Azərbaycan Kibertəhlükəsizlik Mərkəzi və SABAH qrupları

SECURE TOMORROW

Hackathon

Kritik

ZORBOX: MİLLİ SANDBOX LAYİHƏSİ

Hackathon taskının adı: “Zor Box” - Advanced Threat Protection

Kateqoriya: Kibertəhlükəsizlik Alətləri / Sandbox / Malware and Vulnerabilities Analysis

Vaxt: 48 saat

Komanda Tərkibi: Minimum 3 maksimum 5 nəfər

Səviyyə: Orta texniki dərəcə

1. TAPŞIRIĞIN TƏSVİRİ

Komandalar 48 saat ərzində komandaya aid **fayl və URL-dən yüklənən faylın avtomatik analizini icra edən veb platforması** hazırlayacaqlar.

Bu sistemin minimum MVP tələbləri aşağıdakılardır:

- ✓ URL sistemdəki müvafiq bölməyə daxil edildikdə onun daxilindəki faylı endirməlidir.
- ✓ Sistem yüklənmiş faylı sandbox mühitində analiz etməlidir.
- ✓ Əgər həmin fayl şifrəlidirsə şifrə daxil etmə imkanı olmalıdır.
- ✓ Arxa fonda **open-source sandbox inteqrasiyaları** + **Native Engine** sandbox işlənməsi nəzərdə tutulub.
- ✓ Hansı komanda Native Engine-i və open-source inteqrasiyanı daha effektiv və düzgün qurubsa, o daha çox bal alacaq.
- ✓ Layihədə süni intellekt tətbiqi məcburidir. Tətbiqi mexanizmi isə müstəqildir. Ən effektiv və düzgün tətbiq edən daha çox bal alacaq. Tapşırıq icra olunarkən kodun 50%-dan çoxunu Aİ ilə yazan komandaların aşağı bal alma ehtimalı var.
- ✓ Layihə Github mühitinə push edilməlidir. Prosesin gedişində hissə-hissə push edilməlidir.Sonda da bütün layihə tam şəkildə GitHub-da olmalıdır.

2. DƏSTƏKLƏNƏN FAYL TİPLƏRİ – Sistem tərəfindən analiz olunacaq fay tipləri nəzərdə tutulur.

- ✓ .exe (Windows executables)
- ✓ .dll (Windows libraries)
- ✓ .ps1 (PowerShell scripts)
- ✓ .js (JavaScript)
- ✓ .vbs (Visual Basic scripts)
- ✓ .bat, .cmd (Batch scripts)
- ✓ .elf, .bin (Linux binaries)
- ✓ .py (Python scripts)
- ✓ .jar (Java executables)
- ✓ .apk (Android packages)
- ✓ .zip (şifrəli və şifrəsiz – UI-da şifrə girişi hissəsi olmalıdır)
- ✓ .rar (şifrəli və ya şifrəsiz)
- ✓ .7z (şifrəli və ya şifrəsiz)
- ✓ .iso (disk image faylları)
- ✓ .img (USB/məlumat daşıyıcı image-lər)
- ✓ .pdf (şübhəli sənədlər üçün təhlil məqsədilə)
- ✓ .docx, .xls, .ppt (Office sənədləri – makrolu fayllar üçün risk analizi) (Windows executables)
- ✓ .ps1 (PowerShell scripts)
- ✓ .js (JavaScript)
- ✓ .vbs (Visual Basic scripts)
- ✓ .bat, .cmd (Batch scripts)
- ✓ .elf, Linux binaries
- ✓ .zip şifrəli və şifrəsiz (UI-da şifrə girişi hissəsi olmalıdır)

