способны понимать оба пространства имён.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns="urn:ietf:params:xml:ns:forces:lfbmodel:1.1"
   xmlns:lfb="urn:ietf:params:xml:ns:forces:lfbmodel:1.1"
   targetNamespace="urn:ietf:params:xml:ns:forces:lfbmodel:1.1"
   elementFormDefault="qualified" attributeFormDefault="unqualified">
   <xsd:annotation>
      <xsd:documentation xml:lang="en">
         Schema for Defining LFB Classes and associated types
         (frames, data types for LFB attributes, and metadata)[1].
      </xsd:documentation>
   </xsd:annotation>
   <xsd:element name="description" type="xsd:string"/>
   <xsd:element name="synopsis" type="xsd:string"/>
   <!-- Document root element: LFBLibrary -->
   <xsd:element name="LFBLibrary">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="description" minOccurs="0"/>
            <xsd:element name="load" type="loadType"
               minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element name="frameDefs" type="frameDefsType"
               minOccurs="0"/>
            <xsd:element name="dataTypeDefs" type="dataTypeDefsType"
               minOccurs="0"/>
            <xsd:element name="metadataDefs" type="metadataDefsType"
               minOccurs="0"/>
            <xsd:element name="LFBClassDefs" type="LFBClassDefsType"
               minOccurs="0"/>
         </xsd:sequence>
         <xsd:attribute name="provides" type="xsd:Name"
            use="required"/>
      </xsd:complexType>
      <!-- Ограничения для уникальности -->
      <xsd:key name="frame">
         <xsd:selector xpath="lfb:frameDefs/lfb:frameDef"/>
         <xsd:field xpath="lfb:name"/>
      </xsd:key>
      <xsd:key name="dataType">
         <xsd:selector xpath="lfb:dataTypeDefs/lfb:dataTypeDef"/>
         <xsd:field xpath="lfb:name"/>
      </xsd:key>
      <xsd:key name="metadataDef">
         <xsd:selector xpath="lfb:metadataDefs/lfb:metadataDef"/>
         <xsd:field xpath="lfb:name"/>
      </xsd:key>
      <xsd:key name="metadataDefID">
         <xsd:selector xpath="lfb:metadataDefs/lfb:metadataDef"/>
         <xsd:field xpath="lfb:metadataID"/>
      </xsd:key>
      <xsd:key name="LFBClassDef">
         <xsd:selector xpath="lfb:LFBClassDefs/lfb:LFBClassDef"/>
         <xsd:field xpath="lfb:name"/>
      </xsd:key>
```

[1]Схема для определения классов LFB и связанных с ними типов (кадры, типы данных для атрибутов LFB, метаданные).

```
      <xsd:key name="LFBClassDefID">
         <xsd:selector xpath="lfb:LFBClassDefs/lfb:LFBClassDef"/>
         <xsd:field xpath="@LFBClassID"/>
      </xsd:key>
   </xsd:element>
   <xsd:complexType name="loadType">
      <xsd:attribute name="library" type="xsd:Name" use="required"/>
      <xsd:attribute name="location" type="xsd:anyURI"
         use="optional"/>
   </xsd:complexType>
   <xsd:complexType name="frameDefsType">
      <xsd:sequence>
         <xsd:element name="frameDef" maxOccurs="unbounded">
            <xsd:complexType>
               <xsd:sequence>
                  <xsd:element name="name" type="xsd:NMTOKEN"/>
                  <xsd:element ref="synopsis"/>
                  <xsd:element ref="description"
                     minOccurs="0"/>
               </xsd:sequence>
            </xsd:complexType>
         </xsd:element>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:complexType name="dataTypeDefsType">
      <xsd:sequence>
         <xsd:element name="dataTypeDef" maxOccurs="unbounded">
            <xsd:complexType>
               <xsd:sequence>
                  <xsd:element name="name" type="xsd:NMTOKEN"/>
                  <xsd:element name="derivedFrom" type="xsd:NMTOKEN"
                     minOccurs="0"/>
                  <xsd:element ref="synopsis"/>
                  <xsd:element ref="description"
                     minOccurs="0"/>
                  <xsd:group ref="typeDeclarationGroup"/>
                  <!-- Расширение RFC 7408 -->
                  <xsd:element name="defaultValue" type="xsd:token"
                     minOccurs="0"/>
                  <!-- /Расширение RFC 7408 -->
               </xsd:sequence>
            </xsd:complexType>
         </xsd:element>
      </xsd:sequence>
   </xsd:complexType>
   <!-- Предопределенными (встроенными) неделимыми типами данных являются:
