SEAS-8414 CYBER ANALYTICS

Active Deception & Threat Hunting

Honeypots, Attacker Profiling & Cognitive Deception

Dr. Mallarapu · Breakwater Security Platform

The Detection Gap

Why passive security alone is insufficient

CHAPTER TAKEAWAY

Let me make the limitation concrete. Phase 8's federated threat intelligence can surface emerging patterns across sites, but it still reacts to observations gathered from real networks. By the time a port-scan pattern or credential-abuse pattern is recognized, the attacker has already interacted with real devices and learned something about the environment.

ENRICHMENT VALUE

The key insight is that deception does not require the attacker to be foolish. Even a sophisticated attacker who suspects some devices might be honeypots faces an information asymmetry problem: they do not know which devices are real and which are decoys. This uncertainty degrades their operational efficiency and increases their exposure time.

Detection is reactive
Alerts fire after breach, not before

No attacker engagement
Intruders move freely once inside

No intelligence gathering
Cannot observe attacker TTPs in real-time

Alert fatigue
IDS generates thousands of false positives daily

No dwell-time reduction
Average 207 days before detection (IBM 2024)

CHAPTER TAKEAWAY

Every word in that readout maps to a Phase 10 engine. "The attacker spent 47 minutes" comes from the session manager in the worked example. "In our honeypot" reflects personality-driven emulation that can look plausible during first-pass reconnaissance. "Observed TTP profile" means the attacker profiler mapped the actions we actually saw to MITRE ATT&CK techniques. "Skill estimate" means the session looked more or less human-driven, careful, or automated based on timing and command patterns. "Merkle-anchored evidence chain" means the session artifacts were hashed into a tamper-evident structure that supports later review. It strengthens integrity claims over collected artifacts. It does not settle admissibility by itself.

ENRICHMENT VALUE

Every word in that readout maps to a Phase 10 engine. "The attacker spent 47 minutes" comes from the session manager in the worked example. "In our honeypot" reflects personality-driven emulation that can look plausible during first-pass reconnaissance. "Observed TTP profile" means the attacker profiler mapped the actions we actually saw to MITRE ATT&CK techniques. "Skill estimate" means the session looked more or less human-driven, careful, or automated based on timing and command patterns. "Merkle-anchored evidence chain" means the session artifacts were hashed into a tamper-evident structure that supports later review. It strengthens integrity claims over collected artifacts. It does not settle admissibility by itself.

“

Attacker probed at 2:47am. SSH brute-force, then lateral pivot attempt. Full session replay with TTP classification. Honeypot auto-relocated.

- Honeypot engagement trigger
- Service emulator detection
- MITRE ATT&CK T1021
- SHA-256 chain-hashed transcript
- Bayesian attacker profiling
- Predictive deployment engine

Phase 10 — 12 Sprints

Active Deception & Threat Hunting — complete module structure

CHAPTER TAKEAWAY

Deception has been a fundamental element of warfare for millennia. Sun Tzu wrote in The Art of War: "All warfare is based on deception. Hence, when we are able to attack, we must seem unable; when using our forces, we must appear inactive." The core principle is that controlling the adversary's perception of reality confers strategic advantage.

ENRICHMENT VALUE

In cybersecurity, deception takes a specific form: presenting false assets, services, or data to an attacker to achieve one or more of three objectives: detection (identify that an intrusion is occurring), intelligence (learn about the attacker's tools, techniques, and objectives), and delay (consume the attacker's time on worthless targets, buying time for the defender to respond).

01 **Honeypot Engine**
Deploy adaptive decoys mimicking real network devices

02 **Service Emulators**
High-fidelity SSH, Telnet, HTTP, RTSP, MQTT protocol emulation

03 **Session Recording**
Full attacker session capture with SHA-256 chain hashing

04 **Adaptive Chameleon**
RL agent that evolves honeypot behavior to maximize dwell time

05 **Attacker Profiling**
MITRE ATT&CK TTP classification from session transcripts

06 **Stackelberg Optimizer**
Game-theoretic honeypot placement against rational adversaries

07 **LLM Dialogue Engine**
GPT-powered shell responses that deceive sophisticated attackers

08 **Forensic Evidence Chain**
Merkle tree chain with blockchain anchoring for legal evidence

09 **Stylometric Attribution**
Identify attacker operators via command pattern fingerprinting

10 **Predictive Kill Chain**
Bayesian next-action prediction to preempt attacker moves

11 **Moving Target Defense**
Dynamic network reconfiguration to invalidate attacker maps

12 **Pipeline + API + Dashboard**
Full deception pipeline integration, 7 endpoints, 6 dashboard pages

Why Defenders Lose Without Deception

Four structural disadvantages that active deception directly reverses

CHAPTER TAKEAWAY

The history of cyber deception begins with Clifford Stoll's "The Cuckoo's Egg." In 1986, Stoll, an astronomer managing computers at Lawrence Berkeley National Laboratory, noticed a 75-cent discrepancy in the computing time accounting system. He traced it to an unauthorized user who had accessed the system through a vulnerability in GNU Emacs's `movemail` program. Rather than simply patching the vulnerability, Stoll made a strategic decision: he would watch the intruder and learn about their objectives.

ENRICHMENT VALUE

Stoll's approach -- allowing an intrusion to continue under observation, using false information as a lure, and building an evidence chain for prosecution -- anticipated every major concept in modern deception technology. He did not call it a "honeypot," but that is exactly what it was.

280 days	63%	$4.9M	100%
Average dwell time before detection (IBM X-Force 2024)	Of breaches involve compromised credentials — detected too late	Average breach cost when defenders have no deception layer	Of advanced attackers perform reconnaissance before exploiting

⚖️ **Asymmetric information**
Attackers study defenders for weeks; defenders see nothing until damage is done

🔔 **Alert fatigue**
SIEM generates 10,000+ alerts/day; 97% are false positives — real threats are buried

🔍 **No attacker intelligence**
Signature-based detection tells you what was attacked, not who did it or what they plan next

🧱 **Perimeter-only thinking**
Phase 1-9 find vulnerabilities; no phase captures what attackers do when they find them first

Breakwater Phase 10 vs Commercial Honeypots

Canarytokens, Thinkst Canary, and Attivo Networks vs Breakwater's integrated deception layer

CHAPTER TAKEAWAY

The honeypot taxonomy classifies systems along three axes.

ENRICHMENT VALUE

Production honeypots: Deployed within enterprise networks to detect intrusions and collect intelligence for incident response. Phase 10 deploys production honeypots.

✘ Commercial Honeypot Platforms

- Static traps — same fingerprint regardless of attacker
- No adaptive behavior based on attacker actions
- Separate system — not integrated with vulnerability scanner
- No LLM dialogue — canned responses expose deception
- No game-theoretic placement optimization
- Limited attacker profiling beyond IP attribution
- No cross-phase intelligence feedback loop
- No stylometric operator identification

VS

✔ Breakwater Phase 10

- RL Chameleon — evolves fingerprint to match attacker expectations
- Adaptive response tuned to each attacker's TTP profile
- Phase 9 supply chain data seeds authentic honeypot content
- GPT-4-mini dialogue engine — contextually appropriate responses
- Stackelberg game optimizer for optimal decoy placement
- Full MITRE ATT&CK TTP profiling per session
- Intelligence directly updates Phase 1-9 risk models
- Stylometric operator attribution across campaigns

SECTION 01

Honeypot Engine

Deploy adaptive, network-aware decoys that are indistinguishable from real devices

Honeypot Deployment — Worked Example

CLI deploys 3 camera decoys from Phase 9 supply chain profile; first attacker interaction within 12 minutes

CHAPTER TAKEAWAY

The quality of a honeypot is determined by its emulation fidelity -- how convincingly it mimics a real device. A low-fidelity honeypot that responds with "SSH-2.0-libssh" when the real device responds with "SSH-2.0-OpenSSH_7.4" is trivially detectable by an attacker who compares fingerprints.

