https://netcap.io | https://amsterdam-technologies.com | support@amsterdam-technologies.com | 2026-04-08

Netcap: Deep Network Packet Inspection Platform and Intrusion Detection System

A Solution Whitepaper by Amsterdam Technologies

The Problem

Network traffic remains the richest and most authoritative source of evidence for detecting intrusions, investigating incidents, and understanding the behavior of systems at scale. Yet the tools available to security teams for turning raw packet data into actionable intelligence have failed to keep pace with the complexity and volume of modern networks.

The fundamental challenge is structural. Traditional packet capture tools — tcpdump, Wireshark, tshark — produce raw packet data that requires extensive post-processing before it can be analyzed programmatically. Security researchers, SOC analysts, and data scientists routinely spend 60–70% of their project time on data collection, normalization, and transformation rather than on the detection logic itself. A typical network forensics workflow involves capturing traffic with one tool, exporting it to CSV or JSON with another, writing custom parsers for protocol-specific fields, and then manually aligning timestamps, flows, and sessions before any machine learning model or detection rule can be applied.

This gap creates several concrete problems:

Data engineering overhead consumes security resources. Organizations managing network monitoring infrastructure report that the majority of engineering effort goes into data pipelines rather than detection capabilities. Every new protocol, every new data source, every new ML experiment requires rebuilding the extraction and transformation layer.

Raw packet formats are hostile to machine learning. PCAP files are binary, variable-length, and protocol-dependent. Converting them into the high-dimensional, structured feature vectors that anomaly detection and classification algorithms require is a manual, error-prone process. There is no standard intermediate representation that preserves protocol semantics while remaining accessible to data science tooling.

Parsing untrusted input in memory-unsafe languages is dangerous. Many established network analysis tools are written in C or C++, where a malformed packet can trigger buffer overflows, use-after-free vulnerabilities, or memory corruption. When the traffic being analyzed is adversarial — as it is by definition in intrusion detection — the analysis tool itself becomes an attack surface.

Existing solutions force painful tradeoffs. Commercial network detection and response (NDR) platforms offer structured output but lock organizations into proprietary formats, opaque detection logic, and six- or seven-figure annual contracts. Open-source alternatives provide flexibility but require assembling a patchwork of disconnected tools — each with its own data model, configuration language, and operational overhead. Neither approach gives security teams a unified, extensible framework that produces machine-learning-ready structured data from raw traffic.

Scale demands concurrency; most tools are single-threaded. Modern networks generate traffic volumes measured in gigabits per second. Packet analysis tools that cannot exploit multi-core architectures become bottlenecks, forcing operators to sample traffic — and sampling means missed detections.

The landscape has shifted. The proliferation of IoT and industrial devices, the growth of encrypted protocols, the adoption of cloud-native architectures, and the increasing sophistication of adversaries all demand a new approach: one that treats network traffic as structured data from the moment of capture, operates safely on adversarial input, and provides the extensibility and interoperability that modern security workflows require.

Solution Overview

Netcap (NETwork CAPture) is a deep network packet inspection platform and intrusion detection framework developed by Amsterdam Technologies. It addresses the problems outlined above through a single architectural insight: network traffic should be converted into typed, structured audit records at the point of capture — not as an afterthought in a downstream pipeline.

Where traditional tools produce opaque binary captures that require protocol-specific parsers at every stage of analysis, Netcap transforms packet streams into platform-neutral, type-safe structured records encoded with Google Protocol Buffers. These records preserve the full semantic richness of each protocol layer — 66+ audit record types covering TCP, UDP, HTTP, TLS, DNS, SSH, DHCP, and dozens more — while being immediately consumable by any programming language, data science framework, or machine learning pipeline.

The framework is implemented in Go, a language chosen specifically because its garbage-collected, memory-safe runtime eliminates the class of vulnerabilities — buffer overflows, use-after-free, memory corruption — that make parsing adversarial network data in C/C++ tools inherently risky. This is not a convenience choice; it is a security architecture decision.

Netcap's concurrent design exploits multi-core architectures to achieve gigabit-speed capture and processing. Its distributed collection architecture — with dedicated sensor agents and collection servers — scales to enterprise deployments. And its extensible decoder framework allows new protocols to be added without modifying the core pipeline.