   char, uchar, int16, uint16, int32, uint32, int64, uint64, string[N],
   string, byte[N], boolean, octetstring[N], float32, float64 -->
   <xsd:group name="typeDeclarationGroup">
      <xsd:choice>
         <xsd:element name="typeRef" type="typeRefNMTOKEN"/>
         <xsd:element name="atomic" type="atomicType"/>
         <xsd:element name="array" type="arrayType">
            <!-- Расширение RFC 7408 -->
            <!-- Объявление уникальности идентификаторов для ключей -->
            <xsd:key name="contentKeyID">
               <xsd:selector xpath="lfb:contentKey"/>
               <xsd:field xpath="@contentKeyID"/>
            </xsd:key>
            <!-- /Расширение RFC 7408 -->
         </xsd:element>
         <xsd:element name="struct" type="structType">
            <!-- Расширение RFC 7408 -->
            <!-- Объявление ключа для создания componentID,
              уникальных в структуре -->
            <xsd:key name="structComponentID">
               <xsd:selector xpath="lfb:component"/>
               <xsd:field xpath="@componentID"/>
            </xsd:key>
            <!-- /Расширение RFC 7408 -->
         </xsd:element>
         <xsd:element name="union" type="structType"/>
         <xsd:element name="alias" type="typeRefNMTOKEN"/>
      </xsd:choice>
   </xsd:group>
   <xsd:simpleType name="typeRefNMTOKEN">
      <xsd:restriction base="xsd:token">
         <xsd:pattern value="\c+"/>
         <xsd:pattern value="string\[\d+\]"/>
         <xsd:pattern value="byte\[\d+\]"/>
         <xsd:pattern value="octetstring\[\d+\]"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:complexType name="atomicType">
      <xsd:sequence>
         <xsd:element name="baseType" type="typeRefNMTOKEN"/>
```

```
        <xsd:element name="rangeRestriction"
          type="rangeRestrictionType" minOccurs="0"/>
        <xsd:element name="specialValues" type="specialValuesType"
           minOccurs="0">
           <!-- Расширение RFC 7408 -->
           <xsd:key name="SpecialValue">
              <xsd:selector xpath="specialValue"/>
              <xsd:field xpath="@value"/>
           </xsd:key>
           <!-- /Расширение RFC 7408 -->
        </xsd:element>
     </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="rangeRestrictionType">
     <xsd:sequence>
        <xsd:element name="allowedRange" maxOccurs="unbounded">
           <xsd:complexType>
              <xsd:attribute name="min" type="xsd:integer"
                 use="required"/>
              <xsd:attribute name="max" type="xsd:integer"
                 use="required"/>
           </xsd:complexType>
        </xsd:element>
     </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="specialValuesType">
     <xsd:sequence>
        <xsd:element name="specialValue" maxOccurs="unbounded">
           <xsd:complexType>
              <xsd:sequence>
                 <xsd:element name="name" type="xsd:NMTOKEN"/>
                 <xsd:element ref="synopsis"/>
              </xsd:sequence>
              <xsd:attribute name="value" type="xsd:token"/>
           </xsd:complexType>
        </xsd:element>
     </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="arrayType">
     <xsd:sequence>
        <xsd:group ref="typeDeclarationGroup"/>
        <xsd:element name="contentKey" minOccurs="0"
           maxOccurs="unbounded">
           <xsd:complexType>
              <xsd:sequence>
                 <xsd:element name="contentKeyField"
                    type="xsd:string" maxOccurs="unbounded"/>
              </xsd:sequence>
              <xsd:attribute name="contentKeyID" type="xsd:integer"
                 use="required"/>
           </xsd:complexType>
        </xsd:element>
     </xsd:sequence>
     <xsd:attribute name="type" use="optional" default="variable-size">
        <xsd:simpleType>
           <xsd:restriction base="xsd:string">
              <xsd:enumeration value="fixed-size"/>
              <xsd:enumeration value="variable-size"/>
           </xsd:restriction>
        </xsd:simpleType>
     </xsd:attribute>
     <xsd:attribute name="length" type="xsd:integer"
        use="optional"/>
     <xsd:attribute name="maxLength" type="xsd:integer"
        use="optional"/>
  </xsd:complexType>
  <xsd:complexType name="structType">
     <xsd:sequence>
        <xsd:element name="derivedFrom" type="typeRefNMTOKEN"