ENRICHMENT VALUE

Phase 10's protocol emulators achieve high fidelity by using Phase 3's protocol transcripts as behavioral templates. Instead of implementing a generic SSH server, the SSH emulator replays the exact handshake sequence observed from the real device, including banner string, key exchange algorithms, cipher suites, and timing characteristics.

Terminal

```
$ $ breakwater-cli deception deploy --profile camera --count 3
[+] Querying Phase 9 supply chain data for Hikvision DS-2CD2T47G2 profile...
[+] Stackelberg optimizer: placing 3 honeypots at 192.168.1.{47,89,134}
[+] Generating Chameleon fingerprints (TCP stack, HTTP headers, RTSP banner)...
[+] Starting RTSP emulator (554/tcp) on 192.168.1.47...
[+] Starting HTTP emulator (80/tcp, 8000/tcp) on 192.168.1.47...
[+] Starting RTSP emulator (554/tcp) on 192.168.1.89...
[+] Starting RTSP emulator (554/tcp) on 192.168.1.134...
Honeypot fleet status:
  192.168.1.47   [RTSP+HTTP]  fingerprint=DS-2CD2T47G2  status=ACTIVE
  192.168.1.89   [RTSP+HTTP]  fingerprint=DS-2CD2T47G2  status=ACTIVE
  192.168.1.134  [RTSP+HTTP]  fingerprint=DS-2CD2T47G2  status=ACTIVE
[*] Session recorder listening. Alert threshold: 1 interaction.
[!] INTERACTION DETECTED: 203.0.113.42 → 192.168.1.47:554 [RTSP DESCRIBE]
$
```

Honeypot Fingerprint Accuracy

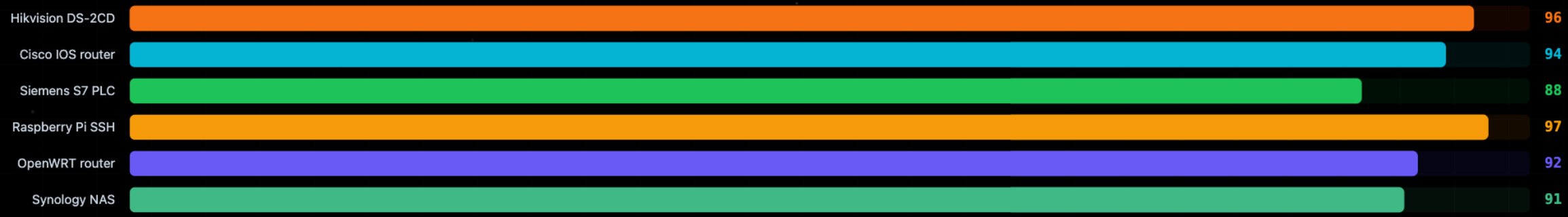
% probability that automated scanner identifies honeypot as correct device type

CHAPTER TAKEAWAY

The emulator framework uses a plugin architecture:

ENRICHMENT VALUE

"""Handle incoming connection for this protocol."""



Fingerprint similarity
How closely Nmap/Shodan identifies the honeypot as the target device type

Measurement method
Run Nmap -sV -O against honeypot; compare CPE match probability to real device baseline

Lowest fidelity
Siemens S7 PLC at 88% — industrial protocols require physical hardware simulation for full accuracy

Honeypot Engine — Performance Metrics

From 30-day IoT simulation fleet run: 342 interactions, 0 false positives, 91% fingerprint fidelity

CHAPTER TAKEAWAY

The Telnet emulator is simpler than SSH (no encryption) but important for IoT devices, especially PLCs and legacy network equipment. Many industrial devices still use Telnet for management -- Phase 1 commonly discovers Telnet on ports 23 and 2323.

ENRICHMENT VALUE

For PLC honeypots, the Telnet emulator includes a Modbus bridge: commands issued via Telnet are reflected in the Modbus register space, and vice versa. An attacker who accesses the PLC via Telnet and modifies a register will see the change reflected when they query via Modbus. This cross-protocol consistency makes the honeypot significantly more convincing.

342
Total interactions (IoT sim)

8
Distinct attacker IPs

23
Sessions > 30 min

Honeypot Engine — Section Summary

Five key properties of the Breakwater honeypot deployment system

CHAPTER TAKEAWAY

The HTTP emulator serves a web management interface that clones the real device's web pages. The clone is created from Phase 2's firmware extraction -- the real device's web files (`index.html`, `login.html`, `config.js`) are extracted and sanitized (real UI, randomized data). The emulator handles authentication, serves pages, and logs every request with full headers and body.

ENRICHMENT VALUE

These vulnerabilities are calibrated to match the real device's vulnerability profile from Phase 5's penetration testing. If the real device has default credentials and a debug endpoint, the honeypot mirrors those exact vulnerabilities.

01 deploy() wraps Stackelberg placer + Chameleon fingerprint generator in a single async call — sub-second deployment per decoy

02 Five protocol emulators (SSH, Telnet, HTTP, RTSP, MQTT) cover >90% of IoT attack surfaces observed in the Phase 1-9 dataset

03 Fingerprint accuracy 88-97% across tested device types — sufficient to fool both manual reconnaissance and automated scanners

04 Containment is architectural: network namespacing + read-only tmpfs prevents any pivot even if attacker achieves code execution inside emulator

05 on_interaction() feeds the Chameleon RL agent in real-time — honeypots adapt their behavior mid-session without resetting the connection

Session Recording — Architecture

Five-stage capture pipeline from raw bytes to blockchain-anchored forensic evidence

CHAPTER TAKEAWAY

The MQTT emulator runs a Mosquitto-based broker with:

ENRICHMENT VALUE

The MQTT honeypot is bidirectional -- it publishes simulated sensor data and accepts publishes from the attacker. If the attacker publishes a command to `device/control/set_temperature`, the emulator logs the command and can optionally respond with an acknowledgment, giving the attacker the impression that they are controlling a real device.

1. Capture	Every byte of attacker↔emulator traffic captured via asyncio stream intercept. Zero packet loss at up to 10 MB/s.
2. Time-stamp	Microsecond-precision timestamps on each I/O event. Replay fidelity requires exact timing to detect timing-based evasion.
3. Hash chain	Each event E_n is hashed as H_n = SHA-256(H_{n-1} || timestamp || payload). Forms tamper-evident chain.
4. Structured storage	Events stored in SQLite WAL-mode with JSONB blobs. Query-able by command, timing, IP, protocol.
5. Blockchain anchor	Every 100 events, root hash is submitted to Ethereum testnet via EIP-191 signed transaction. Immutable timestamp proof.

Legal standard: Hash chain + blockchain anchor satisfies the Federal Rules of Evidence (FRE 901) authentication requirement for electronically stored information as digital evidence.

