NL80211

I. Objectivos

Os objectivos deste trabalho prático são:

- Compreender a estrutura de um programa que use o Netlink para Wifi em C
- Identificar as bibliotecas necessárias para compilar estes programas
- Modificar um programa com vista ao envio de um comando NL80211 e recolha da resposta esperada

II. Duração

Este Trabalho deve durar uma aula (3h).

III. Procedimentos

Este Trabalho irá utilizar:

a) 1x PC do laboratório por grupo de trabalho

1. Preparação

1) No espaço da disciplina no e-learning, faça download dos ficheiros que estão na pasta "Material Guião 2 WLAN";
2) Para poder realizar capturas dos pacotes netfilter trocados entre as aplicações em *userspace* e o *kernel*, é necessário criar uma interface de rede virtual "Netlink Monitor", onde se colocará o Wireshark à escuta. Para criar essa interface, execute os seguintes comandos (i.e., poderá precisar de fazer *sudo*)

```
# ip link add nlmon0 type nlmon
# ip link set nlmon0 up
```

*(nota: se tiver problemas com estes comandos, analise o anexo **V** mais abaixo neste guião)*

3) Certifique-se que tem as bibliotecas de desenvolvimento do Netlink instaladas no seu sistema. Para tal, consulte o anexo **VI** mais abaixo
4) Execute o seguinte comando

```
# iwconfig
```

Após correr esse comando, registe o nome da interface wireless (i.e., "wlp5s0", "wlan0", etc.)

2. Análise e modificação de um programa NL80211 para listagem de atributos de interfaces de rede wireless

3. Abra o ficheiro "show_wifi_interfaces.c" e analise a sua estrutura tentando responder às seguintes questões:
    a. Identifique em que linha está identificado o nome da interface wireless onde abrir a socket netlink; altere para o nome da interface wireless obtido no ponto 4.
    b. Identifique onde estão as funções de callback do código;
    c. Identifique qual o comando Netlink enviado (i.e., estará no formato "NL80211_CMD_...";
    d. Identifique qual o atributo adicionado ao comando antes dele ser enviado
    e. Na função de callback, identifique quais os atributos que são sondados
4. Compile o programa usando o seguinte comando:

```
gcc show_wifi_interfaces.c -o show_wifi_interfaces -
I/usr/include/libnl3/ -lnl-genl-3 -lnl-3
```

5. Execute o programa utilizando o seguinte comando:

```
./show_wifi_interfaces
```

6. Analise o output do programa, que poderá variar mas apresentar informação similar a esta:

```
>>> Getting info for wlan0:
NL80211_ATTR_IFNAME: Interface wlp5s0
NL80211_ATTR_WIPHY: wiphy 0
NL80211_ATTR_MAC: MAC address: b8:27:eb:bf:ce:b9
----------------------------
>>> Program exit.
```

7. Analise o resultado do ecrã, relacionando-o com o código onde estas mensagens são impressas e conclua sobre o método utilizado para obter informação fornecida pelo comando que identificou no ponto 3.c;
8. Ligue o wireshark, e coloque-o em escuta na interface "nlmon0" que criou no ponto 1.2 deste guião
9. Analise as mensagens que estão presentes na captura (que continua a decorrer), e conclua que não são relacionadas com o programa que executou.
10. Volte a correr o programa.
11. Verifique as mensagens "nl80211" que apareceram, concluindo sobre
    f. Qual a mensagem associada ao comando enviado e o atributo que lá foi colocado
    g. Qual a mensagem de resposta, e quais os atributos recebidos.

12. Modifique o programa (voltando depois a compila-lo), de forma a conseguir apresentar os seguintes atributos:
    h. NL80211_ATTR_IFINDEX
    i. NL80211_ATTR_IFTYPE
    j. NL80211_ATTR_WIPHY_FREQ
    k. NL80211_ATTR_CHANNEL_WIDTH
    l. NL80211_ATTR_CENTER_FREQ1
    m. NL80211_ATTR_WIPHY_CHANNEL_TYPE
    n. NL80211_ATTR_WIPHY_TX_POWER_LEVEL

Nota, se tiver mais do que uma interface de rede wireless, pode obter uma listagem geral. Para isso:

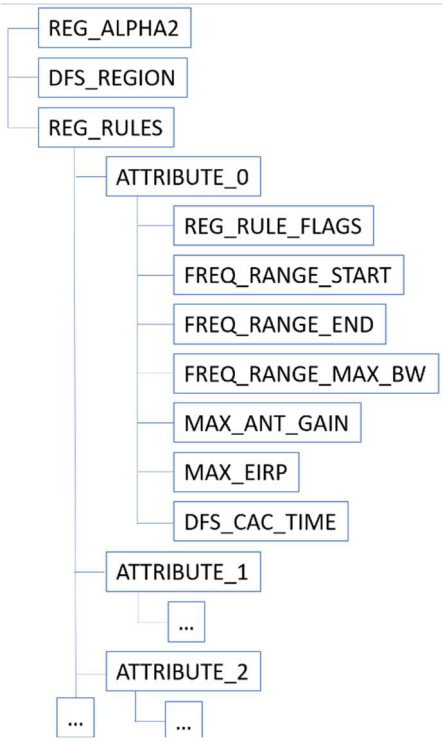
o. Altere o uso da função genlmsg_put() para:

```
genlmsg_put(msg, 0, 0, driver_id, 0, NLM_F_DUMP,
                                    NL80211_CMD_GET_INTERFACE, 0);
```

p. Comente toda a linha onde é feito a adição do atributo NL80211_ATTR_IFINDEX;
q. Comente a linha "nl_wait_for_ack(socket);"

3. Construção de um programa para interpretação de atributos nl80211

4. O ficheiro “simple2.c” envia um comando do tipo “NL80211_CMD_GET_REG” e, para já, apenas imprime o tamanho da mensagem de resposta. Este comando pergunta à placa Wi-Fi qual o domínio regulatório atualmente configurado (i.e., quais os canais suportados), resultando (em “dump”) no envio da informação sobre o domínio regulatório global e todos os domínios regulatórios privados configurados. A resposta tem a estrutura exemplificada na Figura seguinte. Concretamente, os campos representam a seguinte informação:



- **alpha2:** country code de 2 carateres (i.e., “PT”)
- **DFS_Region:** Região para seleção dinâmica de canais
- **REG_RULES:** array “nested” de atributos, cada um deles também um atributo “nested”, contendo uma lista de atributos associados às regras/configurações do domínio regulatório.

Note que:

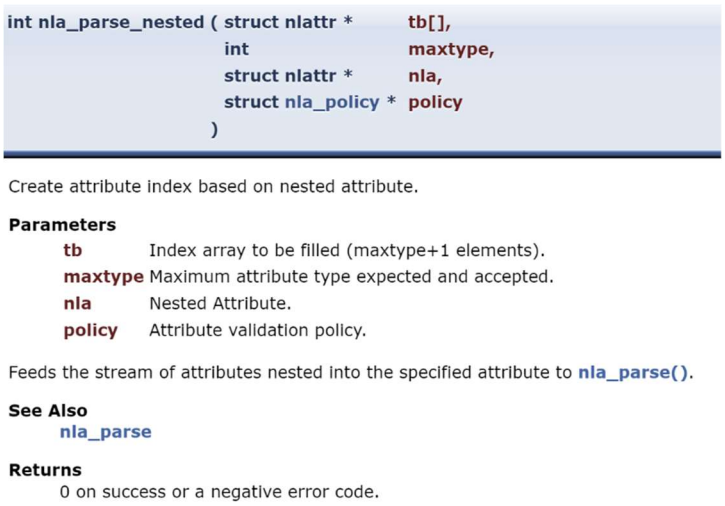
- **EIRP – Effective Isotropic Radiated Power**, que representa a potência efetiva radiada (tendo em conta a potência de transmissão necessária, associada ao ganho obtido pelo uso da antena contra as percas associadas a cabos e conetores da antena)
- **CAC – Channel Availability Check**, tempo durante o qual vai analisar o canal para determinar a sua disponibilidade.