           minOccurs="0"/>
        <xsd:element name="component" maxOccurs="unbounded">
           <xsd:complexType>
              <xsd:sequence>
                 <xsd:element name="name" type="xsd:NMTOKEN"/>
                 <xsd:element ref="synopsis"/>
                 <xsd:element ref="description"
                    minOccurs="0"/>
                 <xsd:element name="optional" minOccurs="0"/>
                 <xsd:group ref="typeDeclarationGroup"/>
              </xsd:sequence>
              <!-- Расширение RFC 7408 -->
              <xsd:attribute name="access" use="optional"
                 default="read-write">
                 <xsd:simpleType>
                    <xsd:list itemType="accessModeType"/>
                 </xsd:simpleType>
```

```
            </xsd:attribute>
            <!-- /Расширение RFC 7408 -->
            <xsd:attribute name="componentID" type="xsd:unsignedInt"
               use="required"/>
         </xsd:complexType>
      </xsd:element>
   </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="metadataDefsType">
   <xsd:sequence>
      <xsd:element name="metadataDef" maxOccurs="unbounded">
         <xsd:complexType>
            <xsd:sequence>
               <xsd:element name="name" type="xsd:NMTOKEN"/>
               <xsd:element ref="synopsis"/>
               <xsd:element name="metadataID" type="xsd:integer"/>
               <xsd:element ref="description"
                  minOccurs="0"/>
               <xsd:choice>
                  <xsd:element name="typeRef" type="typeRefNMTOKEN"/>
                  <xsd:element name="atomic" type="atomicType"/>
                  <!-- Расширение RFC 7408 -->
                  <xsd:element name="array" type="arrayType">
                     <!-- Объявление уникальности идентификаторов ключей -->
                     <xsd:key name="contentKeyID1">
                        <xsd:selector xpath="lfb:contentKey"/>
                        <xsd:field xpath="@contentKeyID"/>
                     </xsd:key>
                     <!-- /Расширение RFC 7408 -->
                  </xsd:element>
                  <xsd:element name="struct" type="structType">
                     <!-- Расширение RFC 7408 -->
                     <!-- Объявление ключей для создания componentID,
                        уникальных в структуре -->
                     <xsd:key name="structComponentID1">
                        <xsd:selector xpath="lfb:component"/>
                        <xsd:field xpath="@componentID"/>
                     </xsd:key>
                     <!-- /Расширение RFC 7408 -->
                  </xsd:element>
               </xsd:choice>
            </xsd:sequence>
         </xsd:complexType>
      </xsd:element>
   </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="LFBClassDefsType">
   <xsd:sequence>
      <xsd:element name="LFBClassDef" maxOccurs="unbounded">
         <xsd:complexType>
            <xsd:sequence>
               <xsd:element name="name" type="xsd:NMTOKEN"/>
               <xsd:element ref="synopsis"/>
               <xsd:element name="version" type="versionType"/>
               <xsd:element name="derivedFrom" minOccurs="0">
                 <xsd:complexType>
                   <xsd:simpleContent>
                     <xsd:extension base="xsd:NMTOKEN">
                       <xsd:attribute name="version"
                         type="versionType" use="optional"/>
                     </xsd:extension>
                   </xsd:simpleContent>
                 </xsd:complexType>
               </xsd:element>
               <xsd:element name="inputPorts"
                type="inputPortsType" minOccurs="0"/>
               <xsd:element name="outputPorts"
                type="outputPortsType" minOccurs="0"/>
               <xsd:element name="components"
                type="LFBComponentsType" minOccurs="0"/>
               <xsd:element name="capabilities"
                type="LFBCapabilitiesType" minOccurs="0"/>
               <xsd:element name="events" type="eventsType"
                  minOccurs="0"/>
               <xsd:element ref="description"
                  minOccurs="0"/>
            </xsd:sequence>
            <xsd:attribute name="LFBClassID" type="xsd:unsignedInt"
               use="required"/>