The project earned 2nd Place at Kaspersky Labs SecurIT Cup 2018 and has its origins in academic research on anomaly-based network intrusion detection. It is offered in three tiers: **Netcap Core**, a GPLv3-licensed open-source CLI framework; **NETCAP Pro**, a cross-platform desktop application with advanced graph analysis, AI-powered threat detection, and professional tool integrations; and **NETCAP Enterprise**, a custom-priced offering for teams and organizations requiring unlimited seats, priority SLA support, and custom integrations.

Netcap is currently in active Beta, with the core framework stable and production-tested, and the Pro desktop application undergoing continuous enhancement.

Key Capabilities

Protocol Decoding and Audit Record Generation

At its core, Netcap transforms raw packet data into structured audit records. The framework ships with **66+ audit record types**, of which 53 are protocol-specific and 5 are flow models, covering the full spectrum of network communication:

- **Transport and Network Layers:** TCP, UDP, IPv4, IPv6, ICMP, ARP, IGMP, ICMPv6
- **Application Protocols:** HTTP, TLS, DNS, DHCP, SSH, FTP, SMTP, NTP, SNMP, SIP
- **Specialized Protocols:** USB, Modbus (industrial), and emerging protocol support
- **Flow Abstractions:** Connection records, bidirectional flows, unidirectional flows, and custom abstractions

Each audit record is a typed Protocol Buffers message with well-defined fields. A DNS audit record, for example, contains the query name, query type, response code, answer records, authority records, and timing information — all as first-class structured fields, not as raw byte arrays requiring downstream parsing.

The decoder architecture is split into two categories: **packet decoders** (75+ individual decoders, one per protocol layer) that process individual packets, and **stream decoders** (40+ TCP stream-based decoders) that handle stateful protocols like TLS, SSH, QUIC, and SMB by reconstructing TCP streams before decoding.

This matters because the alternative — extracting the same information from PCAP files using ad hoc scripts — is brittle, slow, and requires deep protocol expertise for each protocol of interest. Netcap eliminates this entirely: capture once, analyze across all protocols simultaneously.

Deep Packet Inspection

Netcap integrates optional Deep Packet Inspection (DPI) capabilities through nDPI and libprotoident, enabling protocol identification beyond port-based heuristics. This is critical for detecting protocol tunneling, encrypted traffic classification, and identifying applications that use non-standard ports.

The DPI subsystem operates as an optional module (enabled at build time) and enriches audit records with application-level protocol classification. Version information for the DPI libraries (nDPI, libprotoident) is tracked and displayed in the build output, ensuring reproducibility across deployments.

Service Probe Matching and HTTP Header Analysis

Netcap implements nmap-style service probe matching to identify services running on arbitrary ports. When standard service probes fail to identify a service, a fallback mechanism parses HTTP response headers to extract technology information:

- **Server headers** identify web servers (Apache, nginx, IIS, Cloudflare)
- **X-Powered-By headers** reveal frameworks and languages (PHP, ASP.NET, Express)
- **X-Generator headers** identify content management systems (WordPress, Drupal, Jekyll)
- **Via headers** detect proxies and CDNs (Varnish, Squid)

Headers are parsed in priority order, and the first match produces structured Service and Software audit records with product name, version, and vendor information. This layered detection approach ensures that web-facing services are reliably identified even when they do not respond to standard service probes.

Credential Harvesting and Hash Extraction

Netcap extracts authentication credentials and cryptographic hashes from network traffic for forensic analysis and security assessment. Currently implemented harvesters cover:

- **Plaintext protocols:** FTP, Telnet credentials
- **HTTP authentication:** Basic Authentication, Digest Authentication (with enhancement planned for full Hashcat-compatible output)
- **Email protocols:** SMTP (Auth Plain, Auth Login, CRAM-MD5), IMAP (Login, Authenticate Plain, CRAM-MD5)

The roadmap includes high-priority additions for Windows and Active Directory environments: NTLMSSP hash extraction (NTLMv1 and NTLMv2 across SMB, HTTP, IMAP, SMTP, and LDAP), Kerberos AS-REQ pre-authentication hashes, Kerberos AS-REP roasting support, and Kerberos TGS-REP Kerberoasting support. These will produce output in Hashcat-compatible formats for direct offline cracking verification.

