PROVISIONAL PATENT APPLICATION

Title: Cybersecurity-Specific AI Agent Transport Network for Quantum-Resistant Fragment Distribution and Threat Response

Inventor(s): [To be filled]

Application Type: Provisional Patent Application

Filing Date: [To be filled]

Application Number: [To be assigned by USPTO]

TECHNICAL FIELD

This invention relates to cybersecurity-specific distributed AI agent networks that provide secure transport of encrypted data fragments across global locations through autonomous agent coordination, quantum-resistant zero-knowledge protocols, and behavioral threat detection mechanisms designed exclusively for cybersecurity applications.

BACKGROUND OF THE INVENTION

Current Cybersecurity Distribution Systems

Traditional cybersecurity data distribution relies on:

1. VPN-Based Networks: Encrypted tunnels vulnerable to quantum cryptographic attacks

2. CDN Security Models: Content distribution with limited cryptographic protection

3. Distributed Security Operations Centers (SOC): Manual coordination and human oversight

4. Threat Intelligence Sharing: Static data distribution without adaptive threat response

Cybersecurity-Specific Challenges

Quantum Threat Vulnerabilities:

- Current security transport relies on RSA/ECC encryption vulnerable to Shor's algorithm

- Threat intelligence distribution uses mathematical assumptions broken by quantum computers

- Security incident response networks lack quantum-resistant communication protocols

- Cryptographic key distribution systems face quantum computing threats

Cybersecurity Operational Limitations:

- Manual threat intelligence distribution creates delays during critical incidents

- Security operations require human intervention for coordination and decision-making

- Insider threat detection limited in distributed security environments

- Security fragment transport lacks adaptive threat-aware routing

Attack Surface Expansion:

- Traditional security networks provide persistent attack surfaces for adversaries

- Centralized security coordination creates high-value targets

- Static security protocols cannot adapt to emerging quantum threats

- Security agent communication patterns can be analyzed and exploited

Prior Art Analysis - Cybersecurity Focus

WO2021084510A1 (AI Agent Workflows): General AI agent message-based architecture but no cybersecurity-specific threat detection or quantum-resistant protocols.

US20190140913A1 (AI Capabilities at Network Switch): Network-based AI capabilities but limited to infrastructure rather than cybersecurity threat response.

US20240354567A1 (Neuro-Symbolic AI Platform): Knowledge-driven automation platform but not specifically designed for cybersecurity threat response or quantum-resistant operations.

Gap in Prior Art: No existing system provides cybersecurity-specific AI agent networks with quantum-resistant protocols, threat-aware routing, and adaptive security response capabilities.

SUMMARY OF THE INVENTION

The present invention provides a Cybersecurity-Specific AI Agent Transport Network that enables autonomous AI agents to securely transport encrypted security fragments, threat intelligence, and incident response data across global locations using quantum-resistant protocols specifically designed for

cybersecurity operations. The system provides adaptive threat response, quantum-immune communication, and specialized security agent behavioral patterns.

Core Innovation Elements

1. Cybersecurity Agent Specialization: Agents specifically designed for security operations and threat response

2. Quantum-Resistant Security Protocols: Communication protocols immune to quantum cryptographic attacks

3. Threat-Aware Routing: Dynamic routing based on real-time threat intelligence and adversary behavior

4. Security Incident Response Coordination: Specialized coordination for cybersecurity incident management

5. Adaptive Security Behavioral Patterns: Agent behaviors that evolve based on threat landscape changes

Technical Advantages

- Quantum-Immune Security Operations: Cybersecurity networks resistant to quantum computing attacks

- Threat-Adaptive Architecture: Security operations that adapt to emerging threats and attack patterns

- Autonomous Security Response: AI-driven security operations without human coordination delays

- Zero-Knowledge Security Intelligence: Threat intelligence transport without content exposure

- Distributed Security Resilience: Self-healing security networks immune to single points of failure

DETAILED DESCRIPTION OF THE INVENTION

System Architecture

The Cybersecurity-Specific AI Agent Transport Network comprises seven primary components:

1. Security Agent Generation Engine - Creates cybersecurity-specialized AI agents

2. Threat-Aware Mission Coordinator - Assigns security missions based on threat intelligence

3. Quantum-Resistant Communication Framework - Provides quantum-immune agent communication

4. Security Fragment Transport Protocol - Specialized protocols for security data transport

5. Cybersecurity Behavioral Authentication - Authentication methods specific to security operations

6. Threat Intelligence Distribution Network - Real-time threat intelligence sharing

7. Security Incident Response Orchestrator - Coordinates multi-agent security incident response

Component 1: Security Agent Generation Engine

Purpose: Create specialized AI agents designed exclusively for cybersecurity operations with threat-aware behavioral patterns and quantum-resistant communication capabilities.