         </xsd:complexType>
         <!-- Ограничения на ключи для обеспечения уникальности имён
         атрибутов внутри класса -->
         <xsd:key name="components">
            <xsd:selector xpath="lfb:components/lfb:component"/>
            <xsd:field xpath="lfb:name"/>
         </xsd:key>
```

```
            <xsd:key name="capabilities">
               <xsd:selector xpath="lfb:capabilities/lfb:capability"/>
               <xsd:field xpath="lfb:name"/>
            </xsd:key>
            <xsd:key name="events">
               <xsd:selector xpath="lfb:events/lfb:event"/>
               <xsd:field xpath="lfb:name"/>
            </xsd:key>
            <xsd:key name="eventsIDs">
               <xsd:selector xpath="lfb:events/lfb:event"/>
               <xsd:field xpath="@eventID"/>
            </xsd:key>
            <xsd:key name="componentIDs">
               <xsd:selector xpath="lfb:components/lfb:component"/>
               <xsd:field xpath="@componentID"/>
            </xsd:key>
            <xsd:key name="capabilityIDs">
               <xsd:selector xpath="lfb:capabilities/lfb:capability"/>
               <xsd:field xpath="@componentID"/>
            </xsd:key>
            <xsd:key name="ComponentCapabilityComponentIDUniqueness">
               <xsd:selector
                  xpath="lfb:components/lfb:component|
                  lfb:capabilities/lfb:capability|lfb:events"/>
               <xsd:field xpath="@componentID|@baseID"/>
            </xsd:key>
         </xsd:element>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:simpleType name="versionType">
      <xsd:restriction base="xsd:NMTOKEN">
         <xsd:pattern value="[1-9][0-9]*\.([1-9][0-9]*|0)"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:complexType name="inputPortsType">
      <xsd:sequence>
         <xsd:element name="inputPort" type="inputPortType"
            maxOccurs="unbounded"/>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:complexType name="inputPortType">
      <xsd:sequence>
         <xsd:element name="name" type="xsd:NMTOKEN"/>
         <xsd:element ref="synopsis"/>
         <xsd:element name="expectation" type="portExpectationType"/>
         <xsd:element ref="description" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="group" type="xsd:boolean"
         use="optional" default="0"/>
   </xsd:complexType>
   <xsd:complexType name="portExpectationType">
      <xsd:sequence>
         <xsd:element name="frameExpected" minOccurs="0">
            <xsd:complexType>
               <xsd:sequence>
                  <!-- Ссылка должна указывать имя определённого типа кадров -->
                  <xsd:element name="ref" type="xsd:string"
                     maxOccurs="unbounded"/>
               </xsd:sequence>
            </xsd:complexType>
         </xsd:element>
         <xsd:element name="metadataExpected" minOccurs="0">
            <xsd:complexType>
               <xsd:choice maxOccurs="unbounded">
                  <!-- Ссылка должна указывать имя определённых метаданных -->
                  <xsd:element name="ref" type="metadataInputRefType"/>
                  <xsd:element name="one-of"
                     type="metadataInputChoiceType"/>
               </xsd:choice>
            </xsd:complexType>
         </xsd:element>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:complexType name="metadataInputChoiceType">
      <xsd:choice minOccurs="2" maxOccurs="unbounded">
         <!-- Ссылка должна указывать имя определённых метаданных -->
         <xsd:element name="ref" type="xsd:NMTOKEN"/>
         <xsd:element name="one-of" type="metadataInputChoiceType"/>
         <xsd:element name="metadataSet" type="metadataInputSetType"/>
      </xsd:choice>
   </xsd:complexType>
   <xsd:complexType name="metadataInputSetType">
      <xsd:choice minOccurs="2" maxOccurs="unbounded">
         <!-- Ссылка должна указывать имя определённых метаданных -->
         <xsd:element name="ref" type="metadataInputRefType"/>
         <xsd:element name="one-of" type="metadataInputChoiceType"/>
```

```
      </xsd:choice>
   </xsd:complexType>
   <xsd:complexType name="metadataInputRefType">
      <xsd:simpleContent>
         <xsd:extension base="xsd:NMTOKEN">
            <xsd:attribute name="dependency" use="optional"
               default="required">