Machine Learning-Ready Output

Netcap's output formats are designed specifically for consumption by data science and machine learning pipelines:

- **Protocol Buffers** (`.ncap`): The default binary format, providing compact storage, type safety, and cross-language accessibility. Protocol Buffers can be read natively from Python, Java, C++, JavaScript, and dozens of other languages.
- **CSV**: A common interchange format for tools like pandas, scikit-learn, and R.
- **JSON**: Streaming JSON output for integration with Elasticsearch, Splunk, and log aggregation systems.
- **Prometheus Metrics**: Real-time metric export for monitoring dashboards with protocol counters, decoder timing, and packets-per-second gauges.

The labeling tool (`label`) enables creation of labeled CSV datasets from Netcap audit records, a critical capability for supervised machine learning. Researchers can apply label configurations (e.g., attack classifications from datasets like CIC-IDS2018) to produce training data for anomaly detection and threat classification models without manual annotation.

Graph Analysis Engine (Pro)

NETCAP Pro introduces a Maltego-style visual link analysis engine that renders network relationships as interactive, explorable graphs. Analysts can visualize connections between hosts, trace lateral movement patterns, map attack infrastructure, and explore DHCP leases, DNS resolutions, HTTP file extractions, and credential flows as interconnected graph nodes.

This capability transforms the analysis experience from scrolling through tabular data to spatially navigating network relationships — a critical advantage for threat hunting and incident investigation where the connections between entities are often more revealing than the entities themselves.

AI-Powered Analysis (Pro)

NETCAP Pro incorporates intelligent analysis features including automated threat detection, anomaly identification, and report generation. These AI capabilities operate on the structured audit records produced by the core engine, leveraging the high-dimensional feature space that Netcap's 66+ record types provide to identify patterns that manual analysis would miss.

Timeline Analysis (Pro)

The timeline analysis module provides a visual chronological representation of network activity, enabling temporal pattern recognition and forensic reconstruction. Analysts can identify the precise sequence of events during an incident, correlate activities across different protocol layers, and establish the temporal relationship between initial compromise, lateral movement, and data exfiltration.

Tool Integrations (Pro)

NETCAP Pro integrates with the tools that security professionals already use:

- **Wireshark**: Deep-dive into specific packets or streams identified by Netcap analysis
- **Metasploit**: Correlate captured network activity with exploitation frameworks
- **Hashcat and John the Ripper**: Export extracted hashes for offline password cracking
- **BetterCrack**: Additional credential verification workflows
- **Maltego**: OSINT platform integration via the transform plugin for visual intelligence analysis

The Maltego integration is particularly deep, supporting DHCP information extraction, HTTP file extraction, and HTTP parameter command injection analysis directly within the Maltego investigative interface.

35+ Analysis Modules (Pro)

NETCAP Pro ships with over 35 specialized analysis modules organized by domain:

CATEGORY	MODULES
Network Intelligence	Alerts, Connections, Hosts, Domains, Services, Probes
Security Analysis	Credentials, Vulnerabilities, Fingerprints, Rules, DPI
Forensics	Files, Certificates, Software, Devices, Audit Records
Operations	Logs, PCAPs, HTTP, Decoders, Harvesters, BPF Filters
Visualization	Graph, Timeline

Each module provides a focused analytical lens on the captured data, allowing analysts to switch between high-level overview and deep protocol-specific investigation without leaving the application.

Distributed Collection Architecture

For enterprise deployments, Netcap provides a distributed collection architecture with two components:

- **Agents**: Lightweight sensor processes deployed at network capture points (honeypots, network taps, span ports)
- **Collectors**: Central collection servers that receive, aggregate, and store audit records from multiple agents

This architecture enables monitoring of geographically distributed networks, multi-segment enterprise environments, and honeypot arrays from a single analysis interface.

Detection Rules Engine

Netcap supports YAML-based detection rule definitions that can be applied to captured traffic. Rules are organized by category (e.g., streaming protocols, authentication anomalies) and can be executed through the WebUI against live or historical captures. The rules engine includes input sanitization to prevent path traversal attacks when processing rule names containing special characters.

Investigation Notes and Session Management (Pro)

NETCAP Pro allows analysts to annotate hosts, devices, and data fields with investigation notes, creating a persistent record of analytical reasoning. Sessions can be exported and loaded, enabling collaborative investigations where multiple analysts work on the same dataset and share findings.