Technical Implementation:

```python

import time

import hashlib

import secrets

from typing import Dict, List, Tuple

from enum import Enum

class SecurityAgentType(Enum):

THREAT_INTEL_TRANSPORT = "threat_intelligence_transport"

INCIDENT_RESPONSE = "incident_response"

FRAGMENT_SECURITY = "fragment_security_transport"

THREAT_HUNTING = "threat_hunting_coordination"

VULNERABILITY_ASSESSMENT = "vulnerability_assessment"

MALWARE_ANALYSIS = "malware_analysis_transport"

class CybersecurityAgentGenerator:

def __init__(self):
```

```
self.active_security_agents = {}
self.threat_intelligence_db = {}
self.quantum_resistant_protocols = {
'CRYSTALS_Kyber': {'key_size': 3168, 'security_level': 256},
'CRYSTALS_Dilithium': {'signature_size': 2420, 'security_level': 256},
'FALCON': {'signature_size': 690, 'security_level': 256}
}
self.security_behavioral_profiles = {
SecurityAgentType.THREAT_INTEL_TRANSPORT: {
'stealth_priority': 0.9,
'speed_priority': 0.8,
'verification_threshold': 0.95,
'quantum_resistance': 'maximum'
},
SecurityAgentType.INCIDENT_RESPONSE: {
'coordination_priority': 0.9,
'latency_tolerance': 0.1,
'reliability_requirement': 0.99,
'escalation_threshold': 0.7
},
SecurityAgentType.FRAGMENT_SECURITY: {
'confidentiality_priority': 1.0,
'integrity_validation': 'cryptographic',
'transport_obfuscation': 0.95,
'anti_analysis': 'maximum'
}
}
```

```
def generate_security_agent(self, agent_type: SecurityAgentType,
mission_parameters: Dict,
threat_context: Dict) -> 'CybersecurityAgent':
"""Generate specialized cybersecurity AI agent"""
agent_id = self.generate_secure_agent_id()
```

Create threat-aware behavioral profile

```
behavioral_profile = self.create_security_behavioral_profile(
agent_type, mission_parameters, threat_context
)
```

Initialize quantum-resistant communication

```
quantum_keys = self.initialize_quantum_resistant_keys(agent_id)
```

Configure threat detection capabilities

```
threat_detection_config = self.configure_threat_detection(
agent_type, threat_context
)
security_agent = CybersecurityAgent(
agent_id=agent_id,
agent_type=agent_type,
behavioral_profile=behavioral_profile,
quantum_keys=quantum_keys,
threat_detection_config=threat_detection_config,
mission_parameters=mission_parameters
)
```

Register agent for threat intelligence updates

```
self.register_threat_intelligence_subscription(agent_id, agent_type)
```

Initialize security monitoring

```
security_agent.initialize_security_monitoring()
self.active_security_agents[agent_id] = {
'agent': security_agent,
'type': agent_type,
'creation_time': time.time(),
'threat_context': threat_context,
'security_clearance': self.determine_security_clearance(mission_parameters),
'operational_status': 'active'
}
return security_agent
def create_security_behavioral_profile(self, agent_type: SecurityAgentType,
mission_params: Dict,
threat_context: Dict) -> Dict:
"""Create cybersecurity-specific behavioral profile"""
base_profile = self.security_behavioral_profiles[agent_type].copy()
```

Adapt behavior based on current threat level

```
threat_level = threat_context.get('threat_level', 'moderate')
if threat_level == 'critical':
base_profile['stealth_priority'] = min(1.0, base_profile.get('stealth_priority', 0.5)
+ 0.3)
base_profile['verification_threshold'] = min(1.0,
base_profile.get('verification_threshold', 0.8) + 0.15)
```

base_profile['quantum_resistance'] = 'maximum'

Add mission-specific behavioral adaptations

if mission_params.get('classification_level') == 'top_secret':

base_profile['anti_analysis'] = 'maximum'

base_profile['transport_obfuscation'] = 1.0

base_profile['communication_encryption'] = 'quantum_resistant_only'