               <xsd:simpleType>
                  <xsd:restriction base="xsd:string">
                     <xsd:enumeration value="required"/>
                     <xsd:enumeration value="optional"/>
                  </xsd:restriction>
               </xsd:simpleType>
            </xsd:attribute>
            <xsd:attribute name="defaultValue" type="xsd:token"
               use="optional"/>
         </xsd:extension>
      </xsd:simpleContent>
   </xsd:complexType>
   <xsd:complexType name="outputPortsType">
      <xsd:sequence>
         <xsd:element name="outputPort" type="outputPortType"
            maxOccurs="unbounded"/>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:complexType name="outputPortType">
      <xsd:sequence>
         <xsd:element name="name" type="xsd:NMTOKEN"/>
         <xsd:element ref="synopsis"/>
         <xsd:element name="product" type="portProductType"/>
         <xsd:element ref="description" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="group" type="xsd:boolean"
         use="optional" default="0"/>
   </xsd:complexType>
   <xsd:complexType name="portProductType">
      <xsd:sequence>
         <xsd:element name="frameProduced" minOccurs="0">
            <xsd:complexType>
               <xsd:sequence>
                  <!-- Ссылка должна указывать имя определённого типа кадров -->
                  <xsd:element name="ref" type="xsd:NMTOKEN"
                     maxOccurs="unbounded"/>
               </xsd:sequence>
            </xsd:complexType>
         </xsd:element>
         <xsd:element name="metadataProduced" minOccurs="0">
            <xsd:complexType>
               <xsd:choice maxOccurs="unbounded">
                  <!-- Ссылка должна указывать имя определённых метаданных -->
                  <xsd:element name="ref"
                     type="metadataOutputRefType"/>
                  <xsd:element name="one-of"
                     type="metadataOutputChoiceType"/>
               </xsd:choice>
            </xsd:complexType>
         </xsd:element>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:complexType name="metadataOutputChoiceType">
      <xsd:choice minOccurs="2" maxOccurs="unbounded">
         <!-- Ссылка должна указывать имя определённых метаданных -->
         <xsd:element name="ref" type="xsd:NMTOKEN"/>
         <xsd:element name="one-of" type="metadataOutputChoiceType"/>
         <xsd:element name="metadataSet" type="metadataOutputSetType"/>
      </xsd:choice>
   </xsd:complexType>
   <xsd:complexType name="metadataOutputSetType">
      <xsd:choice minOccurs="2" maxOccurs="unbounded">
         <!-- Ссылка должна указывать имя определённых метаданных -->
         <xsd:element name="ref" type="metadataOutputRefType"/>
         <xsd:element name="one-of" type="metadataOutputChoiceType"/>
      </xsd:choice>
   </xsd:complexType>
   <xsd:complexType name="metadataOutputRefType">
      <xsd:simpleContent>
         <xsd:extension base="xsd:NMTOKEN">
            <xsd:attribute name="availability" use="optional"
               default="unconditional">
               <xsd:simpleType>
                  <xsd:restriction base="xsd:string">
                     <xsd:enumeration value="unconditional"/>
                     <xsd:enumeration value="conditional"/>
                  </xsd:restriction>
               </xsd:simpleType>
            </xsd:attribute>
```

```
        </xsd:extension>
      </xsd:simpleContent>
   </xsd:complexType>
   <xsd:complexType name="LFBComponentsType">
      <xsd:sequence>
         <xsd:element name="component" maxOccurs="unbounded">
            <xsd:complexType>
               <xsd:sequence>
                  <xsd:element name="name" type="xsd:NMTOKEN"/>
                  <xsd:element ref="synopsis"/>
                  <xsd:element ref="description" minOccurs="0"/>
                  <xsd:element name="optional" minOccurs="0"/>
                  <xsd:group ref="typeDeclarationGroup"/>
                  <xsd:element name="defaultValue" type="xsd:token"
                     minOccurs="0"/>
               </xsd:sequence>
               <xsd:attribute name="access" use="optional"
                  default="read-write">
                  <xsd:simpleType>
                     <xsd:list itemType="accessModeType"/>
                  </xsd:simpleType>
