

Tolerância a falhas - FFT

...

Augusto Boranga

Jonathan Martins

Lucas Assis

Murilo Wolfart

Fast Fourier Transform

- Algoritmo que transforma sinais no domínio de espaço para o domínio frequência
- Frequentemente utilizado em processamento de imagens
- Apresenta redundâncias inerentes

Implementações

- Algoritmo de Cooley-Tukey em Python
- Algoritmo de Cooley-Tukey em C
- Algoritmo de Cooley-Tukey em C++
- Biblioteca FFTW
- Biblioteca FFTW utilizando 2 threads

Testes

- Todas as injeções foram realizadas sob um arquivo jpg de 1024x1024
- A imagem foi (equivocadamente) linearizada em um vetor unidimensional
- Um algoritmo para a transformada inversa foi implementado mas não utilizado devido ao equívoco anterior, que foi percebido tardiamente

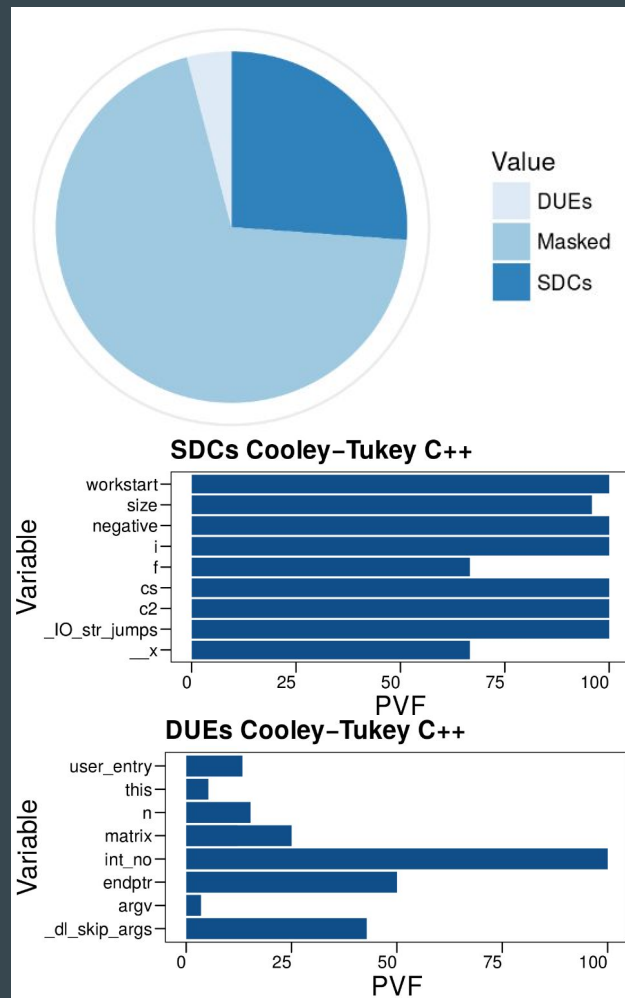
Vulnerabilidade

- Implementação em Python se mostrou extremamente vulnerável possivelmente graças à sua natureza interpretada
- Implementações utilizando a biblioteca FFTW se mostraram mais resilientes, possivelmente graças à verificações internas

Implementation	Injections	SDCs	DUEs	PVF.SDCs	PVF.DUEs
Cooley-Tukey C	3191	830	183	0.26	0.06
Cooley-Tukey CPP	3723	974	152	0.26	0.04
FFTW CPP	4907	1239	32	0.25	0.01
FFTW 2T CPP	3723	803	333	0.22	0.09
Cooley-Tukey Python	457	437	20	0.96	0.04