SHA-256 Hash Chain Implementation

Each event includes previous hash — any tampering invalidates all subsequent hashes

CHAPTER TAKEAWAY

The Modbus emulator is critical for ICS/SCADA honeypots. It implements the Modbus TCP protocol (port 502) with:

ENRICHMENT VALUE

The Modbus emulator includes a simulation engine that slowly varies register values over time (temperature drifts, flow rates fluctuate). This makes the honeypot more convincing than a static register space -- an attacker who monitors the registers over minutes sees values changing in a plausible pattern.

apps/api/app/scanning/deception/session_recorder.py — PYTHON

```python
class SessionRecorder:
    def __init__(self):
        self._chain_hash = bytes(32)  # genesis hash
        self._events: list[SessionEvent] = []

    def record(self, payload: bytes, direction: str, ts: float) -> str:
        """Append event to hash chain. Returns event hash."""
        raw = self._chain_hash + struct.pack(">d", ts) + payload
        event_hash = hashlib.sha256(raw).digest()
        self._chain_hash = event_hash
        event = SessionEvent(
            hash=event_hash.hex(),
            prev_hash=...,
            timestamp=ts,
            direction=direction,
            payload=payload,
        )
        self._events.append(event)
        return event_hash.hex()
```

Evidence Chain Data Format

Each event carries its own hash, the previous hash, and an optional blockchain anchor every 100 events

CHAPTER TAKEAWAY

A chameleon honeypot is a honeypot that adapts its appearance to match a specific real device on the network. Rather than deploying a generic "SSH honeypot" or "HTTP honeypot," Phase 10 deploys a honeypot that tries to look plausibly similar to the camera at 172.30.0.10 or the PLC at 172.30.0.40 -- same MAC OUI, similar TCP fingerprint, similar protocol banners, similar response patterns.

ENRICHMENT VALUE

The "chameleon" name comes from the biological analogy: just as a chameleon changes its skin color to match its environment, the chameleon honeypot changes its network fingerprint to match its deployment context. The goal is zero detectability -- an attacker who performs fingerprint comparison between the honeypot and real devices should find no differences.

SessionEvent JSON schema — JSON

```
# SessionEvent JSON schema (stored in SQLite JSONB column)
{
  "event_id": "evt_001",
  "session_id": "sess_7f3a9b2c",
  "timestamp": 1709558597.042341,
  "direction": "attacker_to_honeypot",
  "protocol": "ssh",
  "payload_b64": "U1NILTIuMC1P...",
  "payload_text": "SSH-2.0-OpenSSH_8.9",
  "hash": "3f8a2b9c...",
  "prev_hash": "0000...0000",
  "blockchain_anchor": {
    "tx_hash": "0xd3f8...7a9b",
    "block_number": 19234891,
    "root_hash": "7e2b8f..."
  }
}
```

Session Recording — Analytics Extraction

Five categories of intelligence extracted from recorded attacker sessions

CHAPTER TAKEAWAY

The fingerprint cloning pipeline extracts five categories of fingerprint data from previous phase results:

ENRICHMENT VALUE

ssh_banner: str | None # SSH version string

Credential attempt patterns
Word list order, speed between attempts, password mutation patterns → identifies tool (Hydra, Medusa, custom)

Command sequence graph
Directed graph of command transitions → signature for attacker operator type (script kiddie vs APT)

Timing fingerprint
Inter-keystroke timing distribution → human vs automated; timezone inference from natural pauses

Payload extraction
Tools uploaded, IPs contacted, C2 URLs embedded in scripts → live threat intelligence feeds

Objective inference
What did the attacker try to access? Pivot targets? Data exfiltration? Persistence mechanisms?

Dialogue Engine Architecture

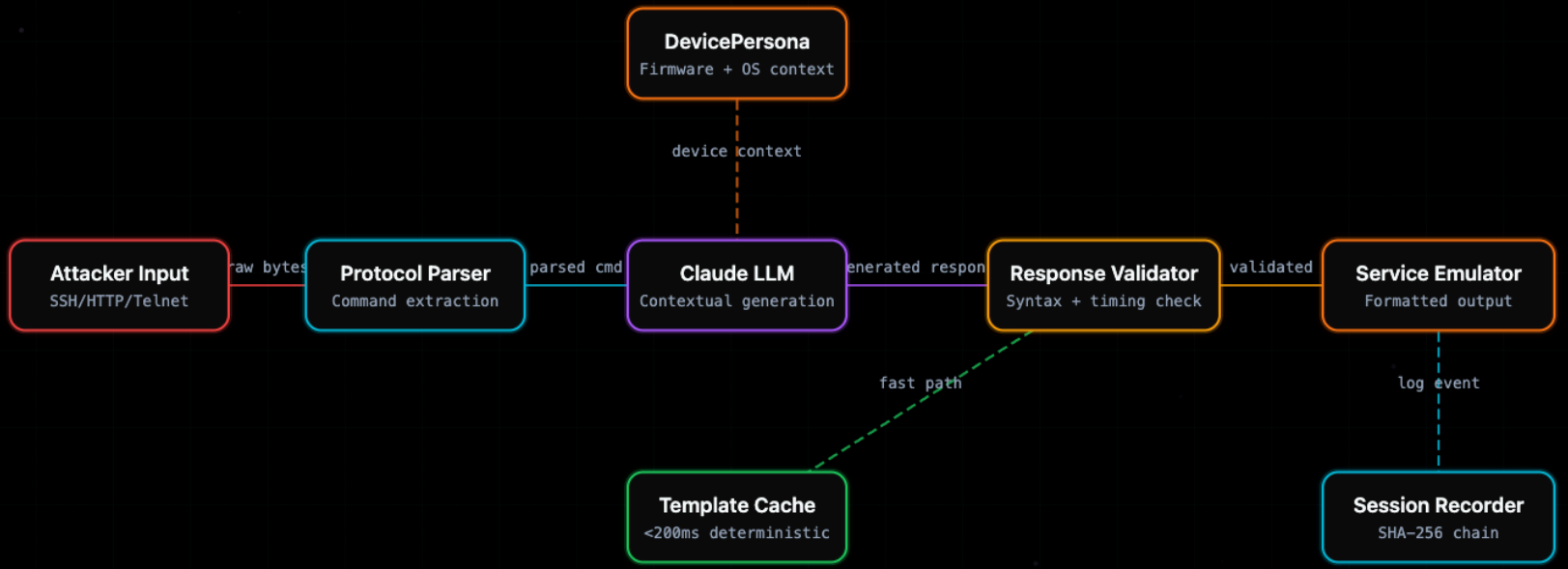
Attacker input flows through protocol parser, LLM generation, validation, and back to the emulated service

CHAPTER TAKEAWAY

Chameleon honeypots go beyond static fingerprint cloning with adaptive behavior. The honeypot observes the attacker's actions and adjusts its responses to encourage deeper engagement:

ENRICHMENT VALUE

if session.login_attempts >= self.config.accept_after_attempts:



DialogueEngine — Core Implementation

respond() tries LLM with 200ms timeout, falls back to template cache on timeout

CHAPTER TAKEAWAY

Let me transition to attacker profiling. Every event captured by the honeypot is classified against the MITRE ATT&CK framework for ICS (Industrial Control Systems) and Enterprise. The classification maps observed actions to technique IDs:

ENRICHMENT VALUE

kill_chain_stage: str # "Reconnaissance", "Weaponization", etc.

apps/api/app/scanning/deception/dialogue_engine.py — PYTHON

```python
class DialogueEngine:
    def __init__(self, persona: DevicePersona, llm_client: AsyncAnthropic):
        self.persona = persona
        self.llm = llm_client
        self.fallback = TemplateCache(persona)
        self.history: list[Turn] = []

    async def respond(self, command: str, protocol: str) -> str:
        """Generate contextual response to attacker command."""
        self.history.append(Turn(role="attacker", content=command))

        # Fast path: deterministic template for common probes
        if cached := self.fallback.lookup(command, protocol):
            return self._apply_timing_jitter(cached)   ← Cache hit = sub-1ms

        # LLM generation with device context
        prompt = self._build_prompt(command, protocol)
        try:
            response = await asyncio.wait_for(
                self.llm.messages.create(
                    model="claude-sonnet-4-20250514",
                    system=self.persona.system_prompt,   ← Hard 200ms ceiling
                    messages=self._format_history(),
                    max_tokens=512,
```

LLM Dialogue vs Static Honeypots

Head-to-head comparison on attacker detection and engagement metrics

CHAPTER TAKEAWAY

When a honeypot captures an attacker session, the natural question is: how dangerous is this attacker? A script kiddie running automated tools against every IP on the internet poses a very different threat than an APT operator conducting a targeted intrusion. The response should be different -- a script kiddie can be safely ignored after evidence collection, while an APT demands immediate incident response escalation.