               </xsd:attribute>
               <!-- Расширение, добавляющее ограничения для componentID -->
               <xsd:attribute name="componentID" use="required">
                  <xsd:simpleType>
                     <xsd:restriction base="xsd:unsignedInt">
                        <xsd:minExclusive value="0"/>
                     </xsd:restriction>
                  </xsd:simpleType>
               </xsd:attribute>
               <!-- Конец расширения -->
            </xsd:complexType>
         </xsd:element>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:simpleType name="accessModeType">
      <xsd:restriction base="xsd:NMTOKEN">
         <xsd:enumeration value="read-only"/>
         <xsd:enumeration value="read-write"/>
         <xsd:enumeration value="write-only"/>
         <xsd:enumeration value="read-reset"/>
         <xsd:enumeration value="trigger-only"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:complexType name="LFBCapabilitiesType">
      <xsd:sequence>
         <xsd:element name="capability" maxOccurs="unbounded">
            <xsd:complexType>
               <xsd:sequence>
                  <xsd:element name="name" type="xsd:NMTOKEN"/>
                  <xsd:element ref="synopsis"/>
                  <xsd:element ref="description" minOccurs="0"/>
                  <xsd:element name="optional" minOccurs="0"/>
                  <xsd:group ref="typeDeclarationGroup"/>
               </xsd:sequence>
               <xsd:attribute name="componentID" type="xsd:integer"
                  use="required"/>
            </xsd:complexType>
         </xsd:element>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:complexType name="eventsType">
      <xsd:sequence>
         <xsd:element name="event" maxOccurs="unbounded">
            <xsd:complexType>
               <xsd:sequence>
                  <xsd:element name="name" type="xsd:NMTOKEN"/>
                  <xsd:element ref="synopsis"/>
                  <xsd:element name="eventTarget"
                    type="eventPathType"/>
                  <xsd:element ref="eventCondition"/>
                  <xsd:element name="eventReports"
                   type="eventReportsType" minOccurs="0"/>
                  <xsd:element ref="description" minOccurs="0"/>
               </xsd:sequence>
               <xsd:attribute name="eventID" type="xsd:integer"
                  use="required"/>
            </xsd:complexType>
         </xsd:element>
      </xsd:sequence>
      <xsd:attribute name="baseID" type="xsd:integer" use="optional"/>
   </xsd:complexType>
   <!-- Группа подстановки для условий события -->
   <xsd:element name="eventCondition" abstract="true"/>
   <xsd:element name="eventCreated"
    substitutionGroup="eventCondition"/>
```

```
   <xsd:element name="eventDeleted"
    substitutionGroup="eventCondition"/>
   <xsd:element name="eventChanged"
    substitutionGroup="eventCondition"/>
   <xsd:element name="eventGreaterThan"
    substitutionGroup="eventCondition"/>
   <xsd:element name="eventLessThan"
    substitutionGroup="eventCondition"/>
   <!-- Расширение RFC 7408 -->
   <xsd:element name="eventBecomesEqualTo"
      substitutionGroup="eventCondition"/>
   <!-- /Расширение RFC 7408 -->
   <xsd:complexType name="eventPathType">
      <xsd:sequence>
         <xsd:element ref="eventPathPart" maxOccurs="unbounded"/>
      </xsd:sequence>
   </xsd:complexType>
   <!-- Группа подстановки для частей пути к событию -->
   <xsd:element name="eventPathPart" type="xsd:string"
      abstract="true"/>
   <xsd:element name="eventField" type="xsd:string"
      substitutionGroup="eventPathPart"/>
   <xsd:element name="eventSubscript" type="xsd:string"
      substitutionGroup="eventPathPart"/>
   <xsd:complexType name="eventReportsType">
      <xsd:sequence>
         <xsd:element name="eventReport" type="eventPathType"
            maxOccurs="unbounded"/>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:simpleType name="booleanType">
      <xsd:restriction base="xsd:string">
         <xsd:enumeration value="0"/>
         <xsd:enumeration value="1"/>
      </xsd:restriction>
   </xsd:simpleType>
</xsd:schema>
```

4. Взаимодействие с IANA

Агентство IANA зарегистрировало новое пространство имён XML в соответствии с RFC 3688 [RFC3688].

