High-Speed Hash Cracker

Distributed Password Hash Cracking Simulator dengan gRPC + Work-Stealing Scheduler

Nama: Sakti, Amor, Reza
Mata kuliah: Bokep

1. Ringkasan Konsep

Proyek ini mendemokan bagaimana proses “cracking” password hash dapat dipercepat dengan menggabungkan beberapa laptop menjadi satu cluster komputasi, menggunakan pembagian kerja berbasis “chunk” dan work-stealing scheduling. Implementasi ditujukan khusus untuk lingkungan lab sendiri sebagai alat edukasi untuk menjelaskan bahwa keamanan password adalah soal waktu dan biaya.

2. Latar Belakang & Motivasi

Hash seperti MD5/SHA-256 masih sering ditemukan pada sistem lama. Jika password lemah, hash tersebut dapat ditebak lewat serangan dictionary/bruteforce atau bahkan pre-computed lookup tables jika hashnya unsalted. Di sisi penyerang, percepatan utama biasanya datang dari paralelisme (banyak core/mesin).

Dengan demo ini, kelas bisa melihat secara nyata bahwa penyerang tidak selalu butuh superkomputer, beberapa perangkat yang bekerja sama bisa memberi peningkatan waktu yang berarti, sehingga pemilihan algoritma hash (mis. bcrypt) dan kebijakan password menjadi sangat penting.

3. Tujuan Pembelajaran

- Memahami konsep dasar password hash cracking (dictionary / brute force).
- Memahami desain distributed computing sederhana seperti master-agent, task queue, dan fault tolerance.
- Mendemokan work-stealing scheduling untuk mengatasi perbedaan performa perangkat (heterogeneous).
- Menghubungkan hasil demo dengan mitigasi defensif seperti hash yang tepat, cost factor berbentuk resource, dan password policy.

4. Desain Sistem (Secara High-Level)

Komponen utama:

Komponen	Peran	Data yang diproses
Master / Coordinator	Menyimpan job, membagi keyspace/wordlist menjadi chunk, memberi task ke agent, mengumpulkan hasil & progress.	Job manifest, chunk range, status lease, progress.
Agent / Worker	Meminta task, memproses chunk, mengirim progress/hasil, melakukan retry jika perlu.	Chunk range, konfigurasi hash, hasil kandidat.
Standalone Worker Pool	Mode cadangan bila jaringan bermasalah: worker berupa goroutine/thread lokal.	Chunk range internal, progress lokal.

5.1 Komunikasi (gRPC)

Komunikasi master-agent dilakukan via gRPC agar ringan, terstruktur, dan mudah dipantau (kalau ada saran yang lebih bagus silahkan wok). API minimal yang dibutuhkan untuk MVP:

- RegisterWorker(capabilities): agent mendaftar dan melaporkan kapasitas (CPU/cores).
- GetTask(): agent meminta chunk pekerjaan (pull-based).
- ReportProgress(task_id, progress): agent mengirim progres berkala.
- ReportResult(task_id, found/none): agent mengirim hasil akhir.

5.2 Work-Stealing Scheduler

Master memecah wordlist/keyspace menjadi banyak chunk kecil. Agent yang lebih cepat akan menyelesaikan chunk lebih cepat dan kembali meminta task berikutnya, sehingga otomatis mengambil porsi kerja lebih banyak. Agent yang lebih lambat tetap berkontribusi tanpa menjadi bottleneck.

5.3 Fault Tolerance

Untuk mencegah demo gagal saat jaringan tidak stabil, setiap task memakai mekanisme lease/timeout: jika agent putus sebelum selesai, chunk dikembalikan ke antrean (requeue) dan bisa dikerjakan agent lain.

6. Nilai Inovasi

- Adaptive chunking berbasis target durasi (mis. 5–10 detik/chunk), sehingga perangkat cepat mendapat chunk lebih besar dan perangkat lambat lebih kecil.
- Mode dual yakni Distributed (3 laptop atau lebih) + Standalone (goroutine pool) dengan kode core yang sama untuk fail-safe demo.
- Audit log & job manifest jadi semua eksekusi tercatat (worker mana mengerjakan chunk apa).

7. Rencana Demo

Alur:

1. Setup job: masukkan hash dummy (dibuat sendiri nanti) + wordlist kecil untuk demo.
2. Jalankan mode Standalone dulu agar audiens paham tentang hash cracking standar.
3. Jalankan mode Distributed: tampilkan 3 laptop terhubung, jelaskan kalau ada perubahan performa yang sangat jelas jika pakai distributed.
4. Jika jaringan bermasalah, switch ke mode Standalone dan tunjukkan konsep paralelisme lokal tetap berjalan.
5. Sampaikan poin hash lambat (bcrypt/Argon2) + password kuat menaikkan resource serangan.

8. Metode Evaluasi

Metode yang diukur:

- Throughput relatif (perbandingan waktu/kecepatan) antara 1 laptop (Standalone) vs 3 laptop (Distributed).
- Work Utilization: jumlah chunk yang diselesaikan tiap worker (menunjukkan work stealing berjalan).
- Stabilitas: ketika 1 worker dimatikan/disconnect, sistem tetap lanjut (requeue/lease).

9. Risiko & Mitigasi

- Wi-Fi kampus memblokir traffic: gunakan hotspot pribadi; siapkan mode Standalone sebagai cadangan.
- Perbedaan performa perangkat: gunakan chunk kecil/adaptive chunking agar yang kentang tidak bottleneck.
- Waktu demo terlalu lama: gunakan dataset kecil untuk demo dan fokus pada visual progress + konsep.

10. Pembagian Tugas Tim

Contoh pembagian:

- Orang 1: Master service (scheduler, job store, lease/requeue, dashboard sederhana).
- Orang 2: Worker/agent + benchmark ringan + progress reporting.
- Orang 3: Dokumentasi, test scenario demo, dan materi presentasi (ethical framing + hasil pengukuran).

Sumber Konsep Awal

Konsep awalnya komunikasi via gRPC, work-stealing dengan chunking, serta mode fail-safe standalone (goroutine pool) jika jaringan demo bermasalah.