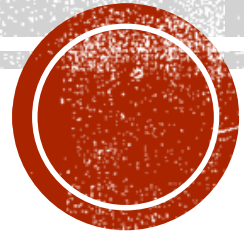
SOFTWARE DEFINED NETWORKING AND SECURITY

CHAPTER 7 MOVING TARGET DEFENSE

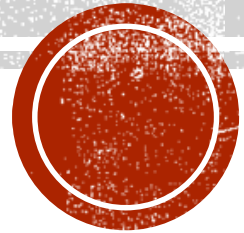
Dijiang Huang, Ankur Chowdhary, and Sandeep Pisharody



OUTLINE

- Foundations of Moving Target Defense
- MTD Classification
- SDN based MTD
- Game Theoretic MTD Models
- Evaluation of MTD Frameworks

FOUNDATION OF MOVING TARGET DEFENSE

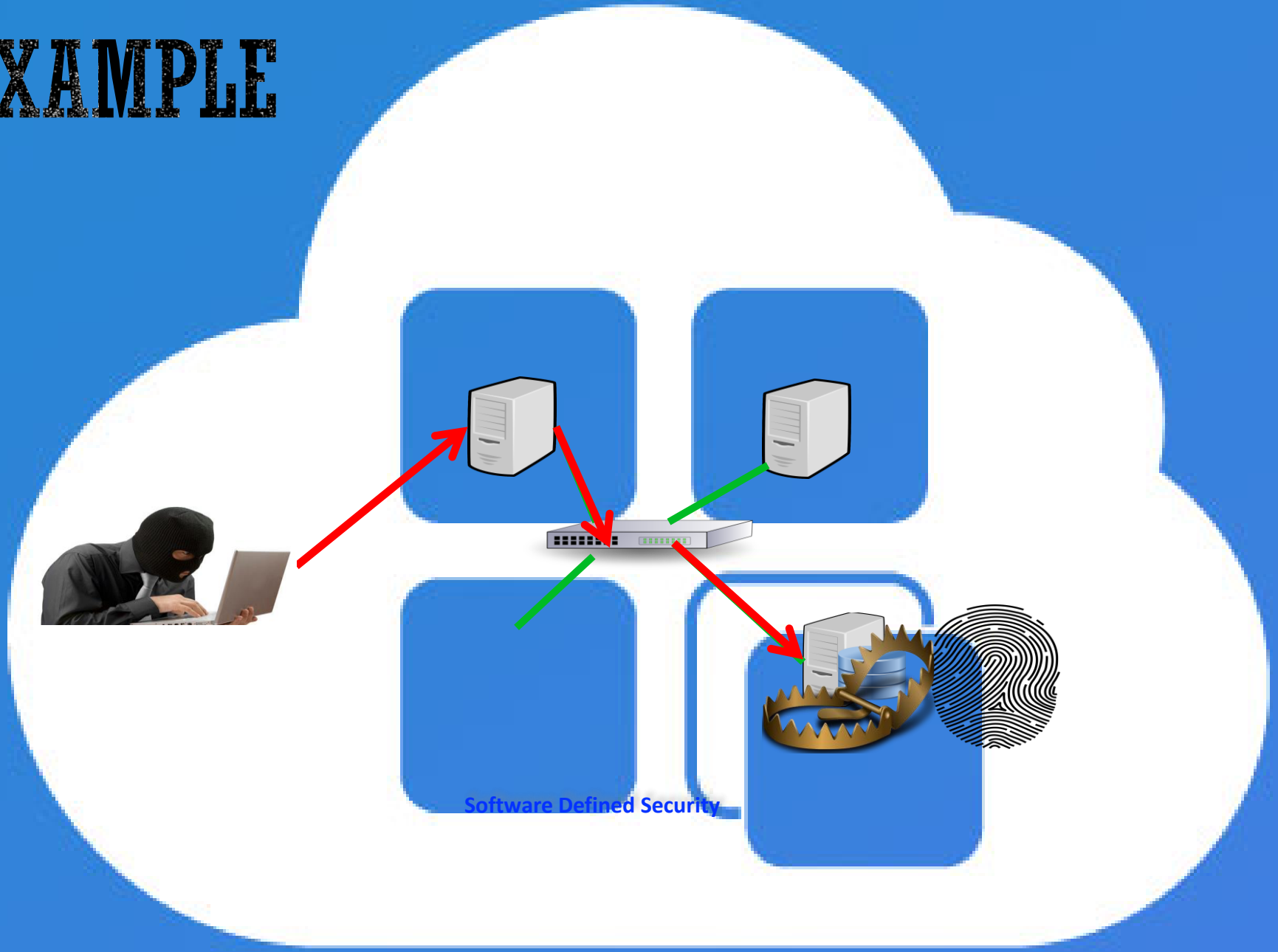


STATIC TARGET

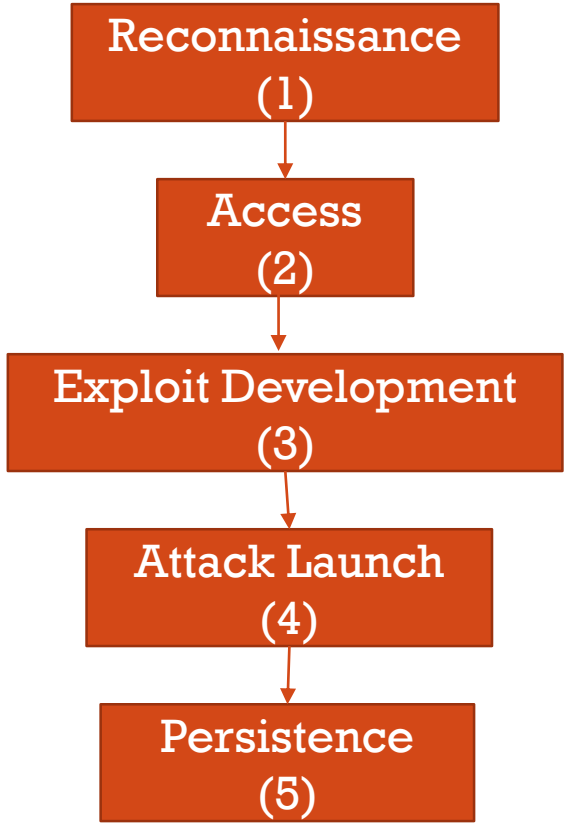
VS

MOVING TARGET

MTD EXAMPLE



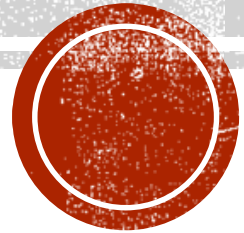
MTD AREAS: CYBER KILL CHAIN MODEL



- The attacker collects useful information about the target.
- The attacker tries to connect or communicate with the target to identify its properties (versions, vulnerabilities, configurations, etc.).
- The attacker develops an exploit for a vulnerability in the system in order to gain a foothold or escalate his privilege.
- The attacker delivers the exploit to the target. This can be through a network connection, using phishing-like attacks, or using a more sophisticated supply chain or gap jumping attack (e.g., infected USB drive).
- The attacker installs additional backdoors or access channels to keep his persistence access to the system.

MTD CLASSIFICATION

Security Modeling based MTD, Protocol Layer based MTD



SECURITY MODELING BASED MTD

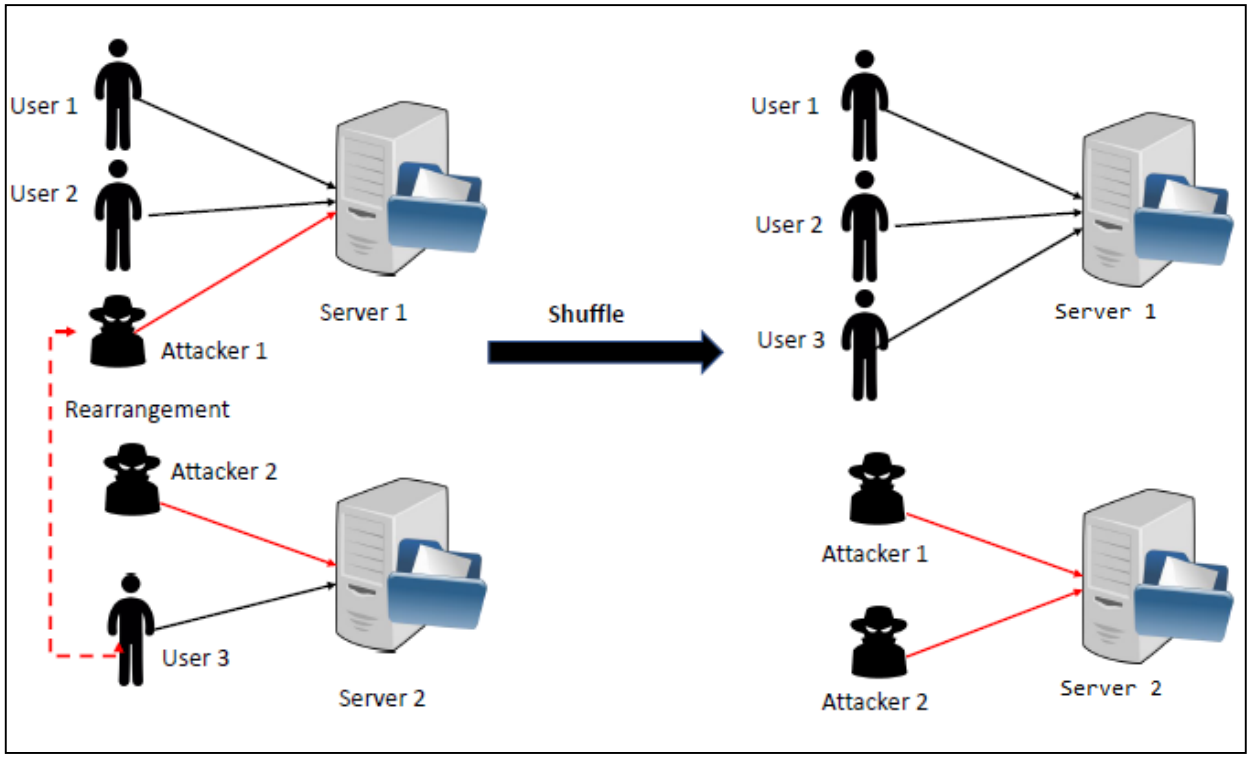
MTD techniques based on security modeling can be classified into:

- Shuffle
- Diversity
- Redundancy

MTD SHUFFLE

- Rearrangement of resources at different layers of protocol stack, e.g., application migration, VM migration, etc.
- An administrator can shuffle the connection for different users connected to the servers.

MTD SHUFFLE

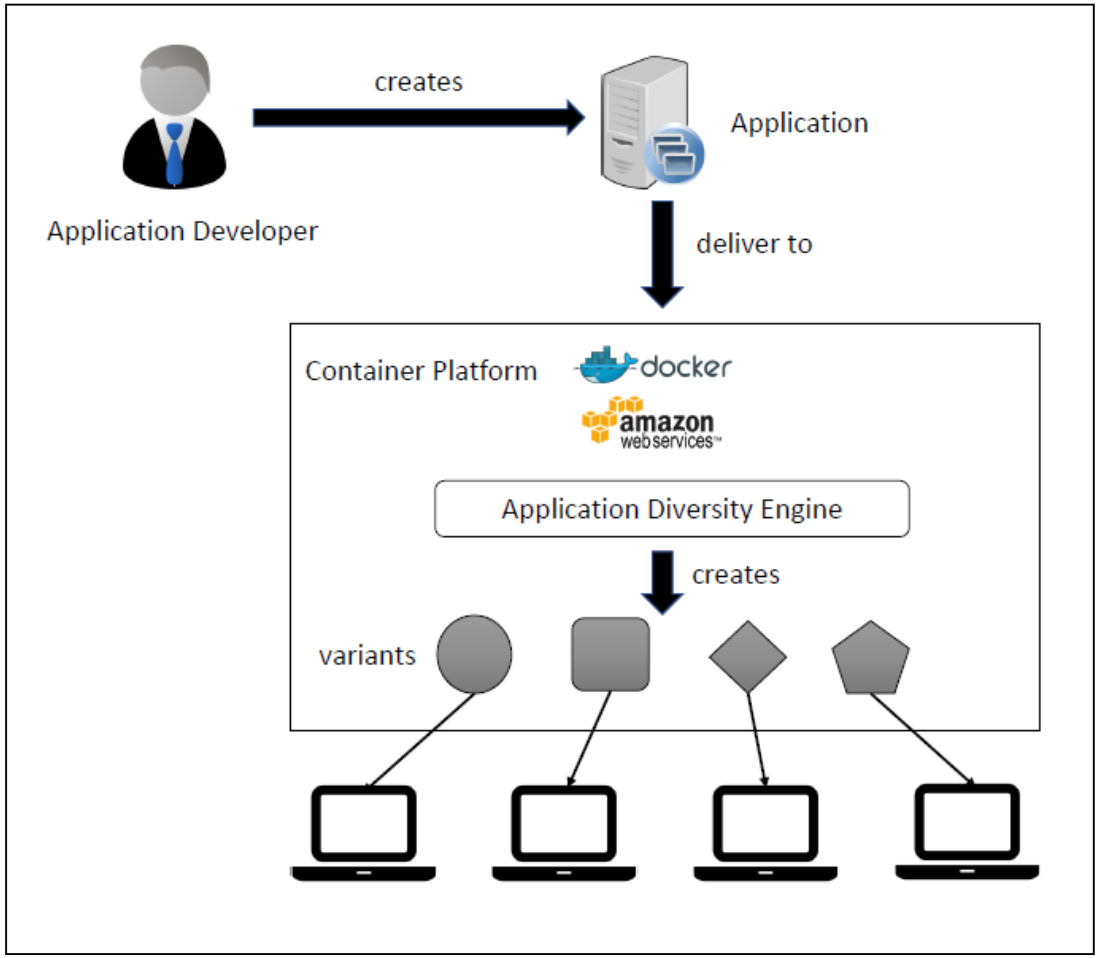


- Both the attackers are connected to Server 2 in the new configuration.
- Shuffling will slow down the attack propagation.

MTD DIVERSITY

- Security attacks like Code Reuse Attacks (CRAs) target known library code and jump-oriented programming techniques.
- Attackers can reverse-engineer the target binary in order to identify the regions of code with vulnerability.
- If the target binary is different, gadget blocks identified with vulnerability become useless on the other binary.
- MTD diversity technique replaces the network function or software targeted by the attacker with an alternate version of software/function having same functionality.
- Diversification can also be applied to Address Space Layout Randomization (ASLR) and Instruction Set Randomization (ISR).

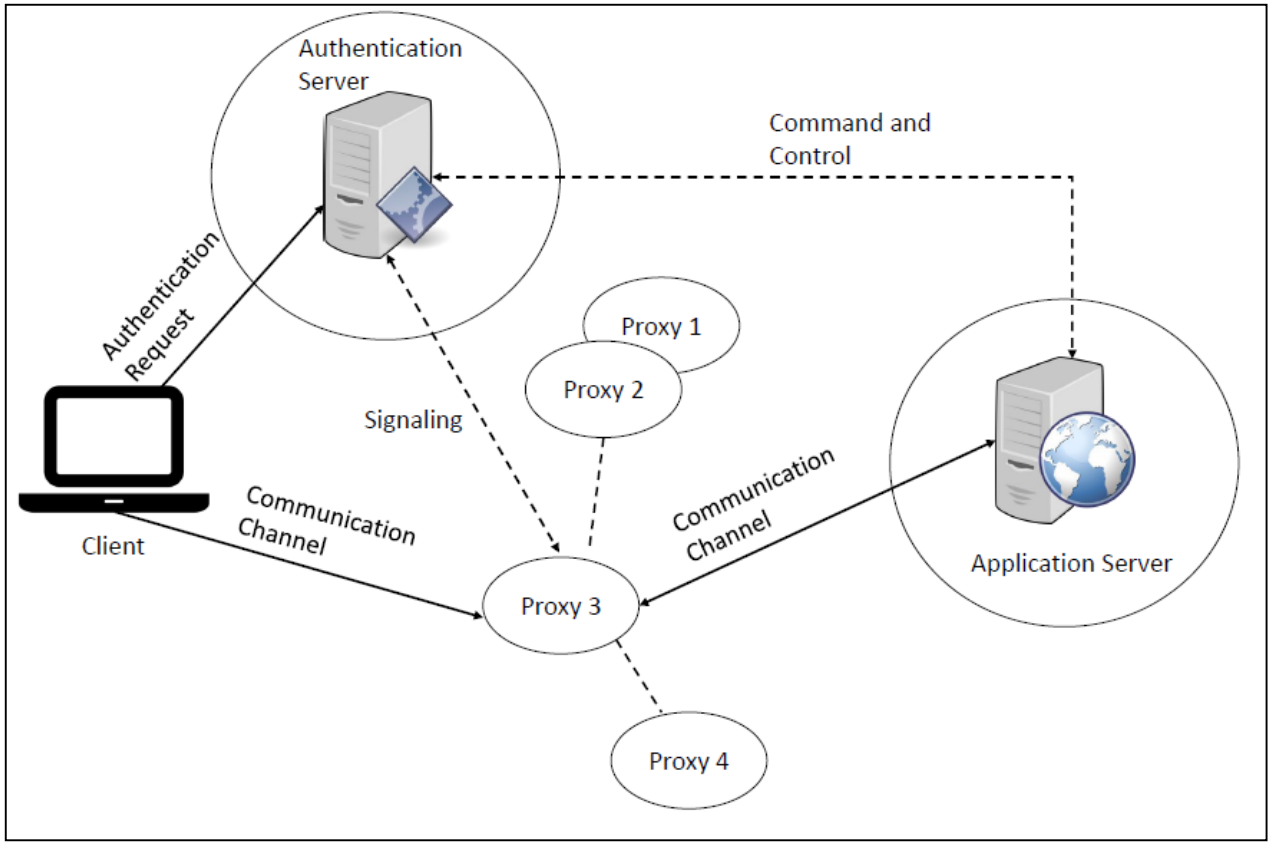
MTD DIVERSITY



REDUNDANCY

- Creates multiple replicas of the target component, in order to maintain an optimal level of service in the case of attack events.
- The goal of the attacker in case of attack vector like DDoS is saturation of network resources.
- With redundancy the attack-goal becomes harder to achieve, since administrator can load-balance the traffic to different proxies in case of traffic surge.

REDUNDANCY



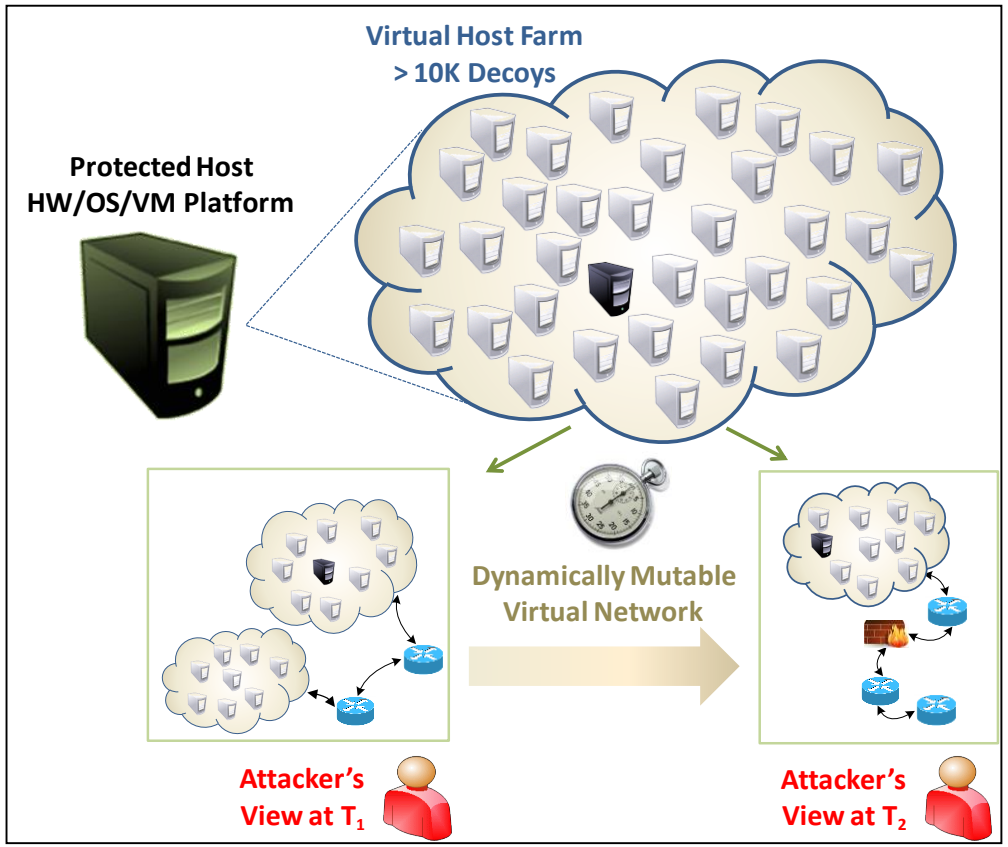
- Layer of proxies is created between the client and the target application.
- Actual IP address of the application server is concealed from the client.
- Once authentication server verifies the client, client can connect to application server via one of the available proxies.