Configure threat-aware routing preferences

base_profile['routing_strategy'] = self.determine_routing_strategy(

threat_context, mission_params

)

Add temporal behavioral variations to prevent pattern analysis

base_profile['timing_variations'] = self.generate_timing_obfuscation_patterns()

return base_profile

def initialize_quantum_resistant_keys(self, agent_id: str) -> Dict:

"""Initialize quantum-resistant cryptographic keys for agent"""

quantum_keys = {}

CRYSTALS-Kyber for key encapsulation

quantum_keys['kyber'] = {

'public_key': self.generate_kyber_keypair()['public'],

'private_key': self.generate_kyber_keypair()['private'],

'algorithm': 'CRYSTALS-Kyber-1024',

'security_level': 256

}

CRYSTALS-Dilithium for digital signatures

```
quantum_keys['dilithium'] = {
'signing_key': self.generate_dilithium_keypair()['private'],
'verification_key': self.generate_dilithium_keypair()['public'],
'algorithm': 'CRYSTALS-Dilithium-5',
'security_level': 256
}
```

FALCON for compact signatures (backup)

```
quantum_keys['falcon'] = {
'signing_key': self.generate_falcon_keypair()['private'],
'verification_key': self.generate_falcon_keypair()['public'],
'algorithm': 'FALCON-1024',
'security_level': 256
}
```

Add key rotation schedule

```
quantum_keys['rotation_schedule'] = {
'current_epoch': int(time.time()),
'rotation_interval': 3600, # 1 hour
'next_rotation': int(time.time()) + 3600
}
return quantum_keys
def configure_threat_detection(self, agent_type: SecurityAgentType,
threat_context: Dict) -> Dict:
"""Configure threat detection capabilities for security agent"""
```

```
threat_detection_config = {
'detection_modules': [],
'threat_signatures': {},
'behavioral_analysis': {},
'quantum_threat_monitoring': {}
}
```

Configure type-specific threat detection

```
if agent_type == SecurityAgentType.THREAT_INTEL_TRANSPORT:
threat_detection_config['detection_modules'] = [
'network_traffic_analysis',
'communication_pattern_analysis',
'quantum_attack_detection',
'man_in_the_middle_detection'
]
threat_detection_config['quantum_threat_monitoring'] = {
'shor_algorithm_indicators': True,
'grover_algorithm_indicators': True,
'quantum_key_distribution_tampering': True,
'post_quantum_crypto_attacks': True
}
elif agent_type == SecurityAgentType.INCIDENT_RESPONSE:
threat_detection_config['detection_modules'] = [
'incident_correlation_analysis',
'attack_vector_identification',
'threat_actor_profiling',
'impact_assessment'
```

```
]
threat_detection_config['behavioral_analysis'] = {
'coordination_anomalies': True,
'response_time_deviations': True,
'communication_compromise_indicators': True
}
elif agent_type == SecurityAgentType.FRAGMENT_SECURITY:
threat_detection_config['detection_modules'] = [
'fragment_integrity_monitoring',
'transport_path_verification',
'quantum_cryptographic_attacks',
'side_channel_attack_detection'
]
```

Add threat context-specific configurations

```
current_threats = threat_context.get('active_threats', [])
for threat in current_threats:
if threat['type'] == 'quantum_computing':
threat_detection_config['quantum_threat_monitoring']['priority'] = 'maximum'
elif threat['type'] == 'nation_state_actor':
threat_detection_config['stealth_detection'] = {
'advanced_persistent_threat': True,
'supply_chain_compromise': True,
'zero_day_exploitation': True
}
return threat_detection_config
class CybersecurityAgent:
```

```
def __init__(self, agent_id: str, agent_type: SecurityAgentType,
behavioral_profile: Dict, quantum_keys: Dict,
threat_detection_config: Dict, mission_parameters: Dict):
self.agent_id = agent_id
self.agent_type = agent_type
self.behavioral_profile = behavioral_profile
self.quantum_keys = quantum_keys
self.threat_detection_config = threat_detection_config
self.mission_parameters = mission_parameters
self.security_state = {
'current_threat_level': 'normal',
'active_threats': [],
'communication_security': 'quantum_resistant',
'operational_security': 'maximum'
}
self.transport_metrics = {
'fragments_transported': 0,
'security_incidents_detected': 0,
'quantum_attacks_prevented': 0,
'successful_missions': 0
}
def transport_security_fragment(self, fragment_data: bytes,
destination_agent: str,
security_classification: str) -> Dict:
"""Transport security fragment using quantum-resistant protocols"""
```