Architecture and Technical Design

Language and Runtime

Netcap is implemented in **Go** (Golang), a choice driven by three requirements:

1. **Memory safety**: Go's garbage-collected runtime eliminates buffer overflows, use-after-free, and memory corruption vulnerabilities — the precise class of bugs that make parsing adversarial network data in C/C++ dangerous. When the input is potentially malicious (as it always is in network security), the analysis tool must not itself become a vulnerability.
2. **Concurrency**: Go's goroutine model and channel-based communication enable efficient multi-core utilization without the complexity of manual thread management. Netcap's concurrent worker pool distributes packet processing across available CPU cores, achieving sustained high throughput.
3. **Cross-platform compilation**: A single codebase compiles to native binaries for Linux, macOS, and Windows, ensuring consistent behavior across platforms.

Processing Pipeline

The data flow through Netcap follows a clearly defined pipeline:

1. **Packet Acquisition**: The Collector reads packets from live network interfaces (via AF_PACKET sockets on Linux, libpcap on macOS/Windows) or from PCAP dump files. Multiple decode strategies are available — eager decoding with buffer copying (safe default), lazy decoding (single-threaded optimization), zero-copy mode (minimal allocation), memory-pooled mode (reduced GC pressure for high throughput), and datagram mode (application-layer decoding without stream reassembly).
2. **Worker Distribution**: Packets are distributed to a configurable worker pool using symmetric flow hashing, ensuring that packets belonging to the same connection are processed by the same worker — critical for stateful protocol decoding.
3. **Protocol Decoding**: Packet decoders and stream decoders convert raw data into typed Protocol Buffers audit records. Stream decoders reconstruct TCP connections before decoding stateful protocols (TLS handshakes, HTTP request/response pairs, SSH sessions).
4. **Enrichment**: The Resolvers subsystem enriches audit records with contextual data — DNS reverse lookups, GeoIP geolocation, and MAC vendor identification.
5. **Output**: Writers serialize audit records to the configured output format (Protocol Buffers, CSV, JSON, or Elasticsearch) with optional gzip compression.

Serialization with Protocol Buffers

All 66+ audit record types are defined in a single Protocol Buffers schema (`netcap.proto`) and generated using `protoc-gen-gogo`, an optimized Protocol Buffers code generator for Go. This design decision provides:

- **Type safety**: Every field in every audit record has a defined type, preventing the schema drift and type confusion that plague CSV-based pipelines.
- **Compact storage**: Protocol Buffers produce significantly smaller output than equivalent JSON or CSV, reducing disk footprint for long-duration captures.
- **Cross-language interoperability**: Protocol Buffers can be deserialized natively in Python, Java, C++, JavaScript, C#, Ruby, and other languages, enabling security teams to use whatever tools best suit their analysis needs.

TCP Stream Reassembly

The reassembly subsystem reconstructs TCP streams from individual packets, handling out-of-order delivery, retransmissions, and fragmentation. This is essential for decoding application-layer protocols that span multiple packets (HTTP responses, TLS handshakes, file transfers). The reassembly engine manages its own page cache with periodic flushing to prevent memory exhaustion during long captures.

Decode Options and Performance Tuning

Netcap exposes five packet decoding strategies, each optimizing for different tradeoffs:

STRATEGY	BUFFER COPY	LAZY DECODE	CONCURRENCY SAFE	GC PRESSURE	BEST FOR
Default	Yes	No	Yes	Moderate	General use
Lazy	Yes	Yes	No	Moderate	Single-threaded analysis
NoCopy	No	No	Yes	Low	Controlled buffer environments
Pool	Pooled	No	Yes	Very Low	High-throughput sustained capture
Datagrams	Yes	No	Yes	Moderate	Stateless protocol analysis

The pool mode is particularly noteworthy for production deployments: it uses memory pooling for packet buffers (up to 1500-byte MTU), recycling allocations to minimize garbage collection pauses during sustained high-throughput capture.

WebUI and Service Mode

The capture subsystem supports a service mode that serves an HTTP API backed by a Next.js frontend, providing a browser-based interface for configuring captures, viewing audit records, managing detection rules, and exploring results. The service mode supports hot-reload during development.

Security Design

Security is embedded throughout the architecture:

- **Memory-safe parsing**: Go's runtime prevents memory corruption from malformed packets.
- **Input sanitization**: User-provided inputs (rule names, decoder configuration names) are sanitized before use in filesystem operations, preventing path traversal attacks.
- **Build reproducibility**: DPI library versions, gopacket versions, and commit hashes are embedded in the binary via linker flags, enabling exact reproduction of any build.
- **Profiling instrumentation**: Built-in pprof support enables goroutine leak detection, memory profiling, and CPU profiling in production, ensuring operational visibility.

