# Detecção de Periodicidade Baseada em BPF no Plano de Dados Utilizando a Transformada Discreta de Wavelet

**Nicolas Kolling Ribas**[1], **Gabriel Kerschner**[1], **Jéferson Campos Nobre**[1], **Marco Aurélio Spohn**[2], **Lisandro Zambenedetti Granville**[1]

[1] Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Porto Alegre, RS – Brasil

[2]Universidade Federal da Fronteira Sul (UFFS) – Chapecó, SC – Brasil

{nicolas.ribas,gabriel.kerschner,jcnobre,granville}@inf.ufrgs.br,
marco.spohn@uffs.edu.br

***Abstract.*** *This work presents an implementation of the Discrete Wavelet Transform (DWT) method for detecting periodic behavior in the network directly in the data plane using BPF (extended Berkeley Packet Filter). We created an algorithm that bypasses several limitations of BPF and performs DWT decomposition entirely in the data plane. To evaluate the performance overhead of our implementation we compared the use of periodicity detection using up to 17 decomposition levels with a baseline scenario. We then demonstrate that our implementation does not impose an overhead greater than 4% on the packet transfer rate, making it very suitable for high-traffic environments.*

***Resumo.*** *Esse trabalho apresenta uma implementação do método de Transformada Discreta de* Wavelet *(DWT) para detecção de atividades periódicas da rede diretamente no plano de dados utilizando* extended Berkeley Packet Filter *(BPF). Criamos um algoritmo que contorna várias limitações do BPF e executa a decomposição do DWT inteiramente no plano de dados. Para avaliar nossa implementação quanto a sobrecarga de desempenho, comparamos o uso da detecção de periodicidade utilizando até 17 níveis de decomposição com um cenário base. Demonstramos, então, que nossa implementação não impõe uma sobrecarga maior que 4% na taxa de transferência de pacotes, sendo assim, muito adequada para ambientes de alto trafego.*

## 1. Introdução

Para realizar tarefas como analisar o trafego de rede em busca de identificar atividades maliciosas, os operadores de redes encontram um ambiente desafiador dado o crescente volume de dados trafegados em decorrência da recente popularização de dispositivos conectados a internet. Em muitos cenários não é viável o armazenamento dos dados de medição de rede para uma análise posterior, sendo assim, devendo ser feita sobre o *streaming* de dados, do qual pode gerado um sinal a partir de característica do mesmo (*e.g.* quantidade de pacotes por intervalo de tempo). Utilizando-se de técnicas de processamento do sinal para analisar o *streaming* de dados podemos detectar atividades periódicas na rede, o que frequentemente indica padrões de uso que podem ser maliciosos ou benignos. As detecções de periodicidade podem, então, ser utilizadas para sinalizar ao operador a necessidade de uma análise mais aprofundada para determinar a causa da atividade.

O processamento de sinal utilizando o método da Transformada Discreta de *Wavelet* (DWT) oferece a capacidade de analisar dados de séries temporais com baixo custo computacional. O método consegue localizar o sinal tanto no tempo quanto na frequência [Roughan et al. 1998], pois se utiliza de uma maior resolução de tempo para componentes de maior frequência do que os com menor frequência, efetivamente analisando o sinal em diferentes escalas simultaneamente. Embora o método de DWT tenha baixa sobrecarga, para efetivamente atingirmos o processamento na taxa de linha, a aplicação de tecnologias como o *extended Berkeley Packet Filter* (BPF) se mostram promissoras ao permitirem o processamento eficiente dos pacotes de rede ao nível de *Kernel*. Juntamente com o *eXpress Data Path* (XDP), o BPF ainda possibilita o processamento diretamente no dispositivo de hardware de rede. Apesar disso, BPF torna a implementação mais difícil ao não suportar operações aritméticas mais avançadas.

Esse artigo apresenta *bpfwavelet*: uma solução que implementa, em BPF, a detecção de periodicidade baseada no método DWT diretamente no plano de dados. A *bpfwavelet* computa a transformada, localizando o sinal em tempo e frequência, e analisa os diferentes níveis de decomposição computando a função de energia. Por fim, a função de energia é utilizada em heurística baseada em limiar para detectar a periodicidade. Para contornar as limitações do BPF, realizamos mudanças matemáticas, assim eliminando operações aritméticas complexas não suportadas. Nossa solução tem um consumo de memória pequeno e se mostrou possuir uma baixa sobrecarga (não excedendo 4%) em cenário extremo de pacotes com tamanho mínimo de 48 bytes.