Kritik

MVP istiqamətində fəaliyyət üçün köməkçi cədvəl

Fayl tipi	Prioritet	Niyə vacibdir (qısa əsaslandırma)	MVP-də minimal analiz funksiyası (48h)	Qeyd / Risk
.exe	Must-have for MVP	Windows-da ən çox istifadə olunan zərərli fayl tipi; real-world hücumlarda geniş yayılıb.	Hash (MD5/SHA256), PE metadata (imports/exports), Packer/obfuscator əsas göstəriciləri, yara signature scan (yaxud YARA), VT/AV lookup (əgər API varsa).	Sandboxing tam olmayacaqsa, statik analiz yetərlidir; sandbox varsa təhlükəsiz icra.
.dll	Must-have for MVP	DLL-based yükleyicilər və side-loading hücumları üçün vacib; PE analiz eyni mexanizmdən istifadə olunur.	PE metadata, exported functions, suspicious importlar, YARA.	.exe ilə eyni analiz boru xətti.
.ps1	Must-have for MVP	Script-based hücumlar (post-exploitation, persistence) üçün çox istifadə olunur.	Static string/IOCs çıxarımı, suspicious cmdlet/obfuscation deteksiya (base64, encoded commands), basic sandbox (PowerShell constrained) əvəzinə regex-based IOC match.	Kod icrası riskli — sandbox-laşdırılmamış icra olmamalıdır.
.js	Must-have for MVP	Malicious web/launcher skriptləri; makro və drive-by scenarioları üçün əhəmiyyətli.	Static token/URL extraction, obfuscation detection (eval, unescape), basic YARA.	Browser-sandboxing çətindir; statik analiz prioritetdir.
.py	Nice-to-have	Açıq mənbə/script hücumları, məktəblər/CTF üçün faydalıdır.	Static AST-based suspicious API detection, IOC extraction.	Python bytecode dəstəyi əlavə oluna bilər.
.bat, .cmd, .vbs	Nice-to-have	Sadə persistence və script hücumları; təlim məqsədi üçün faydalıdır.	String/command extraction, suspicious command patterns (powershell, wmic).	Sadə analizlə kifayətlənmək olar.
.docx, .xls, .ppt (makrolu fayllar)	Must-have for MVP	Makro vasitəli məhvedici hücumlar hələ də geniş yayılıb.	Extract embedded objects, check for VBA macros, macro presence flag, basic macro IOC scanning (strings/URLs).	Makroları icra etməyin — əvəzinə extract+scan.
.pdf	Nice-to-have	Exploitable sənədlər və embedded JS üçün məqsədəuyğundur.	Extract JavaScript, embedded files, suspicious actions, metadata & embedded URLs.	Kompleks exploit analizini MVP-də saxlağı məsləhət deyil.
.zip, .rar, .7z (şifrəli/sifrəsiz)	Must-have for MVP	Arxivlar üzərindən fayl ötürmələri, şifrəli arxivlar eksfil üçün istifadə olunur. UI-da **şifrə girişi** sahəsi vacibdir.	Archive listing, nested extraction (şifrə yoxdursa), hash və mime-type mismatch, password prompt UI.	Şifrəli halda istifadəçidən şifrə tələb et; bruteforce etmə.
.apk	Nice-to-have	Mobil təhdidlər üçün əhəmiyyətli; əlavə resurs tələb edir (aapt, jadx).	APK manifest extract, permissions list, basic string/URL extraction.	Daha geniş dinamik analiz üçün vaxt/alat tələb edir.
.jar	Optional	Java-based məhvedici fayllar; web app explotasiya və malware.	Java class listing, suspicious reflective usage, embedded native libs.	MVP üçün geriyə çəkilə bilər əgər resurs məhdudsa.
.elf, .bin	Optional / Advanced	Linux server-side zərərli fayllar — real-world amma analiz çətinliyi daha yüksək.	ELF headers, sections, strings, basic imports (ldd) əvəzinə static strings.	Dynamic sandboxing (qemu) olmayan halda məhdud analiz.
.iso, .img	Optional	Disk images içində kompleks məzmun ola bilər — forensik əhəmiyyət.	File listing (mount/loop), notable large binaries detection.	Mounting təhlükəsizliyi və hüquq problemləri; MVP-də təxirə salın.

Nəticə — Prioritet xülasəsi

- **MUST-HAVE (MVP, ilk 48 saat):** .exe, .dll, .ps1, .js, .docx/.xls/.ppt (makrolar), .zip/.rar/.7z (şifrə sahəsi ilə).
- **NICE-TO-HAVE (əslində faydalı, əgər vaxt qalarsa):** .py, .bat/.cmd/.vbs, .pdf, .apk.
- **OPTIONAL / ADVANCED (sonrakı iterasiyalar üçün):** .jar, .elf/.bin, .iso/.img.