Use Cases and Scenarios

Security Researcher Training ML Models for Anomaly Detection

A university research group studying anomaly-based intrusion detection needs to train classification models on labeled network traffic. They process the CIC-IDS2018 dataset through Netcap, using the `capture` tool to generate Protocol Buffers audit records and the `label` tool to apply attack classification labels from a YAML configuration. The resulting labeled CSV dataset — with 66+ feature dimensions per record — feeds directly into their scikit-learn and TensorFlow pipelines. What previously required weeks of custom PCAP parsing and feature engineering now takes hours. The structured, high-dimensional output enables experiments with novel detection algorithms rather than data wrangling.

SOC Analyst Investigating a Suspected Breach

A security operations center analyst receives an alert about unusual outbound traffic from a database server. Using NETCAP Pro, they load the relevant PCAP file and immediately access 35+ analysis modules. The Connections module reveals unauthorized SSH sessions to an external IP. The Credentials module surfaces extracted NTLM hashes from SMB traffic, indicating lateral movement. The Graph Analysis engine visualizes the full attack chain — from initial HTTP exploitation to privilege escalation via credential theft to data exfiltration over DNS tunneling. The Timeline module establishes the precise sequence of events. Investigation notes document each finding for the incident report. The entire investigation, from PCAP to documented conclusions, takes place within a single application.

Healthcare Organization Monitoring Medical IoT Devices

A hospital network security team needs to monitor traffic from hundreds of medical devices — infusion pumps, patient monitors, imaging systems — many running embedded operating systems with proprietary protocols. They deploy Netcap agents on network taps at each clinical segment, with a central collector aggregating audit records. The memory-safe Go runtime ensures stability when processing traffic from devices that occasionally produce malformed packets. The Devices and Services modules in NETCAP Pro identify every device on the network, its communication patterns, and any unauthorized protocol usage. The DPI module detects protocol anomalies that port-based monitoring would miss.

Penetration Testing Team Capturing Credentials During an Engagement

A penetration testing team conducting an internal network assessment uses NETCAP Pro to passively capture and analyze network traffic. The credential harvesters extract FTP passwords, HTTP Basic Authentication tokens, SMTP authentication exchanges, and Telnet sessions in real time. Captured hashes are exported in Hashcat-compatible formats and forwarded to the cracking workstation. Integration with Wireshark allows deep-dive into specific sessions, while Metasploit integration correlates captured credentials with exploitable services. The Graph module maps the relationship between compromised credentials and accessible systems, revealing the most efficient path to domain compromise.

Honeypot Operator Analyzing Attacker Behavior

A threat intelligence team operates a distributed network of honeypots across multiple geographic regions. Netcap agents deployed on each honeypot stream audit records to regional collectors. The structured output — with protocol-specific fields for SSH brute-force attempts, HTTP exploitation payloads, and DNS beaconing patterns — feeds into an automated analysis pipeline that classifies attacker behavior, identifies toolkits, and generates indicators of compromise. The Prometheus metrics export drives real-time dashboards showing attack volume, protocol distribution, and geographic origin across the entire honeypot network.

Incident Response Firm Conducting Network Forensics

A digital forensics firm receives a 200GB PCAP file from a client's compromised network. Using NETCAP Pro, they process the capture through the concurrent multi-core engine, generating structured audit records across all protocol layers simultaneously. The Files module extracts transferred files for malware analysis. The Certificates module identifies suspicious TLS certificates used by command-and-control infrastructure. The Vulnerabilities module flags services with known weaknesses. The Software module catalogs every application and version observed on the network. The AI-powered analysis generates an initial threat assessment, which analysts refine using investigation notes. The complete forensic timeline is exported for inclusion in the legal report.

Pricing and Plans

Netcap follows a tiered model that provides the open-source core for free while offering professional and enterprise capabilities for security teams that need advanced analysis features.