PROTOCOL LAYER BASED MTD

MTD techniques based on IP protocol layers can be classified into:

- Network Level MTD
- Host Level MTD
- Application Level MTD

NETWORK LEVEL MTD

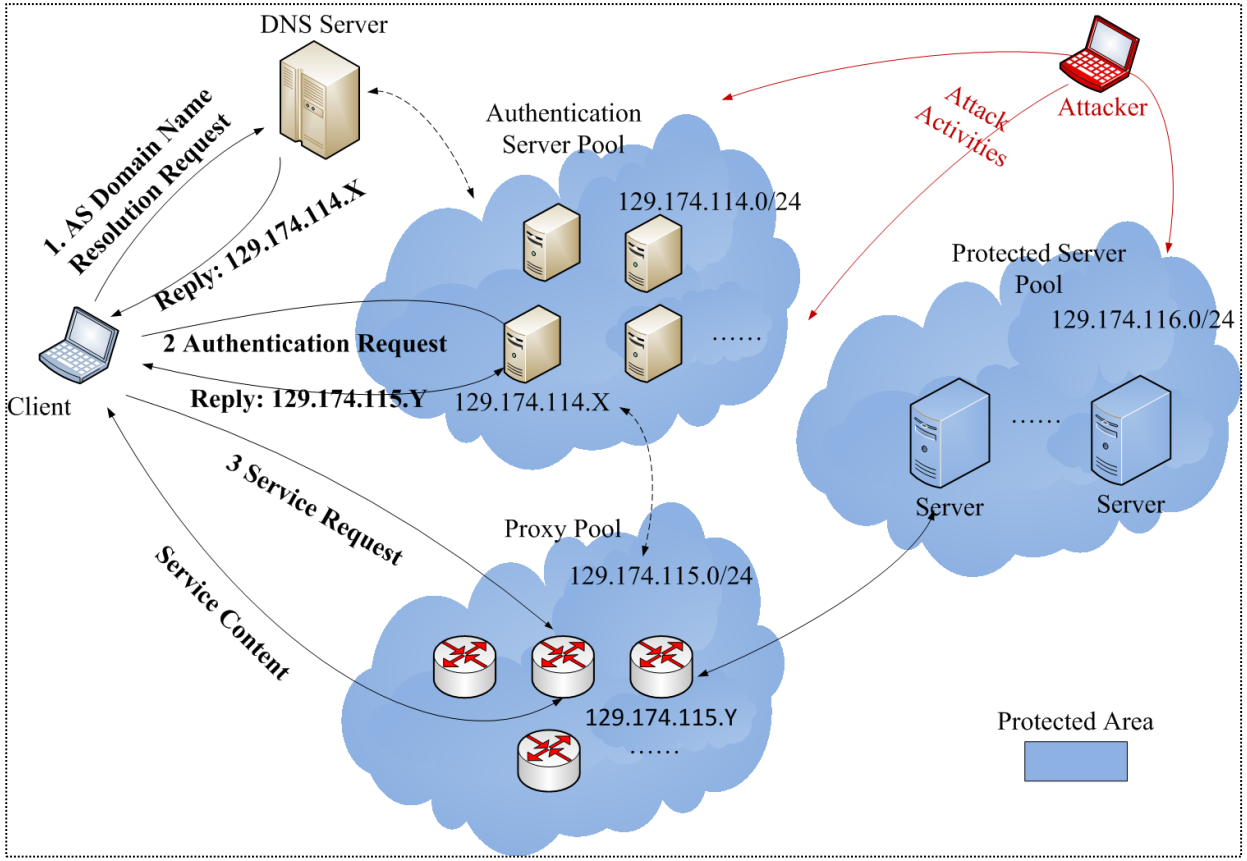


- Change in network parameters.
- Network Reconfiguration, Network Connection Obfuscation, Routing Randomization.
- Key security issues such as security vulnerability exploits, zero-day attacks can be mitigated using NMTD.

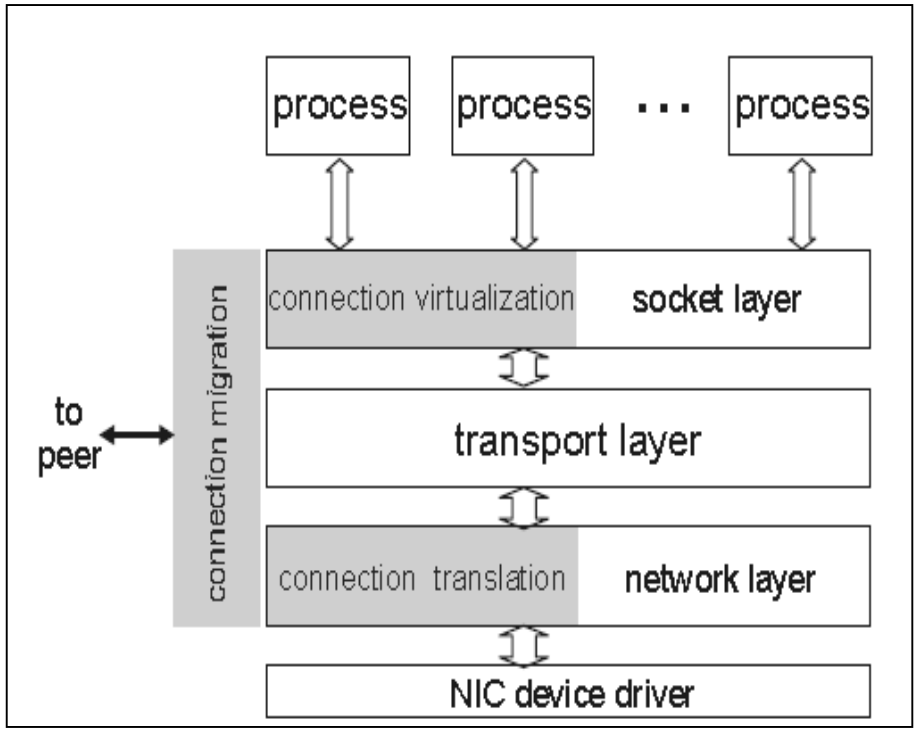
NMTD CHALLENGES

1. Service availability
   - Authenticated clients should always know the new IP address/port number.
   - When the IP and Port changes, the connection still maintained, minimizing service downtime.
2. Service Security
   - Only the authenticated users can access the service.
   - How to mitigate insider attacks?

AUTHENTICATION FRAMEWORK



SEAMLESS TCP CONNECTION MIGRATION



- Keep end-to-end transport connection alive through separating transport endpoint identification from network endpoint identification.

- Three components
  - Connection virtualization
  - Connection translation
  - Connection migration

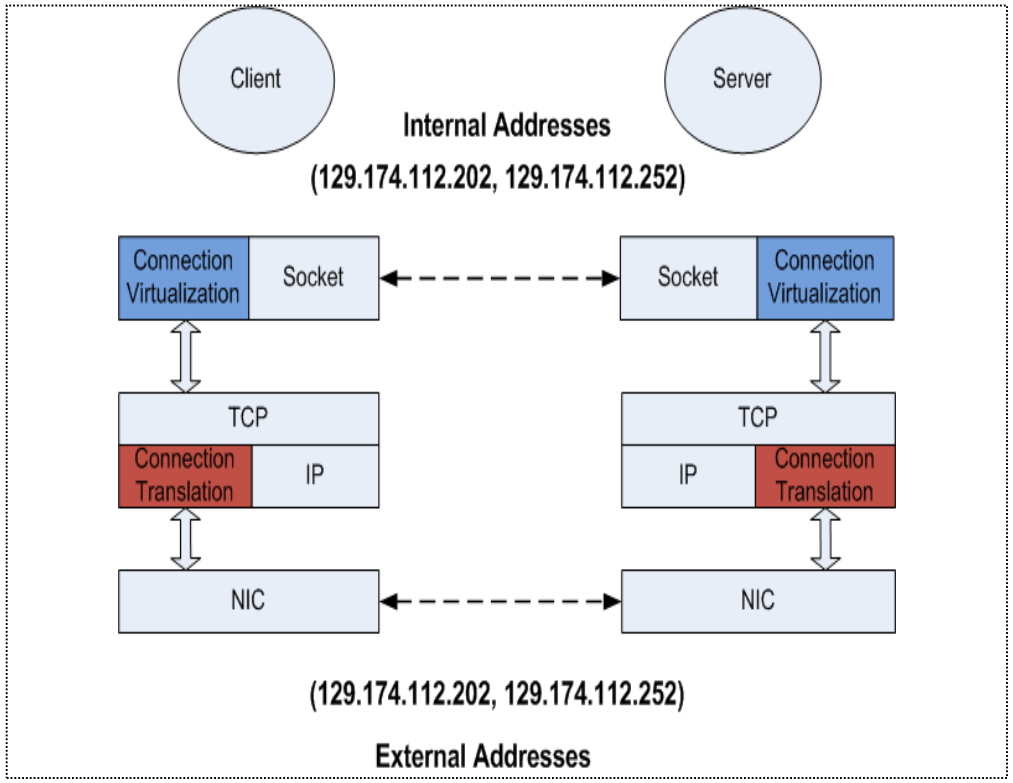
CONNECTION VIRTUALIZATION

- Internal address for applications;
  - IP and Ports
  - never changes for one connection
- External address for communications
  - IP and Ports
  - may change according to MTD requirements
- A map to translate between Internal address and External address

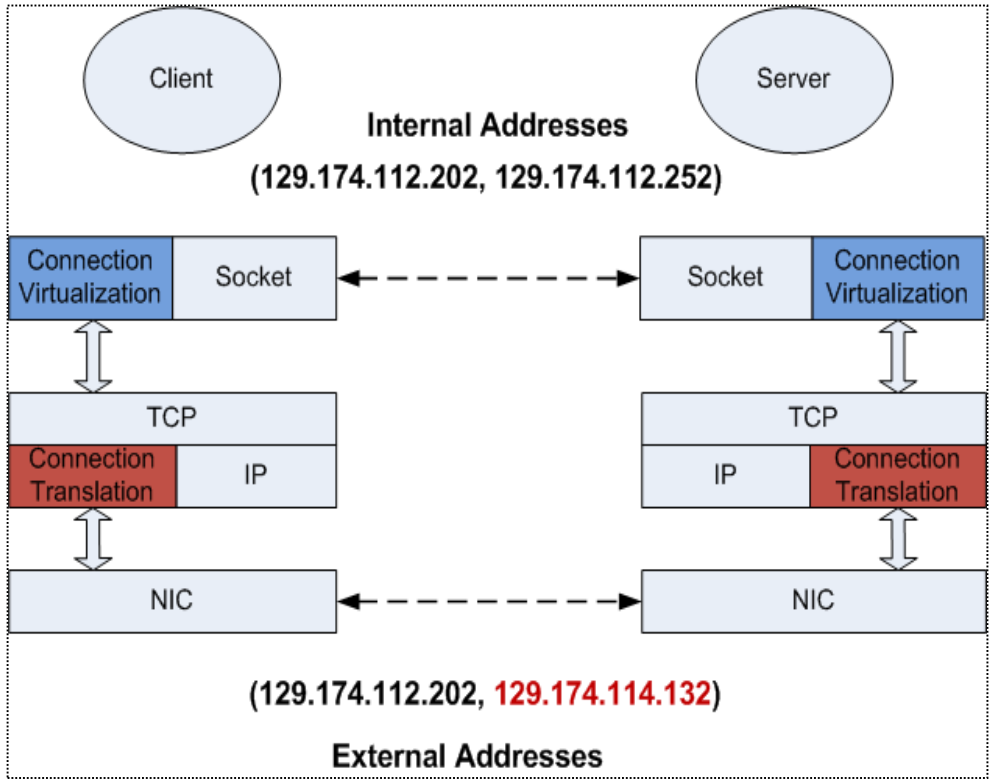
CONNECTION TRANSLATION

At beginning,

internal address == external addresses



Server changes its IP address



CONNECTION MIGRATION

- After the server changes its IP address and port, it will inform the client to update the internal-external address mapping.
- Migration Steps: protected by a shared secret key
  - Suspend a connection
    - Keep connection alive
  - Resume a connection
    - Update internal-external endpoints mappings
    - Server sends UPDATE packet
    - Client sends UPDATE_ACK packet
- Both endpoints need to know the same internal address pair.

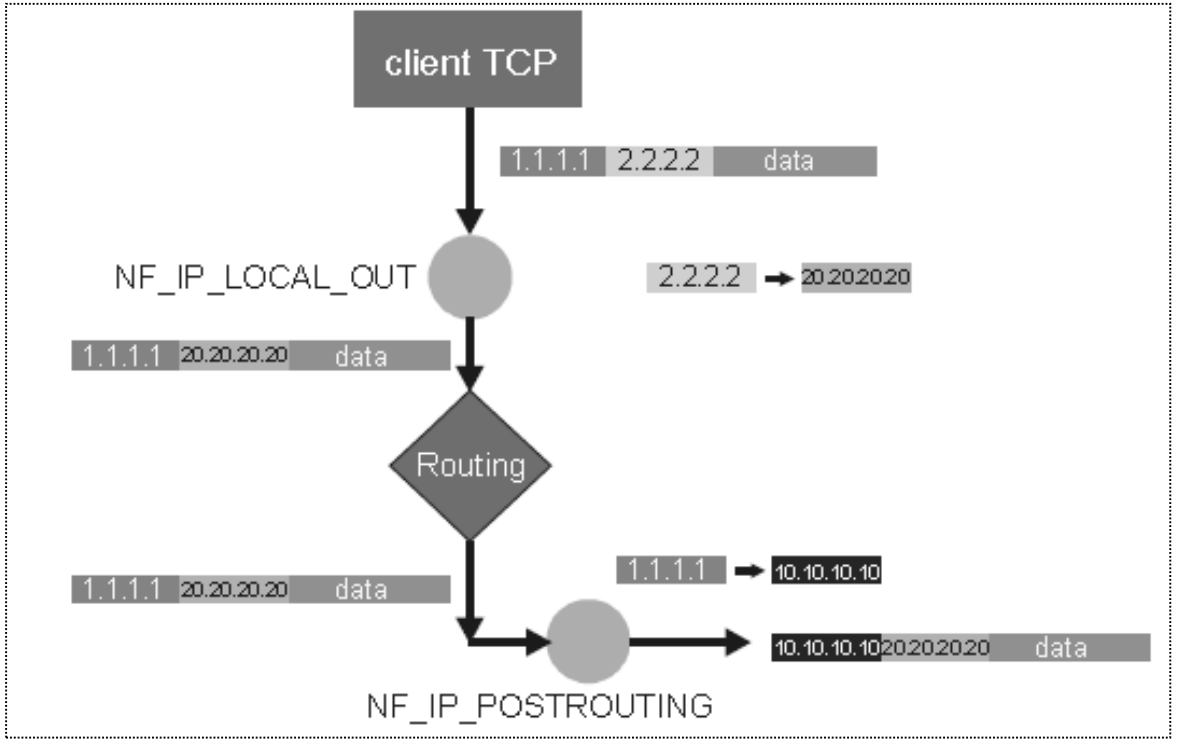
IMPLEMENTATION

- All in a kernel module in Linux
- Support both client and server mobility
- Connection Virtualization
  - Intercept socket system calls
- Connection Translation
  - Instrument Netfilter hooks
- Connection Migration
  - Migration daemon

INTERCEPT SYSTEM CALLS

- Overwrite the function pointers in the system call table
- Intercept
  - Accept()
  - Connect()
  - Close()
  - Getsockname()
  - Getpeername()

INSTRUMENT NETFILTER HOOKS

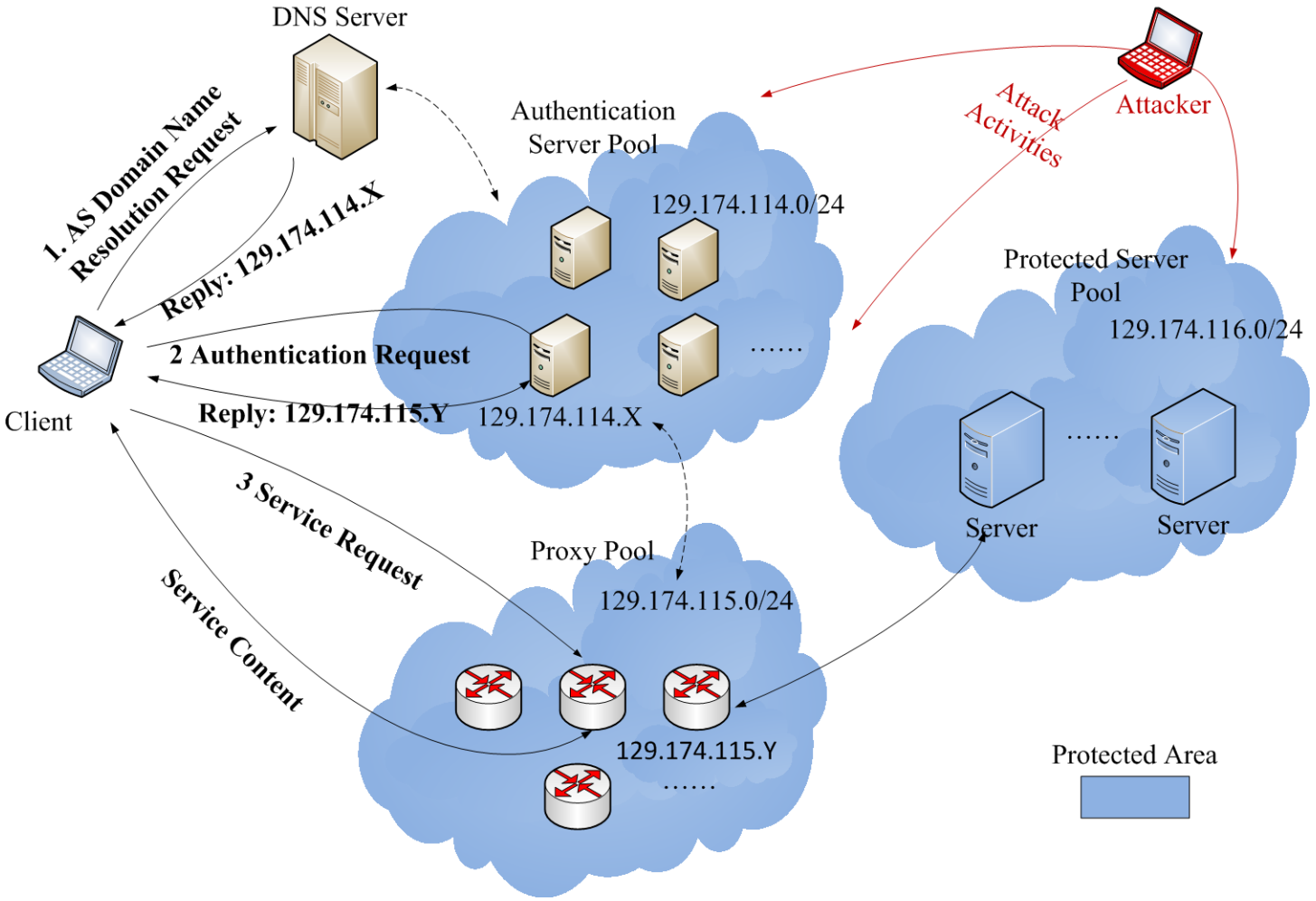


- For outgoing traffic
  - NF_IP_LOCAL_OUT for destination address translation
  - NF_IP_POSTROUTING for source address translation
- For incoming traffic
  - NF_IP_PREROUTING for destination address translation
  - NF_IP_LOCAL_IN for source address translation

MIGRATION DAEMON

- A Kernel thread as a server process
- Initiate the suspension after receiving a suspend event from an Access Policy Manager (APM)
  - Active the connection migration helper
- Restore the connection after receiving a resumption event from APM
  - Exchange UPDATE and UPDATE_ACK packets to update the internal to external address mapping

CASE STUDY: MOBILE VPN



FAST AND LIGHTWEIGHT VPN SESSION RESUMPTION

- In a traditional VPN (like OpenVPN), both the VPN client and server are expected to maintain the same physical IP address during the entire VPN session.
- In mobile environment, mobile clients will most likely obtain new IP addresses after a handover.
- VPN server's IP address can change too in MTD environments (IP hopping).

FAST AND LIGHTWEIGHT VPN SESSION RESUMPTION

- OpenVPN ties each VPN session with the IP address of the VPN client, and uses that IP as the identifier of the VPN session.
- Change of the IP address of the client → inability to locate the VPN session the VPN server → A new VPN session has to be renegotiated.

NETWORK HANDOVER DELAYS

- OpenVPN

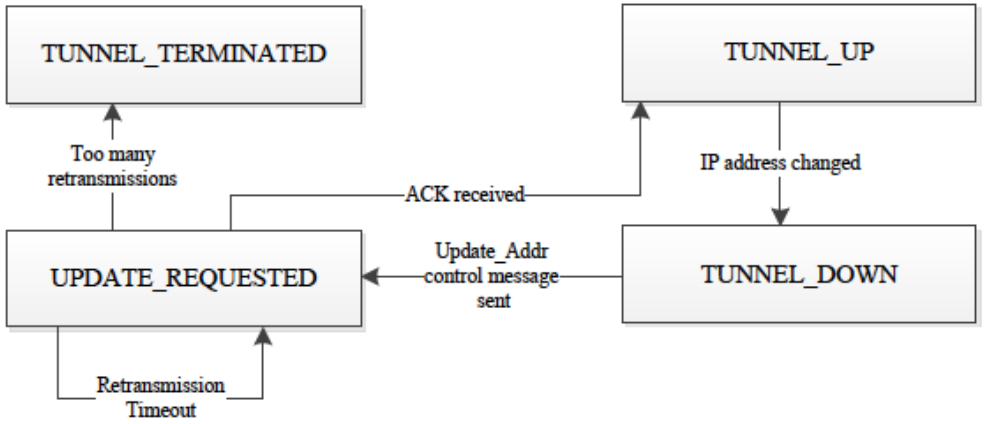
Delay	Network Handover (L2/L3)	VPN session termination	VPN renegotiation
Causes	Joining Network, authentication, DHCP etc	Inactivity Timeout (60s recommended) (Controlled by VPN server)	Full TLS handshake, pushing VPN configuration, pushing routing options etc.