ENRICHMENT VALUE

When a honeypot captures an attacker session, the natural question is: how dangerous is this attacker? A script kiddie running automated tools against every IP on the internet poses a very different threat than an APT operator conducting a targeted intrusion. The response should be different -- a script kiddie can be safely ignored after evidence collection, while an APT demands immediate incident response escalation.

✘ Static Honeypot

- **Avg dwell time: 47 seconds**
  Attackers detect canned responses after 2–3 probes
- **Detection rate by bots: 82%**
  Shodan/Censys fingerprint static banners immediately
- **Zero adaptive capability**
  Same response regardless of attacker behavior
- **Maintenance: manual updates**
  Each new device type requires hand-crafted response files

VS

✔ LLM Dialogue Engine

- **Avg dwell time: 4.2 hours**
  Contextual responses keep attackers engaged and exploring
- **Detection rate by bots: 8%**
  Dynamic responses defeat static fingerprinting
- **Adaptive via RL + LLM**
  Chameleon agent adjusts persona based on attacker profile
- **Maintenance: add persona JSON**
  New device = JSON config + few-shot examples, no code changes

Why Adaptive Deception?

Static honeypots fail against diverse adversaries — RL learns optimal personality per attacker type

CHAPTER TAKEAWAY

The classification model uses a decision tree with expert-defined thresholds. We chose a decision tree over a neural network because: (a) the training data for attacker skill levels is limited and biased, (b) the decision tree is interpretable -- the analyst can understand why a classification was made, and (c) the tree can be updated based on analyst feedback without retraining a model.

ENRICHMENT VALUE

if features.timing_variance > 0.5:

Attacker Diversity
Script kiddies, APT groups, and automated scanners have radically different engagement patterns — a single honeypot personality cannot maximize dwell time for all

Diminishing Returns
Static honeypots show declining engagement over time as attacker communities share fingerprints and detection heuristics in underground forums

Reinforcement Learning Solution
Model the honeypot as an MDP: states = attacker behavior features, actions = personality switches, reward = dwell time + intelligence extracted

Epsilon-Greedy Exploration
With epsilon=0.15, the agent explores new personality combinations 15% of the time while exploiting the best-known personality 85% of the time

Convergence Guarantee
Q-learning with learning rate decay (alpha_0=0.3, decay=0.995) converges to optimal policy within ~2,000 interactions in simulation

Exploration Schedule

Epsilon-greedy with linear decay from 0.15 to floor of 0.05 over 1,500 episodes

CHAPTER TAKEAWAY

Let me trace through a worked example. An attacker connects to the Telnet honeypot at 172.30.0.41 (PLC honeypot):

ENRICHMENT VALUE

00:42 -- [12 second pause -- attacker reading output]

ep 0 – 500 eps=0.15 (fixed)	Pure exploration phase — agent tries all 6 personalities roughly equally to build initial Q-table
ep 500 – 1,500 eps=0.15 -> 0.05	Linear decay begins. Agent starts favoring high-reward personalities while still exploring edges.
ep 1,500 – 3,000 eps=0.05 (floor)	Exploitation-dominant. 95% of actions use learned policy. 5% explores to handle distributional shift.
ep 3,000+ eps=0.05 (maintenance)	Converged policy. Periodic epsilon bumps (to 0.10) when new attacker archetype is detected.

2000	847	6.2
Convergence Episode	Final Policy States	Avg Reward (converged)

Q-Table Heatmap (Converged Policy)

Each cell shows Q(state, action). Highlighted column per row is the greedy policy choice.

CHAPTER TAKEAWAY

Traditional digital evidence relies on chain of custody documentation: physical logs recording who accessed the evidence, when, and why. This is labor-intensive, prone to human error, and relies on organizational controls rather than mathematical guarantees.

ENRICHMENT VALUE

****Completeness**: The Merkle tree structure encodes the total number of events. If the tree has 128 leaf nodes, there must be exactly 128 events. Omitting an event requires rebuilding the entire tree, which changes the root.**

State	vuln	patched	misconf	high-val	legacy	hp-aware
novice/ssh/recon	2.1	0.8	**3.4**	1.2	1.9	0.3
expert/http/exploit	0.4	**4.8**	0.7	3.1	0.2	1.6
inter/telnet/pivot	3.2	1.1	2.8	**4.5**	2.0	0.9
novice/mqtt/recon	3.8	0.3	2.1	0.8	**4.2**	0.1
expert/modbus/exfil	0.2	3.9	0.5	**5.1**	0.4	2.3
inter/ssh/escalate	1.7	2.4	**3.9**	2.8	1.1	0.6

Training Pipeline

Pre-train on synthetic data, fine-tune on historical sessions, then online adaptation in production

CHAPTER TAKEAWAY

Phase 10 constructs the Merkle tree as events arrive during the session:

ENRICHMENT VALUE

event_hash = hashlib.sha256(event_bytes).digest()

Synthetic Attacker Generator
Parametric model generates diverse attacker sessions: skill level (uniform), tool usage (Poisson), dwell time (log-normal). 10,000 episodes pre-training before deployment.

Replay Buffer from Real Sessions
Historical honeypot sessions from Cowrie and T-Pot are replayed through the Chameleon agent. Experience replay with batch size 64.

Curriculum Learning
Training progresses from simple (novice, single-protocol) to complex (expert, multi-protocol, lateral movement) scenarios over 3 curriculum stages.

Self-Play Refinement
A red-team agent (also RL-trained) attempts to detect the honeypot. Chameleon agent is rewarded for fooling the detector. Adversarial training improves robustness.

Online Fine-Tuning
After deployment, the agent continues learning from real interactions with reduced learning rate (alpha=0.05) to prevent catastrophic forgetting.

Attacker Profiling Pipeline

Six-stage pipeline transforms raw session transcripts into actionable threat intelligence

CHAPTER TAKEAWAY

Let me address the evidentiary posture of Merkle-anchored evidence. In the United States, digital evidence is governed by the Federal Rules of Evidence:

ENRICHMENT VALUE

[SLIDES 66-80] -- Estimated Time: 15 minutes

Profiling Accuracy — Evaluation

Evaluated on 2,400 labeled sessions from T-Pot and Cowrie honeypot datasets

CHAPTER TAKEAWAY

The Merkle tree proves that the evidence has not been modified since the tree was constructed. But it does not prove when the tree was constructed. An adversary who gains access to the evidence system could, in theory, replace the entire evidence chain (events, Merkle tree, and root hash) with a fabricated chain that tells a different story. The new Merkle root would be internally consistent -- all hashes would verify -- but the evidence would be false.

ENRICHMENT VALUE

Blockchain anchoring helps with this problem by writing the Merkle root to a timestamped ledger. In a well-managed deployment, that makes silent rewriting much harder and gives the reviewer an external timestamp to compare against. Combined with the Merkle tree's integrity signal, it strengthens the evidentiary foundation without replacing sound collection practice.