Optional / Advanced	Must-have for MVP	Nice-to-have		

Kritik

3. SİSTEM AXINI

1. Fayl yüklə və ya URL daxil et. - Must-have for MVP
2. URL əlavə edilibsə, sistem faylı avtomatik endirir. - Must-have for MVP
3. Fayl sandbox əlavə olunur: - Must-have for MVP

Open-source sandbox mühərrikləri:

- ✓ Strelka (metadata, yara scan, hash, file carving) - Must-have for MVP
- ✓ Litterbox (dinamik davranış simulyasiyası) - Nice-to-have
- ✓ Cuckoo Sandbox (klassik sandbox icrası və API əsaslı analiz) - Must-have for MVP
- ✓ CAPEv2 (malduck) (fork of Cuckoo, x64 + shellcode + VBA dəstəyi) - Must-have for MVP
- ✓ Firejail-based Simple Sandbox (bash+seccomp təcridli icra) - Must-have for MVP
- ✓ StuxnetBox / Flare VM (reverse engineering əsaslı interaktiv sandboxlar) - Optional / Advanced
- ✓ Viper (analiz + malware kolleksiya və YARA integrasiyası) - Nice-to-have
- ✓ Joe Sandbox Community Edition (bulud əsaslı limitli versiya) - Optional / Advanced
- ✓ Detux (Linux executable analiz üçün sandbox) - Optional / Advanced
- ✓ Sandsifter (low-level instruction trace analiz) - Optional / Advanced
- ✓ Qiling Framework (Python-da scriptable emulyator sandbox) - Nice-to-have

Native Engine (komanda tərəfindən yazılmış syscall/network/memory-based analiz) - Must-have for MVP

4. Risk dərəcəsi verilir (AI score, rule-based scoring) - Must-have for MVP
5. Nəticə: PDF/JSON/STIX formatda export + UI-də visual olaraq göstərilir - Must-have for MVP

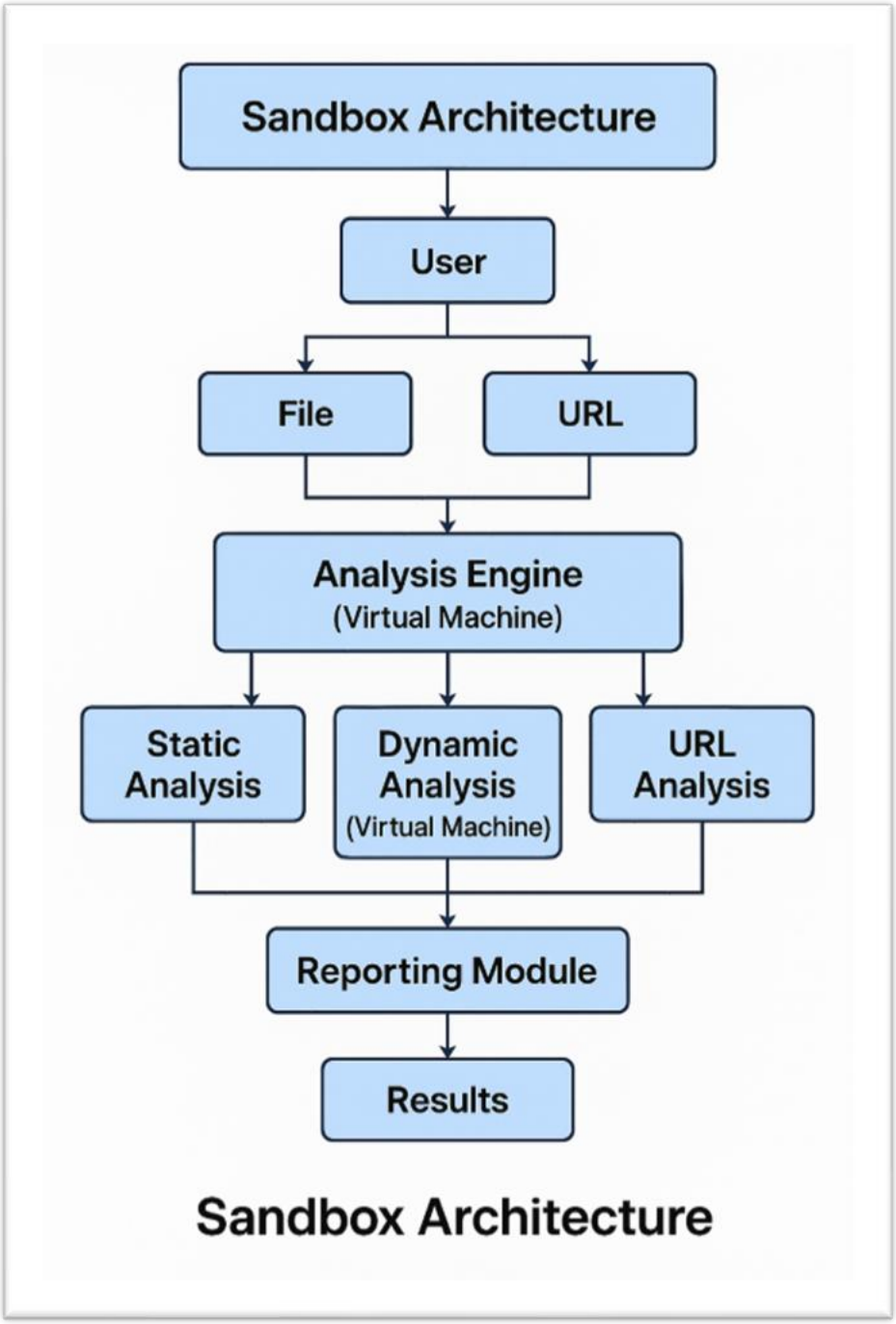
MVP nəticə üçün proses sistem axınının təsviri