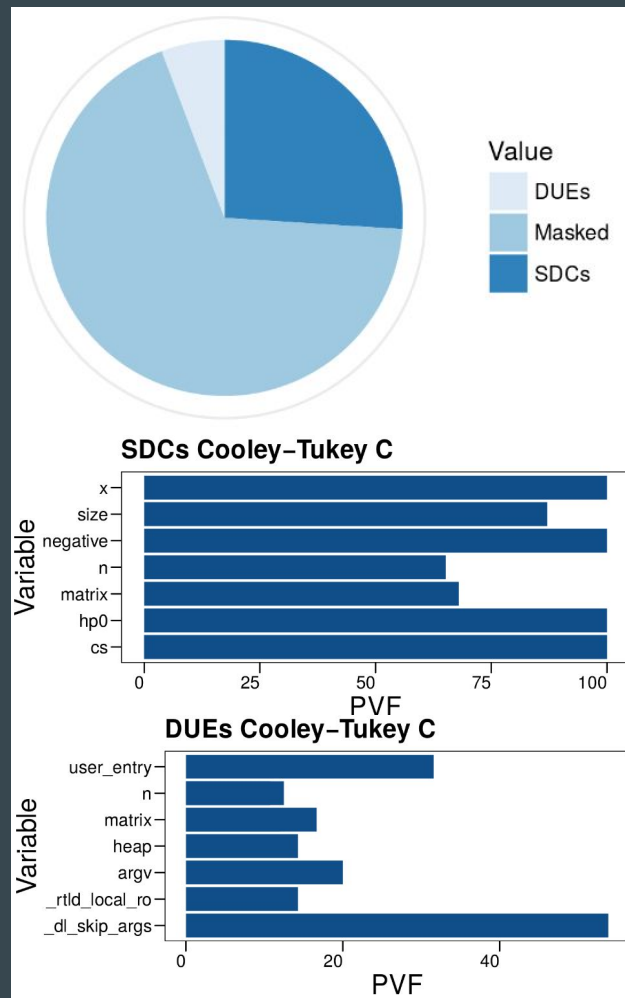
Cooley-Tukey C++

- Em SDCs, aparecem apenas duas variáveis pertencentes à aplicação
- Sendo variáveis de controle, a duplicação facilmente detecta a maioria das falhas sem grande prejuízo de tempo/memória
- Em DUEs, variáveis internas são incomparavelmente mais vulneráveis



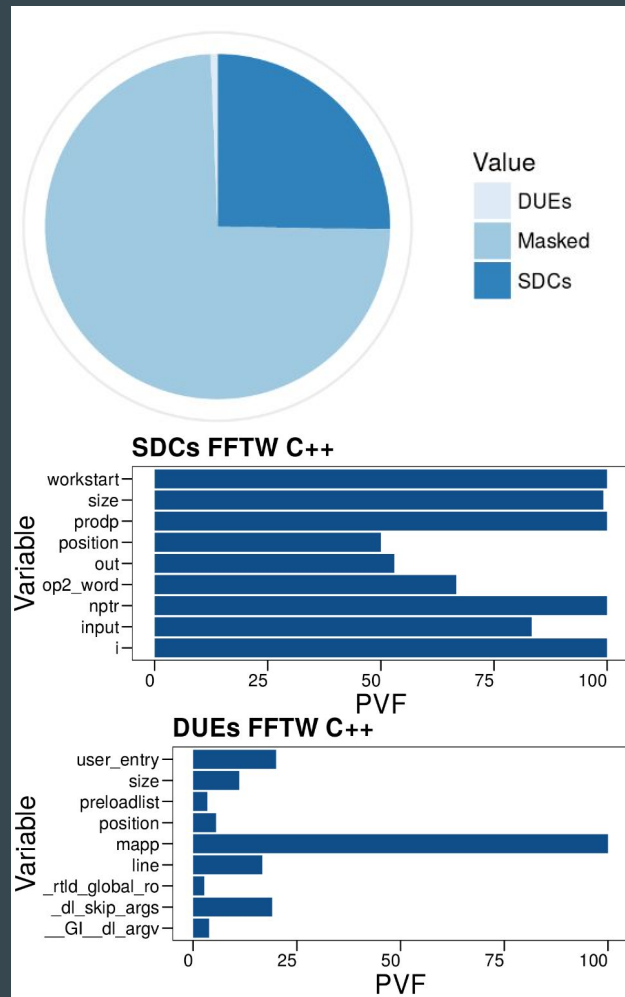
Cooley-Tukey C

- Em SDCs, além das variáveis de controle, passa a aparecer a matriz resultante, que precisa de técnicas mais elaboradas para ser protegida
- Em DUEs, variáveis internas continuam significativamente mais relevantes