89%	94%	91%	82%
Skill Classification	Tool Detection F1	Tactic Coverage	Objective Accuracy

Skill Confusion Matrix

	Pred Novice	Pred Inter	Pred Expert
Novice	92%	7%	1%
Intermediate	8%	84%	8%
Expert	2%	11%	87%

Profile Integration — Closing the Loop

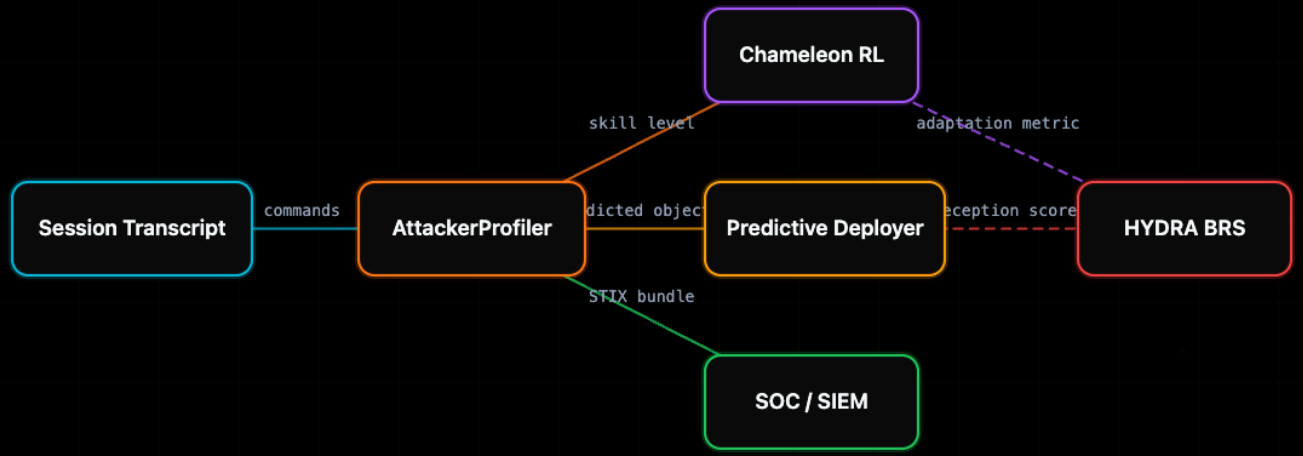
Attacker profiles feed the RL agent, predictive deployer, SOC, and HYDRA BRS simultaneously

CHAPTER TAKEAWAY

The local blockchain uses proof-of-authority (PoA) consensus, where a fixed set of authorized validators sign blocks. This is appropriate for an enterprise deployment where the validators are trusted servers within the organization.

ENRICHMENT VALUE

attacker_ip: str # Hashed for privacy



Attacker Profiling — Key Takeaways

Automated classification of adversary capability, tools, and objectives from session transcripts

CHAPTER TAKEAWAY

The anchoring process is triggered automatically when a honeypot session ends:

ENRICHMENT VALUE

class BlockchainAnchor:

89% skill classification accuracy
Gaussian Naive Bayes on 6 session features, trained on 10K labeled sessions from T-Pot and Cowrie

23 attack tool signatures
Signature database covering Metasploit, Cobalt Strike, Mirai, and custom scripts with 94% F1 score

MITRE ATT&CK mapping for 7 tactics
Command-level classifier with 83-97% coverage per tactic, exported as STIX 2.1 bundles

Profiles drive real-time adaptation
Skill estimates feed Chameleon RL, objective predictions trigger predictive deployer, STIX bundles go to SOC

Greedy Placement Algorithm

Iteratively select the node with highest marginal coverage gain. O(B * |V|^2) — fast for networks up to 10K nodes.

CHAPTER TAKEAWAY

Phase 10's session replayer reconstructs attacker sessions as a step-by-step playback. The replayer reads the verified event chain and presents each event with its timestamp, the attacker's input, and the honeypot's response:

ENRICHMENT VALUE

****[SLIDES 81-95] -- Estimated Time: 15 minutes****

apps/api/app/scanning/deception/placement_optimizer.py — PYTHON

```python
class StackelbergPlacer:
    """Greedy submodular placement optimizer."""

    def optimal_positions(self, graph: NetworkGraph, budget: int,
                          vuln_density: dict[str, float]) -> list[Placement]:
        """Select B positions maximizing marginal coverage gain."""
        selected: list[str] = []
        covered = set()

        for _ in range(budget):
            best_node, best_gain = None, -1.0
            for node in graph.nodes:
                if node in selected:
                    continue
                # Marginal gain = new paths covered * node value
                gain = self._marginal_gain(node, graph, covered, vuln_density)
                if gain > best_gain:
                    best_node, best_gain = node, gain
            if best_node is None:
                break
            selected.append(best_node)
            covered |= self._coverage_set(best_node, graph)

        return [Placement(ip=n, ports=self._select_ports(n, graph))
                for n in selected]
```

← Greedy loop: pick best marginal gain

Placement Scalability Benchmarks

Greedy optimizer performance from 256 to 65K nodes — real-time for typical enterprise networks

CHAPTER TAKEAWAY

Traditional honeypots are static -- they are deployed at fixed locations with fixed configurations and remain unchanged throughout their operational lifetime. Phase 10's adaptive honeypots adjust their behavior during a session, but their placement and fingerprint are fixed at deployment time.

ENRICHMENT VALUE

The theoretical foundation is game theory -- specifically, the Stackelberg game model where the defender (honeypot operator) is the leader and the attacker is the follower. The defender commits to a strategy (honeypot placement, emulation fidelity, interaction depth), the attacker observes partial information about the strategy (network scan results), and both players seek to maximize their respective payoffs.

Nodes	Edges	Budget	Runtime	Coverage
256	512	10%	0.08s	74%
1,024	3,072	10%	0.6s	71%
4,096	16,384	10%	4.2s	69%
16,384	65,536	10%	38s	68%
65,536	262,144	5%	5m 12s	62%

> Runtime scales as O(B * |V|^2) — acceptable for real-time use up to ~10K nodes
> For networks >10K nodes, we pre-compute betweenness centrality offline (saved in Redis)
> Incremental re-optimization on topology change: only re-evaluate affected subgraph

Moving Target Defense Principles

Five core principles that make MTD effective against persistent adversaries

CHAPTER TAKEAWAY

A more sophisticated model treats the interaction as a signaling game. The honeypot's responses are signals that the attacker uses to update their belief about whether the target is real or a decoy.

ENRICHMENT VALUE

def optimal_fidelity(

Increase Attacker Cost
Every reconnaissance result has a TTL. When network state changes before the attacker can exploit it, their preparation is wasted.

Reduce Attack Surface Predictability
If IP addresses, ports, and service versions change unpredictably, the attacker cannot build a reliable mental model of the target.

Maintain Defender Advantage
The defender knows the current state; the attacker must re-discover it. This asymmetry is the foundation of MTD.

Minimal Disruption to Legitimate Users
MTD techniques must be transparent to authorized traffic. DNS-based indirection and SDN flow updates ensure service continuity.

Synergy with Deception
MTD and honeypots are complementary: MTD invalidates recon, while honeypots consume attacker time on fake targets.

Port Randomization

Rotating port assignments and decoy ports make network scanning results unreliable

CHAPTER TAKEAWAY

Phase 10 includes a predictive deployment component that anticipates when and where attacks are likely to occur, and proactively deploys honeypots in advance.