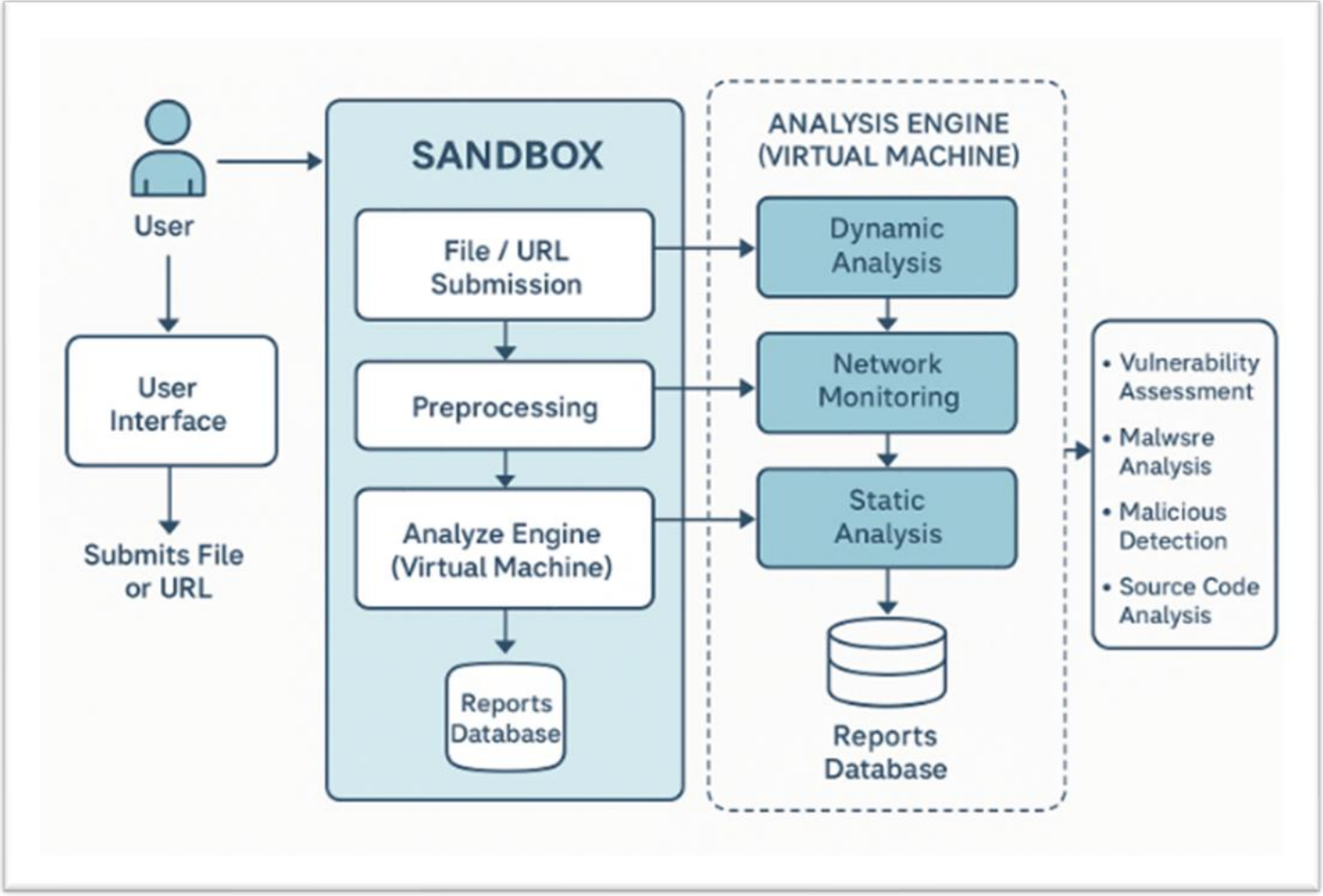
1. Fayl yüklə və ya URL daxil et.
2. URL-dən avtomatik yükləmə (əgər URL təqdim edilibsə).
3. Fayl storage və safepath-a yerləşdirilir; ilkin qarantiya yoxlamaları (size, mime, hash).
4. Fayl sandbox- və/ya statik-analiz boru xəttinə qoyulur (**strelka / cuckoo / CAPEv2 / native**).
5. Analiz nəticələri YARA/IOC/heuristic və AI-score ilə qiymətləndirilir.
6. Risk balı hesablanır (rule-based + AI hybrid).
7. Hesabatlar yaradılır (UI + PDF + JSON + STIX).
8. İstifadəçi və ya administrator interface-də nəticələr görsənir və feedback əlavə edə bilir.

