

## NDN-DPDK 项目解析（四）：自定义去 IP 化分组修改

下面是讲解如何在兴趣包中添加自定义字段

### 一、添加编码类型

假设我们要在兴趣包中添加一个 CustomField 字段，该字段进行 TLV 编码的编码类型为 TtCustomField，首先我们要对该类型编号。

编号范围参考：<https://docs.named-data.net/NDN-packet-spec/0.3/types.html>

这里我们选择 0x8000 作为 TtCustomField 的编号

- 打开 ndn/an/tlv-type.go
- 在底部添加：TtCustomField = 0x8000

```
57 | TtNotAfter      = 0x00FF
58 |
59 | TtSafeBag       = 0x80
60 | TtSafeBagEncryptedKey = 0x81
61 | TtCustomField = 0x8000
62 |
63 | _ = "enumgen"
```

保存更改的 tlv-type.go

### 二、修改兴趣包

打开/ndn-dpdk/ndn/interest.go 文件，修改里面的代码

**修改 Interest 结构体：**在 Interest 结构体中添加新的自定义字段。

CustomField []byte

```
18 // Interest represents an Interest packet.
19 type Interest struct {
20     packet      *Packet
21     Name        Name
22     CanBePrefix bool
23     MustBeFresh bool
24     ForwardingHint ForwardingHint
25     Nonce        Nonce
26     Lifetime     time.Duration
27     HopLimit     HopLimit
28     AppParameters []byte
29     SigInfo       *SigInfo
30     SigValue      []byte
31     CustomField   []byte
32 }
```

**更新 Field() 方法：**在 Interest 的 Field() 方法中，将自定义字段编码为 TLV 格式，并将其添加到字段列表中。

```
...
if len(interest.CustomField) > 0 {
    fields = append(fields, tlv.TLVBytes(an.TtCustomField, interest.CustomField))
}
...
```

```

184 // Field implements tlv.Fielder interface.
185 func (interest Interest) Field() tlv.Field {
186     fields := []tlv.Fielder{interest.Name}
187     if interest.CanBePrefix {
188         fields = append(fields, tlv.TLV(an.TtCanBePrefix))
189     }
190     if interest.MustBeFresh {
191         fields = append(fields, tlv.TLV(an.TtMustBeFresh))
192     }
193     if len(interest.ForwardingHint) > 0 {
194         fields = append(fields, interest.ForwardingHint)
195     }
196     nonce := interest.Nonce
197     if nonce.IsZero() {
198         nonce = NewNonce()
199     }
200     fields = append(fields, nonce)
201
202     if lifetime := interest.Lifetime; lifetime != 0 && lifetime != DefaultInterestLifetime {
203         if lifetime < MinInterestLifetime {
204             return tlv.FieldError(ErrLifetime)
205         }
206         fields = append(fields, tlv.TLVNNI(an.TtInterestLifetime, lifetime/time.Millisecond))
207     }
208     if len(interest.CustomField) > 0 {
209         fields = append(fields, tlv.TLVBytes(an.TtCustomField, interest.CustomField))
210     }
211     if interest.HopLimit != 0 {
212         fields = append(fields, interest.HopLimit)
213     }
214     fields = append(fields, interest.encodeParamsPortion()...)
215     return tlv.TLVFrom(an.TtInterest, fields...)
216 }

```

**更新 UnmarshalBinary() 方法：**在解码函数中，添加对自定义字段的处理。

ooo

case an.TtCustomField:

interest.CustomField = de.Value

ooo

```

219 func (interest *Interest) UnmarshalBinary(wire []byte) (e error) {
220     for _, de := range d.Elements() {
221         case an.TtInterestLifetime:
222             // interest.Lifetime = time.Duration(de.UnmarshalNNI(0))
223             interest.Lifetime *= time.Millisecond
224         case an.TtHopLimit:
225             if e := de.UnmarshalValue(&interest.HopLimit); e != nil {
226                 return e
227             }
228             // 添加对自定义字段的解析
229         case an.TtCustomField:
230             interest.CustomField = de.Value
231         case an.TtAppParameters:
232             interest.AppParameters = de.Value
233             paramsPortion = de.WireAfter()
234         case an.TtSigInfo:
235             var si SigInfo
236             if e := de.UnmarshalValue(&si); e != nil {
237                 return e
238             }
239             interest.SigInfo = &si
240         case an.TtSigValue:
241             interest.SigValue = de.Value
242         default:
243             if de.IsCriticalType() {
244                 return tlv.ErrCritical
245             }
246     }
247 }

```

保存更改的 interest.go，到这一步在兴趣包中添加字段已经完成了，后面介绍如何在发送兴

趣包的时候在这个字段写入内容，以及收包的时候如何解析出这个字段里面的内容

### 三、在发包时填充自定义字段：

打开 cmd/ndndpdk-godemo/ping.go 修改里面的代码

在调用 MakeIntere 方法制作兴趣包 interest 后，直接对 interest.CustomField 赋值为想要传输的内容，这里我们传输一个字符串 “Custom410”

```
interest.CustomField = []byte("Custom410")
```