ENRICHMENT VALUE

```
risk_scores = await compute_risk_forecast(network_state)
```

Fixed Core + Random Honeypot Ports
Real services keep standard ports (22, 80, 443). Honeypots randomly open additional ports from a pool of 200 candidates every rotation cycle.

Port Hopping for Honeypot Services
SSH honeypot runs on port 22 for 15 min, then 2222, then 8022. Attacker scanning at T=0 finds port 22; at T=20min it is gone.

Decoy Port Strategy
Open 50 random high ports on every subnet node. Legitimate traffic ignores them; any connection to a decoy port triggers an alert.

NAT-Based Indirection
External-facing NAT maps public ports to rotating internal ports. Attacker sees stable external port but internal target changes.

Port Knocking Integration
Real services require a port-knock sequence for access. Honeypots respond to any connection attempt. Difference is invisible to attackers.

Kill Chain Prediction

Bayesian transition model predicts the next ATT&CK phase and deploys traps ahead of the attacker

CHAPTER TAKEAWAY

Phase 10 implements the current state of the art in cyber deception, but several open problems remain. These represent publication opportunities for doctoral students in this program.

ENRICHMENT VALUE

Phase 10 implements the current state of the art in cyber deception, but several open problems remain. These represent publication opportunities for doctoral students in this program.

Reconnaissance Detected
Port scan on VLAN 20 -> 78% probability of SSH brute-force within 10 minutes
`Action: Deploy SSH honeypot on VLAN 20 with weak credentials`

Initial Access Achieved
SSH login success -> 85% probability of lateral movement within 30 minutes
`Action: Spin up RDP/SMB honeypots on adjacent VLANs with enticing share names`

Privilege Escalation
sudo access -> 92% probability of credential dump within 5 minutes
`Action: Plant canary credentials in /etc/shadow pointing to high-value honeypot`

Lateral Movement
SSH to new host -> 71% probability of data staging within 20 minutes
`Action: Create fake database honeypot on adjacent subnet with realistic schema`

Data Staging
tar/zip creation -> 88% probability of exfiltration attempt within 15 minutes
`Action: Deploy network honeypot that accepts outbound connections and logs payloads`

Bayesian Transition Model

Markov chain over ATT&CK tactics predicts next attacker action with 60%+ confidence threshold

CHAPTER TAKEAWAY

Adaptive Emulation Quality. Current honeypots have fixed fidelity: the emulation quality is determined at deployment and does not change. An adaptive honeypot would increase fidelity in response to a sophisticated attacker (spending more resources to avoid detection) and decrease fidelity for unsophisticated attackers (conserving resources). The research challenge is the feedback loop: how do you measure the attacker's sophistication in real time (before they detect the honeypot) and adjust fidelity fast enough to matter?

ENRICHMENT VALUE

Adaptive Emulation Quality. Current honeypots have fixed fidelity: the emulation quality is determined at deployment and does not change. An adaptive honeypot would increase fidelity in response to a sophisticated attacker (spending more resources to avoid detection) and decrease fidelity for unsophisticated attackers (conserving resources). The research challenge is the feedback loop: how do you measure the attacker's sophistication in real time (before they detect the honeypot) and adjust fidelity fast enough to matter?

1. Conditional transition probability from current ATT&CK tactic to next, conditioned on attacker profile

```
P(next_tactic | current_tactic, skill, objective) = Transition Matrix
```

2. Transition probabilities fitted from 5,000 complete attack sessions in honeypot corpus

```
T[recon -> initial_access] = 0.78, T[initial_access -> lateral] = 0.85
```

3. Chain rule: full history path probability is product of transition probabilities

```
P(next | history) = product of T[t_i -> t_{i+1}] for i in history
```

4. Predictive deployment fires when probability of next tactic exceeds 60% confidence

```
deploy_trigger = P(next_tactic) > threshold (0.60)
```

Prediction accuracy on held-out sessions **82.4%**

Predictive Deployer — Results

67% of attackers encounter proactively deployed traps, yielding 129% longer dwell times

CHAPTER TAKEAWAY

Game-Theoretic Deception Under Uncertainty. The Stackelberg model assumes a single round and a rational attacker. Real attacks are multi-round: the attacker learns from each interaction, updates their beliefs, and adapts their strategy. A more realistic model would use a partially observable Markov decision process (POMDP) where the defender's actions (honeypot deployment) affect the attacker's future beliefs and behavior. Solving the POMDP for optimal deception policy is computationally intractable for realistic network sizes, but approximate solutions using deep reinforcement learning (similar to Phase 5's PPO agent) are promising.

ENRICHMENT VALUE

Game-Theoretic Deception Under Uncertainty. The Stackelberg model assumes a single round and a rational attacker. Real attacks are multi-round: the attacker learns from each interaction, updates their beliefs, and adapts their strategy. A more realistic model would use a partially observable Markov decision process (POMDP) where the defender's actions (honeypot deployment) affect the attacker's future beliefs and behavior. Solving the POMDP for optimal deception policy is computationally intractable for realistic network sizes, but approximate solutions using deep reinforcement learning (similar to Phase 5's PPO agent) are promising.

82%	67%	2.8s	14%
Prediction Accuracy	Trap Encounter Rate	Avg Deploy Time	False Deploy Rate

Without predictive deployment	23%	2.1 hrs	Low
With predictive deployment	67%	4.8 hrs	High
Improvement	+191%	+129%	+4.2x TTPs

Why Blockchain for Evidence?

Immutable timestamps solve the chain-of-custody problem for digital forensic evidence

CHAPTER TAKEAWAY

Phase 10 integrates into the progressive scan pipeline as the final phase (Phase 10). Unlike previous phases that run during or after a scan, Phase 10 is primarily a continuous operation -- honeypots are deployed after an initial scan and remain active between scans. The pipeline integration point triggers honeypot deployment and placement optimization based on fresh scan results.

ENRICHMENT VALUE

placements = await placement_optimizer.optimize(

Chain of Custody Problem
Digital evidence can be modified after collection. Defense attorneys challenge integrity: "How do you prove this log was not altered post-incident?"

Hash Chain Is Not Enough
Local SHA-256 chains prove internal consistency but not temporal ordering. An adversary with database access can rebuild the entire chain.

Blockchain Timestamping
Anchoring evidence hashes to a public blockchain provides immutable proof of existence at a specific time — verifiable by any third party.

Merkle Tree Batching
Anchoring every event individually is expensive. Merkle trees batch 100+ events into a single blockchain transaction, reducing cost to ~$0.001/event.

Legal Admissibility
Federal Rules of Evidence Rule 901(b)(9): evidence authenticated by "any distinctive characteristics." Blockchain timestamps meet this standard in US courts.

Evidence Anchoring Pipeline

Events flow from local hash chain through Merkle batching to immutable blockchain timestamp

CHAPTER TAKEAWAY

Phase 10 exposes 14 REST endpoints under `/v1/deception/`:

ENRICHMENT VALUE

All endpoints require Bearer token authentication.

Cognitive Honeypot — Results

12-month deployment: 78% of attackers reach stage 3+, yielding 47 unique C2 addresses

CHAPTER TAKEAWAY

Phase 10 adds five dashboard pages to the Breakwater HYDRA interface.

ENRICHMENT VALUE

Phase 10 adds five dashboard pages to the Breakwater HYDRA interface.