- MobiVPN

Delay	Network Handover (L2/L3)	VPN resumption
Causes	Joining Network, authentication, DHCP etc	Light-weight VPN resumption

EFFECT OF VPN SERVER'S IP ADDRESS CHANGE

- OpenVPN:
  - The VPN server is Passive. Connected clients have to timeout first, before they can renegotiate new VPN sessions.



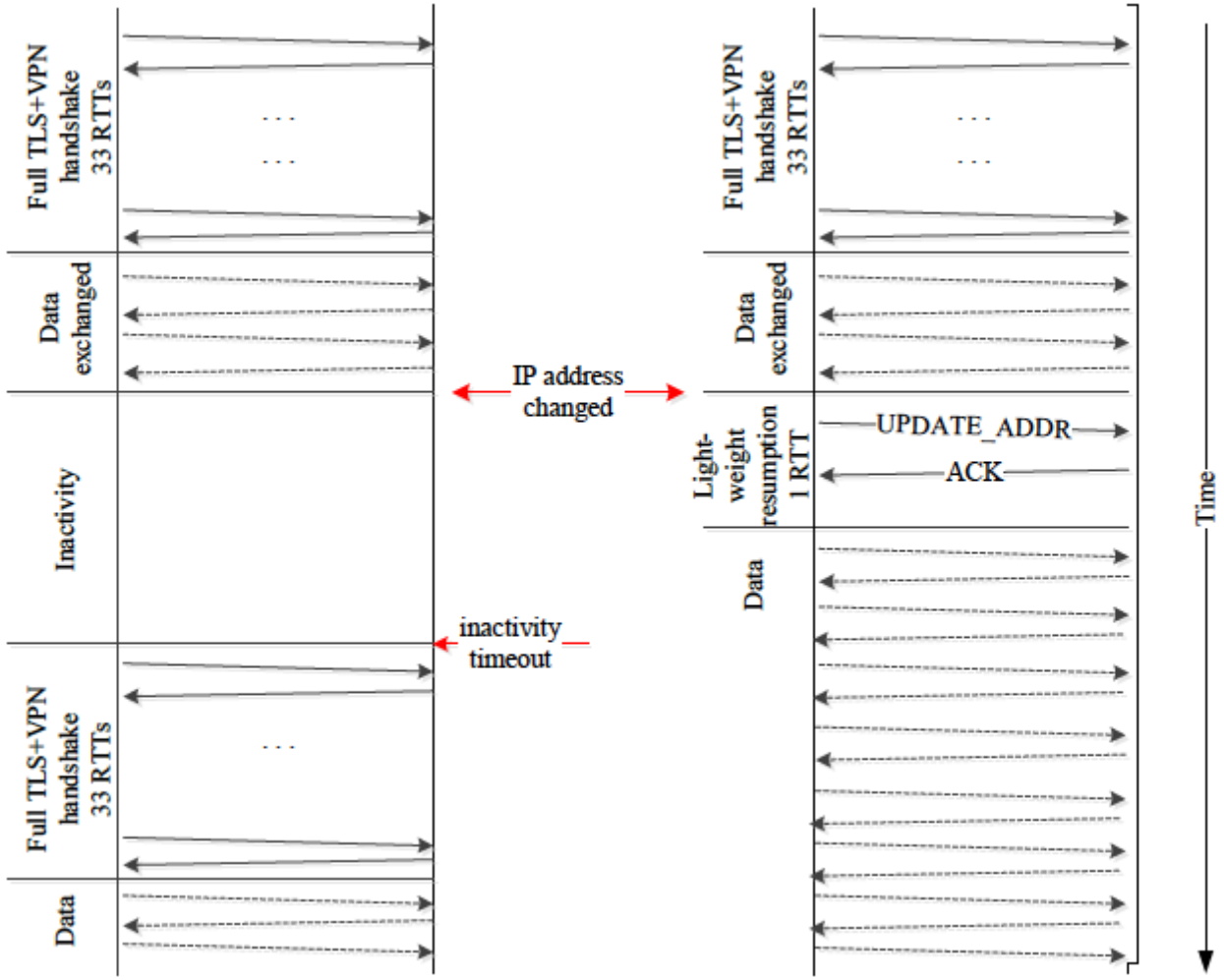
- MobiVPN
  - The VPN server is Active. It can uses our light-weight VPN resumption protocol to update all connecting VPN clients with its new IP address eliminating the need to renegotiate new VPN sessions.

MOBIVPN HASH TABLES

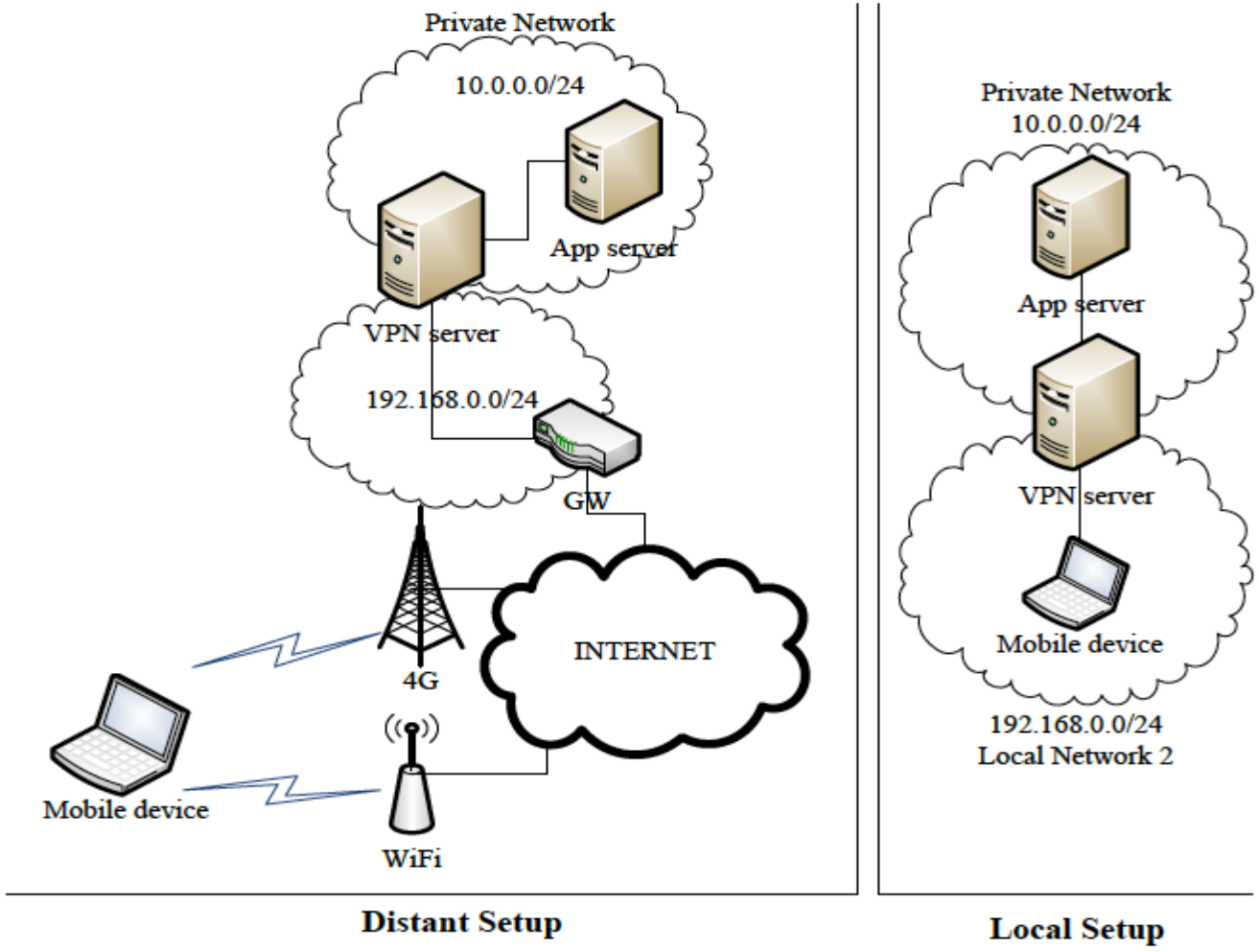
IP	VPN Instance
192.168.100.2	struct VPN_instance { .. }
192.168.100.10	struct VPN_instance { … }
….	…..

Session ID	IP
XYZ123	192.168.100.2
ABC456	192.168.100.10
….	…..

OPENVPN VS MOBIVPN

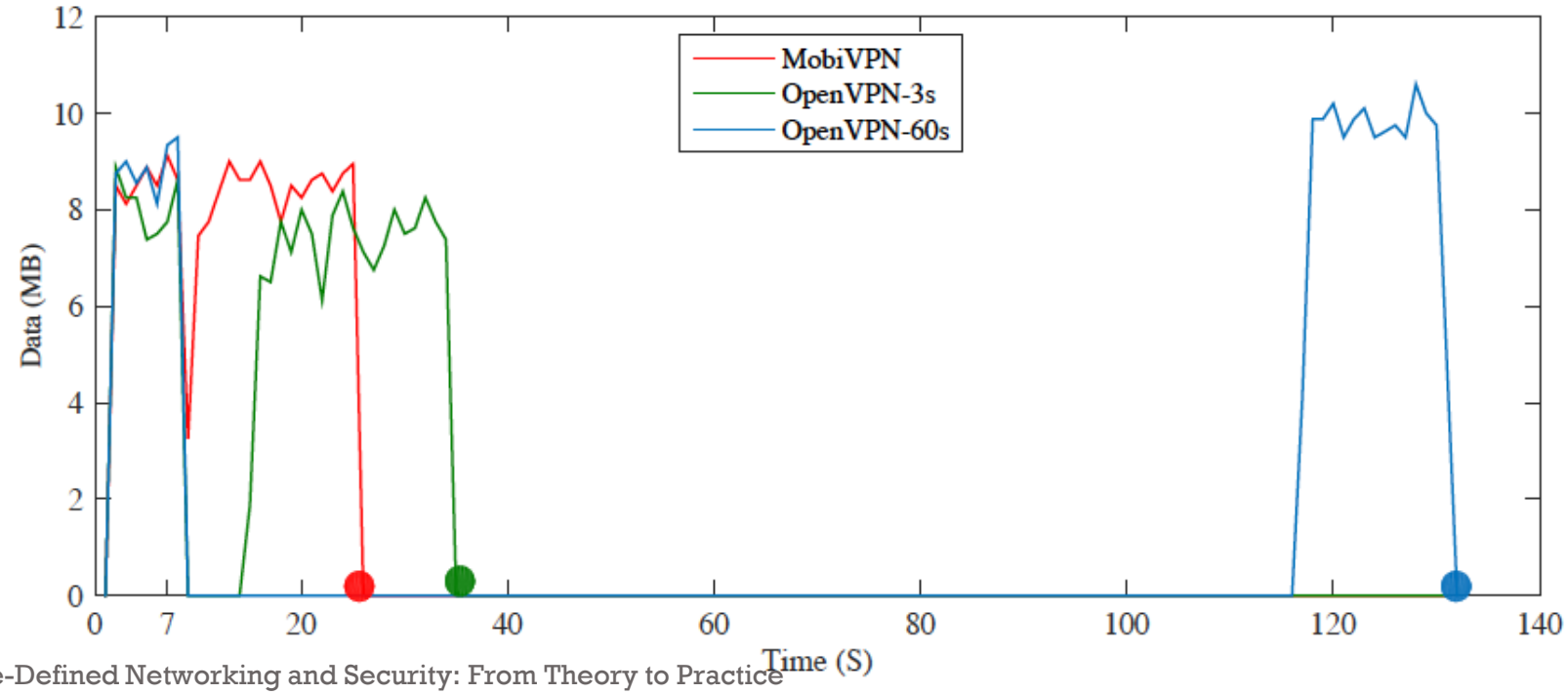


EVALUATION TESTBED



EFFECT OF FAST VPN RESUMPTION ON DATA TRANSFER - LOCAL TESTBED

- iperf is used to transmit 200MB of data between mobile device and application server through the VPN tunnel.
- IP of physical NIC was changed after 7 seconds.



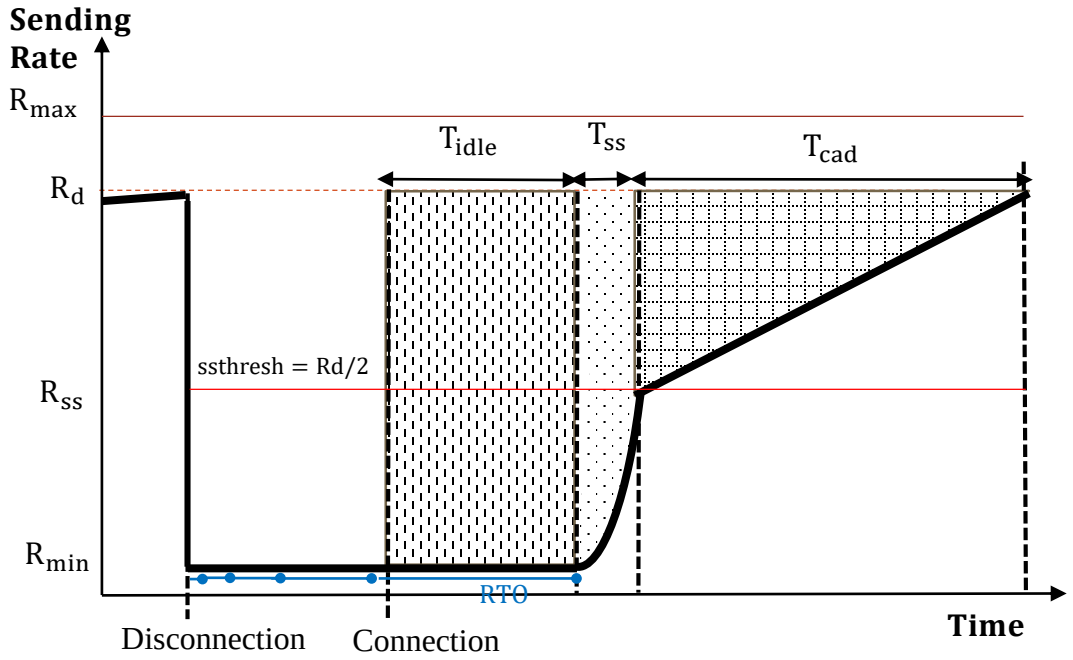
PERFORMANCE MEASURES WHEN THE VPN CLIENT CHANGES ITS IP ADDRESS

Testbed	Measured Metric	MobiVPN	OpenVPN -3s	% Decrease by MobiVPN	OpenVPN -60s	% Decrease by MobiVPN
Local	VPN Unavailability Time	214 ms	4.498 s	95.24%	61.711 s	99.65%
	VPN Session Resumption Time	8 ms	2.409 s	99.67%	2.412 s	99.67%
	Total Time To Resume VPN	222 ms	6.907 s	**96.79%**	64.123 s	**99.65%**
	Data Transfer Time (200MB)	24 s	33.1 s	**27.49%**	130.1 s	**81.55%**
Distant	VPN Unavailability Time	512 ms	4.710 s	89.13%	61.927 s	99.17%
	VPN Session Resumption Time	46 ms	3.462 s	98.67%	3.447 s	98.67%
	Total Time To Resume VPN	558 ms	8.172 s	**93.17%**	65.374 s	**99.15%**
	Data Transfer Time (40MB)	87 s	104 s	**16.35%**	207 s	**57.97%**

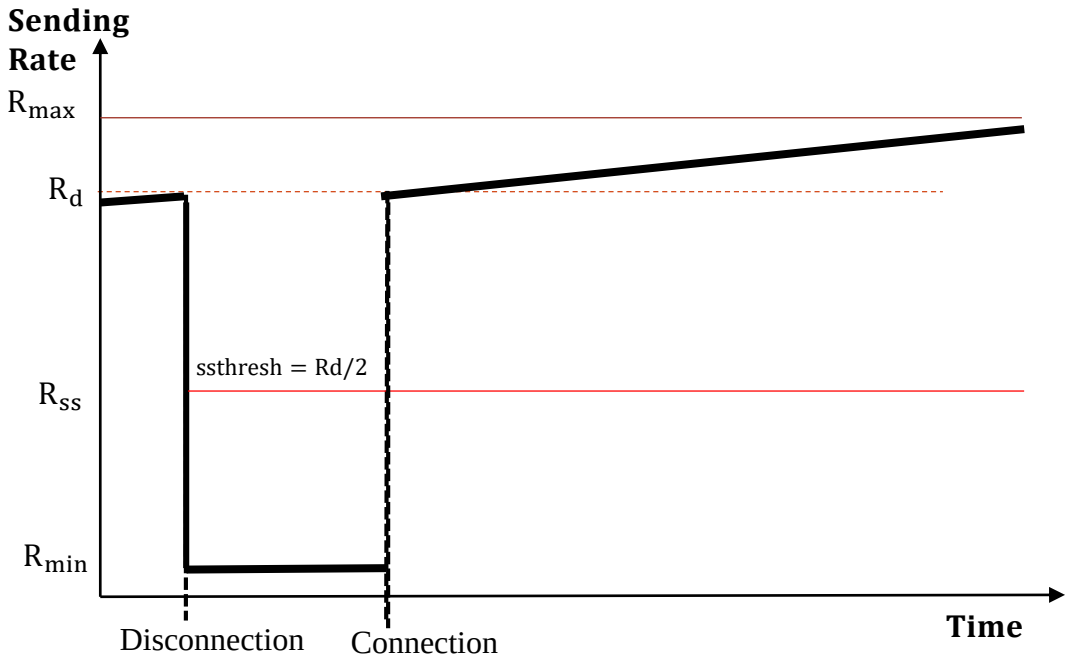
PERSISTENCE AND FAST RESUMPTION OF TCP BASED APPLICATIONS

- Problem?
- Application sessions in the mobile node must be kept alive by hiding the VPN tunnel breakage and re-establishment from the applications.