使用 fmt.Printf 可以打印出内容。

```
75 func init() {  
79     defineCommand(&cli.Command{  
108         Action: func(c *cli.Context) error {  
120             var nData, nErrors atomic.Int64  
121             for {  
122                 select {  
123                     case <-c.Context.Done():  
124                         return nil  
125                     case timestamp := <-ticker.C:  
126                         go func(t0 time.Time, s uint64) {  
127                             interest := ndn.MakeInterest(fmt.Sprintf("%s/%016X", name, s), ndn.MustBeFreshFlag, lifetime)  
128                             interest.CustomField = []byte("Custom410")  
129                             fmt.Printf("CustomField: %v\n", interest.CustomField)  
130                             e := endpoint.Consume(ctx, interest, endpoint.ConsumerOptions{  
131                                 Verifier: verifier,  
132                             })  
133                             rtt := time.Since(t0)  
134                             if e == nil {  
135                                 nDataL, nErrorsL := nData.Add(1), nErrors.Load()  
136                                 log.Printf("%.2f%% D %016X %6dus", 100*float64(nDataL)/float64(nData+nErrorsL), s, rtt.M:  
137                             } else {  
138                                 nDataL, nErrorsL := nData.Load(), nErrors.Add(1)  
139                                 log.Printf("%.2f%% E %016X %v", 100*float64(nDataL)/float64(nData+nErrorsL), s, e)  
140                             }  
141                         }(timestamp, seqNum)  
142                         seqNum++  
143                     }  
144             }  
145         }  
146     }  
147 }
```

### 四、收包解析

兴趣包是由 consumer 发送给 producer 的，所以下面我们修改 producer 让它能打印输出 interest 里面的 CustomField 字段。

打开 ndn/endpoint/producer.go

添加下面的代码：

```
fmt.Printf("Interest.CustomField: %v\n", interest.CustomField)
```

```
130 func (p *producer) handleInterest(ctx context.Context, wg *sync.WaitGroup, pkt *ndn.Packet) {  
131     defer wg.Done()  
132  
133     interest := pkt.Interest  
134     fmt.Printf("Interest.CustomField: %v\n", interest.CustomField)  
135     if !p.Prefix.IsPrefixOf(interest.Name) {  
136         return  
137     }  
138 }
```

注意：使用 fmt.Printf 打印需要先引入 "fmt" 包

```
3 import (  
4     "context"  
5     "errors"  
6     "io"  
7     "sync"  
8     "fmt"  
9 )
```

### 五、编译运行

创建两个虚拟机：虚拟机 A、虚拟机 B

虚拟机 A 作为 client 发送兴趣包

虚拟机 B 作为 server 接收兴趣包并解析

按前面的步骤修改完代码后开始编译：

```
cd /home/lwj/Desktop/ndn-dpdk
```

```
NDNDPDK_MK_RELEASE=1 make
```

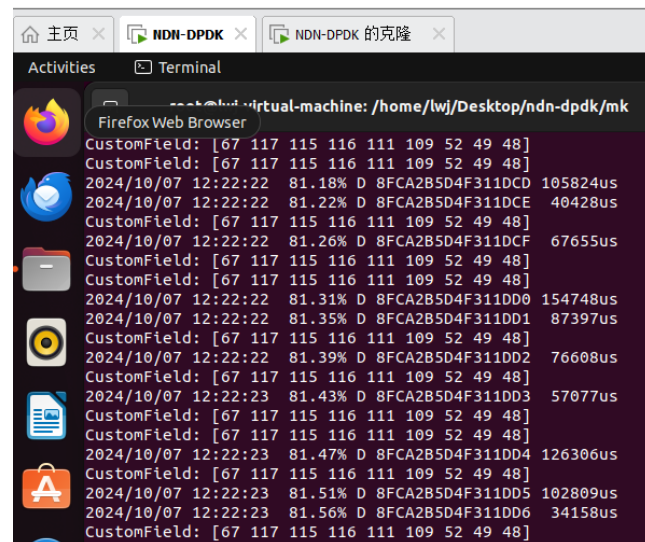
```
cd mk/
```

```
sudo ./install.sh
```