№	Addım	Cavabdeh rollar	Inputs	Alətlər / Modul	Məhsul (Output)	Validation / Qəbul meyarı	Hackathon tapşırığı (İzlənilən)
1	Fayl yükləmə / URL təqdimatı	Frontend dev, UX	Fayl (upload) və ya URL	Frontend form, size limit, token auth	Upload job ID, metadata (filename, size, content-type)	Fayl 0 < size ≤ 10 MB; mime-type uyğun gəlir	Frontend komponent: upload + URL sahəsi + password-for-archive input. UI: progress & job-id qaytar.
2	URL-dən avtomatik yükləmə	Backend (fetcher)	URL	Downloader (requests/axios), content-type check, redirect handling	Yerli fayl kopyası, initial headers, HTTP status	HTTP 200; content-length mövcud və limit daxilində	Implement retry(3), timeout(30s), domain allowlist/blocklist.
3	İlk sanity yoxlaması	Backend	Fayl path	file magic (libmagic), hash (SHA256/MD5), size, mime	metadata record (hash,mime,size)	Hash generasiya uğurlu; mime != suspicious mismatch	Write DB entry; əgər mime-hint ilə extension mismatch → flag.
4	Karantin & Storage	Infra/Backend	Fayl	Isolated storage (no-exec), immutable perms	Safe path, retention policy	Fayl no-exec; owner=svc_user; storage quota yoxlanır	Implement safe path + ACLs; log əskikdirsə reject.
5	Archive handling & password prompt	Backend + Frontend	Archive fayllar (.zip/.7z/.rar)	libarchive, 7z, unzip; frontend password input	Extracted file list / nested archive tree	Extraction depth ≤ N; password required for encrypted archives	UI: password field & "provide later" flag. No bruteforce.
6	Static analiz boru xətti	Analyzer team	Fayl + type	Strelka (carving/YARA), PE parsers (pefile), oletools, pdf-parser, jsbeautify	Static report: strings, imports, macros, embedded objs, YARA hits	Parsers run without crash; YARA rules format valid	Implement modular static analyzers as microservices; output JSON schema.
7	Dynamic / Sandbox analiz (opsional)	Sandbox team	Fayl	Cuckoo, CAPEv2, Firejail sandbox, or Native engine	Behavioral report: network calls, processes, registry/files, syscalls	Sandbox completed within time-limit; network isolated	If infra limited, run minimal emulation (Cuckoo minimal VM) or drop dynamic and mark as "no-dynamic".
8	Native engine augmentasiya	Core devs (advanced)	Binary/syscall traces	Custom syscalls monitor, ptrace/qemu, memory dumper	Syscall trace, suspicious sequences, memory IOCs	Trace consistent; no host compromise	Implement as optional module to feed into scoring.
9	IOC / YARA / Threat intelligence enrichment	TI team	Static+dynamic outputs	YARA DB, TI feeds (local), VirusTotal API (if available)	Enriched IOCs, reputation tags	At least top-5 IOCs extracted; YARA rules matched	Build mapping: indicator -> confidence score.