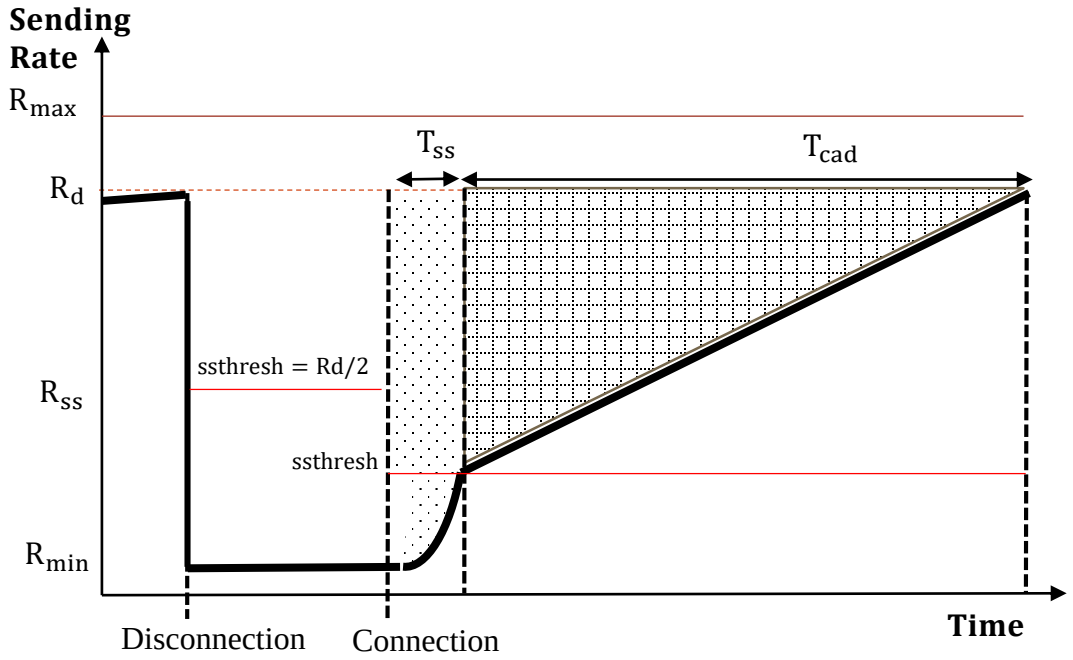
EFFECT OF MOBILITY ON TCP RENO SENDING RATE



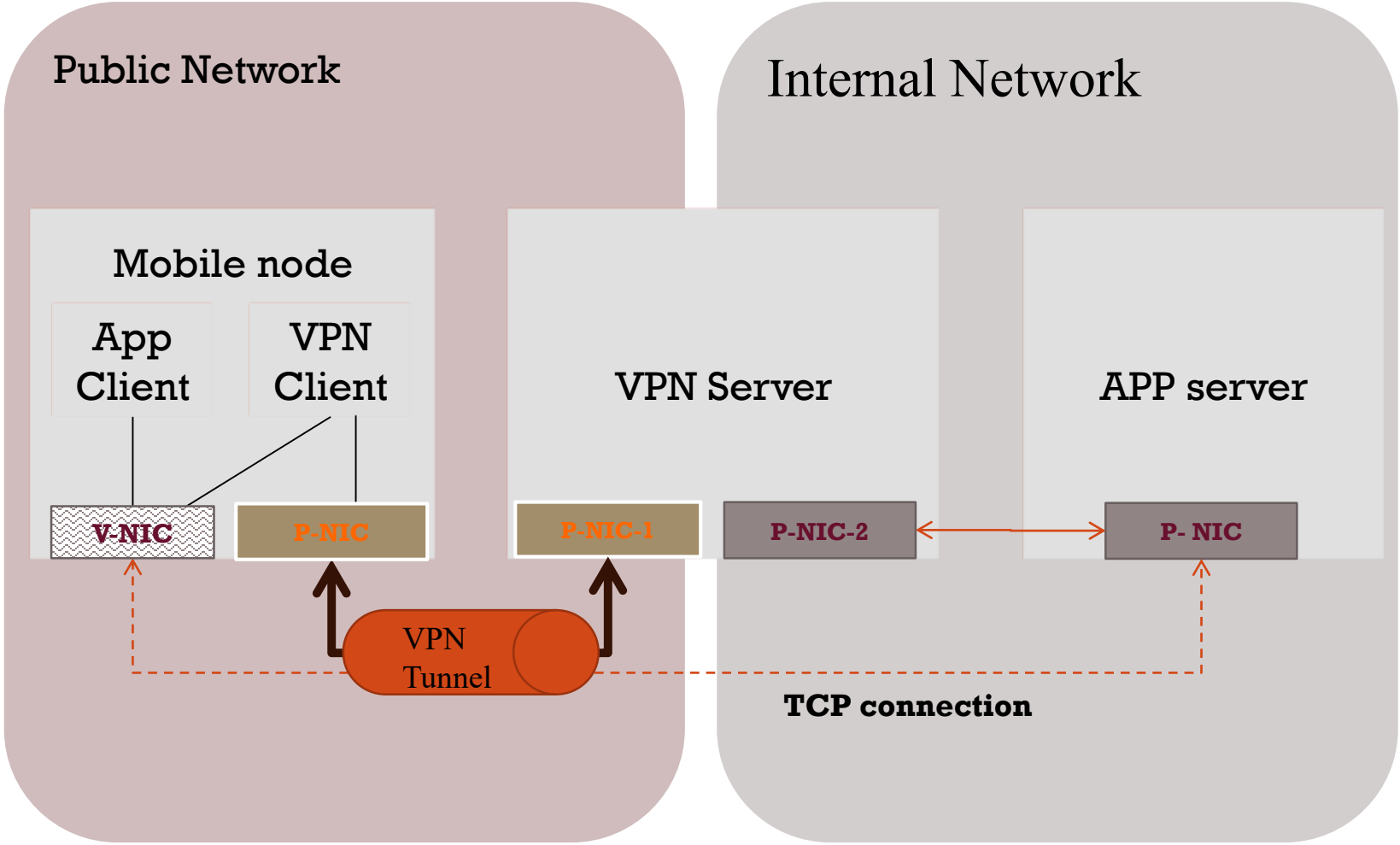
TCP SENDING RATE – MOBIVPN WITH BUFFERING



TCP SENDING RATE – MOBIVPN WITHOUT BUFFERING



MOBIVPN APPROACH



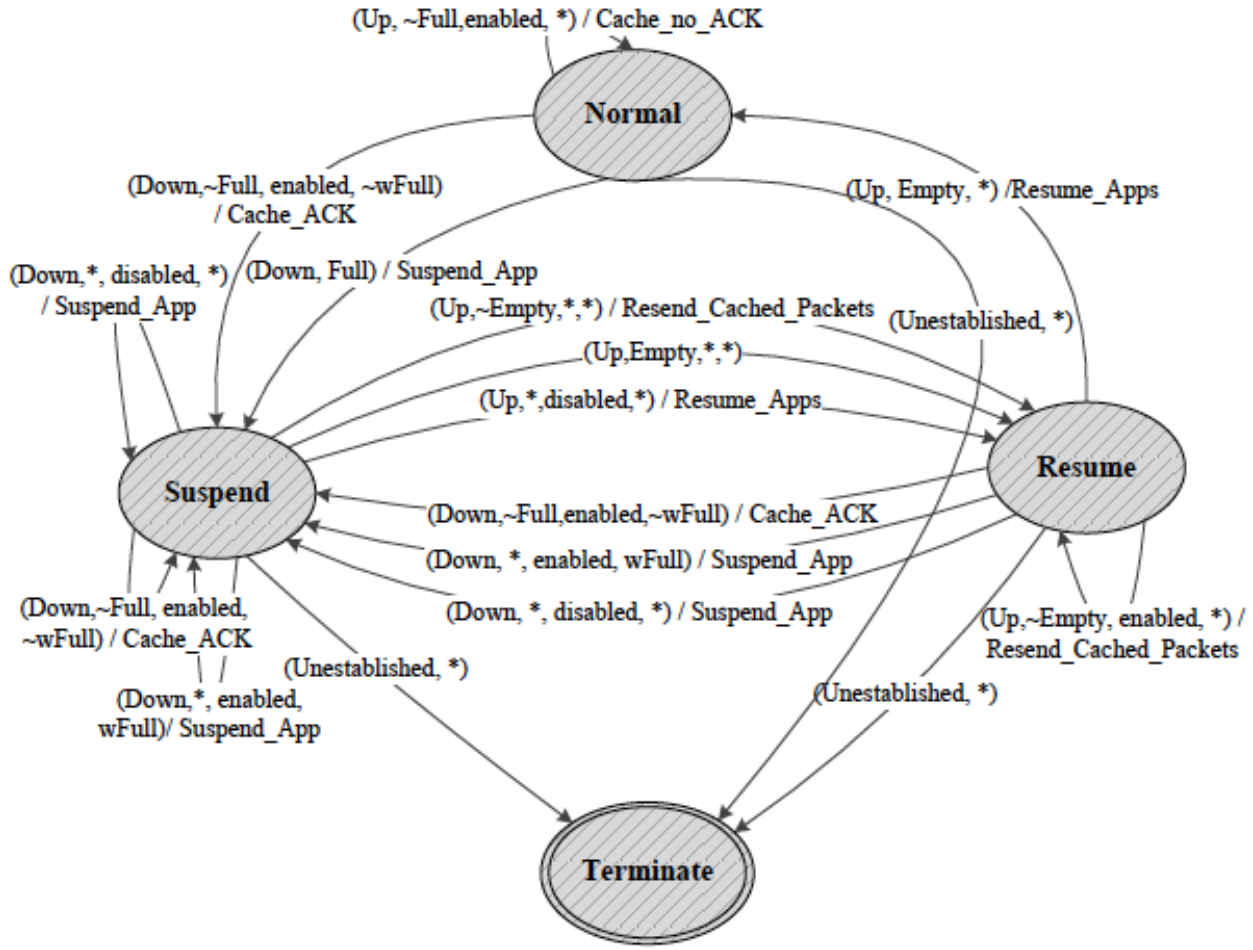
MOBIVPN SYSTEM MODEL

- ***Modeled as a Finite State Transducer with these states:***
- ***Normal state****:* the VPN tunnel is healthy and the applications' TCP sessions behave as normal.
- ***Suspend state****: when* VPN tunnel fails due to network disconnection.
  - MobiVPN suspend non-buffered flows.
  - For buffered flows, MobiVPN buffers and acknowledges packets from the applications,
  - and eventually suspends the application sessions when the buffer is full by sending a Zero-Window packet,
  - This causes TCP to suspend its timers preventing it from terminating the TCP socket.
  - It also prevents TCP sockets from dropping the congestion window, avoiding slow-start upon reconnection.

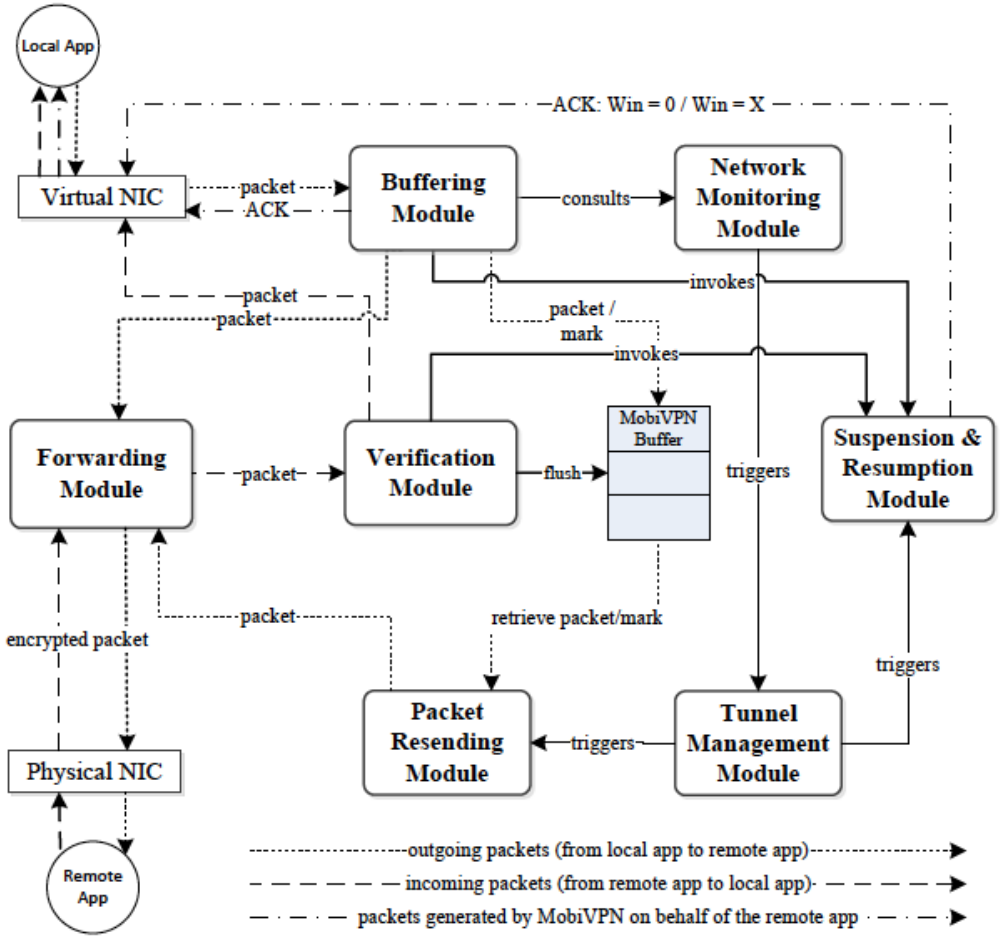
MOBIVPN SYSTEM MODEL

- ***Resume state:*** when the VPN tunnel is restored.
  - Non-buffered flows are resumed using Triple-ACK by the Suspension & Resumption (S&R) Module.
  - The packet resending module sends out buffered packets to the intended recipients until the buffer is cleared .
  - The verification module responsible for intercepting the packets received from the other end and flush the acknowledged packets from the buffer, while forwarding any incoming data to the local app.
  - The S&R module is invoked to resume the flows whose all buffered packets are acknowledged.