```
   URI: The URI for this namespace is:
      urn:ietf:params:xml:ns:forces:lfbmodel:1.1
   Registrant Contact: IESG
   XML: none, this is an XML namespace
```

5. Вопросы безопасности

Эта спецификация лишь добавляет несколько конструкций к исходной модели [RFC5812], а именно - новое событие, несколько новых свойств и способ определения дополнительных типов доступа и комплексных метаданных. Документ также разъясняет и решает проблему наследования путём добавления версии класса LFB в derivedFrom. Эти конструкции и обновление наследования не меняют природы исходной модели.

Таким образом, вопросы безопасности из [RFC5812] применимы и к этой спецификации, а внесённые здесь изменения являются просто конструкциями для записи определений библиотеки XML как в [RFC5812]. Эти изменения, а также разъяснение модели наследования не влияют на семантику безопасности протокола.

Что касается повсеместного мониторинга (PM), описанного в [RFC7258], эта спецификация определяет способы показать более полную информацию (метаданные), передаваемую между LFB, а также между CE и FE, и новое событие. Эти изменения не влияют или слабо влияют на упрощение PM и возможности атакующего, который контролирует LFB. Новые метаданные могут сделать PM более элегантным, но не дают новых способов (вредоносного) использования LFB для PM. То же самое относится к новому событию.

Документ не задаёт спецификации протокола и поэтому не указывает способов передачи данных между объектами. Таким образом, снижение возможностей PM для пассивного атакующего, который просто хочет перехватывать данные, выходит за рамки документа.

6. Литература

6.1. Нормативные документы

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

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004, <http://www.rfc-editor.org/info/rfc3688>.

[RFC5810] Doria, A., Hadi Salim, J., Haas, R., Khosravi, H., Wang, W., Dong, L., Gopal, R., and J. Halpern, "Forwarding and Control Element Separation (ForCES) Protocol Specification", RFC 5810, March 2010, <http://www.rfc-editor.org/info/rfc5810>.

[RFC5812] Halpern, J. and J. Hadi Salim, "Forwarding and Control Element Separation (ForCES) Forwarding Element Model", RFC 5812, March 2010, <http://www.rfc-editor.org/info/rfc5812>.

[RFC7121] Ogawa, K., Wang, W., Haleplidis, E., and J. Hadi Salim, "High Availability within a Forwarding and Control Element Separation (ForCES) Network Element", RFC 7121, February 2014, <http://www.rfc-editor.org/info/rfc7121>.

[RFC7258] Farrell, S. and H. Tschofenig, "Pervasive Monitoring Is an Attack", BCP 188, RFC 7258, May 2014, <http://www.rfc-editor.org/info/rfc7258>.

6.2. Дополнительная литература

[OpenFlowSpec1.1] ONF, "OpenFlow Switch Specification", February 2011, <https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.1.0.pdf>.

Благодарности

Спасибо Joel Halpern, Jamal Hadi Salim и Dave Hood за комментарии и дискуссии, которые помогли улучшить документ. Отдельная благодарность Joel Halpern за решение вопроса с принятыми по умолчанию значениями, Adrian Farrel за рецензию AD, Ben Campbell за рецензию Gen-ART и Tom Yu за обзор безопасности - это сделало документ лучше. Улучшить финальный вариант документа помогли рецензии и замечания членов IESG - Stephen Farrel, Barry Leiba, и Ted Lemon. Автор благодарен Julian Reschke за предложение по смене пространства имён и Joel Halpern за помощь в этом.

Адрес автора

Evangelos Haleplidis

University of Patras

Department of Electrical and Computer Engineering

Patras 26500

Greece

EMail: ehalep@ece.upatras.gr

Перевод на русский язык

Николай Малых

nmalykh@protokols.ru