Kritik

10	AI scoring & rule-based scoring	ML/Rules team	Features (imports, YARA, behavior, TI)	Lightweight ML model (logistic/reg tree) + rules engine	AI score (0-100), rule triggers	Model returns score + top features; rules evaluable	Provide transparent explainability: which features raised score.
11	Final risk aggregation	Core devs	AI score + rule hits + reputation	Weighted aggregator (configurable weights)	Final risk level (Low/Medium/High/Critical) + numeric score	Aggregator outputs stable mapping; unit tests pass	Expose weights in admin UI for hackathon tuning.
12	Report generation & export	Backend + Frontend	Aggregated analysis	PDF generator (wkhtmltopdf/reportlab), JSON schema, STIX serializer	PDF, JSON, STIX, UI report	Exports valid JSON & STIX; PDF readable	Provide example PDF template + downloadable files.
13	UI visualization & drill-down	Frontend	Report data	Charts, timeline, IOC list, YARA hits	Interactive UI page	All major data accessible; filters work	Implement timeline of behaviour, YARA hits list, download button.
14	Feedback loop & reanalysis	Ops + User	User feedback, false-positive flags	Re-run queue, manual override	Re-analysis job; adjusted rules	Re-analysis completes; rules updated in sandbox	UI: "Mark FP / Request reanalysis".
15	Audit & logging	Infra/Security	All actions	Central logging (ELK/Graylog), audit trails	Immutable logs	All actions logged with user/job-id	Provide simple audit viewer for admins.





Texniki detallar — hər modul üzrə MVP keyfiyyətinə çatmaq üçün tövsiyələr

A. Frontend

Elementlər: Upload, URL field, Archive password field, Job status, Results page, Download buttons.
Validasiya: Maks fayl ölçüsü (məsələn 10 MB), allowed extensions configurable.
UX: Job ID, progress bar, estimated queue position (opsional).

B. Downloader

Follow redirects, limit overall download size, domain blocklist/allowlist, SNI/SSL validation.
Save HTTP headers for TI enrichment.

C. Storage & Security

No-exec mount, read-only for analysis nodes; mount namespaces; AV scanning on store.
Limit nested extraction depth (məs. max 3).

D. Static analiz

PE: pefile — imports/exports, compile timestamp, sections, entropy (high entropy→packed).
Office: oletools — macro extraction, macro tokenization, suspicious keywords (AutoOpen, CreateObject).
Scripts: detect base64/encoded payloads, long obfuscated strings, suspicious eval/exec.

PDF: extract JS, /OpenAction, embedded files.
Archive: list files, mime detection per entry.

E. Dynamic analiz / Sandbox

Network: simulate internet with controlled sinkhole (no real outbound).
Snapshots: pre/post FS snapshot with file diff.
Time limit per job (məs. 120–300s).
Resources: CPU/memory quotas, seccomp / apparmor / firejail.

F. Scoring (məsləhət olunan metod)

Rule-based: YARA matches (weight per rule severity), presence of macros, suspicious imports, entropy.
AI-based: small explainable model (logistic regression) over features: yara_count, entropy_score, num_urls, suspicious_api_calls, reputation_score.
Aggregation: final_score = 0.6*rule_score + 0.4*ai_score (konfiqurasiya edilə bilər).
Map numeric score: 0–29 Low, 30–59 Medium, 60–79 High, 80–100 Critical.

G. Export schemas (minimal)

JSON: { job_id, filename, sha256, mime, yara_hits:[], macros:[], behaviors:[], ai_score, final_score, risk_level }
STIX: Create malware / observed-data / indicator objects for top IOCs.
PDF: Human-readable executive summary + technical annex (YARA hits, timeline).