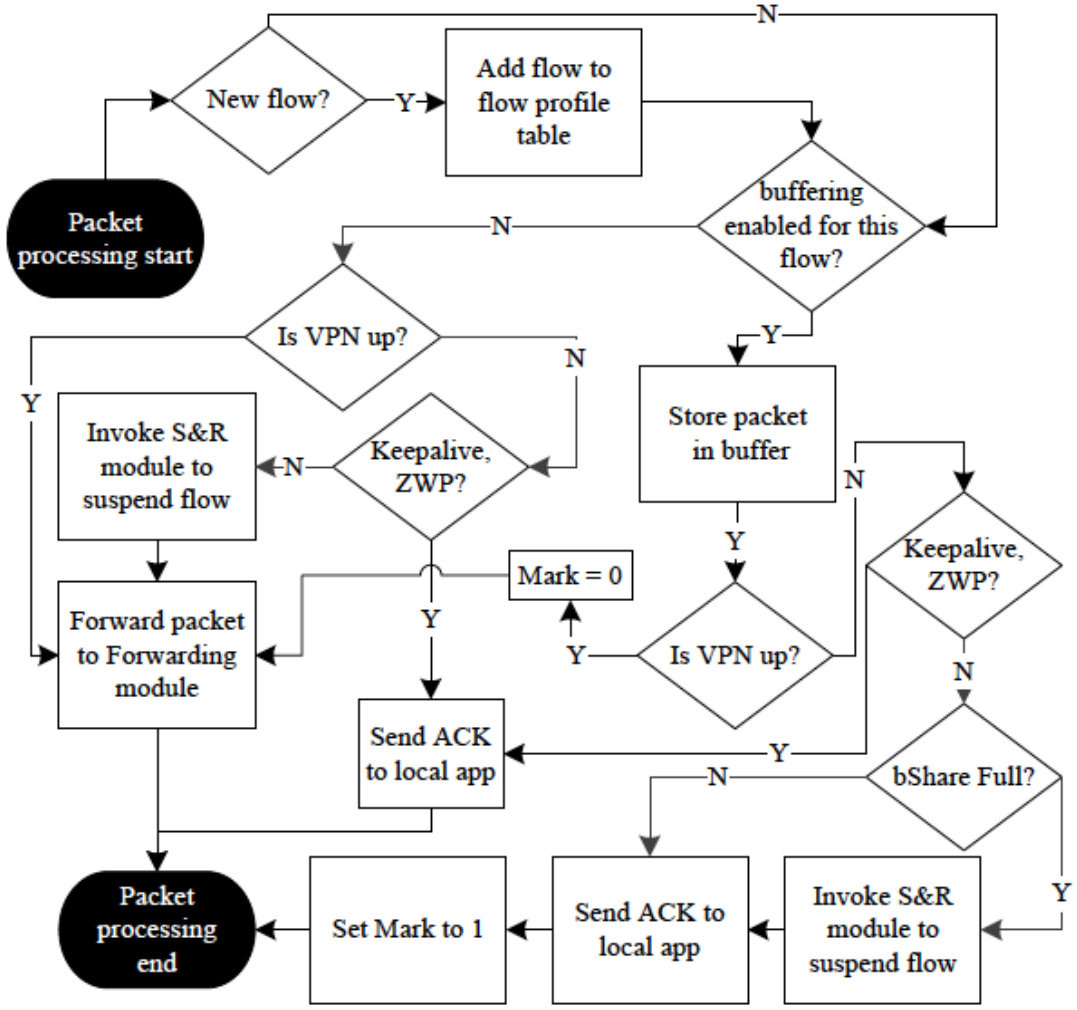
MOBIVPN STATE TRANSITION DIAGRAM



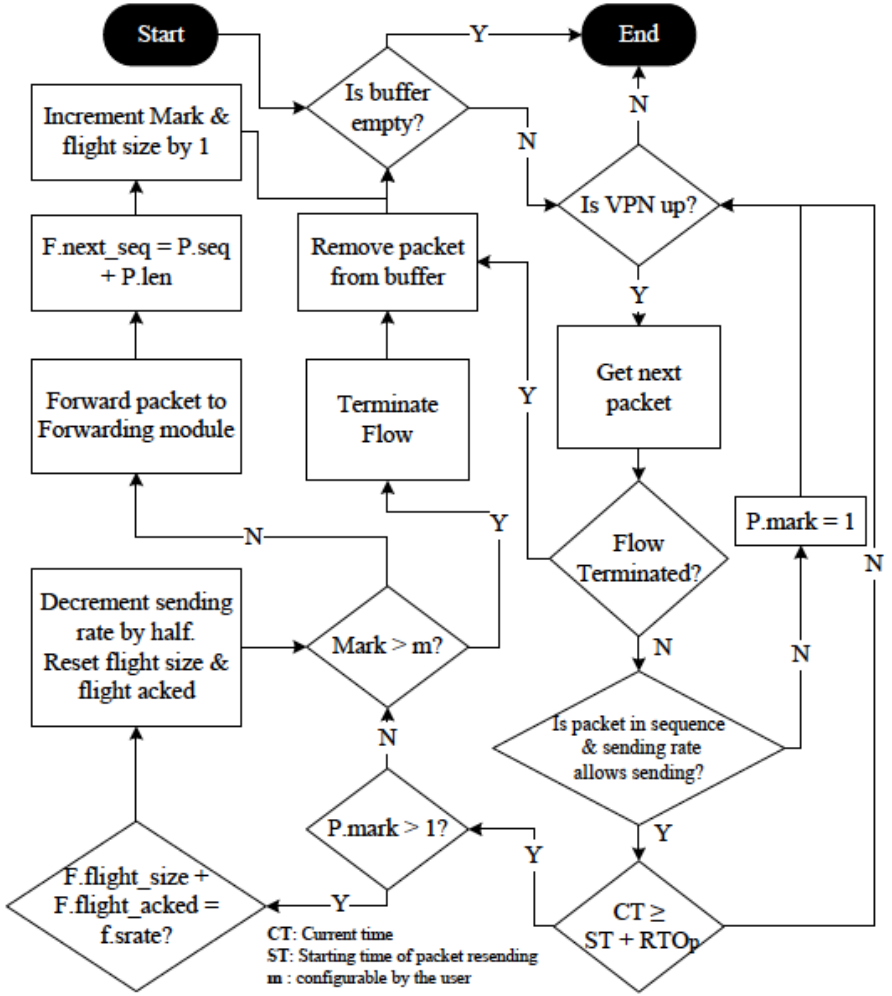
MOBIVPN MODULE RELATIONS



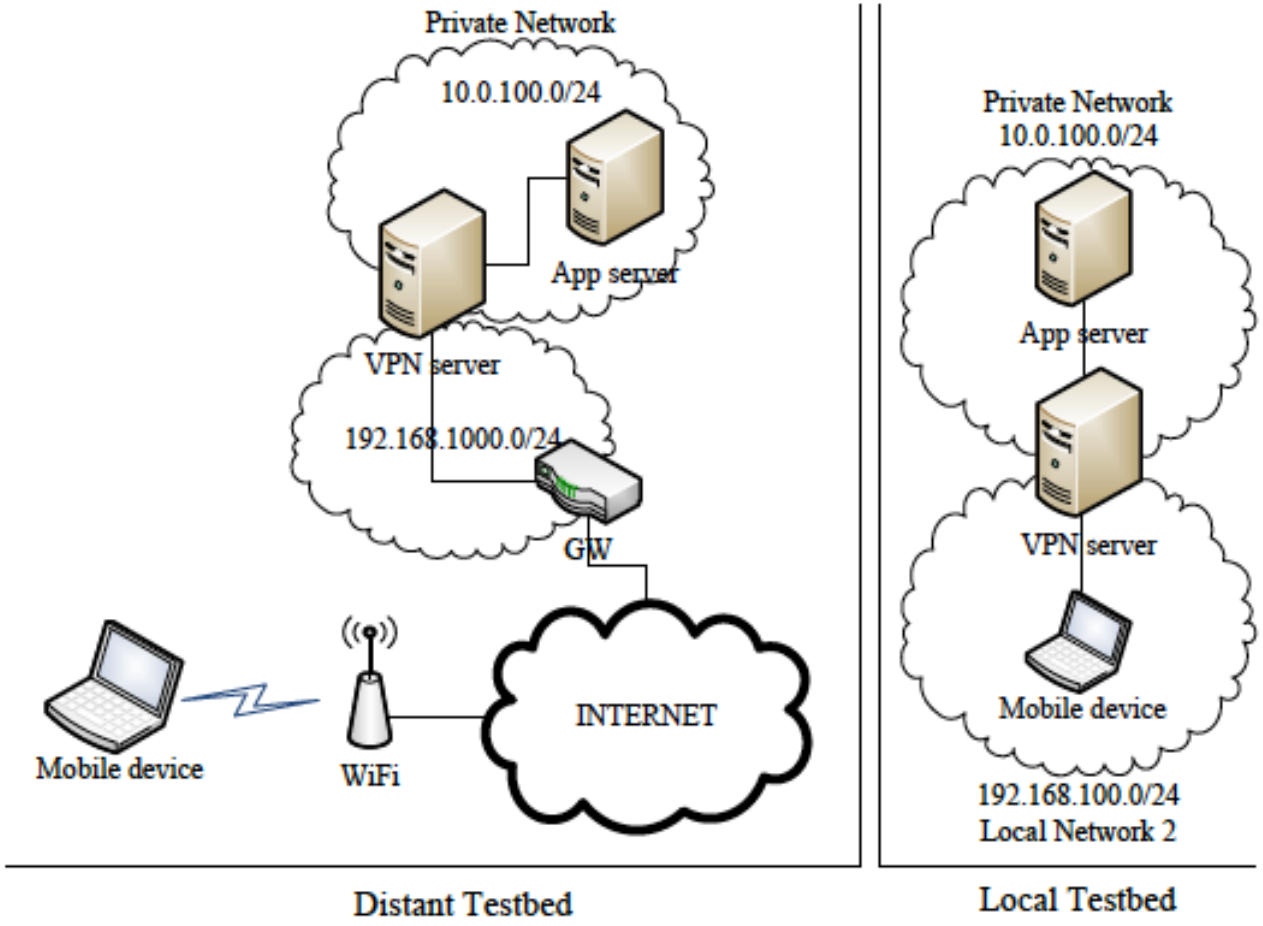
BUFFERING MODULE



PACKET RESENDING MODULE



MOBIVPN EVALUATION TESTBED



PERSISTENCY EVALUATION

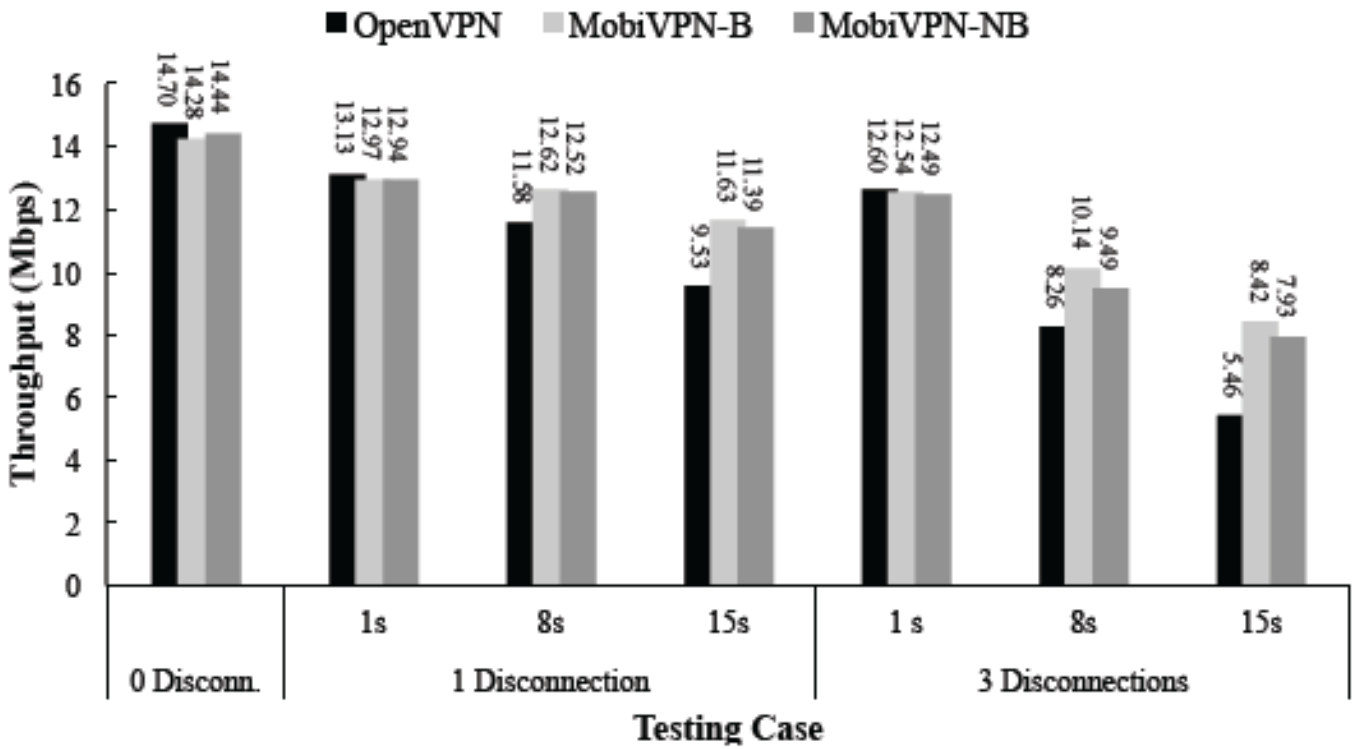
- Transmitted a 1GB text file using a simple file transfer program that uses a TCP socket: *socket(AF_INET,SOCK_STREAM,0)*
- Disconnected the network interface for X seconds after a few seconds from the beginning of file transfer.

Transfer direction	Server's TCP settings	Disconnection length	Transfer Completed?	
			OpenVPN	MobiVPN
S → C	*tcp_retries2 = 6*	10 seconds	✓	✓
		45 seconds	✗	✓
C → S	*tcp_keepalive_time=10s* *tcp_keepalive_intvl=5s* *tcp_keepalive_probes=3*	10 seconds	✓	✓
		45 seconds	✗	✓

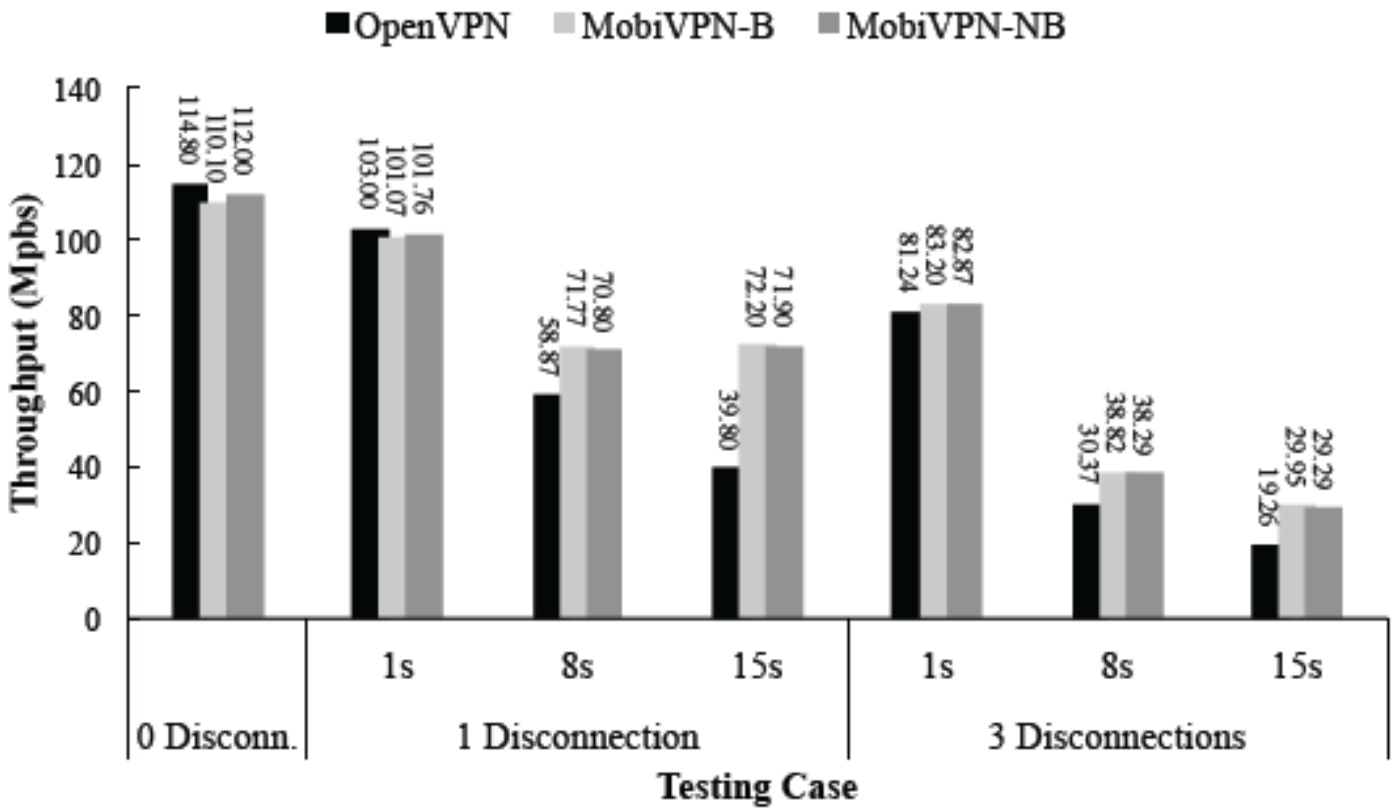
MOBIVPN PERFORMANCE EVALUATION

- We used *iperf* to measure the average throughput of OpenVPN vs. MobiVPN by streaming 200 MB (in local testbed) and 100MB (in distant testbed) of data from the client to the application server using several scenarios varying the number of disconnections and their length.
- No network switching was performed in order to eliminate the factor of the VPN tunnel resumption which we discussed earlier.

RESULTS – DISTANT TESTBED



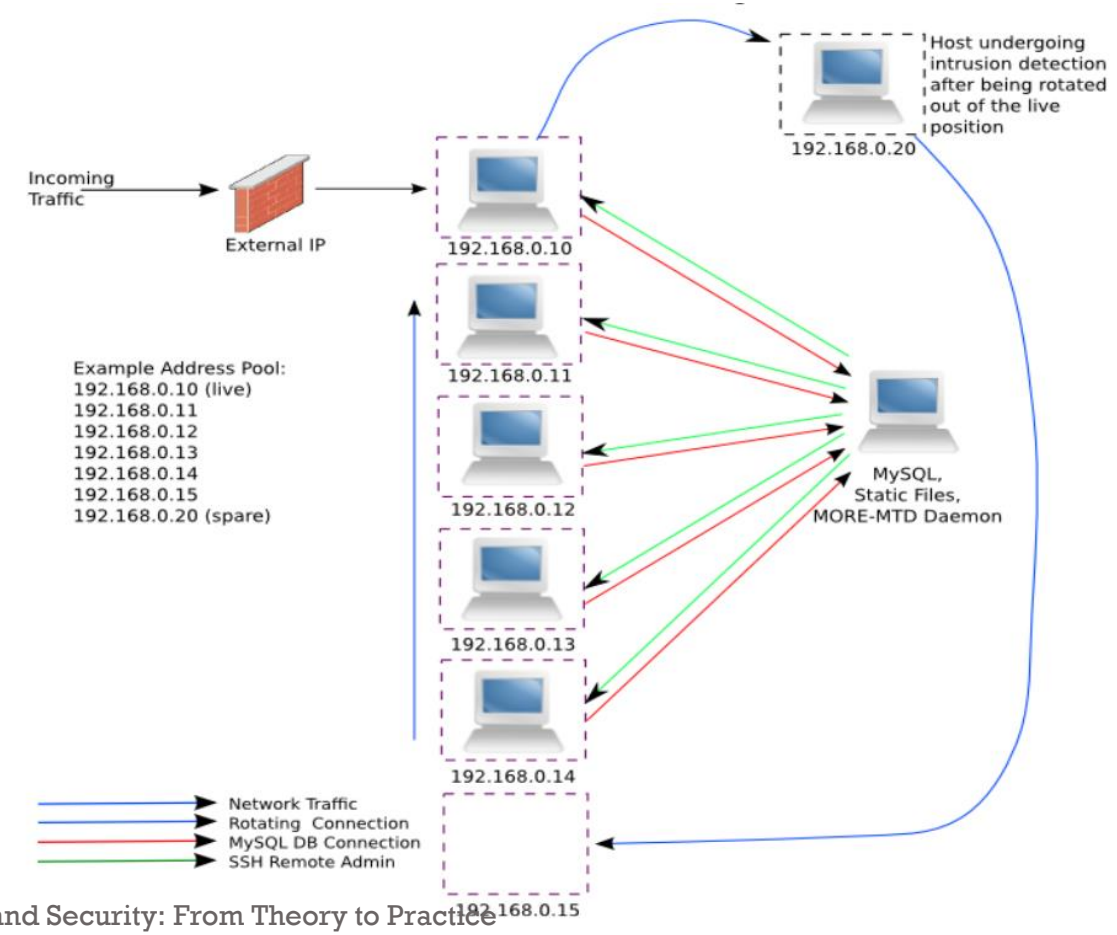
RESULTS – LOCAL TESTBED



HOST LEVEL MTD

- Host Level MTD requires a change in host resources, OS, renaming of configurations.
- Multi-OS Rotation Environment (MORE) is an example of host-level MTD. System consists of two sets of IP addresses, i.e., Live IP and Spare IP.
- The live IP address can be accessed by the clients connected to the host, whereas the spare IP address is used when the host is being rotated.
- The host which is selected during the rotation phase is analyzed for the presence of malware, evidence of intrusion attempts, etc., and is replaced with a clean version of OS.

HOST LEVEL MTD: MULTI-OS ROTATION ENVIRONMENT (MORE)



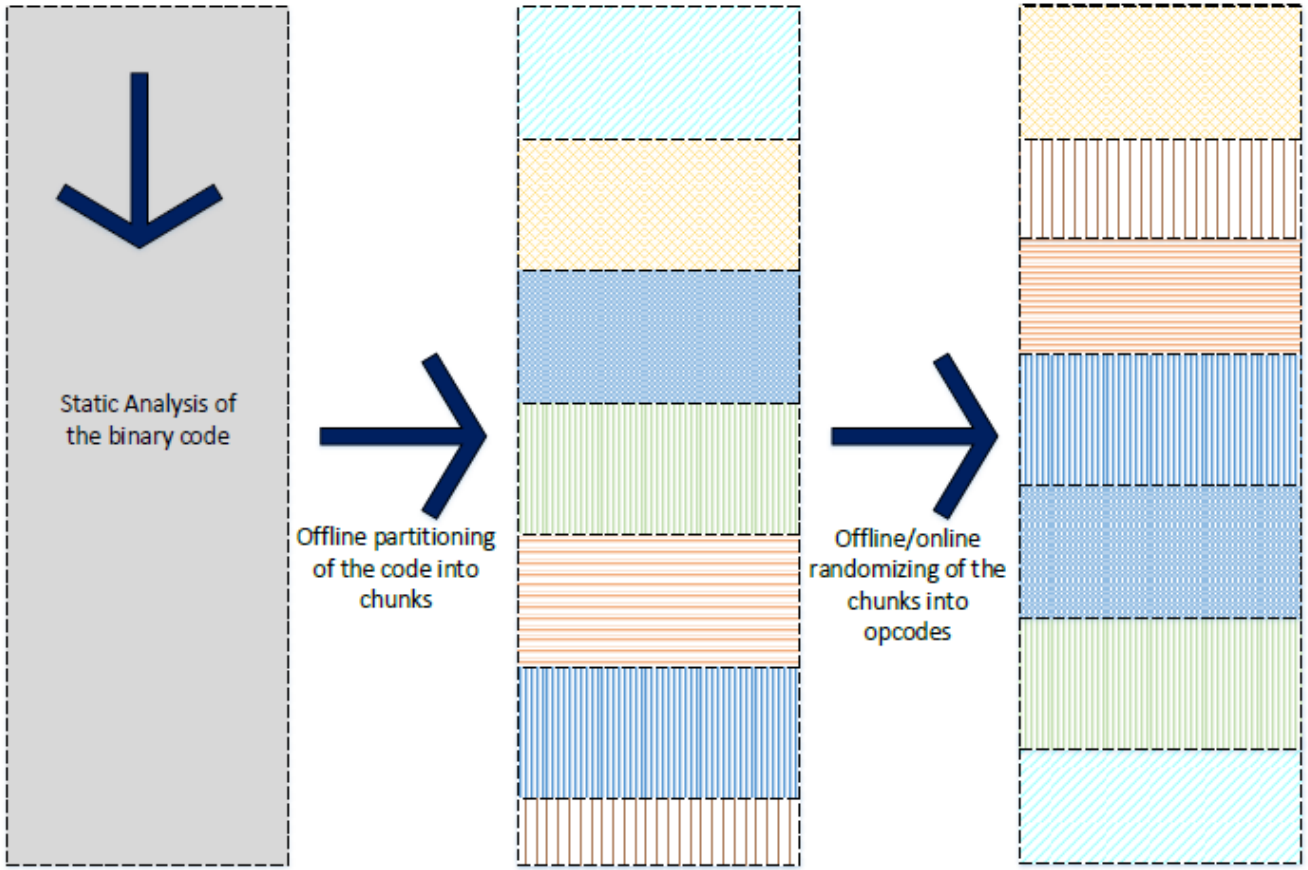
APPLICATION LEVEL MTD

- Application Level MTD involves a change in the application required, source code, memory mapping, software version.
- Address Space Layout Randomization (ASLR) is a technique in OS to hide the base address of the software gadgets running on the OS.
- ASLR protection forces the attacker to guess the location of the gadget.
- Some parts of the le are not randomized, leaving weak spots in the application for the attacker, which can be used by him to invoke malicious code.

APPLICATION LEVEL MTD AGAINST ROP

- ROP is an exploit technique which allows an attacker to take control of the program ow by smashing the call stack and inserting the malicious instruction sequence in the program.
- ROP is able to bypass Data Execution Prevention (DEP), a technology used in hardware and software to prevent code injection and executions.
- MTD can be used to prevent ROP attacks.

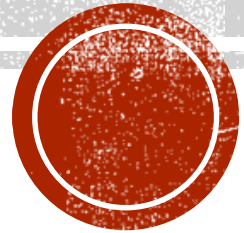
APPLICATION LEVEL MTD AGAINST ROP



APPLICATION LEVEL MTD AGAINST ROP

- The attackers leveraging the ROP attacks assume that the gadgets they are targeting are in absolute address or shifted by some constant bytes.
- MTD solution can be used to ensure that even if the attacker can detect the byte shift of one gadget, he is not able to guess byte shift for rest of the gadgets.
- The Process Executable (PE)/ELF file for the application binary can be analyzed and the chunks of different memory blocks can be rearranged in order to provide MTD based security.
- The reorganized binary le provides the same functionality as the original application binary.

SDN BASED MTD

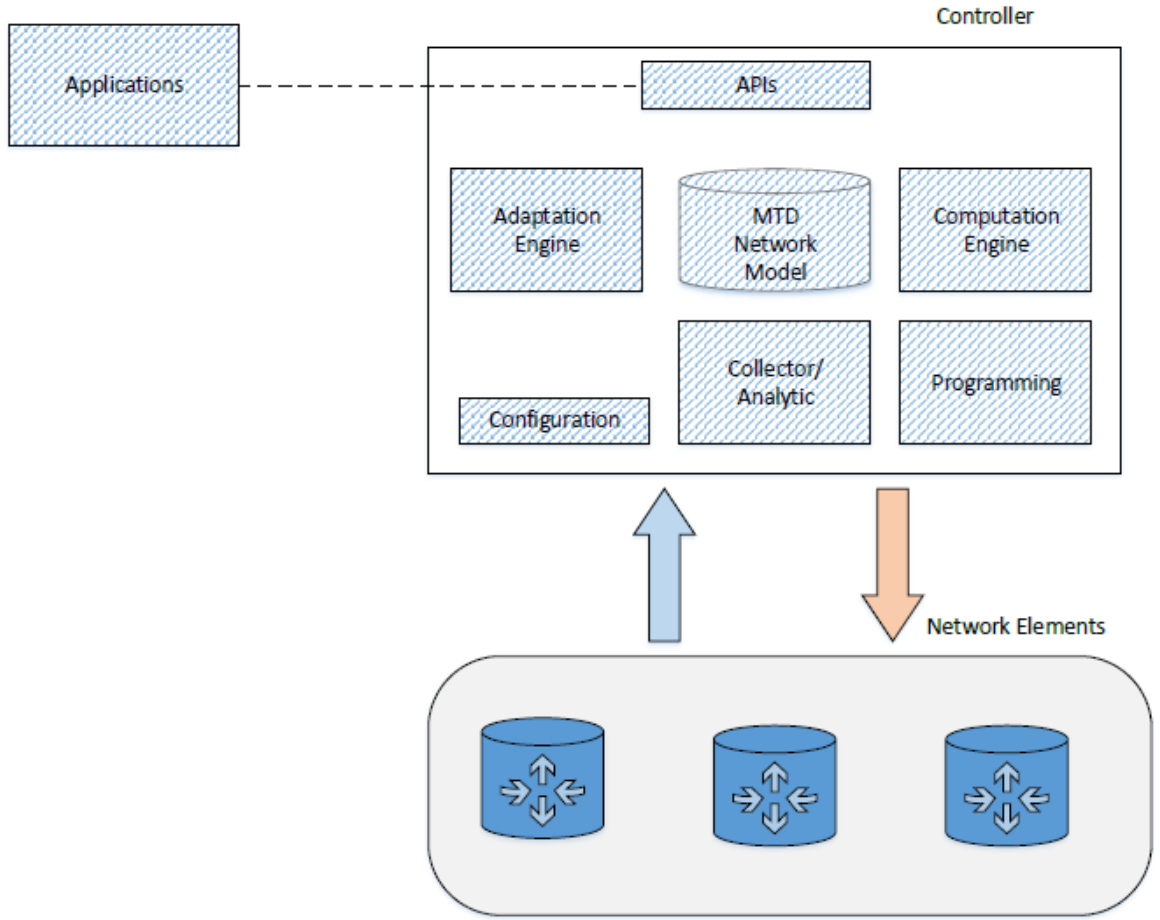


Network Mapping and Reconnaissance Protection, OpenFlow Random Host Mutation, Frequency Minimal MTD for SDN, SDN based Scalable MTD in Cloud.

SDN BASED MTD SOLUTION

- The flexible programmable network solution offered by SDN makes it an ideal candidate for provisioning MTD solutions.
- Cloud network managed by SDN, the SDN controller can be notified by security analysis tools about active threat in the network.
- SDN controller such as POX, ODL, Floodlight can take preventive methods to deal with the situation.
- Some SDN based countermeasures include IP address hopping, reconfiguration of routes for network traffic or changing the host endpoint to delay the attack propagation.

SDN BASED NETWORK MAPPING AND RECONNAISSANCE PROTECTION



SDN BASED NETWORK MAPPING AND RECONNAISSANCE PROTECTION

- Most scanning tools make use of ICMP, TCP or UDP scans to identify the connectivity and reachability.
- The Time to Live (TTL) information can also help in the identification of a number of hops to the attack target.
- SDN-enabled devices can use MTD adaptations can be used to delay the attack propagation by masquerading the real response and replying back with a fake response to confuse the attacker.

