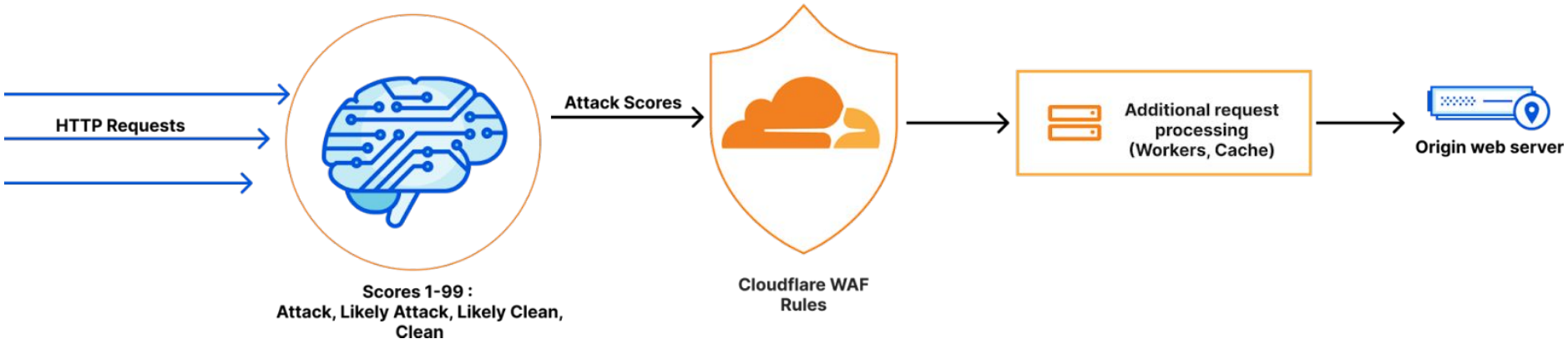
CLOUDFLARE

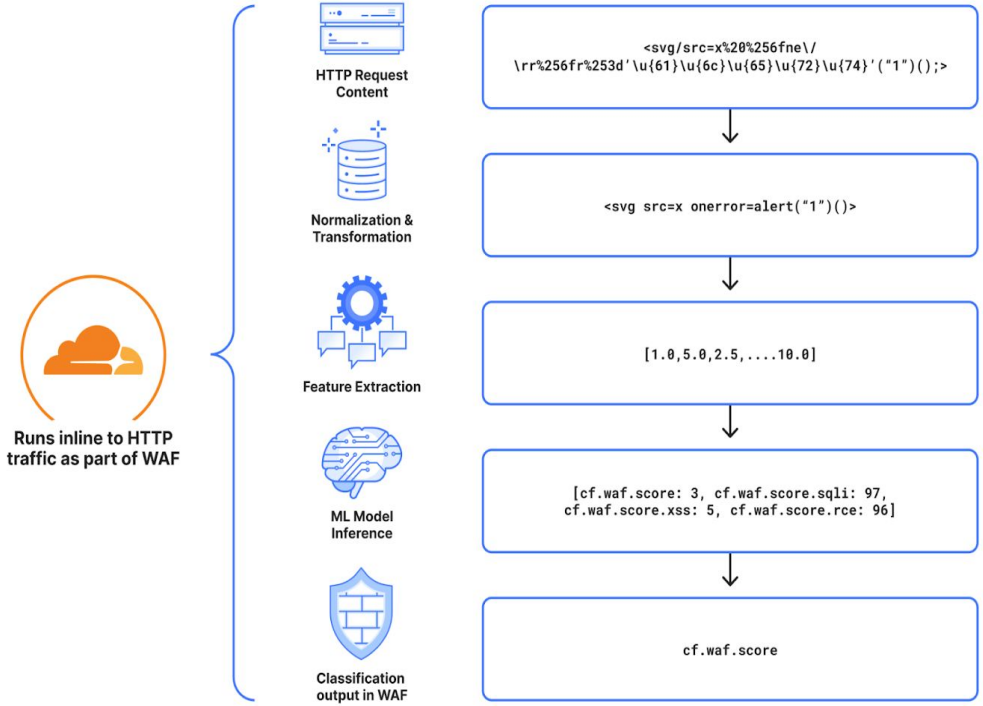
How to Scale your Machine Learning Dragon!

Denzil Correa
Senior Engineering Manager
Cloudflare
LeadDev Berlin 2025

WAF Attack Score detects 0-Day Before 0-Day at 84 Mn RPS!