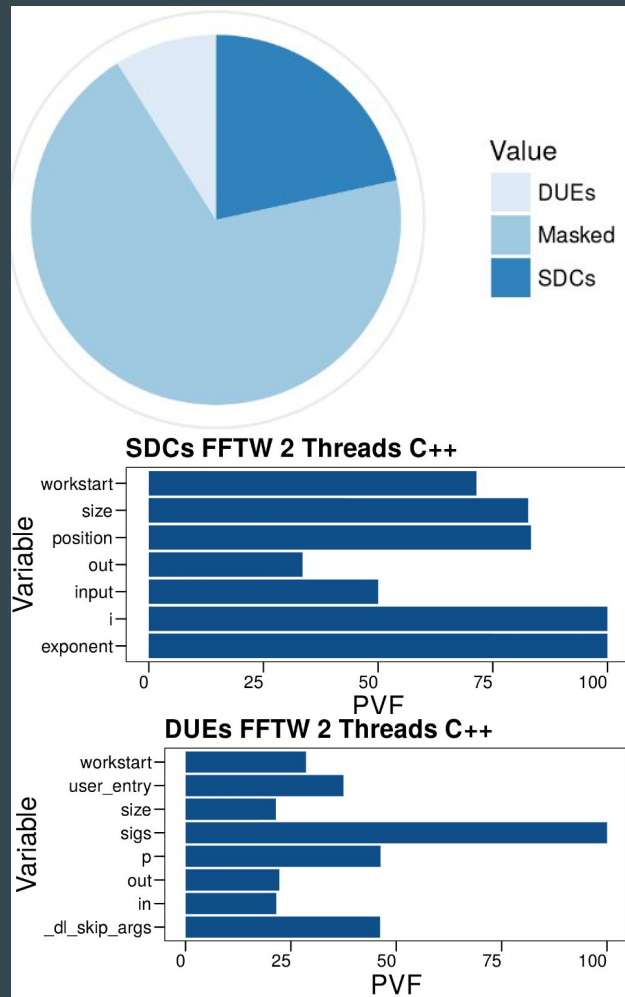
FFTW

- Em SDCs passam a aparecer o vetor original e o resultado final
- Pouquíssimos DUEs, possivelmente devido à verificações internas
- Novamente temos variáveis internas notavelmente mais vulneráveis em DUEs



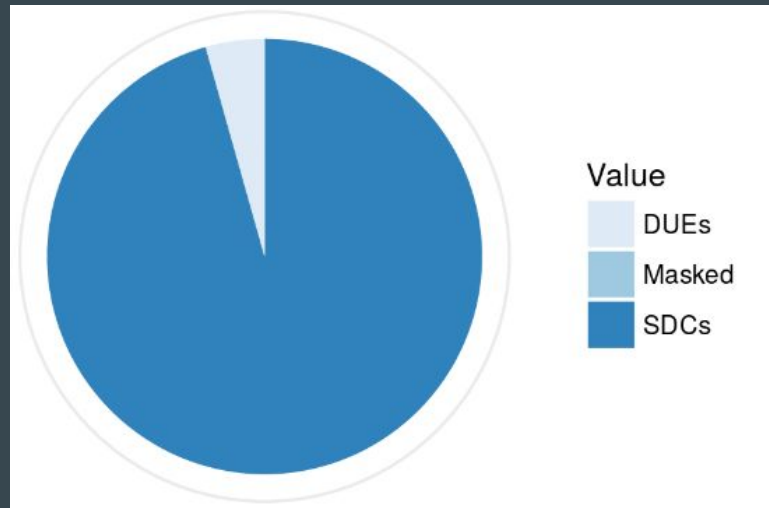
FFTW 2 Threads

- Novamente o vetor inicial e final aparecem como potenciais causadores de SDCs
- Maior percentual de DUEs entre todos os testes, possivelmente indicando que injeções são mais nocivas à uma execução paralela
- Dominância novamente das variáveis de controle dentre as maiores causadoras de DUEs



Cooler-Tukey Python

- Nenhum mascaramento em mais de 500 injeções
- Todas variáveis afetadas fazem parte do interpretador Python
- Análise abandonada