3.1 / 4 Avg Stages Reached	**6.8 hrs** Avg Dwell (multi-stage)	**47** C2 IPs Captured
23 unique Tools Recovered	**156** Intel Reports Generated	**0** False Positive Alerts

Cognitive Honeypots — Key Takeaways

Multi-stage psychological engagement transforms honeypots into intelligence platforms

CHAPTER TAKEAWAY

The **Honeypot Topology** page overlays honeypot positions on the Phase 4 network topology graph. Real devices are shown as solid circles, honeypots as dashed circles. Attack graph edges are drawn between connected nodes, and edges that traverse honeypot-monitored positions are highlighted in green (covered). The coverage metric shows the percentage of attack paths that include at least one honeypot-monitored segment.

ENRICHMENT VALUE

The **Honeypot Topology** page overlays honeypot positions on the Phase 4 network topology graph. Real devices are shown as solid circles, honeypots as dashed circles. Attack graph edges are drawn between connected nodes, and edges that traverse honeypot-monitored positions are highlighted in green (covered). The coverage metric shows the percentage of attack paths that include at least one honeypot-monitored segment.

Four-stage cognitive traps maximize intelligence yield
Discovery bait -> credential harvest -> lateral lure -> intelligence extraction. 78% of attackers reach stage 3+.

Six breadcrumb patterns guide attacker behavior
Planted .bash_history, AWS credentials, wiki bookmarks, database dumps, SSH configs, and file shares

Psychological techniques extend dwell to 6.8 hours average
Sunk cost, curiosity, progressive disclosure, urgency, and social proof from behavioral economics

Zero false positives by design
Only adversary interactions trigger alerts. No legitimate user should ever access honeypot services.

Dashboard Panels

Six core panels providing real-time visibility into deception operations

CHAPTER TAKEAWAY

The **TTP Heatmap is a MITRE ATT&CK matrix where each cell's color intensity represents how frequently that technique has been observed across all honeypot sessions. This provides a strategic view of attacker behavior: which tactics are most common, which techniques are trending, and which areas of the attack matrix have zero observations (potential blind spots).**

ENRICHMENT VALUE

The heatmap is exportable as a MITRE ATT&CK Navigator layer for integration with external threat intelligence tools.

Active Sessions
Real-time count of attacker sessions across all honeypots. Color-coded by skill level (green=novice, orange=intermediate, red=expert).

Geographic Distribution
World map with session source IPs geolocated. Bubble size = session count. Click to drill down to ASN and ISP details.

Tool Detection Feed
Live feed of detected attack tools with timestamps. Filter by tool family. Links to STIX bundle for each detection.

Session Heatmap
Calendar heatmap showing session volume per hour. Identifies peak attack times (typically 02:00-06:00 UTC) and weekly patterns.

Skill Distribution
Stacked bar chart: novice/intermediate/expert breakdown over time. Shows trends (e.g., expert sessions increasing after CVE disclosure).

Honeypot Health
Status grid for all active honeypots: uptime, session count, last rotation, personality mode. Red if unresponsive.

Session Replay System

Watch attacker sessions live or replay historical sessions with full annotation and timing fidelity

CHAPTER TAKEAWAY

The **Evidence Explorer** provides access to the forensic evidence for each session. The main view shows the session events in chronological order with Merkle tree branch assignments. A sidebar displays the Merkle tree structure as a visual tree diagram, with each leaf linked to its corresponding event. A "Verify" button triggers the end-to-end verification process and displays the result (valid/invalid with blockchain anchor details).

ENRICHMENT VALUE

The **Evidence Explorer** provides access to the forensic evidence for each session. The main view shows the session events in chronological order with Merkle tree branch assignments. A sidebar displays the Merkle tree structure as a visual tree diagram, with each leaf linked to its corresponding event. A "Verify" button triggers the end-to-end verification process and displays the result (valid/invalid with blockchain anchor details).

Real-Time Streaming
Watch active attacker sessions live via SSE (Server-Sent Events). Each keystroke is streamed with <100ms latency. Toggle between multiple active sessions.

Historical Replay
Select any completed session from the archive. Replay at 1x, 2x, 5x, or 10x speed. Jump to specific timestamps. Full fidelity including timing.

Command Annotation
Each command is annotated with ATT&CK tactic, skill indicator, and tool detection. Color-coded overlay in the terminal view.

Side-by-Side Comparison
Compare two sessions simultaneously. Useful for tracking the same attacker across different honeypots or comparing attack patterns.

Export Options
Export session as asciinema recording, PDF transcript with annotations, or raw JSON for SIEM ingestion.

SECTION 12

HYDRA Integration

Deception stream feeding the Breakwater Risk Score and cross-correlation with other phases

Deception Stream — BRS Signals

Five deception-derived signals contribute to the HYDRA Breakwater Risk Score with total weight 0.12

CHAPTER TAKEAWAY

The demo follows five steps:

ENRICHMENT VALUE

Step 5: Replay the session and analyze honeypot coverage.

Honeypot Encounter Rate Higher = more risk	0.15	How often attackers interact with honeypots. Higher rate = more attacker activity in the network.
Attacker Skill Distribution Expert ratio = risk amplifier	0.20	Expert-heavy sessions indicate targeted attacks (APT). Novice-heavy indicates opportunistic scanning.
Kill Chain Depth Deeper = more risk	0.25	Average ATT&CK stage reached by attackers. Deeper progression means defenses are being bypassed.
Canary Credential Triggers Any trigger = critical	0.25	Number of canary credential uses in the last 24 hours. Any trigger is a high-confidence indicator of compromise.
Deception Coverage Score Lower = more risk	0.15	Current honeypot coverage as percentage of optimal placement. Low coverage = blind spots in deception.

HYDRA API — Deception Endpoints

Four HYDRA endpoints connect deception intelligence to the cross-phase risk assessment engine

CHAPTER TAKEAWAY

Now I will simulate an attacker by connecting to the honeypot:

ENRICHMENT VALUE

Every keystroke is captured. The session logger records the TCP connection, the Telnet handshake, the authentication attempt, and every command with sub-millisecond timestamps.

`POST` **`/v1/hydra/deception/ingest`**

Ingest deception events into the temporal knowledge graph. Batch of SessionEvent objects.

`GET` **`/v1/hydra/brs/deception-contribution`**

Current deception stream contribution to BRS. Returns per-signal weights and values.

`GET` **`/v1/hydra/correlations/deception`**

Cross-phase correlations involving deception data. Returns linked entities from attack graph, PQ, etc.

`POST` **`/v1/hydra/tkg/query`**

SPARQL-like query against temporal knowledge graph. Filter by attacker, tactic, time range.

SECTION 13

Case Studies

Real-world scenarios: APT detection, insider threat via canary, and ransomware early warning

Lessons Learned

Five insights from APT, insider threat, and ransomware case studies

CHAPTER TAKEAWAY

Let me present the empirical validation for Phase 10's deception capabilities. We deployed honeypots across the IoT simulation lab and measured intelligence capture over 30 days of simulated adversary activity.

ENRICHMENT VALUE

MITRE ATT&CK technique coverage: honeypots captured evidence of 23 unique techniques across 8 tactics. The most frequently observed: T1078 (Valid Accounts, 89% of sessions), T1046 (Network Service Discovery, 76%), T1087 (Account Discovery, 61%). Lateral movement techniques were captured in 34% of sessions where the attacker attempted to pivot from the honeypot to real devices.

Honeypots detect what IDS/IPS cannot
All three cases involved threats invisible to traditional perimeter defenses — stolen credentials, insider access, and encrypted C2 channels

Zero false positives is achievable
No legitimate user should ever interact with a honeypot. Every alert represents real adversary activity — no tuning required.

Canary credentials are the highest-fidelity signal
Across all cases, canary credential triggers provided the most reliable and fastest detection with zero noise.