SDN BASED NETWORK MAPPING AND RECONNAISSANCE PROTECTION

- The attack traffic can be dropped by SDN based Firewall, and response the appears to be legitimate attack reply can be sent back to the attacker.
- The attacker's workload will be increased because he will have to distinguish the fake reply from the real traffic reply.
- The SDN-enabled devices can also introduce random delays in TCP handshake request, that will disrupt the identification of TCP services.

SDN BASED SERVICE VERSION AND OS HIDING

- The attacker needs to identify the version of OS or vulnerable service in order to mount an attack.
- For instance, the attacker can send HTTP GET request to Apache Web Server, and the response can help in identification of vulnerability associated with a particular version of the Apache software.
- An SDN-enabled solution can override the actual service version with a bogus version of the Apache Server.

SDN BASED SERVICE VERSION AND OS HIDING

- OS Fingerprinting is another attack vector, where the attacker tries to discover the version of the operating system which is vulnerable.
- Modern OS can generate a random response to TCP and UDP requests.
- The way in which TCP sequence numbers are generated can help an attacker in the identification of OS version.

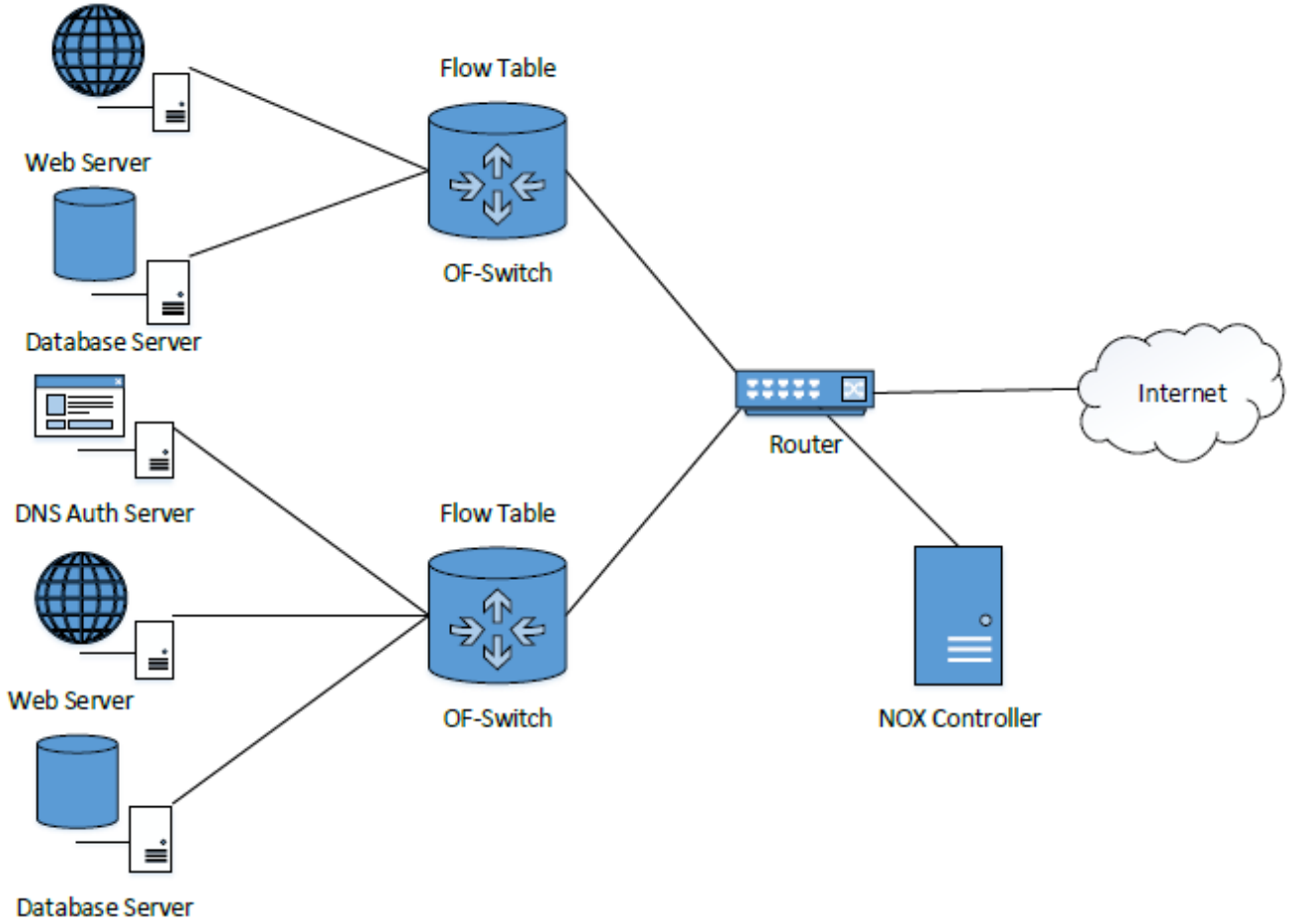
SDN BASED SERVICE VERSION AND OS HIDING

- In an SDN-enabled solution, the OS version can also be obfuscated by the generation of a random response to the probes from a scanning tool.
- SDN can introduce a layer of true randomization for the transit traffic to the target.
- The SDN controller can manage a list of OS proles and send a reply resembling TCP sequence of a bogus OS, thus misguiding the attacker.

OPENFLOW RANDOM HOST MUTATION

- SDN makes use of OpenFlow protocol for control plane traffic.
- OpenFlow enabled MTD architecture can be used to mutate IP address with a high degree of unpredictability while keeping a stable network configuration and minimal operational overhead.
- The mutated IP address will be transparent to the end host.

OPENFLOW RANDOM HOST MUTATION



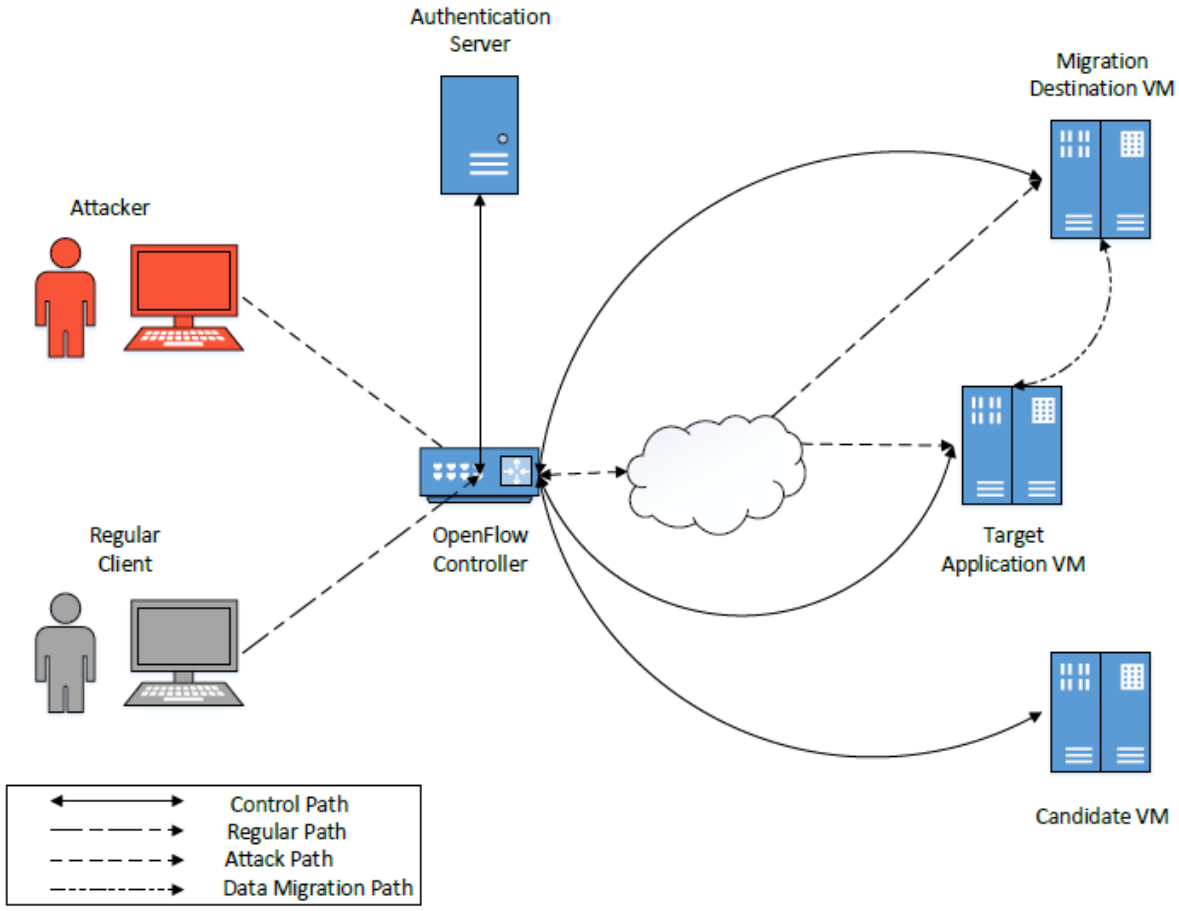
OPENFLOW RANDOM HOST MUTATION

- The actual IP address of the host called real IP (rIP) is linked with a short-lived virtual IP address (vIP) at regular interval.
- The translation of rIP-vIP happens at the gateway of the network, and a centralized SDN controller performs the mutation across the network.
- A Constraint Satisfaction Problem (CSP) is formulated in order to maintain mutation rate and unpredictability constraints.
- Sensitive hosts have a higher mutation rate compared to the regular hosts in this scheme.

FREQUENCY MINIMAL MTD USING SDN: DESIGN GOALS

- What is the optimal frequency at which proactive VM migration should take place, provided wastage of cloud resources is minimal?
- What should be the preferred location for VM migration which can be selected without impacting the application performance?

FREQUENCY MINIMAL MTD USING SDN



FREQUENCY MINIMAL MTD USING SDN

- The normal clients can access the services hosted in the cloud network via regular path.
- The attack path represents the path along which the attacker tries to exploit the target application.
- The VMs periodically share their resource information such as storage, compute capacity with the controller along the control path.
- Depending upon the level of threat, the migration of VM can be proactive or reactive.

FREQUENCY MINIMAL MTD USING SDN

- The data path shows the path along which VM application and its back-end database are migrated.

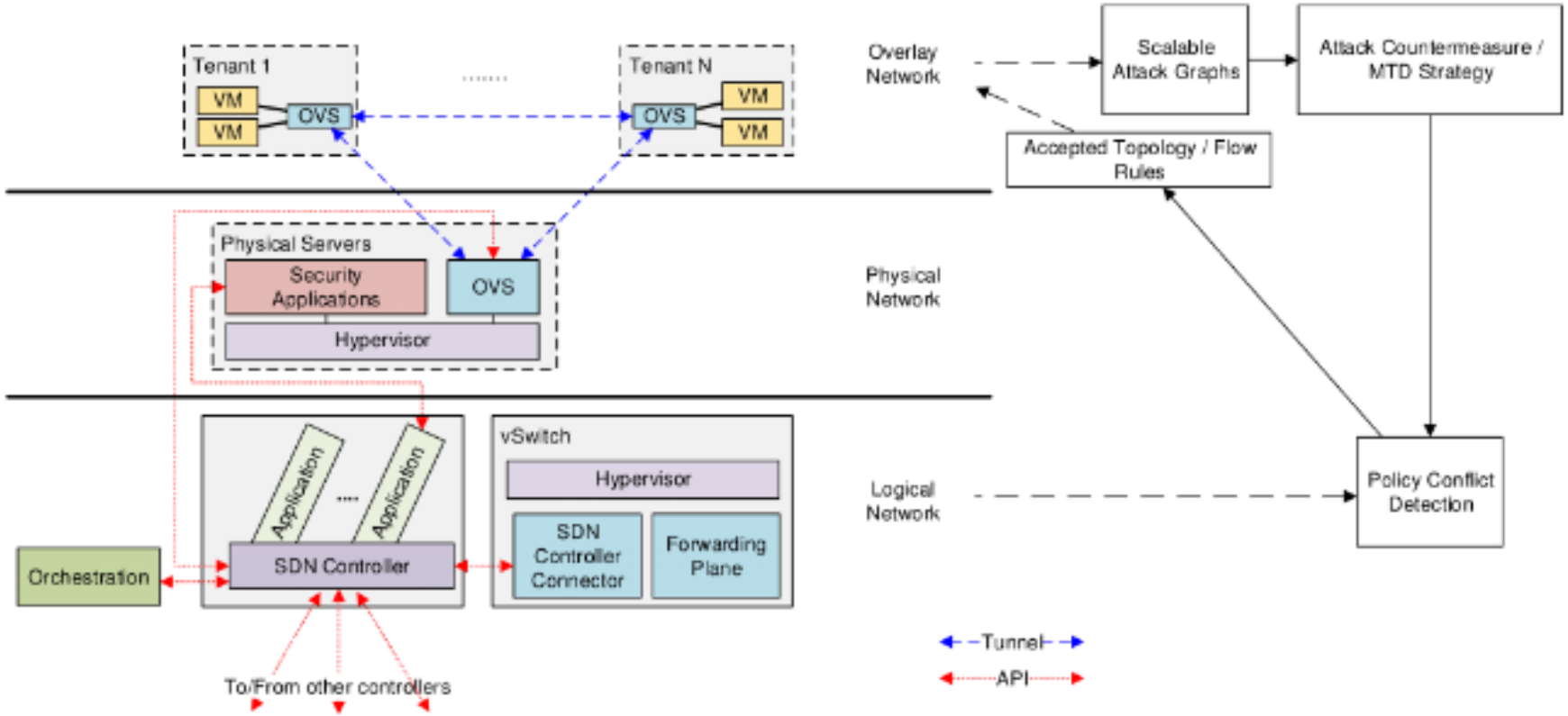
FREQUENCY MINIMAL MTD USING SDN: VM MIGRATION CONSIDERATIONS

- VM Capacity: This parameter considers the capacity of migration target in terms of computing resources available.
- Network Bandwidth: The lower the network bandwidth between the source and target VM, slower will be the migration process and longer would be the exposure period.
- VM Reputation: This is the objective indicator of VM robustness to deter future cyber attacks. The history of VM in terms of cyber attacks launched against the VM.

SDN BASED SCALABLE MTD IN CLOUD

- Attack Graph-based approach to perform the security assessment of a large scale network.
- Based on the security state of the cloud network, MTD countermeasures are selected.
- The overlay network is responsible for vulnerability analysis, attack graph generation.

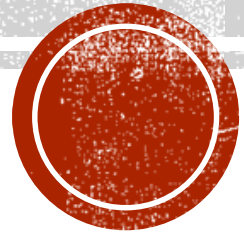
SDN BASED SCALABLE MTD IN CLOUD



SDN BASED SCALABLE MTD IN CLOUD

- Attack Graphs help in identification of possible attack scenarios that can lead to exploitation of vulnerabilities in the cloud network.
- Parallel Hypergraph Partitioning in order to create a scalable attack graph in real time and select MTD countermeasure - VM migration.
- The MTD strategy considers the available space and the threat level of the destination server before performing the VM migration.

GAME THEORETIC MTD MODELS



IP Randomization, Game Theoretic Multi-stage MTD, Software Diversity, Markov Game based MTD

GAME THEORETIC MODELING OF MTD

- Moving Target Defense can be considered as a game between the defender and the attacker.

VERY VERY SHORT GAME THEORY TUTORIAL

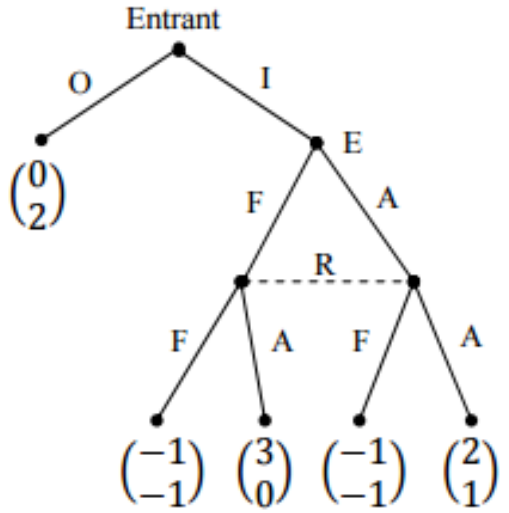
- Type of game (Static, dynamic, single player, multi-player)
- Information (No information, partial information, complete information)
- Players, States, Actions, Utilities
- Modeling – Normal form, Extensive form.

Prisoner's Dilemma – Payoff Matrix

ROW \ COL	Co-operate	Defect
Co-operate	(3, 3)	(0, 5)
Defect	(5, 0)	(1, 1)

Preference to Move Based on Higher Payoff

Nash Equilibrium



GAME THEORETIC MODELING OF MTD

- The goal of the attacker is the exploit critical services in the network with a minimum possible cost of attack.
- The goal of the defender is to invest the security budget in such a way that the cost of attack is maximized for the attacker.
- The defensive strategy should stop or slow down the attack propagation.

GAME THEORETIC IP RANDOMIZATION

- Real Nodes that the attacker wants to target.
- Decoy nodes, which have been added to the system by the defender in order to distract the attacker.
- A game theoretic model can used to study the strategic interactions between the attacker and the decoy nodes.
- The goal of an adversary in the attack model is to discover the real node's IP address in order to select the appropriate attack against services running on that VM.

GAME THEORETIC IP RANDOMIZATION

- The information set of the system can be described by the current number of valid sessions, Y(t).
- Decoy node fraction scanned by the adversary at time t denoted by D(t).
- The goal of the system administrator is to minimize the cost function composed of an information set by choosing an optimal random variable R.

GAME THEORETIC IP RANDOMIZATION

- Goal of admin $\min_{R} E\big(D(R) + \beta Y(R)\big)$.
- The β denotes delay introduced by migration of a connection to the IP address of the real node from decoy node.
- The cost of the connection is, $\beta Y(t)$.
- The randomization policy is the mapping of the information space (Y(t), D(t)) to {0,1} Radom variable.

GAME THEORETIC FEEDBACK DRIVEN MULTI-STAGE MTD

- A multi-stage defense mechanism can be used to manipulate the attack surface in order to provide a proactive defense mechanism against attacks.
- The attacker needs to learn the network setup continuously, and change his attack vector accordingly.
- The defender, on the other hand, needs to consider the reconfiguration cost of shifting the attack surface.

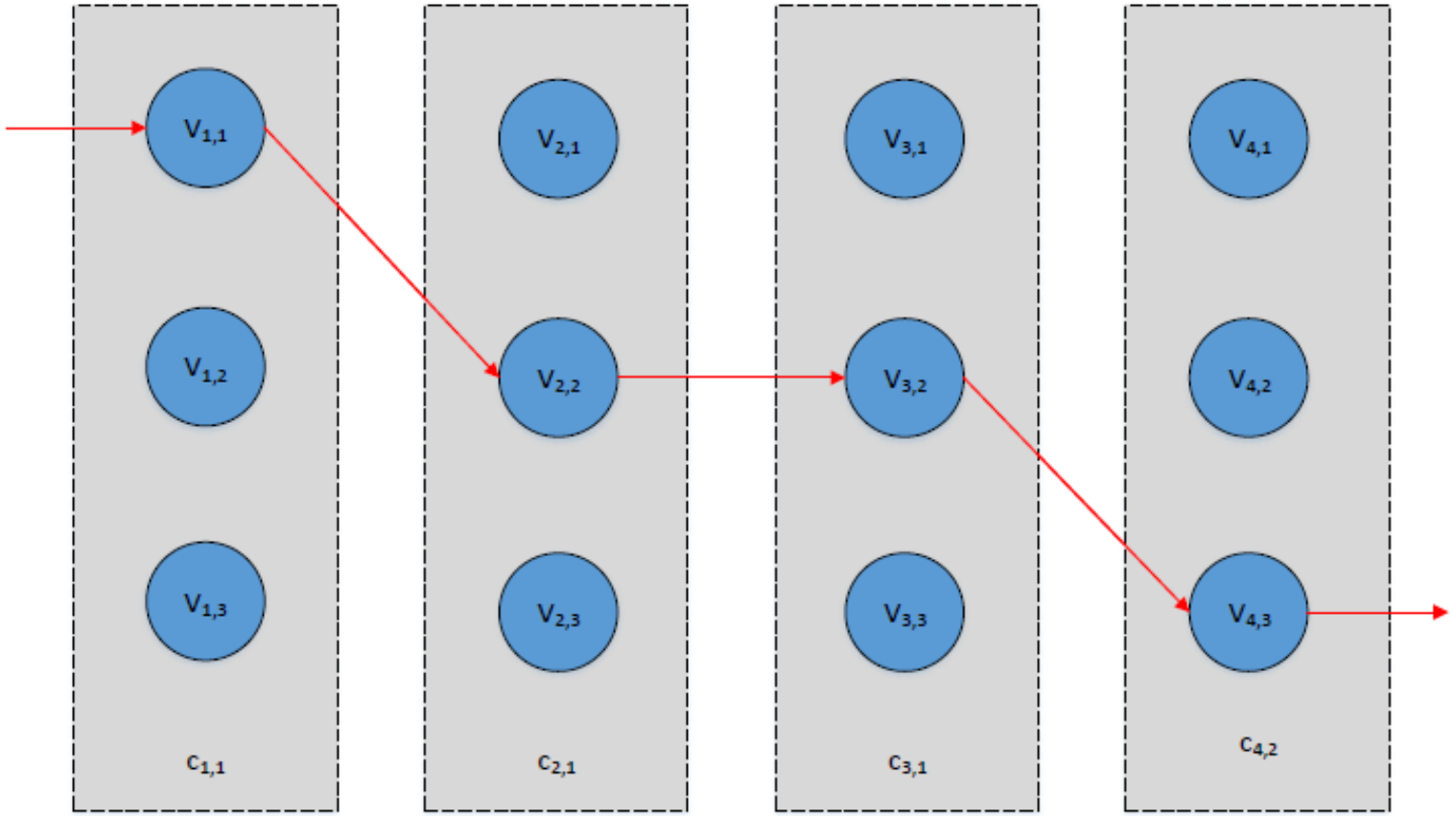
GAME THEORETIC FEEDBACK DRIVEN MULTI-STAGE MTD

- Thus, the defender has to find the optimal configuration policy in order to achieve the desired objective.
- Consider a system to be partitioned into several layers, l=1,2,3,..,N.
- Vulnerabilities at each layer $V_l := \{v_{\{l,1\}}, v_{\{l,2\}}, \ldots, v_{\{l,n_l\}}\}$.
- Vulnerability V_l is the common knowledge for both players.