编译完成后的运行步骤可以参考 NDN-DPDK 项目解析三中的转发器实验

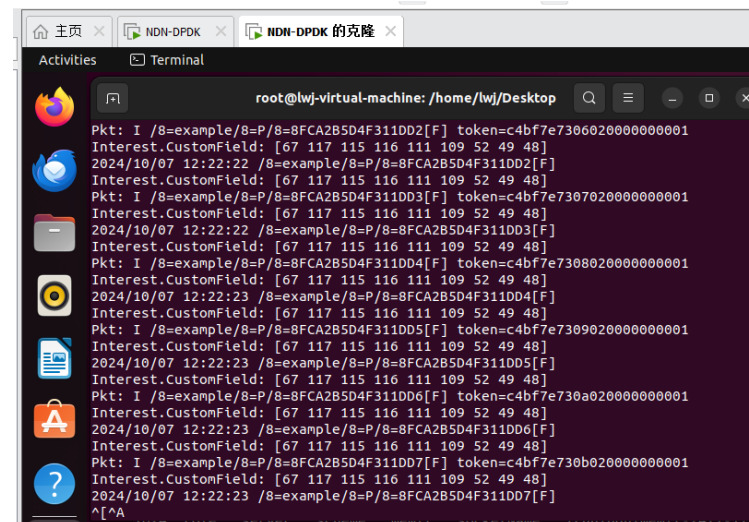
下面是运行结果：

主机 A client 发送兴趣包



```
CustomField: [67 117 115 116 111 109 52 49 48]
CustomField: [67 117 115 116 111 109 52 49 48]
2024/10/07 12:22:22 81.18% D 8FCA2B5D4F311DDC 105824us
2024/10/07 12:22:22 81.22% D 8FCA2B5D4F311DCE 40428us
CustomField: [67 117 115 116 111 109 52 49 48]
2024/10/07 12:22:22 81.26% D 8FCA2B5D4F311DCF 67655us
CustomField: [67 117 115 116 111 109 52 49 48]
CustomField: [67 117 115 116 111 109 52 49 48]
2024/10/07 12:22:22 81.31% D 8FCA2B5D4F311DD0 154748us
2024/10/07 12:22:22 81.35% D 8FCA2B5D4F311DD1 87397us
CustomField: [67 117 115 116 111 109 52 49 48]
2024/10/07 12:22:22 81.39% D 8FCA2B5D4F311DD2 76608us
CustomField: [67 117 115 116 111 109 52 49 48]
2024/10/07 12:22:23 81.43% D 8FCA2B5D4F311DD3 57077us
CustomField: [67 117 115 116 111 109 52 49 48]
CustomField: [67 117 115 116 111 109 52 49 48]
2024/10/07 12:22:23 81.47% D 8FCA2B5D4F311DD4 126306us
CustomField: [67 117 115 116 111 109 52 49 48]
2024/10/07 12:22:23 81.51% D 8FCA2B5D4F311DD5 102809us
2024/10/07 12:22:23 81.56% D 8FCA2B5D4F311DD6 34158us
CustomField: [67 117 115 116 111 109 52 49 48]
```

主机 B server 接收兴趣包并打印 CuotomField 字段



```
Pkt: I /8=example/8=P/8=8FCA2B5D4F311DD2[F] token=c4bf7e7306020000000001
Interest.CustomField: [67 117 115 116 111 109 52 49 48]
2024/10/07 12:22:22 /8=example/8=P/8=8FCA2B5D4F311DD2[F]
Interest.CustomField: [67 117 115 116 111 109 52 49 48]
Pkt: I /8=example/8=P/8=8FCA2B5D4F311DD3[F] token=c4bf7e7307020000000001
Interest.CustomField: [67 117 115 116 111 109 52 49 48]
2024/10/07 12:22:22 /8=example/8=P/8=8FCA2B5D4F311DD3[F]
Interest.CustomField: [67 117 115 116 111 109 52 49 48]
Pkt: I /8=example/8=P/8=8FCA2B5D4F311DD4[F] token=c4bf7e7308020000000001
Interest.CustomField: [67 117 115 116 111 109 52 49 48]
2024/10/07 12:22:23 /8=example/8=P/8=8FCA2B5D4F311DD4[F]
Interest.CustomField: [67 117 115 116 111 109 52 49 48]
Pkt: I /8=example/8=P/8=8FCA2B5D4F311DD5[F] token=c4bf7e7309020000000001
Interest.CustomField: [67 117 115 116 111 109 52 49 48]
2024/10/07 12:22:23 /8=example/8=P/8=8FCA2B5D4F311DD5[F]
Interest.CustomField: [67 117 115 116 111 109 52 49 48]
Pkt: I /8=example/8=P/8=8FCA2B5D4F311DD6[F] token=c4bf7e730a020000000001
Interest.CustomField: [67 117 115 116 111 109 52 49 48]
2024/10/07 12:22:23 /8=example/8=P/8=8FCA2B5D4F311DD6[F]
Interest.CustomField: [67 117 115 116 111 109 52 49 48]
Pkt: I /8=example/8=P/8=8FCA2B5D4F311DD7[F] token=c4bf7e730b020000000001
Interest.CustomField: [67 117 115 116 111 109 52 49 48]
2024/10/07 12:22:23 /8=example/8=P/8=8FCA2B5D4F311DD7[F]
```

从接收端打印结果中可以看到 “Interest.CustomField: [67 117 115 116 111 109 52 49 48]”。

“67 117 115 116 111 109 52 49 48” 刚好是 “Custom410” 的 ASCII 编码，说明可以成功解析出 CustomField 字段。