## 2. Fundamentação Teórica

A ideia fundamental por trás do DWT envolve a utilização de funções chamadas *wavelets*, que são pequenas ondas ou pulsos localizados no tempo. Para decompor um sinal, o DWT utiliza um filtro de passa-baixa (também conhecido como função de escala ou *wavelet* pai) e um filtro de passa-alta (também conhecido como função *wavelet* ou *wavelet* mãe). Esses filtros são convulsionados com $k$ pontos de dados de cada vez (dependendo do tamanho dos filtros), codificando informações de alta e baixa frequência em dois níveis distintos de decomposição e efetivamente fazendo uma sub-amostragem no sinal original pela metade. Os pontos de dados codificados gerados pelos filtros passa-alta e passa-baixa são chamados de coeficientes de detalhe e de aproximação, respectivamente. Podemos aplicar a decomposição DWT recursivamente $m$ vezes usando os coeficientes de aproximação no nível $j-1$ como entrada para o nível $j$ $(1 \leq j \leq m)$ para analisar frequências com uma granularidade mais fina. O sinal original corresponde ao nível zero.

Dentre as *wavelets* disponíveis, utilizaremos a *wavelet* de Haar, a qual é um das mais simples e frequentemente serve como introdução ao conceito de DWT [Bartlett et al. 2011]. Sendo assim, definimos os seus filtros de passa-alta e passa-baixa como sendo $(1/\sqrt{2}, -1/\sqrt{2})$ e $(1/\sqrt{2}, 1/\sqrt{2})$ respectivamente. Considerando a série temporal $X_{0,k}, k = 0, 1, 2, ...$ a qual corresponde ao sinal de entrada então os coeficientes de aproximação e detalhe são calculados pelas equações 1 e 2, respectivamente.

$$X_{j,k} = \frac{1}{\sqrt{2}}(X_{j-1,2k} + X_{j-1,2k+1}) \tag{1}$$

$$d_{j,k} = \frac{1}{\sqrt{2}}(X_{j-1,2k} - X_{j-1,2k+1}) \tag{2}$$

Para analisar padrões recorrentes na rede a função de energia dos coeficientes de detalhe tem sido utilizados [Huang et al. 2001]. A função de energia $E_j$ é definida como:

$$E_j = \frac{1}{N_j} \sum_k |d_{j,k}|^2, \quad j = 1, 2, \ldots, m \tag{3}$$

onde j representa o nível de decomposição e $N_j$ é o número de coeficientes no nível $j$. Calcular a energia dos coeficientes de detalhe em cada nível de decomposição nos permite examinar as propriedades temporais do sinal, indo das frequências mais altas para as mais baixas à medida que o nível de decomposição aumentam.

Ademais, [Feldmann et al. 1999] demonstrou que traçar a função $g(j) = \log_2(E_j)$ pode ser usada como um método para detectar periodicidades em um sinal. À medida que cada nível de decomposição filtra uma faixa de frequência específica no sinal original, uma periodicidade particular se manifesta como uma diminuição repentina em $g(j)$.

## 3. Decomposição Online do DWT

Para a nossa aplicação de detecção de atividades periódicas de rede, consideramos um sinal onde as amostras representam a quantidade de pacotes de um fluxo de rede em um intervalo de tempo fixo. Um fluxo se refere a todos os pacotes que correspondem a uma regra especificada pelo operador de rede, como todos os pacotes com uma determinada porta de destino ou endereço IP. Implementar a transformação DWT para esse tipo sinal em BPF traz um principal desafio, BPF não oferece suporte a todas as operações aritméticas necessárias para calcular as Equações 1, 2 e 3. Além disso, calcular e armazenar o sinal e os coeficientes de aproximação em diferentes níveis pode acarretar sobrecarga significativa de processamento e armazenamento.

À primeira vista, a Equação 3 apresenta desafios adicionais para sua implementação em BPF, incluindo a divisão por um número $N_j$ que varia ao longo do tempo para cada nível da decomposição e operações de ponto flutuante (divisão por $\sqrt{2}$). No entanto, ao expandirmos a equação para diferentes níveis, podemos simplificá-la a ponto de não serem mais necessárias essas operações. Inicialmente, supomos, que o comprimento do sinal é uma potência de dois, ou seja, $N = 2^n$ para algum $n$, de modo que $N_j = N/2^j$, onde $j$ representa o nível da decomposição ou o sinal quando $j = 0$. A Equação neste caso é representada por:

$$NE_j = 2^{j-1} \sum_k (X_{j-1,2k} - X_{j-1,2k+1}) \tag{4}$$

Para detectar a periodicidade no sinal utilizamos a eurística onde dividimos dois níveis de energia adjacentes e verificamos se o valor passa de um limiar. Se a energia cair de repente de um nível para outro então $\frac{E_{j-1}}{E_j}$ é maior que o limite. Mais especificamente checamos se $\frac{E_{j-1}}{E_j} > \frac{\alpha}{\beta}$. Valores para $\alpha$ e $\beta$ foram determinados empiricamente previamente por [Huaytalla et al. 2022]. Definindo $S_j = \sum_k (X_{j-1,2k} - X_{j-1,2k+1})$. Com intuito de evitar divisões que podem gerar ponto flutuante, podemos reescrever a equação que detecta periodicidade assim:

$$\beta S_{j-1} > 2\alpha S_j \tag{5}$$

**Algoritmo 1** Online DWT e calculo de energia

```
1:  procedure XDPFUNC
2:      count ← count + 1
3:      if first == true then
4:          first ← false
5:          SCHEDULE(Interval, CollectProcessSample)
6:      end if
7:  end procedure
8:  procedure COLLECTPROCESSSAMPLE
9:      SCHEDULE(Interval, CollectProcessSample)
10:     x ← count
11:     count ← 0
12:     k ← index
13:     for j ← 0 to levels and j ≤ MAX_LEVELS do
14:         if ISEVEN(k) then
15:             w_j ← x
16:             break
17:         end if
18:         if j ≠ 0 then
19:             s_j ← s_j + (w_j − x)^2
20:             if j ≥ 2 then
21:                 if β · s_{j−1} > 2 · α · s_j then
22:                     GENERATEALARM(j − 1)
23:                 end if
24:             end if
25:         end if
26:         x ← x + w_j
27:         k ← k/2
28:     end for
29:     index ← index + 1
30: end procedure
```

Para implementar o DWT de forma online, criamos um algoritmo que armazena apenas o último coeficiente de aproximação ($X_j$) calculado em cada nível, a soma acumulada $S_j$ de cada nível, o índice da amostra atual e um contador de pacotes. Sendo assim, nosso algoritmo armazena um total de apenas 2*MAX_LEVEL + 2 valores inteiros por fluxo, onde MAX_LEVEL é o número máximo de níveis de decomposição. O pseudo código da nossa implementação pode ser visualizado no Algoritmo 1. Para reprodutibilidade o código-fonte do nosso trabalho estão disponíveis para livre acesso. [1].

A primeira função *XDPFunc* é disparada pelo gancho XDP do BPF, sendo assim, é executada a cada chegada de pacote. Nela incrementa-se o contador de pacotes e, caso seja o primeiro pacote do fluxo, agenda-se a execução da função *CollectProcessSample* assim que se passarem $Interval$ nanosegundos. A função *CollectProcessSample* é executada a cada $Interval$ nanosegundos. Sua primeira execução é disparada pelo primeiro

[1] https://github.com/nicolaskribas/bpfwavelet

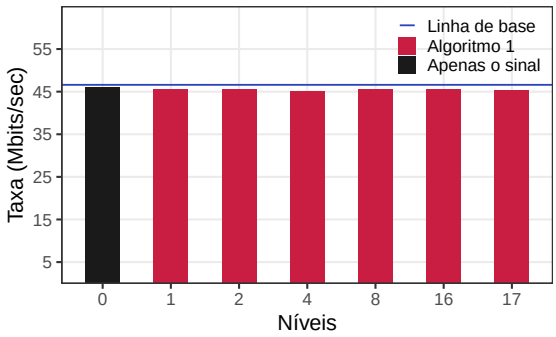


**Figura 1. Análise da taxa de transferência por nível de decomposição**

pacote chegado e as subsequentes execuções são agendadas por ela mesma (Linha 9). Nessa função, a contagem de pacotes é coletada, sendo essa, efetivamente, uma amostra do sinal. A decomposição é iniciada sendo feita para todos os níveis (Linha 13), até que o índice $k$ seja par (precisa-se de dois coeficientes—ou amostras—para se calcular a próxima decomposição), caso o índice do último coeficiente for par não se pode calcular uma nova decomposição. Enquanto o índice for impar a decomposição avança um nível. A soma $S_j$ é calculada na Linha 19, e na Linha 21 é utilizada em conjunto com $S_{j-1}$ para se identificar a periodicidade utilizando a Equação 5. Linhas 26 e 27 calculam o coeficiente de aproximação ($X_j$) do próximo nível e prepara o valor do índice do próximo coeficiente.