GAME THEORETIC FEEDBACK DRIVEN MULTI-STAGE MTD

- The feasible system configurations at layer l, $C_l: \{c_{\{l,1\}}, c_{\{l,2\}}, \dots, c_{\{l,m_l\}}\}$.
- The function π_l maps each system configuration with vulnerability set $\pi_l: C_i \rightarrow 2^{\{V_l\}}$.

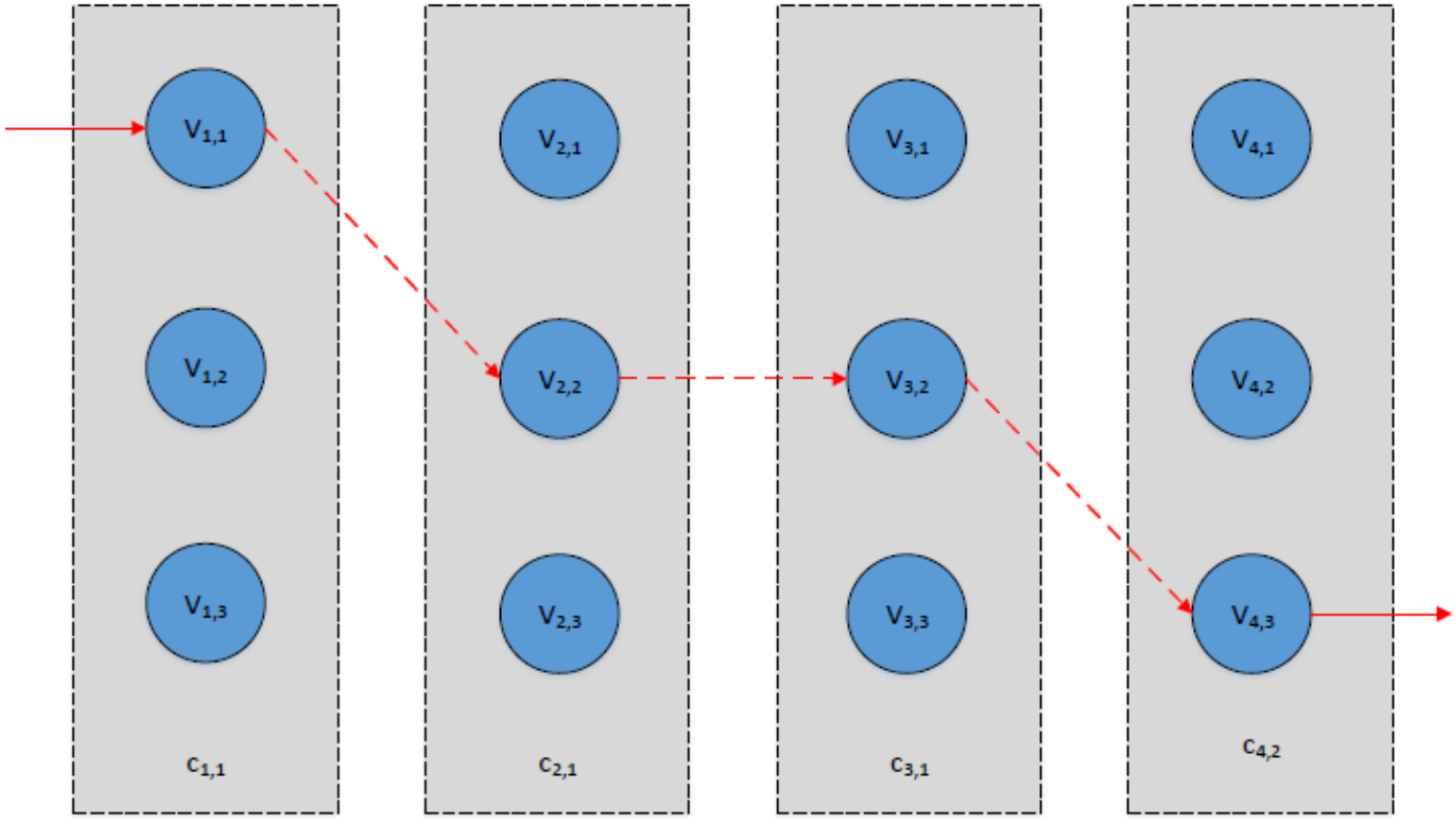
STATIC CONFIGURATION SEQUENCE AND ATTACK PROPAGATION



STATIC CONFIGURATION SEQUENCE AND ATTACK PROPAGATION

- Attack Surface: l=1,2,3,4 and an attack surface with vulnerability $V_l = \{v_{\{l,1\}}, v_{\{l,2\}}, v_{\{l,3\}}\}$.
- Two possible configurations for defender at layer 1, $C_1 = \{c_{\{1,1\}}, c_{\{1,2\}}\}$.
- The configurations in the example above allows attacker to launch a multi-stage attack $\{v_{\{1,1\}} \rightarrow v_{\{2,2\}} \rightarrow v_{\{2,3\}} \rightarrow v_{\{4,3\}}\}$.

RANDOMIZED CONFIGURATION FOR ATTACK SURFACE



RANDOMIZED CONFIGURATION FOR ATTACK SURFACE

- The attack-defense interaction can be formulated as two-player Zero-sum game.
- The defender can randomize the attack surface at each layer to thwart Multi-Stage Attacks.

GAME THEORY BASED SOFTWARE DIVERSITY

- Over-reliance of a certain version of the software and an Operating system can lead to security compromise.
- The hosts in the network and vulnerabilities can be represented by a bipartite graph.
- Software vulnerabilities in original configuration vs vulnerabilities in diversified configuration can be studied as an anti-coordination game.

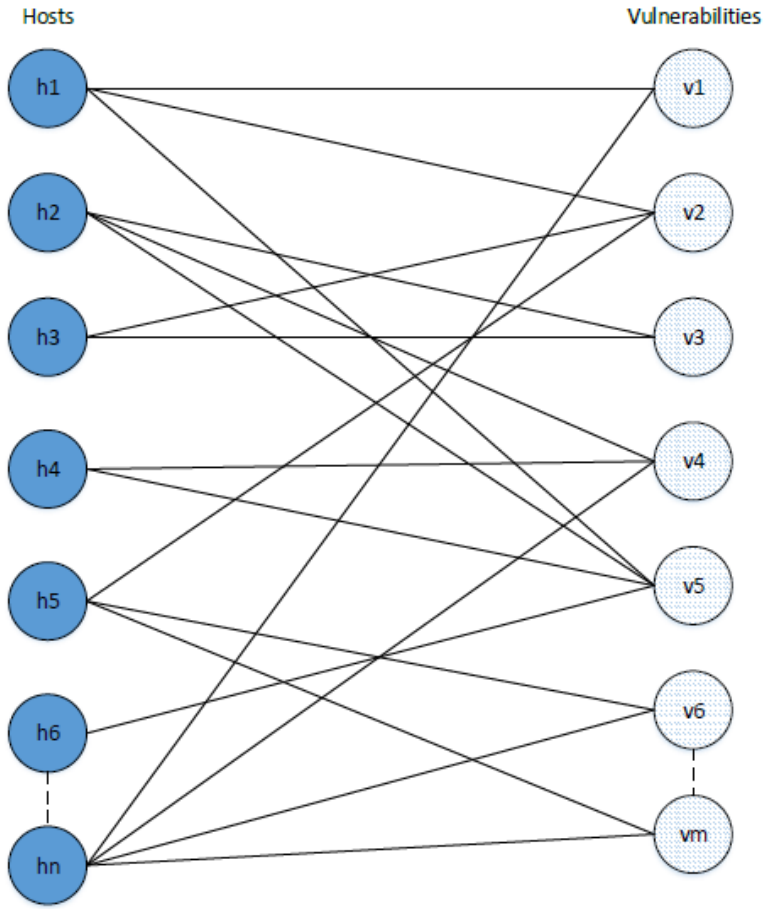
GAME THEORY BASED SOFTWARE DIVERSITY

- The model assumes that there are k vulnerabilities per host, $h_i \in H$.
- A compromised vulnerability affects nk/m hosts.
- Exploit probability of a host chosen at random, which is connected to vulnerability $v_i, p_i = \frac{\deg(v_i)}{kn}$.
- $N_a = (\sum_{\{i=1\}}^{m} p_i^m)^{\frac{1}{1-a}}$, where N_a is the diversity number.

GAME THEORY BASED SOFTWARE DIVERSITY

- In the equation, $N_1 = \lim_{\{a \to 1\}} N_a$, which is same as Shannon Entropy, i.e., $\log(N_1) = H = -\sum_{\{i=1\}}^{m} p_i \log(p_i)$.
- As the value of a increase from $-\infty$ to $+\infty$, the N_a changes weigtage assigned from least to most connected vulnerabilities.

ANTI COORDINATION GAME: SOFTWARE DIVERSITY



ANTI COORDINATION GAME: SOFTWARE DIVERSITY

- Anti Coordination Game for two players can be considered in terms of two hosts h_1, h_2 with vulnerabilities v_1, v_2.
- Each host can stay with current vulnerability v_1 or switch to vulnerability v_2.
- The switching cost c_w is the cost of user getting familiar with new version of software.

ANTI COORDINATION GAME: SOFTWARE DIVERSITY

- If both hosts h_1 and h_2 plan to stay with original software, the cost incurred due to attack would be c_2, and $c_w < c_2$.
- The intrinsic cost of one vulnerability is c_0.

	Stay	Switch
Stay	$-c_2, -c_2$	$0, -c_w$
Switch	$-c_w, 0$	$-(c_2 + c_w), -(c_2 + c_w)$

GAME THEORY BASED SOFTWARE DIVERSITY

- The game consists of two pure strategy Nash Equilibrium (Stay, Switch) and (Switch, Stay).
- The game also comprises a mixed strategy Nash Equilibrium with cost $\frac{c_2+c_w}{2c_2}$.
- Anti-Coordination game shows that some hosts may choose to stay with their original software.
- An effective diversity technique based on Anti-Coordination Game analysis and Renyi Entropy would be to reduce the overlap of vulnerabilities across different hosts.

MARKOV GAME BASED MTD

- Markov Game-based MTD modeling can be used for analyzing security capacity of the system and the attack strategy employed by the adversary for defeating the MTD strategy.
- A = $\{\epsilon, a_1, a_2\}$ denotes the strategy of the attacker.
- D = $\{\epsilon, d_1, d_2\}$ denotes the defenders strategy.
- G = { (D_i, A_i, w_i)}, depicts algorithm G, where $w_i = \{0,1\}$ denotes the winner attacker/defender 1/0 in the step i.
- Random coin-toss based attack defense using algorithm 'M' in the next time-step.
- Consider game end in 'k' time steps. A $(k+1) \times (k+1)$ Markov Game matrix 'M' can be used show attack-defense moves.

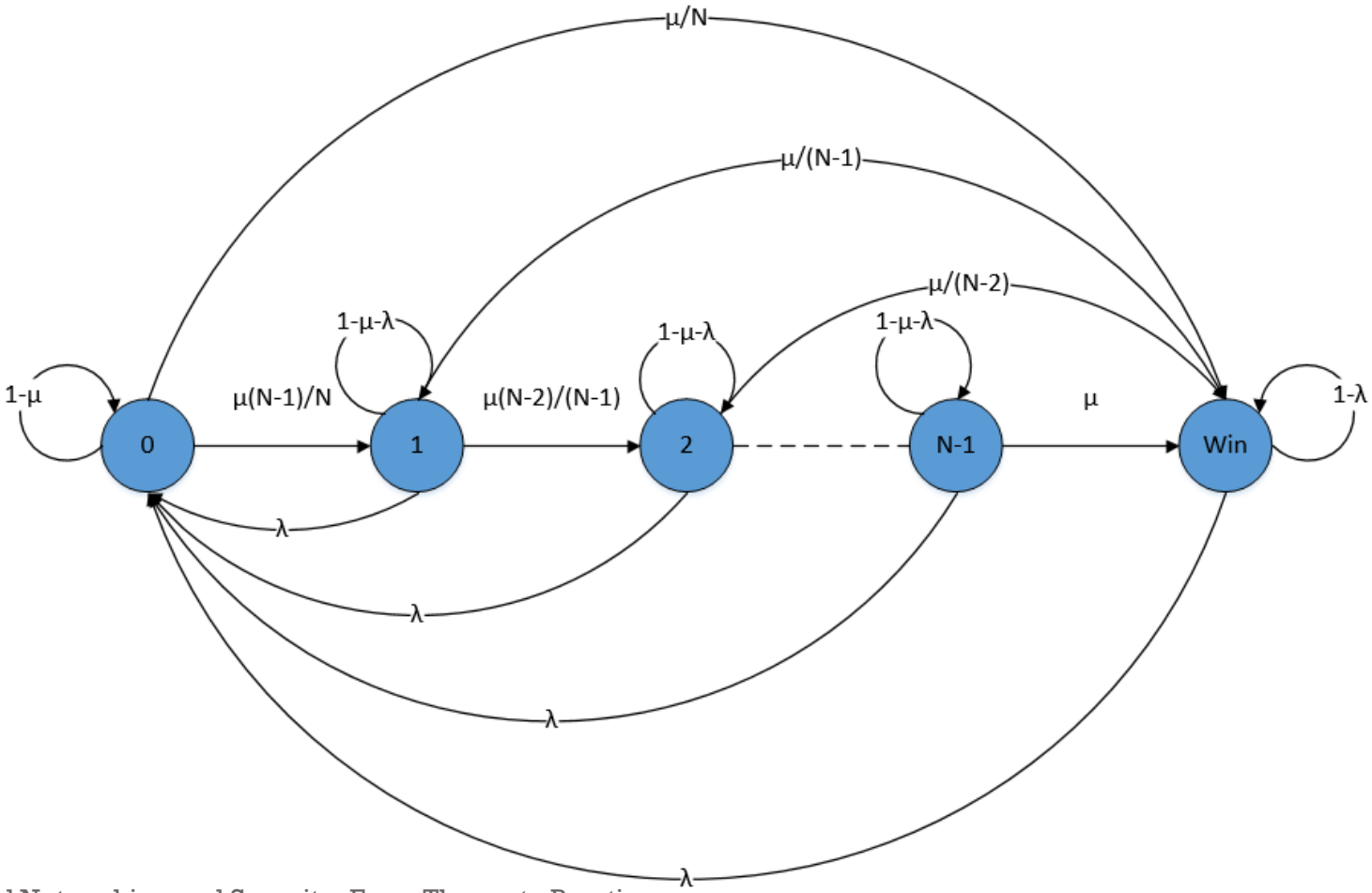
MARKOV GAME BASED MTD

- Parameters λ, μ represent the state of play for the attacker/defender, such that $0 \leq (\lambda + \mu) \leq 1$.
- Transition matrices for attacker and defender $M^A, M^D \; for \; (i, j \in \{0, \ldots, k\})$. The MTD game (M^D, λ, M^A, μ) can be described using $M = \lambda M^D + \mu M^A + (1 - \lambda - \mu)\, I_{\{k+1\}}$, where $I_{\{k+1\}}$ is the identity matrix.

IP HOPPING IN MARKOV GAME

- Continually changing the IP address of the host on the network that attacker is trying to target.
- Adversary has to continually rescan the network to discover the target hosts.
- TCP communication involves 3-way handshake. The attacker sends the 'SYN' packet. If the adversary replies with 'ACK', adversary successfully discovers target.
- Rate of scan by adversary is μ and defender mutates the IP address at a rate λ.

IP HOPPING IN MARKOV GAME



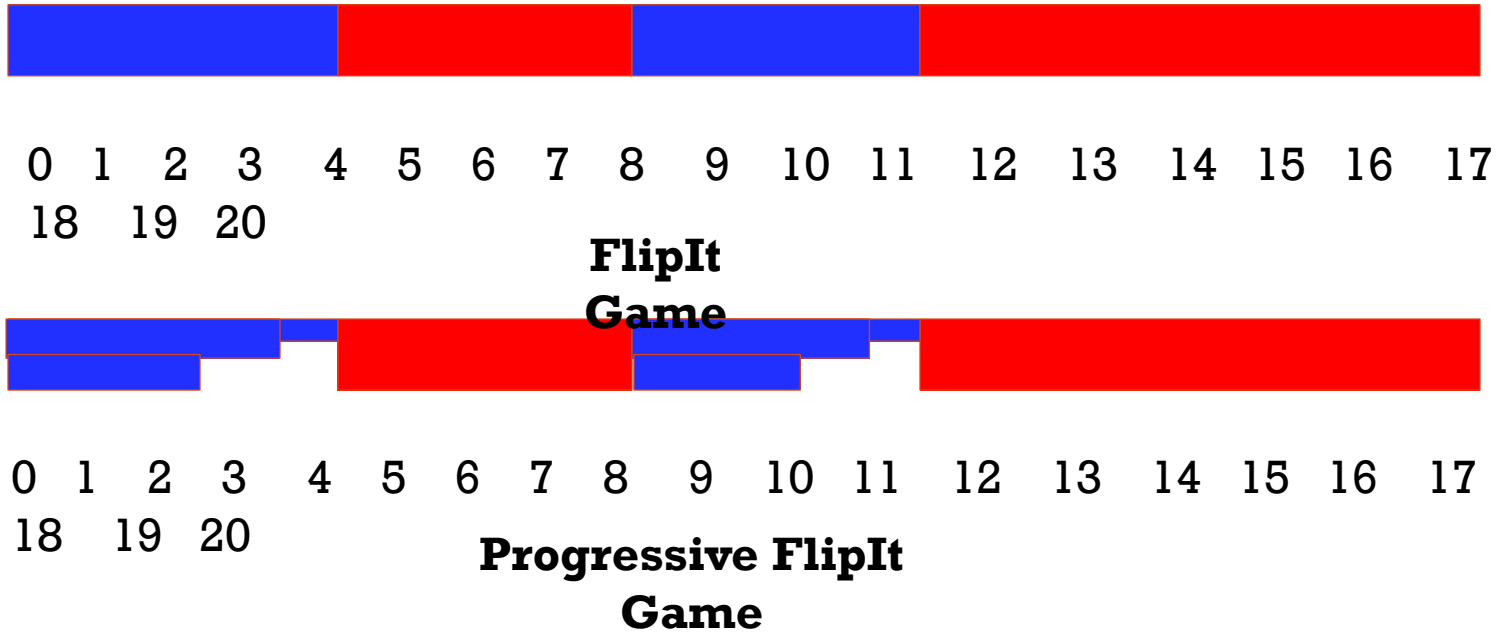
IP HOPPING IN MARKOV GAME

- Markov Game for N+1 states, where each state represents the IP randomization. 'k' represents the winning state for the adversary.
- $M_{\{q,0\}} = \lambda$. The defender randomizes IP address with probability rate λ and the game transitions back to state '0'.
- $M_{\{q,q+1\}} = \frac{\mu(N-q-1)}{(N-q)}$. The attacker tries IP address with probability μ and with probability $1 - \frac{\mu}{N-q}$ the IP address is incorrect.
- $M_{\{q,N\}} = \frac{\mu}{N-q}$. The attacker tries IP address with probability $\frac{1}{N-q}$. The IP address is correct and the attacker wins the game.

IP HOPPING IN MARKOV GAME

- $M_{\{q,q\}} = \lambda$. Both attacker and defender do nothing with probability $(1 - \lambda - \mu)$.
- $M_{\{N,0\}} = \lambda$, $M_{\{N,N\}} = (1 - \lambda)$. The attacker doesn't need to try new IP address. Only if the defender randomizes the IP address, attacker will be kicked out of winning state.
- Stationary distribution for M-MTD game is $\pi M = \pi$, with '0' as initial state representing worst case scenario for the attacker. Probability that attacker wins after 'k' time steps is $P(attacker - win) \leq T \times \pi(k)$.