WAF Attack Score protects against CVEs before they are known



No central bottleneck - Global Distributed Architecture

335

cities in 125+ countries, including mainland China

13,000

networks directly connect to Cloudflare, including every major ISP, cloud provider, and enterprise

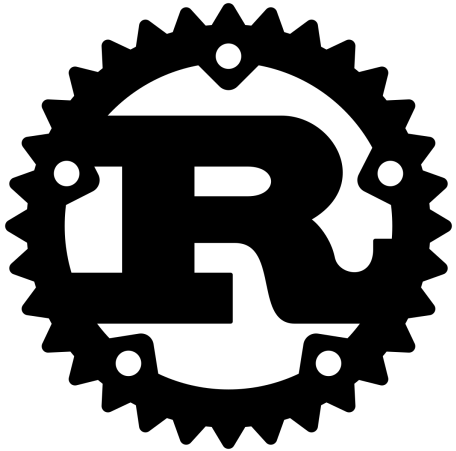
348 Tbps

global network edge capacity, consisting of transit connections, peering and private network interconnects

~50 ms

from 95% of the world's Internet-connected population

Pick the tools that enforce your performance envelope



TensorFlow Lite

Optimise your feature extraction pipelines!

	Description	Inference time (request body)	Gain
Baseline	Rust `HashMap` lookups	248.46 µs	

Optimise your feature extraction pipelines!

	Description	Inference time (request body)	Gain
Baseline	Rust `HashMap` lookups	248.46 µs	
Optimisation 1	`Aho Corasick`	248.46 µs → 129.59 µs	47.84% or 1.64x

Optimise your feature extraction pipelines!

	Description	Inference time (request body)	Gain
Baseline	Rust `HashMap` lookups	248.46 µs	
Optimisation 1	`Aho Corasick`	248.46 µs → 129.59 µs	47.84% or 1.64x
Optimisation 2	Rust `match`	248.46 µs → 112.96 µs	54.54% or 2.2x

Optimise your feature extraction pipelines!

	Description	Inference time (request body)	Gain
Baseline	Rust `HashMap` lookups	248.46 µs	
Optimisation 1	`Aho Corasick`	248.46 µs → 129.59 µs	47.84% or 1.64x
Optimisation 2	Rust `match`	248.46 µs → 112.96 µs	54.54% or 2.2x
Optimisation 3	Regex `WindowedReplacer`	248.46 µs → 51 µs	79.47% or 4.87x

Optimise your feature extraction pipelines!

	Description	Inference time (request body)	Gain
Baseline	Rust `HashMap` lookups	248.46 µs	
Optimisation 1	`Aho Corasick`	248.46 µs → 129.59 µs	47.84% or 1.64x
Optimisation 2	Rust `match`	248.46 µs → 112.96 µs	54.54% or 2.2x
Optimisation 3	Regex `WindowedReplacer`	248.46 µs → 51 µs	79.47% or 4.87x
Optimisation 4	Branchless ngram lookups	248.46 µs → 21.53 µs	91.33% or 11.54x

Push your model inference capabilities envelope

	Description	Inference time (request body)	Gain
Baseline	`TensorFlow Lite 2.6.0`	246.31 µs	

Push your model inference capabilities envelope

	Description	Inference time (request body)	Gain
Baseline	`TensorFlow Lite 2.6.0`	246.31 µs	
Optimisation 1	`SIMD AVX2`	248.46 µs → 130.07 µs	47.19% or 1.89x