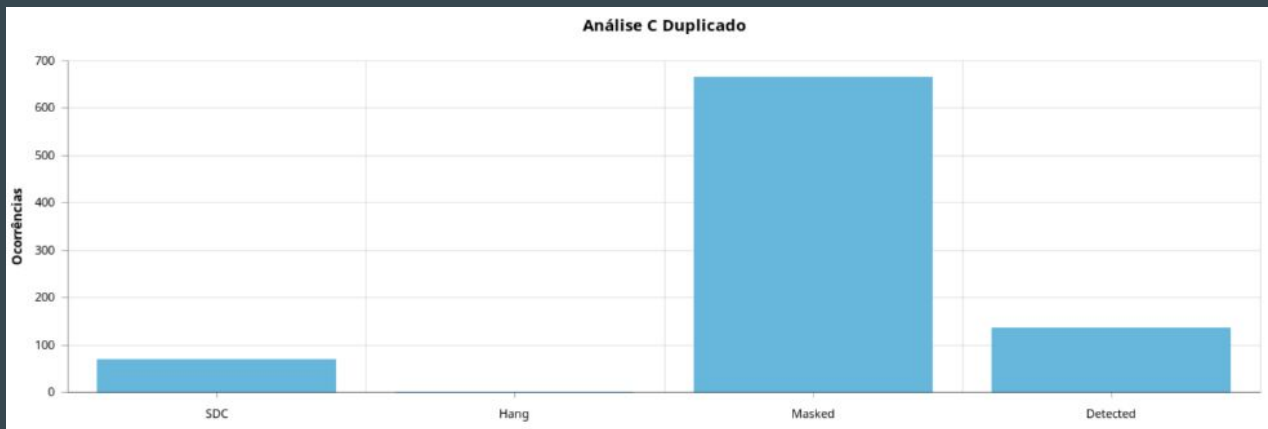


Duplicação do Código

- Realizada apenas para as implementações em C e C++
- Conta com um arquivo externo utilizado para sinalização de erros detectados internamente
- Compara com o resultado original apenas o primeiro resultado
- Não é capaz de correlacionar o resultado da injeção com a detecção interna do erro

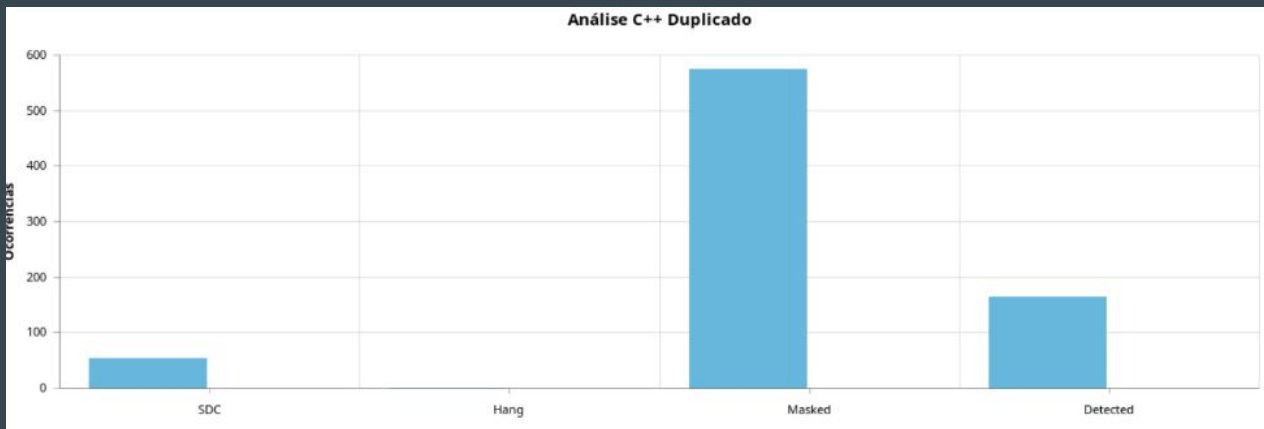
Duplicação do Código - C

- Foram detectados erros o suficiente para supor que todos os SDCs seriam detectados
- Alguns erros que não causaram SDCs foram detectados como errados, possivelmente por acontecerem na segunda execução