GitHub modul strategiyası

Modul / Repo	Funksiya / Məqsəd	Əsas İş	Monitorinq Nöqtəsi	GitHub Push Tövsiyəsi
uploader-service	Fayl upload və URL submit	Fayl qəbul et, job yarat, metadata saxla	uploader.requests_total, upload_size_bytes, errors_total	README + env.example, unit test, PR şablon
fetcher-service	URL-dən avtomatik fayl yükləmə	URL qəbul edildikdə faylı endir, storage-a yönləndir	fetcher.download_duration_seconds, fetcher.download_failures_total	README, unit tests, PR şablon
storage-service	Safe storage və ilkin yoxlamalar	Faylı storage-a yerləşdir, hash/mime/size yoxlaması	storage.files_stored_total, storage.hash_time_seconds	README + example DB config, PR şablon
archive-handler	Arxiv və şifrəli fayl emalı	ZIP/RAR/7z arxivləri aç, şifrəli faylları flag et	archive.extractions_total, archive.encrypted_count	README, PR şablon
static-analyzer	Statik analiz (YARA/heuristics/IOCs)	PE/ELF/Office/Script faylları analiz et, YARA qaydalarını tətbiq et	analyzer.jobs_processed_total, analyzer.yara_hits_total	README + sample fixtures, PR şablon
sandbox-integration	Dinamik analiz	Cuckoo/CAPEv2/Firejail icra, native engine	sandbox.runs_total, sandbox.timeouts_total	README + infra notes, PR şablon
scoring-service	Risk balı hesablama	Rule-based + AI score, final risk səviyyəsi	scoring.requests_total, scoring.latency_seconds	README + model example, PR şablon

Kritik

orchestrator	Boru xətti koordinasiyası	Storage → Static → Sandbox → Scoring → Reporter workflow	orchestrator.jobs_in_state{state}, queue length	README + JSON schema, PR şablon
reporter-service	Hesabatların yaradılması	JSON, PDF, STIX reportları hazırlayıb storage-a göndər	reporter.reports_generated_total, reporter.pdf_generation_time_seconds	README, PR şablon
frontend-ui	İstifadəçi / administrator interfeysi	Upload, status, report viewer, feedback	RUM metrics, frontend errors	README, build + unit test, PR şablon
ti-enrichment	Threat intelligence	Domain/IP reputation, TI DB və optional VirusTotal API	ti.queries_total, ti.reputation_unknown_total	README, API keys GitHub Secrets-də, PR şablon
infra	IaC (Infrastructure as Code) və monitoring	k8s manifests (Kubernetes manifestləri — YAML formatında yazılmış fayllardır), docker-compose, Prometheus/Grafana stack	Alerts: disk usage, job queue, errors	README + monitoring docs, PR şablon

Qeyd:

- Hər modulun Dockerfile .github/workflows/ci.yml həmçinin ansible faylı olmalıdır.
- Monitorinq üçün Prometheus metrics və logs hər modulda expose edilməlidir.
- Orchestrator mərkəzi "job state" izləyicisi kimi istifadə olunur, digər modulların statuslarını toplamaq üçün.
- Frontend UI-da istifadəçi və admin nəticələri görür, feedback göndərə bilir; backend modullar bu feedback-ləri alıb iş axınına təsir edə bilər.

4. QİYMƏTLƏNDİRMƏ MEYARLARI (100 BAL)

Kateqoriya	Detal	Bal
Prototipin işləkliyi	Fayl/URL daxil edilib analiz çıxmalıdır	0 - 13
Native Engine sandbox	Davranış, syscall, scoring işləyirsə	0 - 13
Open-source sandbox inteqrasiyası	Ən azı **4** sandbox düzgün işləyir	0 - 13
Təhlükəsizlik	Fayl təcridi, icazə nəzarəti	0 - 13
UI və istifadənin rahatlığı	Dashboard + file/result interfeysi	0 - 12
Export mexanizmi	JSON və PDF çıxışı	0 - 12
Kod strukturu və sənədləşdirmə	GitHub, README, docker istifadə	0 - 14
Əlavə xüsusiyyətlər	STIX, mobil uyğunluq, Şəxsi yazılmış AI scoring modeli	0 - 10

5. BONUS BAL

- ✓ Fayl davranışlarını vizualizasiya edən qrafik – +3 bal
- ✓ REST API endpoint ("/analyze", "/result") – +2 bal
- ✓ 3 və ya daha çox open-source sandbox düzgün işləyirsə – +5 bal
- ✓ Mənbə kodlarını AI minimum istifadə edərək yazan komandalara - +10 bal