Figura 1 - Resposta exemplificativa para o comando

1. Procure o ficheiro “nl80211.h” no seu sistema (deverá estar em “/usr/include/linux/nl80211.h”) e determine o significado dos restantes atributos
2. Compile o ficheiro, inicie uma captura na interface nlmon0, e execute o programa. Depois de ter corrido o programa, interrompa a captura.
3. Analise a captura, procurando pelos dois pacotes do protocolo “nl80211”. Analise o conteúdo, e compare a estrutura da resposta obtida à da figura anterior.
4. No wireshark, analise o parâmetro “Alpha2: PT” do atributo “NL80211_ATTR_REG_ALPHA2”, na disposição hexadecimal apresentada na janela do canto inferior direito (do wireshark). Conclua quanto ao código ASCII das letras “P” e “T”, e como uma “string” é um array de “chars” mais o caracter especial de terminação de “string” (representado em código como “/0” e em ASCII como 00, ou seja NULL).
5. Utilizando a estratégia de obtenção dos valores dos atributos vista no código do programa anterior, altere o programa do ficheiro “simple2.c”, para obter a string associada ao domínio regulatório “PT”, usando para tal a função “nla_get_string” (em vez de nal_get_u32), armazenando

o resultado num array de chars (char *). *Nota: a impressão de strings em C usando o printf tem o modificador %s*.

Adicionei as linhas de código na função de callback, para o efeito.



```
int nla_parse_nested ( struct nlattr *      tb[],
                       int                  maxtype,
                       struct nlattr *      nla,
                       struct nla_policy *  policy
                     )
```

Create attribute index based on nested attribute.

Parameters

tb	Index array to be filled (maxtype+1 elements).
maxtype	Maximum attribute type expected and accepted.
nla	Nested Attribute.
policy	Attribute validation policy.

Feeds the stream of attributes nested into the specified attribute to **nla_parse()**.

See Also

nla_parse

Returns

0 on success or a negative error code.

6. O acesso aos atributos dos diferentes domínios regulatórios privados, requer a análise de atributos "nested". Para tal, é necessário utilizar a função nla_parse_nested(). Nesta função, precisamos de fornecer um array de elementos com a estrutura "struct nlattr", com um número de elementos capaz de suportar o máximo de regras regulatórias possíveis "NL80211_MAX_SUPP_REG_RULES".

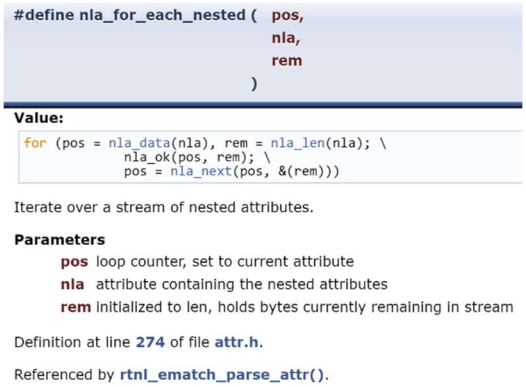
Figura 2 - Função nla_parse_nested

Neste exercício em particular, temos que ter em conta que dentro do nosso atributo "nested" (de nome "NL80211_ATTR_REG_RULES") existem outros atributos "nested". Assim, para cada atributo recebido no array destino indicado no primeiro parâmetro da função nla_parse_nested(), voltamos a ter que executar uma nova função nla_parse_nested() para chegarmos aos atributos individuais do domínio de regulação.

```
if (tb_msg[NL80211_ATTR_REG_RULES]) {


        struct nlattr * nested[NL80211_MAX_SUPP_REG_RULES+1];
        int err = nla_parse_nested(nested,NL80211_MAX_SUPP_REG_RULES,
                        tb_msg[NL80211_ATTR_REG_RULES],NULL);


        struct nlattr * inner[NL80211_MAX_SUPP_REG_RULES+1];
        int err2 = nla_parse_nested(inner,NL80211_MAX_SUPP_REG_RULES,
                        nested[0],NULL);


      if (inner[NL80211_ATTR_FREQ_RANGE_START]){
                printf("Range Start:");
                int type = nla_get_u32(inner[NL80211_ATTR_FREQ_RANGE_START]);
                printf(" %d\n",type);
        }
}
```