## 4. Avaliação

Para avaliar a nossa implementação quanto à sobrecarga que ela apresenta, executamos testes em um ambiente com dois servidores diretamente conectados, em uma configuração de *sender-receiver*. O servidor *sender* foi responsável por enviar pacotes para o servidor *receiver*, que processou esses pacotes utilizando nossa implementação de DWT em BPF. O objetivo era verificar como a implementação se comportaria sob diferentes níveis de decomposição da DWT e avaliar seu desempenho em relação ao cenário base, onde o DWT não foi usado.

Avaliamos com 0, 1, 2, 4, 8, 16 e 17 níveis de decomposição, coletando amostras com intervalo de $250\mu s$. Isso nos permitiu observar como o desempenho da implementação responde a diferentes complexidades de transformação. Utilizamos pacotes de 58 bytes em todos os testes. Esse tamanho foi escolhido para simular um cenário de pacotes com tamanho mínimo. Cada cenário de avaliação foi executado por 15 minutos. Os resultados da taxa de transmissão em cada cenário podem ser visualizado na Figura 1.

Como pode se observar, a sobrecarga foi muito pequena em comparação com o cenário base, independente da quantidade de níveis de decomposição, não passando de 4%. Isso pode ser atribuído ao fato de que nossa implementação utiliza-se do gancho XDP apenas para contar os pacotes, enquanto a transformada, calculo de energia e detecção de periodicidade propriamente ditas são executadas assincronamente utilizando as funções de temporizadores e *callbacks* do BPF, sendo assim não acarretam atrasos expressivos no processamento dos pacotes no momento que são recebidos.

## 5. Trabalhos Relacionados

No trabalho de [Huaytalla et al. 2022] é proposto uma implementação, em linguagem P4, do método de DWT para a detecção de atividades periódicas no sinal de rede. Em sua avaliação, com cenário similar ao testados neste trabalho (intervalo de amostra de $250\mu s$ e pacotes de 54 bytes), seu algoritmo demonstrou sobrecarga de 17%. O fato da implementação em P4 não ter a possibilidade de calcular as decomposições de forma assíncrona pode explicar esse valor tão alto em comparação com nossa implementação em BPF.

## 6. Conclusão e Trabalhos Futuros

Este estudo apresentou *bpfwavelet*: uma implementação eficiente do algoritmo online de DWT para detectar periodicidades na taxa de pacotes de rede. Ao adotar uma a abordagem utilizando BPF, conseguimos realizar essa análise em tempo real sem causar uma sobrecarga significativa no sistema, mantendo-a abaixo de 4%. Esse resultado demonstra a viabilidade do uso do BPF como uma ferramenta para a análise de tráfego de rede em tempo real, fornecendo informações sobre a periodicidade na chegada dos pacotes.

Em última análise, esta implementação do algoritmo online de DWT usando BPF demonstrou ser uma ferramenta promissora para a detecção de periodicidades na taxa de pacotes de rede, com potencial para melhorias adicionais e aplicações mais amplas na área de monitoramento de rede em tempo real.

É importante notar que este estudo não explorou todo o potencial do BPF, uma vez que não foi testado com hardware capaz de realizar o XDP *offload*, o que poderia otimizar ainda mais o desempenho do BPF em cenários de alto tráfego de rede. Portanto, há espaço para futuras pesquisas e otimizações, visando aproveitar ao máximo as capacidades do BPF para análise de tráfego em tempo real.

## Referências


Bartlett, G., Heidemann, J., and Papadopoulos, C. (2011). Low-rate, flow-level periodicity detection. In *IEEE Conference on Computer Communications Workshops*, pages 804–809.

Feldmann, A., Gilbert, A. C., Huang, P., and Willinger, W. (1999). Dynamics of ip traffic: A study of the role of variability and the impact of control. *SIGCOMM Comput. Commun. Rev.*, 29(4):301–313.

Huang, P., Feldmann, A., and Willinger, W. (2001). A non-instrusive, wavelet-based approach to detecting network performance problems. In *1st ACM SIGCOMM Workshop on Internet Measurement*, IMW '01, page 213–227, New York, NY, USA. Association for Computing Machinery.

Huaytalla, B. R., Jacobs, A. S., Silva, M. V. B., Carvalho, F. B., Ferreira, R. A., Willinger, W., and Granville, L. Z. (2022). Dwt in p4: Periodicity detection in the data plane. In *IEEE Global Communications Conference*, pages 6343–6348.

Roughan, M., Veitch, D., and Abry, P. (1998). On-line estimation of the parameters of long-range dependence. In *IEEE GLOBECOM*, pages 3716–3721.