Validate fragment security requirements

```
if not self.validate_security_clearance(security_classification):
return {'status': 'failed', 'reason': 'insufficient_clearance'}
```

Apply quantum-resistant encryption

```
encrypted_fragment = self.quantum_encrypt_fragment(
fragment_data, security_classification
)
```

Create transport manifest with integrity validation

```
transport_manifest = {
'fragment_id': self.generate_fragment_id(),
'destination_agent': destination_agent,
'security_classification': security_classification,
'transport_time': time.time(),
'integrity_hash': self.compute_quantum_resistant_hash(fragment_data),
'transport_route': self.calculate_secure_route(destination_agent)
}
```

Monitor for threats during transport

```
threat_monitoring_result = self.monitor_transport_threats(transport_manifest)
if threat_monitoring_result['threats_detected']:
```

Implement evasive routing

```
alternative_route = self.calculate_evasive_route(
destination_agent, threat_monitoring_result['threats']
)
transport_manifest['transport_route'] = alternative_route
```

Execute secure transport

transport_result = self.execute_quantum_resistant_transport(

encrypted_fragment, transport_manifest

)

Update security metrics

if transport_result['status'] == 'success':

self.transport_metrics['fragments_transported'] += 1

self.transport_metrics['successful_missions'] += 1

return transport_result

def quantum_encrypt_fragment(self, fragment_data: bytes,

classification: str) -> bytes:

"""Encrypt fragment using quantum-resistant algorithms"""

Select encryption algorithm based on classification

if classification in ['top_secret', 'classified']:

Use CRYSTALS-Kyber for key encapsulation + AES-256-GCM

encapsulated_key = self.kyber_encapsulate(self.quantum_keys['kyber']['public_key'])

encrypted_data = self.aes_gcm_encrypt(fragment_data, encapsulated_key)

Add Dilithium signature for authentication

signature = self.dilithium_sign(encrypted_data, self.quantum_keys['dilithium']['signing_key'])

return encrypted_data + signature

else:

Use standard quantum-resistant encryption for lower classifications

return self.quantum_resistant_encrypt(fragment_data)

```

### Component 2: Threat-Aware Mission Coordinator

Purpose: Assign cybersecurity missions to specialized agents based on real-time threat intelligence, incident priority, and security requirements.

Technical Implementation:

```python

class ThreatAwareMissionCoordinator:

def __init__(self):

self.mission_queue = []

self.threat_intelligence_feed = {}

self.agent_capabilities = {}

self.incident_priority_matrix = {

'critical': {'priority': 1, 'max_response_time': 60}, # 1 minute

'high': {'priority': 2, 'max_response_time': 300}, # 5 minutes

'medium': {'priority': 3, 'max_response_time': 1800}, # 30 minutes

'low': {'priority': 4, 'max_response_time': 3600} # 1 hour

}

def assign_security_mission(self, mission_request: Dict) -> Dict:

"""Assign cybersecurity mission based on threat context and agent capabilities"""

## Analyze threat context

threat_analysis = self.analyze_current_threats(mission_request)

## Determine mission requirements

```
mission_requirements = self.calculate_mission_requirements(
mission_request, threat_analysis
)
```

Select optimal agent for mission

```
selected_agent = self.select_security_agent(mission_requirements)
if not selected_agent:
return {'status': 'failed', 'reason': 'no_suitable_agent'}
```

Configure mission-specific security protocols

```
mission_config = self.configure_mission_security(
mission_requirements, threat_analysis
)
```

Assign mission to agent

```
mission_assignment = {
'mission_id': self.generate_mission_id(),
'agent_id': selected_agent['agent_id'],
'mission_type': mission_request['type'],
'security_requirements': mission_requirements,
'threat_context': threat_analysis,
'mission_config': mission_config,
'deadline': time.time() + mission_requirements['max_duration']
}
```

Monitor mission execution

```
self.initialize_mission_monitoring(mission_assignment)
```

return {'status': 'assigned', 'mission_assignment': mission_assignment}

def analyze_current_threats(self, mission_request: Dict) -> Dict:

"""Analyze current threat landscape relevant to mission"""

threat_analysis = {

'quantum_threat_level': self.assess_quantum_threats(),

'nation_state_indicators': self.check_nation_state_activity(),

'active_incidents': self.get_active_security_incidents(),

'network_threat_level': self.assess_network_threats(),

'recommended_security_level': 'high'

}

Adjust security level based on mission classification

if mission_request.get('classification') == 'top_secret':

threat_analysis['recommended_security_level'] = 'maximum'

Check for mission-specific threats

mission_type = mission_request.get('type')

if mission_type == 'threat_intelligence_transport':

threat_analysis['intelligence_compromise_risk'] = self.assess_intelligence_risks()

elif mission_type == 'incident_response':

threat_analysis['incident_escalation_risk'] = self.assess_incident_risks()

return threat_analysis

```

### Component 3: Quantum-Resistant Communication Framework

Purpose: Provide quantum-immune communication protocols specifically designed for cybersecurity operations and threat response coordination.

Technical Implementation:
```

```python
class QuantumResistantSecurityCommunication:
def __init__(self):
self.post_quantum_algorithms = {
'kyber': 'CRYSTALS-Kyber-1024',
'dilithium': 'CRYSTALS-Dilithium-5',
'falcon': 'FALCON-1024',
'sphincs': 'SPHINCS+-256s'
}
self.security_protocols = {}
self.threat_aware_routing = {}
def establish_secure_channel(self, source_agent: str,
destination_agent: str,
security_level: str) -> Dict:
"""Establish quantum-resistant secure communication channel"""
```

Generate session keys using quantum-resistant KEM

```
session_keys = self.generate_quantum_session_keys(security_level)
```

Configure threat-aware routing

```
secure_route = self.calculate_threat_aware_route(
source_agent, destination_agent, security_level
)
```

Establish channel with forward secrecy

```
channel_config = {
'channel_id': self.generate_channel_id(),
```

'encryption': session_keys['encryption'],

'authentication': session_keys['authentication'],

'forward_secrecy': True,

'quantum_resistance': 'verified',

'threat_monitoring': 'active',

'route': secure_route

}

return channel_config

```

## CLAIMS

### Independent Claims

Claim 1: A cybersecurity-specific AI agent transport network method comprising:

- generating specialized AI agents configured exclusively for cybersecurity operations including threat intelligence transport, incident response, and security fragment distribution;

- implementing quantum-resistant communication protocols using CRYSTALS-Kyber, CRYSTALS-Dilithium, and FALCON algorithms for cybersecurity data protection;

- coordinating threat-aware mission assignment based on real-time security threat intelligence and incident priority matrices;

- providing zero-knowledge security fragment transport where agents transport encrypted cybersecurity data without knowledge of content or purpose;

- executing adaptive security behavioral patterns that evolve based on cybersecurity threat landscape changes and quantum computing attack indicators.

Claim 2: A cybersecurity AI agent transport system comprising:

- a security agent generation engine configured to create cybersecurity-specialized AI agents with threat detection capabilities and quantum-resistant communication protocols;

- a threat-aware mission coordinator configured to assign security missions based on incident priority, threat intelligence, and agent security clearances;
```

- a quantum-resistant communication framework configured to provide quantum-immune communication channels specifically for cybersecurity operations;

- a security incident response orchestrator configured to coordinate multi-agent cybersecurity incident response and threat mitigation;

- wherein the system is specifically designed and limited to cybersecurity applications and threat response operations.

Claim 3: A method for quantum-resistant cybersecurity agent coordination comprising:

- creating agent behavioral profiles specific to cybersecurity operations including stealth priority, verification thresholds, and quantum resistance requirements;

- implementing threat detection modules for quantum cryptographic attacks, advanced persistent threats, and nation-state actor indicators;

- coordinating security fragment transport using threat-aware routing that adapts to real-time cybersecurity threat intelligence;

- providing cybersecurity incident response coordination through autonomous AI agents with specialized security behavioral patterns.

Dependent Claims

Claim 4: The method of claim 1, wherein cybersecurity agent types include threat intelligence transport agents, incident response agents, fragment security agents, threat hunting coordination agents, vulnerability assessment agents, and malware analysis transport agents.

Claim 5: The system of claim 2, wherein quantum-resistant protocols include CRYSTALS-Kyber for key encapsulation, CRYSTALS-Dilithium for digital signatures, and FALCON for compact signatures with 256-bit security levels.