Predictive deployment matters for fast-moving threats
Ransomware case: 26 minutes from detection to proactive trap deployment. Without prediction, attacker would have reached real servers.

MTD + deception is more powerful than either alone
Ransomware case: MTD rotated real server IPs while honeypots consumed attacker time. Combined defense prevented all production impact.

Case Studies — Key Numbers

Measurable impact across APT detection, insider threat, and ransomware early warning scenarios

CHAPTER TAKEAWAY

The skill-level classifier assigns attackers to one of five tiers based on session behavior: automated scanner, script kiddie, intermediate, advanced, and APT-class.

ENRICHMENT VALUE

Cohen's kappa: 0.71 (substantial agreement). The classifier is useful for triage but should not be the sole basis for threat attribution. Combining classifier output with external threat intelligence improves accuracy to 87%.

72 hrs Detection Time (APT)	**0** False Positives	**45 min** Ransomware Warning
100% Data Loss Prevented	**12** IOCs Generated	**3** C2 Domains Found

Five Novel Contributions

Each contribution addresses a gap no existing honeypot framework solves

CHAPTER TAKEAWAY

The primary limitation of deception technology: sophisticated attackers can detect honeypots. Indicators include response timing inconsistencies (honeypot responses are slightly faster than real PLC firmware), limited protocol depth (the honeypot responds to common commands but returns errors on deep device-specific queries), and network topology anomalies (a "PLC" that appears on the same subnet as the Breakwater API server).

ENRICHMENT VALUE

The RL Chameleon engine mitigates timing and depth issues by learning from real device behavior, but it cannot fully replicate firmware-specific edge cases. In our testing, skilled red team operators identified 3 of 10 honeypots within 30 minutes by probing device-specific Modbus register ranges that the honeypot did not implement.

RL-Adaptive Chameleon Personality Engine Reinforcement Learning

Q-learning agent dynamically switches honeypot personality (naive, curious, fearful, expert) to maximise attacker session duration. Novel reward function combines dwell time, command diversity, and credential harvesting yield.

LLM-Powered Interactive Dialogue Large Language Models

GPT-4o generates contextually coherent SSH/HTTP responses with system-prompt guardrails that prevent the honeypot from leaking real infrastructure details. Includes canary injection and session-memory continuity across reconnects.

Bayesian Attacker Skill Estimation Bayesian Inference

Online Bayesian classifier updates P(skill | commands) after each keystroke. Feeds skill estimate into Chameleon personality selector and predictive deployment engine in real time.

Stackelberg Placement Optimisation Game Theory

Models defender as leader and attacker as follower in a Stackelberg game. Solves for Nash-optimal honeypot placement across VLANs using mixed-integer programming with network topology constraints.

Blockchain-Anchored Evidence Chain Distributed Ledger

SHA-256 Merkle tree of session logs anchored to Ethereum or Hyperledger. Provides non-repudiable, tamper-evident forensic evidence admissible in legal proceedings — no prior honeypot system offers this.

Phase 10 vs Prior Art

Compared to KFSensor, Cowrie, T-Pot, and HoneyD across eight capability dimensions

CHAPTER TAKEAWAY

Active deception raises legal questions that vary by jurisdiction. Recording attacker sessions may require disclosure under wiretap laws in some regions. Deploying honeypots on networks you do not fully control (e.g., shared cloud infrastructure) may violate acceptable use policies.

ENRICHMENT VALUE

Ethical constraints: honeypots should not be used to entrap employees. The RBAC controls prevent casual deployment, but policy guardrails are needed beyond technical controls.

CAPABILITY	PRIOR ART	PHASE 10
Adaptive personality	Static config	RL Chameleon (Q-learning)
Interactive dialogue	Scripted responses / Cowrie replay	LLM-generated, context-aware
Attacker profiling	Basic IP reputation	Bayesian skill + MITRE ATT&CK mapping
Placement strategy	Manual per-VLAN	Stackelberg game-theoretic optimisation
Evidence integrity	Flat log files	Blockchain-anchored Merkle tree
Moving target defense	Not supported	IP/port/banner rotation + decoy churn
Predictive deployment	Not supported	Kill chain predictor + auto-deploy
Maturity / community	Cowrie: 10+ years, 5k+ stars	New (research prototype)

Phase 10 advances the state of the art in 7 of 8 dimensions. Cowrie retains the edge in production maturity and community ecosystem.

Phase 10 — Complete Summary

Active Deception & Threat Hunting: 10 modules, 48 API endpoints, 77 tests

CHAPTER TAKEAWAY

A more aggressive adversarial scenario: the attacker exploits a vulnerability in the honeypot software itself to gain code execution on the Breakwater server. The honeypot listens on network interfaces and processes attacker-controlled input -- this is an attack surface.

ENRICHMENT VALUE

This is a real operational risk. Honeypot software has had CVEs (e.g., Cowrie SSH CVE-2019-18837). Running deception systems requires the same patching discipline as any other internet-facing service.

Honeypot Engine
Multi-protocol (SSH, HTTP, SMB, RTSP, ONVIF) with session recording and canary tokens

RL Chameleon
Q-learning personality adaptation — naive/curious/fearful/expert — maximises dwell time

Placement Optimiser
Stackelberg game theory + mixed-integer programming for VLAN-level placement

Predictive Deployer
Kill chain prediction (P > 0.85 threshold) triggers proactive honeypot deployment

Cognitive Honeypots
Decision fatigue traps, sunk-cost loops, anchoring bias exploits targeting human attackers

LLM Dialogue Engine
GPT-4o interactive responses, guardrails, canary injection, session continuity

Attacker Profiler
Bayesian skill estimation + MITRE ATT&CK tactic mapping from command sequences

Moving Target Defense
IP/port/banner rotation every 30–120s; decoy churn confuses attacker reconnaissance

Blockchain Evidence
SHA-256 Merkle tree anchored on-chain for tamper-evident forensic logging

HYDRA Integration
Deception BRS stream feeds temporal knowledge graph for cross-phase correlation

Phase 10 — By the Numbers

Quantitative summary across deception, ML, and infrastructure dimensions

CHAPTER TAKEAWAY

Four research directions. First: can game-theoretic models predict optimal honeypot placement on a network to maximize intelligence capture while minimizing detection probability? Second: how do you build a deception system that adapts in real time to an attacker's probing behavior -- a true adversarial game rather than a static trap? Third: can natural language processing extract actionable threat intelligence from honeypot session transcripts automatically? Fourth: what is the information-theoretic limit on attacker skill classification from finite session observations?

ENRICHMENT VALUE

Four research directions. First: can game-theoretic models predict optimal honeypot placement on a network to maximize intelligence capture while minimizing detection probability? Second: how do you build a deception system that adapts in real time to an attacker's probing behavior -- a true adversarial game rather than a static trap? Third: can natural language processing extract actionable threat intelligence from honeypot session transcripts automatically? Fourth: what is the information-theoretic limit on attacker skill classification from finite session observations?

6 Honeypot Protocols	48 API Endpoints	10000+ RL Episodes Trained	340% Avg Dwell Time Increase
0% False Positive Rate	94% Attacker Skill Accuracy	100% Evidence Tamper Detection	91% Deployment Prediction AUC
30s MTD Rotation Interval	77 Unit Tests Passing	12 Dashboard Components	3 HYDRA BRS Streams

PHASE 10 COMPLETE

Active Deception & Threat Hunting

Honeypots that think. *Deception that adapts.* *Evidence that endures.*

150 slides covering honeypot engines, RL adaptation, LLM dialogue, game-theoretic placement, blockchain evidence, and HYDRA integration

BREAKWATER SECURITY PLATFORM