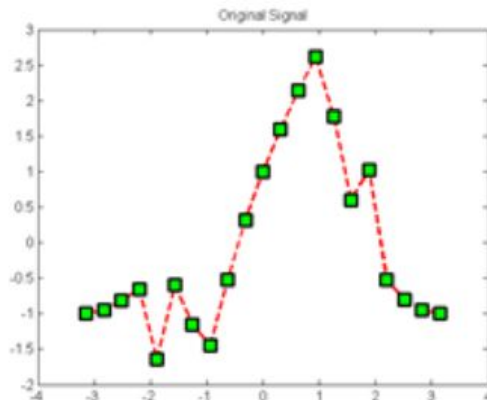
Duplicação do Código - C++

- Novamente, foram detectados erros o suficiente para supor que todos os SDCs seriam detectados
- Ainda mais erros não contabilizados pelo CAROL-FI foram contabilizados, reforçando sua possível ocorrência na segunda execução



Fast Fourier Transform - Redundância

■ DFT(S)



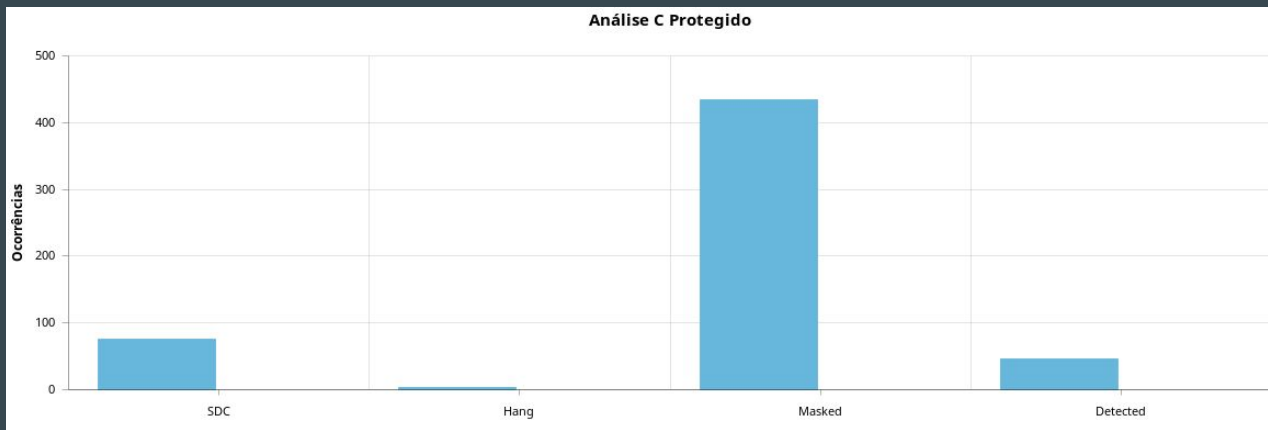
$$|F[u]| = |F[-u]| = \sqrt{R^2[u] + I^2[u]}$$

Same color indicates complex conjugates

	F(0)	F(1)	F(2)	F(3)	F(4)	F(5)	F(6)	F(7)	F(8)	F(9)	F(10)	F(11)	F(12)	F(13)	F(14)	F(15)	F(16)	F(17)	F(18)	F(19)	F(20)
coefficients	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	1.0	12.3	2.72	0.23	0.15	0.70	2.23	0.35	2.58	0.63	1.64	1.64	0.63	2.58	0.35	2.23	0.70	0.15	0.23	2.72	12.3
		400	92	74	26	64	28	11	23	44	37	37	44	23	11	28	64	26	74	92	400
	+	-	+	-	-	+	+	+	-	-	+	-	+	+	-	+	+	+	-	+	-
	9.97	7.57	0.80	0.10	0.70	1.81	0.22	1.00	0.13	0.12	0.12	0.13	0.13	1.00	0.22	1.81	0.70	0.10	0.80	7.57	9.97
	10i	67i	33i	12i	12i	39i	22i	21i	88i	51i	51i	88i	21i	22i	39i	12i	12i	33i	67i	10i	

“Selective Hardening” - C

- Não foram detectados erros o suficiente para cobrir todos SDCs
- Tempo de execução diminuiu em 30% em comparação com a duplicação, mas os erros detectados internamente foram reduzidos drasticamente



Conclusão

- A duplicação, além de ter um overhead enorme, cobre possivelmente mais erros do que seria necessário
- Apesar da existência da redundância na FFT, é preciso explorá-la mais cuidadosamente para que seu uso se torne vantajoso
- Ainda existe a possibilidade de que outras propriedades da FFT sejam utilizadas para realizar um selective hardening efetivo, mas o grupo não conseguiu explorá-las nesse trabalho