PLAN	PRICE	BILLING	DESCRIPTION
Core	Free	Forever	Open-source CLI framework (GPLv3). 66+ audit record types, Protocol Buffers & CSV export, live capture, distributed collection, Prometheus metrics, Maltego integration.
Pro Monthly	€49/month	Monthly	Full-featured cross-platform desktop application. Everything in Core plus Graph Analysis Engine, AI-powered threat detection, Timeline Analysis, tool integrations (Wireshark, Metasploit, Hashcat, John), investigation notes, session export & loading, email support.
Pro Yearly	€490/year	Annual	Identical to Pro Monthly with annual billing (save €98/year).
Enterprise	Custom	Custom	Everything in Pro plus unlimited team seats, priority support with SLA guarantee, custom integrations, on-site training, dedicated account manager, custom feature development, security audit assistance, volume licensing, dedicated instance option, SSO/SAML.

The pricing philosophy reflects Amsterdam Technologies' commitment to the security research community: the core analytical engine remains free and open-source under GPLv3, ensuring that researchers, students, and open-source projects always have access to production-quality network analysis capabilities. The Pro tier packages professional workflow features — visual analysis, AI assistance, tool integration — into a desktop application priced accessibly for individual practitioners. The Enterprise tier provides the organizational support, scale, and customization that teams and enterprises require.

A **14-day free trial** of NETCAP Pro is available with full access to all features and no credit card required. Educational discounts of 50% are available for students and educators with valid `.edu` email addresses.

Frequently Asked Questions

Can I evaluate NETCAP Pro before purchasing? Yes. NETCAP Pro includes a 14-day free trial with full access to all features, requiring no credit card. Additionally, a live demo is available at try.netcap.io, providing browser-based access to the application with pre-loaded example data for immediate evaluation.

What platforms does Netcap support? Netcap Core compiles to native binaries for Linux, macOS, and Windows. NETCAP Pro is available as a native desktop application for all three platforms. Docker images are also available for containerized deployments of the core framework.

How does Netcap handle data privacy and sensitive network traffic? Netcap processes and stores all data locally — on the machine or infrastructure where it is deployed. There is no cloud telemetry, no data exfiltration, and no external dependencies during operation. Audit records remain under the operator's full control. For sensitive environments, the Enterprise tier supports fully on-premise deployment.

What output formats are supported for integration with existing tools? Netcap produces output in Protocol Buffers (compact binary, cross-language), CSV (data science tools, spreadsheets), JSON (log aggregation, Elasticsearch), and Prometheus metrics (monitoring dashboards). Protocol Buffers can be read natively from Python, Java, C++, JavaScript, and dozens of other languages.

Is Netcap suitable for production network monitoring at scale? Yes. The concurrent multi-core processing engine, configurable decode strategies (including memory-pooled mode for sustained high throughput), and distributed collection architecture with dedicated agents and collectors are designed for enterprise-scale deployments. The pool decode mode minimizes garbage collection pressure for sustained gigabit-speed capture.

Do you offer educational discounts? Yes. Amsterdam Technologies offers 50% off NETCAP Pro for students and educators. Contact support with a valid `.edu` email address for verification.

Why Amsterdam Technologies

Netcap reflects Amsterdam Technologies' engineering-first approach to cybersecurity tooling. Rather than building a marketing-driven product that wraps a thin feature set in dashboard aesthetics, Amsterdam Technologies invested in solving the foundational problem: transforming network traffic into structured, type-safe, machine-learning-ready data at the point of capture. The result is a framework that serves equally well as a research platform, a forensic analysis tool, and an operational security monitoring system.

The company is headquartered in Amsterdam, Netherlands, and maintains a product portfolio spanning cybersecurity, developer tools, and productivity applications. This breadth reflects a consistent engineering philosophy: build tools that solve real problems with technical rigor, make them accessible across platforms, and price them so that individuals and teams can adopt them without procurement friction.

Netcap's roadmap includes completion of the advanced credential harvesting capabilities (NTLMSSP, Kerberos AS-REQ/AS-REP/TGS-REP), eBPF-based kernel-accelerated packet capture for 10Gbps+ throughput on Linux, expanded AI-powered detection capabilities in NETCAP Pro, and continued growth of the protocol decoder library. The framework's open-source core ensures that the broader security research community contributes to and benefits from these advances.

- **Website:** https://netcap.io
- **Contact:** support@amsterdam-technologies.com
- **Company:** amsterdam-technologies.com

Amsterdam Technologies B.V.
Overtoom 456-1, 1054JW Amsterdam
The Netherlands

KVK: 98875728
VAT: 653976148B01

support@amsterdam-technologies.com
https://amsterdam-technologies.com