O código aqui presente é um exemplo que pode usar e integrar diretamente no programa "simple2.c" para conseguir aceder aos parâmetros individuais, do primeiro atributo "nested" obtido de dentro do atributo nested "NL80211_ATTR_REG_RULES"). **Integre este código no seu programa, compile e teste. Altere o programa para imprimir os restantes atributos individuais. Compare com a totalidade da informação que verifica existir na resposta obtida no wireshark. Comclua sobre a ineficiência deste método.**

7. A melhor estratégia para extração de atributos "nested" (sobretudo no caso em que os atributos contêm outros atributos nested) é utilizar a função de iteração nla_for_each_nested()", com o protótipo seguinte:



```
#define nla_for_each_nested ( pos,
                              nla,
                              rem
                            )
```

Value:

```
for (pos = nla_data(nla), rem = nla_len(nla); \
     nla_ok(pos, rem); \
     pos = nla_next(pos, &(rem)))
```

Iterate over a stream of nested attributes.

Parameters

pos loop counter, set to current attribute
nla attribute containing the nested attributes
rem initialized to len, holds bytes currently remaining in stream

Definition at line **274** of file **attr.h**.

Referenced by **rtnl_ematch_parse_attr()**.

Para esta função, é necessário fornecer um array de estruturas do tipo "struct nlattr" para armazenar os atributos existentes no atributo "nested".

Figura 3 - Função nla_for_each_nested()

1. Altere o código feito na alínea anterior, pelo seguinte código que faz uso da função nla_for_each_nested(), e compare os resultados, concluindo sobre o uso desta função.

```
struct nlattr * nl_rule;
int rem_rule;

        nla_for_each_nested(nl_rule, tb_msg[NL80211_ATTR_REG_RULES],rem_rule){

                struct nlattr * tb_rule[NL80211_REG_RULE_ATTR_MAX + 1];
                int flags, start_freq_khz, end_freq_khz, max_bw_khz;

                nla_parse(tb_rule, NL80211_REG_RULE_ATTR_MAX, nla_data(nl_rule),
                                nla_len(nl_rule),
                                NULL);

                start_freq_khz = nla_get_u32(tb_rule[NL80211_ATTR_FREQ_RANGE_START]);

                printf("freq: %d\n",start_freq_khz);
```

4. Análise de um programa NL80211 para scan de pontos de acesso

2. Abra o ficheiro "scan_access_points.c" e analise a sua estrutura tentando responder às seguintes questões:
   a. Identifique em que linha está identificado o nome da interface wireless onde abrir a socket *netlink*; altere para o nome da interface wireless obtido no ponto 4 do primeiro exercício.
   b. Identifique onde estão as funções de callback do código;

   c. Identifique que comandos Netlink são enviados (i.e., estão no formato "NL80211_CMD_..." e em que ordem eles são enviados;
   d. Identifique o comportamento de cada função de callback em função do comando enviado

3. Desenvolva um diagrama onde esquematiza a relação entre os elementos que analisou na alínea anterior.

V. Links úteis

libnl

- https://www.infradead.org/~tgr/libnl/
- https://www.infradead.org/~tgr/libnl/doc/core.html (Core Library Developer's Guide)

nl80211

- https://git.kernel.org/pub/scm/linux/kernel/git/linville/wireless.git/tree/include/uapi/linux/nl80211.h?id=HEAD (Kernel Documentation)

Debugging Netlink Sockets

- http://0x90.at/post/netlink-debugging

VI. Suporte NLMON para capturas com netfilter no Wireshark

Se estiver a replicar este trabalho em casa e obter erros a criar esta interface, – "*Operation not supported*" – poderá não ter esta funcionalidade disponível no *kernel* da sua distribuição Linux. O primeiro procedimento a tentar é ativar o módulo da driver NLMON, caso esteja compilado no *kernel* mas não ativo:

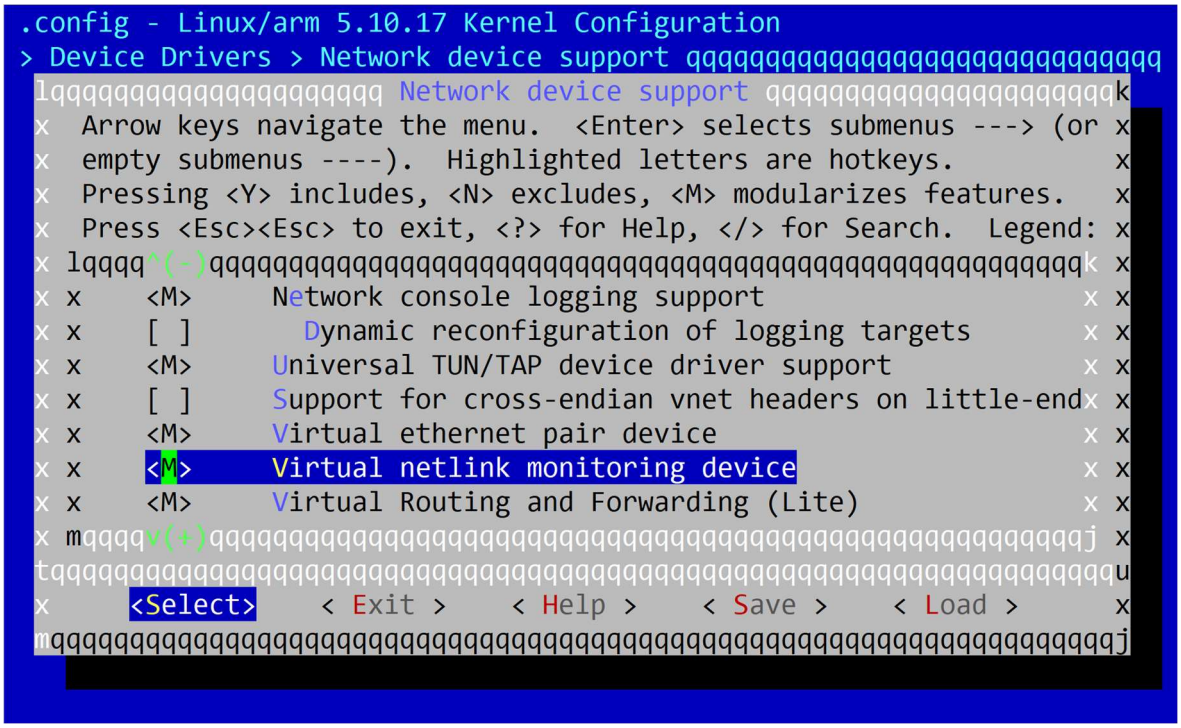
```
# modprobe nlmon
```

Caso continue a ter erro, significa que precisa de compilar o *kernel* com essa opção, e fazer *boot* a partir do mesmo. É preciso ter em conta que o NLMON só está disponível em *kernels* a partir da versão 3.11. Note que os PC's do laboratório estão a utilizar o Ubuntu 20.04, que já vem com esta funcionalidade compilada. Se utilizar outra distribuição, precisará de verificar qual o procedimento para compilar o kernel, garantindo que providencia suporte para a seguinte opção do ficheiro .config:

```
CONFIG_NLMON=m
```

Caso esteja a configurar as opções de compilação através do menu usando o comando "make menuconfig", a opção a ativar está dentro da seguinte estrutura:

```
Linux Kernel Configuration
└─> Device Drivers
      └─> Network device support
            └─> Virtual netlink monitoring device
```



Depois de compilar e arrancar com o novo *kernel*, deverá realizar o comando "modprobe nlmon" indicado acima, e os comandos de "ip link (…)" já irão funcionar.

VII. Instalação das bibliotecas libnl necessárias

Para compilar com sucesso os programas com libnl necessários para este guião, precisa de instalar a versão de desenvolvimento das bibliotecas "libnl" e "libnl-gen", ambas na versão 3. As versões de desenvolvimento (Linux) instalam no sistema os cabeçalhos (i.e., ficheiros .h) e os objetos compilados das bibliotecas

estáticas/dinâmicas (i.e., ficheiros libnl.a e libnl.so). No caso do Ubuntu (ou qualquer outra distribuição Linux baseada em Debian), as bibliotecas necessárias podem ser utilizadas através do seguinte comando:

```
# apt-get install libnl-3-dev libnl-genl-3-dev
```

No Ubuntu, este processo irá instalar os cabeçalhos das bibliotecas (i.e., os "#include …" no código) na pasta "/usr/include/libnl3", e irá instalar as bibliotecas estáticas e dinâmicas na pasta "/lib/<seu sistema>/…".