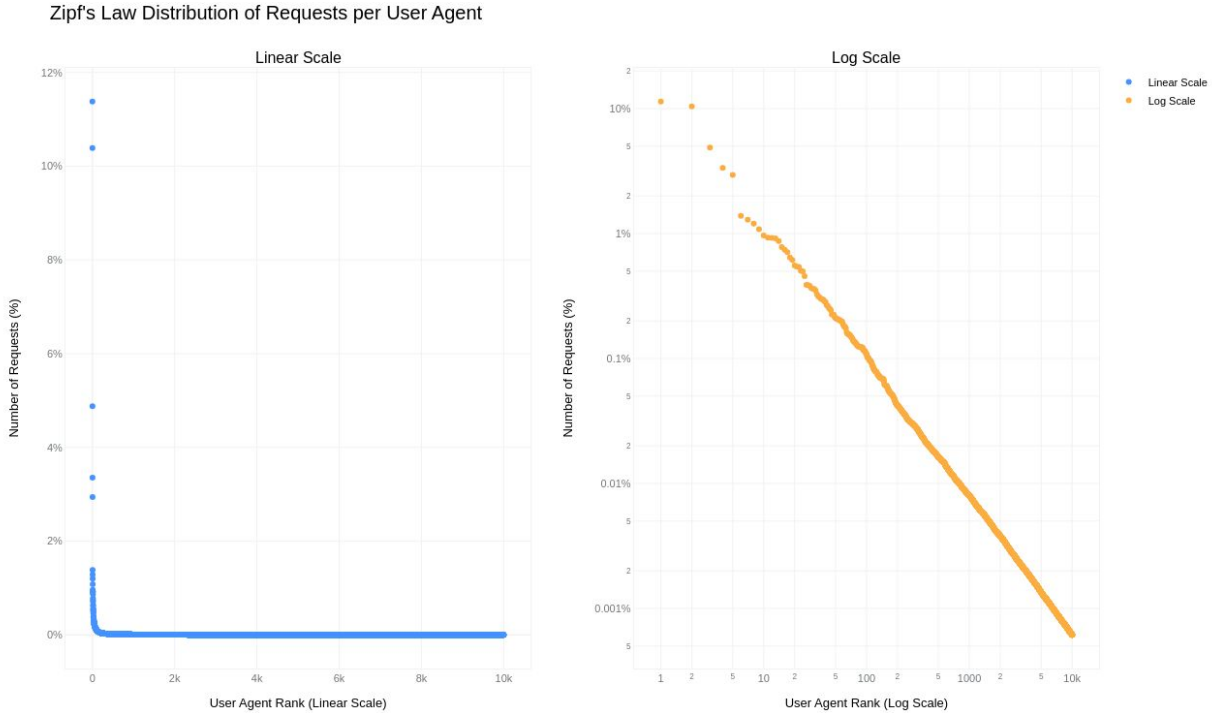
Push your model inference capabilities envelope

	Description	Inference time (request body)	Gain
Baseline	`TensorFlow Lite 2.6.0`	246.31 µs	
Optimisation 1	`SIMD AVX2`	248.46 µs → 130.07 µs	47.19% or 1.89x
Optimisation 2	`TF Lite 2.6.0 → 2.16.1 + SIMD AVX2`	248.46 µs → 115.17 µs	53.64% or 2.1x

Push your model inference capabilities envelope

	Description	Inference time (request body)	Gain
Baseline	`TensorFlow Lite 2.6.0`	246.31 µs	
Optimisation 1	`SIMD AVX2`	248.46 µs → 130.07 µs	47.19% or 1.89x
Optimisation 2	`TF Lite 2.6.0 → 2.16.1 + SIMD AVX2`	248.46 µs → 115.17 µs	53.64% or 2.1x
Optimisation 3	`TF Lite 2.16.1 + SIMD AVX2 + XNNPack`	248.46 µs → 56.22 µs	77.17% or 4.38x

Zipfian Traffic + LRU Cache → 70% Cache Hit Ratio



286.32 years of processing time saved every single day!

	Total Inference Time
Baseline	1519 µs

286.32 years of processing time saved every single day!

	Total Inference Time
Baseline	1519 µs
SIMD	884 µs

286.32 years of processing time saved every single day!

	Total Inference Time
Baseline	1519 µs
SIMD	884 µs
TFLite Upgrade + XNN Pack + Preprocessing optimisation	552 µs

286.32 years of processing time saved every single day!

	Total Inference Time
Baseline	1519 µs
SIMD	884 µs
TFLite Upgrade + XNN Pack + Preprocessing optimisation	552 µs
LRU Caching	275 us

286.32 years of processing time saved every single day!

	Total Inference Time
Baseline	1519 µs
SIMD	884 µs
TFLite Upgrade + XNN Pack + Preprocessing optimisation	552 µs
LRU Caching	275 us
Total Gain	**81.9% or 5.5x faster**

How to Scale your Machine Learning Dragon!

- **Architect for Scale** - Distributed by default design, bring the compute to edge
- **Performant tools** for Internet scale like TFLite and Rust
- **Performance Optimization** - Every CPU cycle and byte counts, optimise it!
- **Inference Optimization** - Quantisation, Vectorisation, Version checks etc.
- **Caching** is always an option, use the right cache strategy

Thank you

1 888 99 FLARE

denzil@cloudflare.com

cloudflare.com