Claim 6: The method of claim 3, wherein threat-aware routing calculates secure routes based on active threat indicators, nation-state actor presence, quantum computing attack indicators, and advanced persistent threat signatures.

Claim 7: The system of claim 2, wherein security agent behavioral profiles adapt based on threat levels including critical (0.9+ stealth priority), high (0.95+ verification threshold), and maximum quantum resistance requirements.

Claim 8: The method of claim 1, wherein security incident response coordination includes incident correlation analysis, attack vector identification, threat actor profiling, and automated impact assessment.

Claim 9: The system of claim 2, wherein security clearance levels determine agent mission assignments including unclassified, confidential, secret, and top secret cybersecurity operations.

Claim 10: The method of claim 3, further comprising quantum threat monitoring including Shor's algorithm indicators, Grover's algorithm indicators, quantum key distribution tampering detection, and post-quantum cryptographic attack prevention.

PROTOTYPE SYSTEM DESIGN AND PROJECTED CAPABILITIES

Cybersecurity Agent Architecture Framework

Threat Intelligence Transport Design:

- Quantum-Resistant Encryption Overhead: System designed to minimize latency impact per fragment while maintaining quantum security

- Threat Detection Architecture: Framework designed for high detection rates for quantum cryptographic attacks

- Zero-Knowledge Transport Verification: System designed to ensure successful transport without content exposure

- Adaptive Routing Framework: Architecture intended to provide effective threat avoidance using dynamic routing

Incident Response Coordination Framework:

- Response Time Optimization: System designed to significantly reduce incident response coordination time

- Multi-Agent Coordination Design: Framework intended for successful multi-agent incident response operations

- Threat Escalation Prevention: Architecture designed to reduce incident escalation through proactive agent coordination

- Security Communication Reliability: System designed for reliable quantum-resistant agent communication

Security-Specific Architecture

Quantum Resistance Framework:

- CRYSTALS-Kyber Integration: System designed to utilize 256-bit security level with compact key sizes

- CRYSTALS-Dilithium Signatures: Framework designed to support post-quantum digital signatures with high security

- FALCON Compact Signatures: Architecture intended to provide compact signatures while maintaining quantum security

- Quantum Attack Prevention: System designed to resist Shor's and Grover's algorithm attacks through post-quantum cryptography integration

INDUSTRIAL APPLICABILITY

Cybersecurity Applications

Security Operations Centers (SOC): Multi-location SOC coordination with quantum-resistant communication and automated threat intelligence distribution.

Incident Response Teams: Distributed incident response coordination with specialized AI agents for different phases of cybersecurity incident management.

Threat Intelligence Sharing: Secure distribution of classified threat intelligence across organizational boundaries with zero-knowledge transport protocols.

Critical Infrastructure Protection: Quantum-resistant security coordination for power grids, financial networks, and government systems requiring highest security levels.

Commercial Advantages

Quantum-Immune Cybersecurity: First cybersecurity-specific AI agent network providing quantum resistance through specialized behavioral patterns and communication protocols.

Adaptive Threat Response: Security operations that automatically adapt to emerging threats and attack patterns through AI agent behavioral evolution.

Zero-Knowledge Security Operations: Cybersecurity coordination without exposing sensitive information or operational details to individual agents or network components.

CONCLUSION

The Cybersecurity-Specific AI Agent Transport Network provides quantum-resistant security operations through specialized AI agents designed exclusively for cybersecurity applications. By focusing on threat-aware coordination, quantum-immune communication, and adaptive security behavioral patterns, the system enables autonomous cybersecurity operations that evolve with the threat landscape while maintaining operational security and quantum resistance.

Key Technical Innovations:

1. Cybersecurity-specialized AI agent behavioral profiles and threat detection capabilities

2. Quantum-resistant communication protocols specifically designed for security operations

3. Threat-aware mission coordination based on real-time cybersecurity intelligence

4. Zero-knowledge security fragment transport with quantum-immune encryption

5. Adaptive security behavioral patterns that evolve based on cybersecurity threat indicators

END OF PROVISIONAL PATENT APPLICATION

Filing Status: Ready for USPTO submission with cybersecurity-specific limitations

Priority Date: [To be established upon filing]

Related Applications: Integrates with Temporal Constraint-Based Quantum-Safe Security Architecture

International Filing: PCT application planned within 12 months