FLIPIT GAME EXAMPLE



GAME SPECIFICATIONS

- M = 10 instances.
- The attacker may attempt to wrest control of a server through a probe action, which succeeds with some probability
- Otherwise increases the success probability of subsequent probes
- The defender may at any time reimage a server

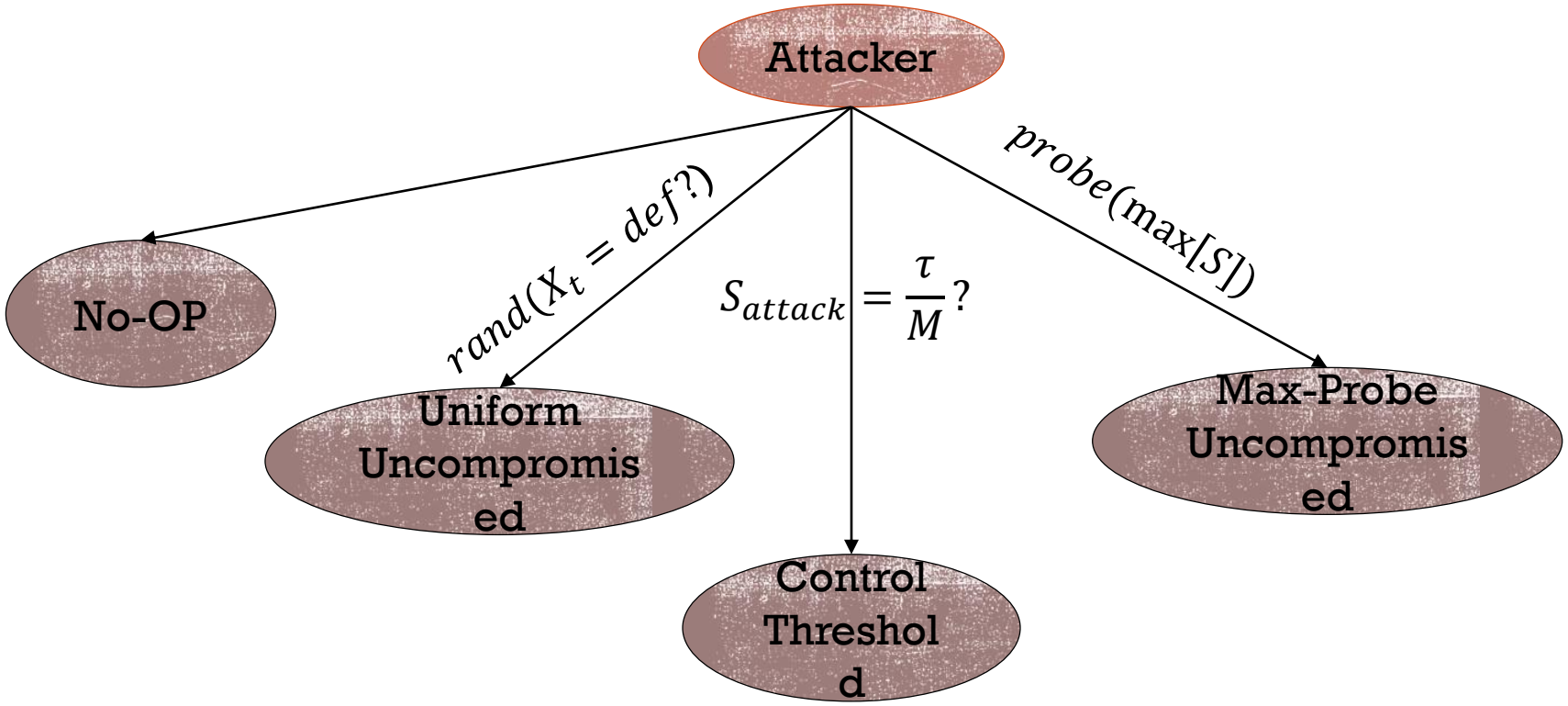
GAME STATES AND ACTIONS

- Server State $< \chi, \nu, \rho >$
- $\chi = \{att, def\}$ – who controls the server.
- $\nu \in \{up\} \cup [0, T]$ – server is up/down from a reimage initiated at [0,T].
- ρ number of attack probes since the last defender reimage action.

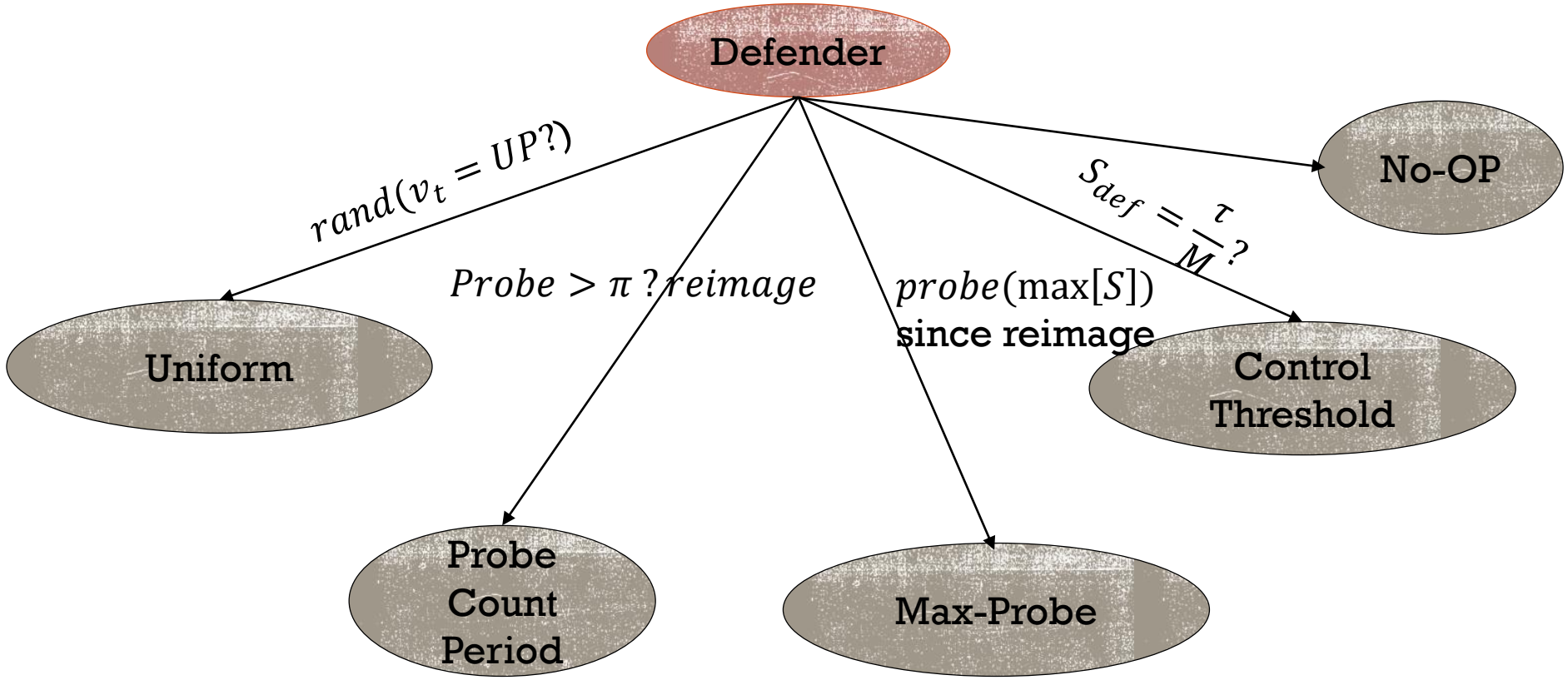
UTILITY

- Each player accrues utility based on the number of server up and in their control and the number of servers that are down
- Attacker – {disrupt, control}
- Defender – {confid, avail}
- $\boldsymbol{c_A} = \{0.2, 0.5, 1\}$, $\boldsymbol{Utility} = \{low, majority, high\}$

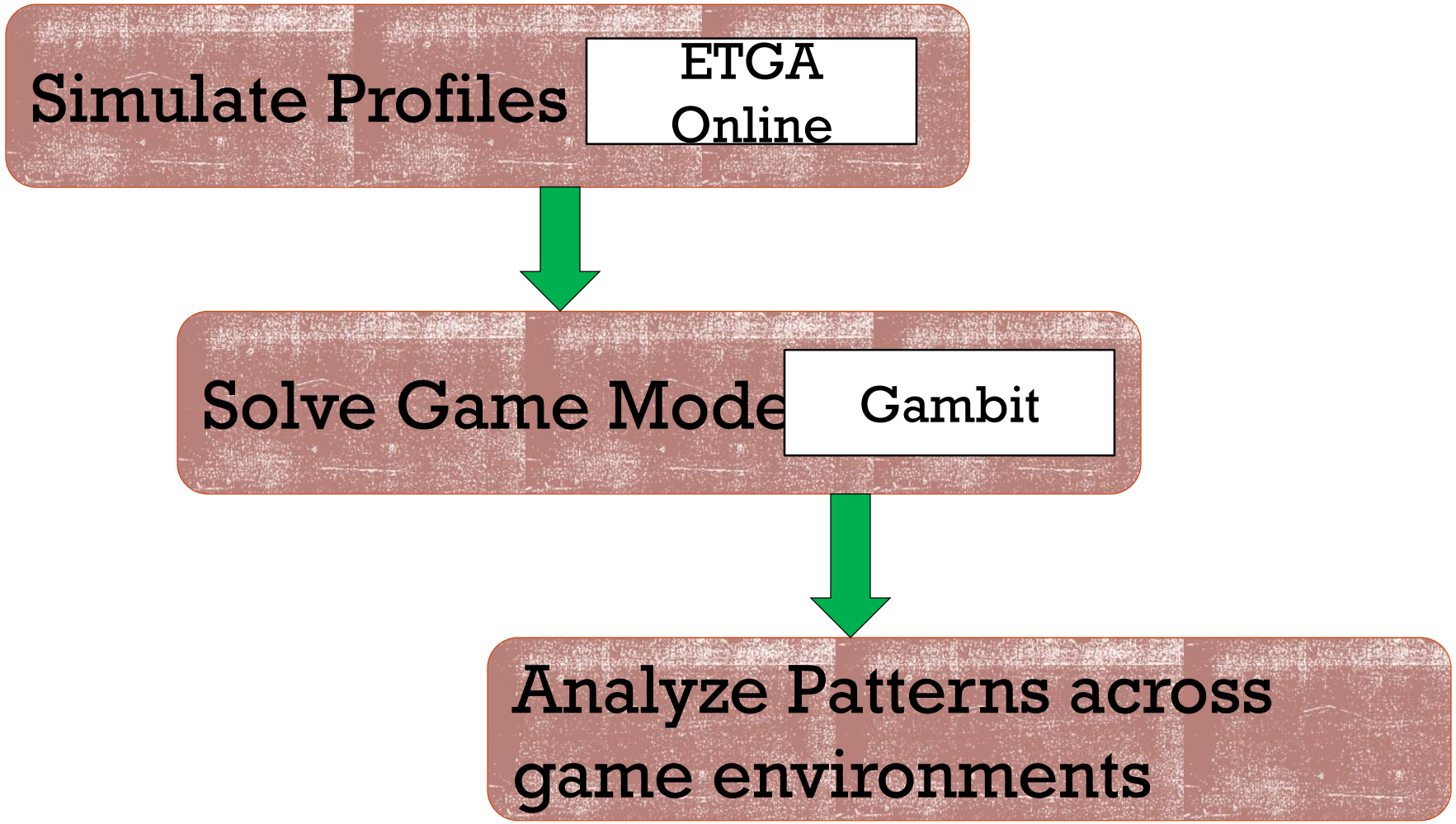
ATTACK STRATEGIES



DEFENSE STRATEGIES



EGTA PIPELINE

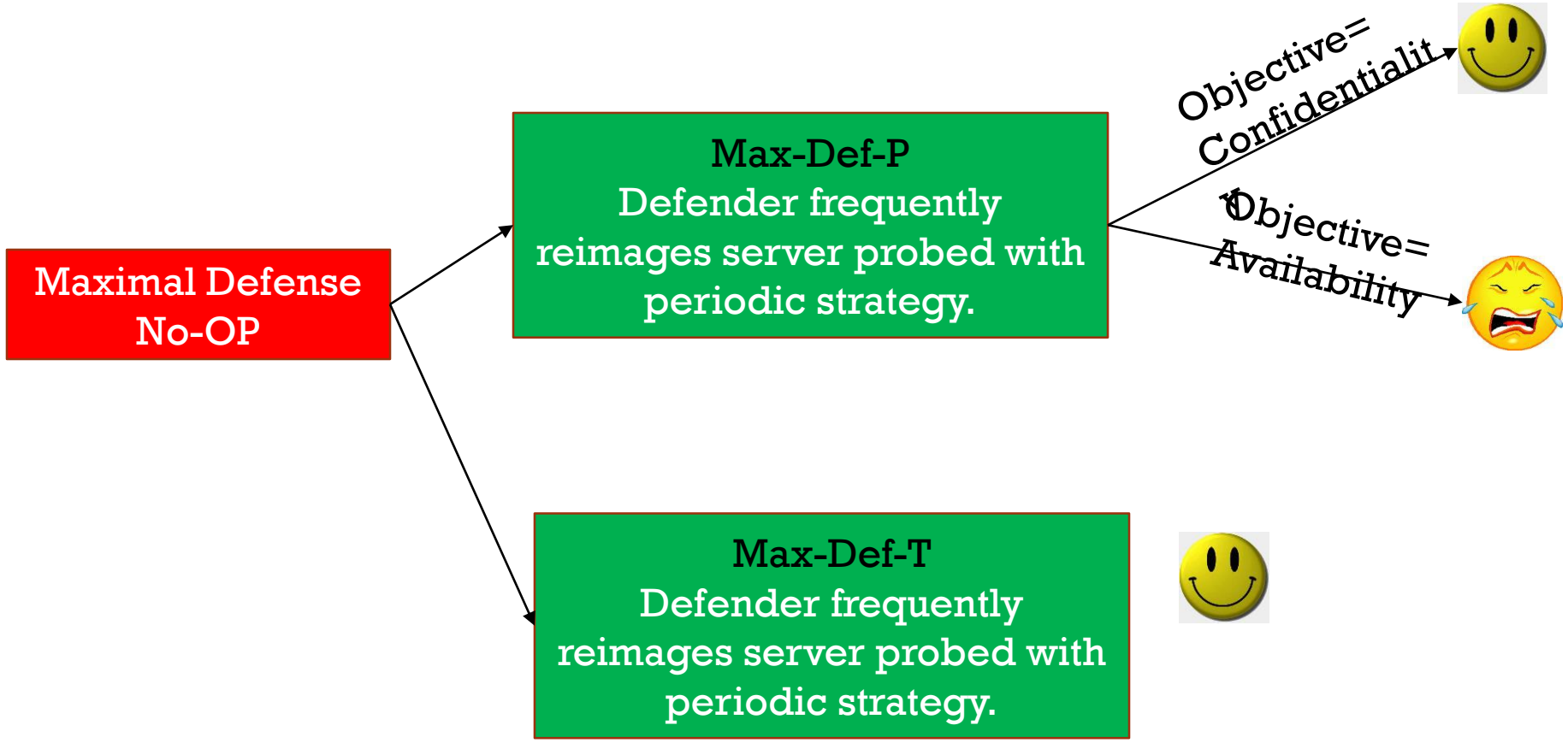


RESULTS: PERFECT PROBE DETECTION ENVIRONMENT

		Utility Environments			
Utility threshold	c_A	disrupt/avail	disrupt/confid	control/avail	control/confid
low	1	MaxDef-T, Share	*MaxDef-P*	MaxDef-T, Share	*MaxDef-P*
low	0.5	Share	*MaxDef-P*	MaxDef-T	*MaxDef-P*
low	0.2	Share	*MaxDef-P*	MaxDef-T, Share	*MaxDef-P*
majority	1	MaxDef-T	MaxDef-P	MaxDef-T	MaxDef-P
majority	0.5	Fight	MaxDef-P	MaxDef-T	*MaxDef-P*
majority	0.2	Fight	MaxDef-P	MaxDef-T, MaxAtt	*MaxDef-P*
high	1	MaxDef-T, MaxAtt	*MaxDef-P*	MaxDef-T, MaxAtt, Fight	*MaxDef-P*
high	0.5	MaxDef-T, MaxAtt, Fight	*MaxDef-P*	MaxDef-T, MaxAtt, Fight	*MaxDef-P*
high	0.2	MaxDef-T, MaxAtt	*MaxDef-P*	MaxDef-T, MaxAtt, Fight	*MaxDef-P*

Table 4: Qualitative Nash equilibria for the thirty-six perfect probe detection environments. Cells in italics indicate games not actually simulated, but with obvious equilibria given the confid defense objective.

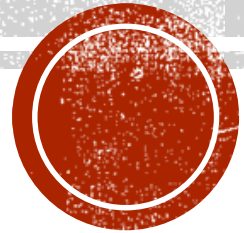
RESULTS: PERFECT PROBE DETECTION ENVIRONMENT



KEY INSIGHTS

- With perfect probe detection, **maximal defense** is always an equilibrium when attackers have **control** objectives, and pervasive as well for **disrupt** objectives.
- Maximal attack is **occasionally in equilibrium** among others with **perfect probe detection**, but becomes significantly more **prevalent** once **probe detection degrades**.
- **Fight** equilibria are generally **pervasive**, except when **contention** for servers is **particularly weak**.
- The **Control** strategies **appear widely** in equilibrium configurations.

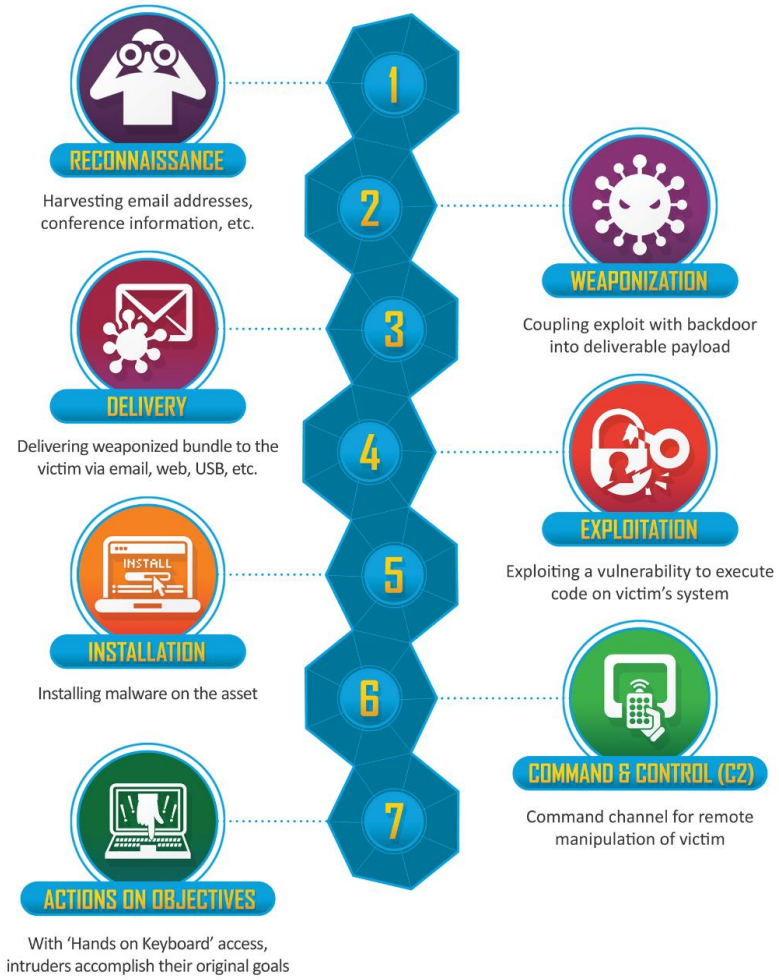
EVALUATION OF MTD



Quantitative Metrics for MTD Evaluation, MTD Analysis and Evaluation Framework (MASON).

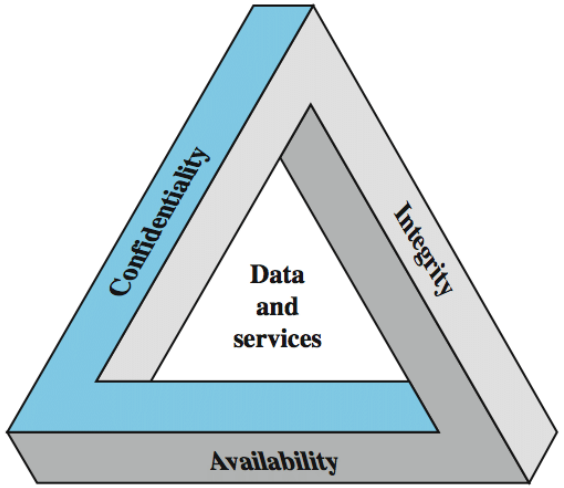
MTD EFFECTIVENESS EVALUATION

- Goal is to understand the effectiveness of countermeasures during the intrusion process.



TRADITIONAL METRICS

- CIA Triad: Confidentiality, Integrity, Availability.
- Designed to assist in policy creation to protect data.
- Limitations
  - Quantitative scale unclear.
  - Limited distinction between attacker and defender.
  - Doesn't reflect increased information about an attacker.



PROPOSED METRICS

- Assigns 0 to 1 for each item indicating their relative value.
- Metrics assigned independently to both defender (mission) and attacker.
  - Productivity – Tasks performed over time.
  - Success - Ability to complete tasks.
  - Confidentiality – Visibility of task activity.
  - Integrity – Accuracy of task output.

EXPERIMENT DETAILS

- Intended to verify the usefulness of the metrics.
- Network topologies of 10 to 20 nodes built on VMWare.
- Tested on networks with two different methods of MTD.
- Experiment metric is the weighted average for each task.
- Assumptions:
  - Other metrics encapsulated within these metrics (IE network surface area being covered by Attack Productivity and Mission Confidentiality).
  - Network is continually running tasks to achieve an output.
  - Attacks are trying to compromise information (CIA Triad).

ARCSYNE

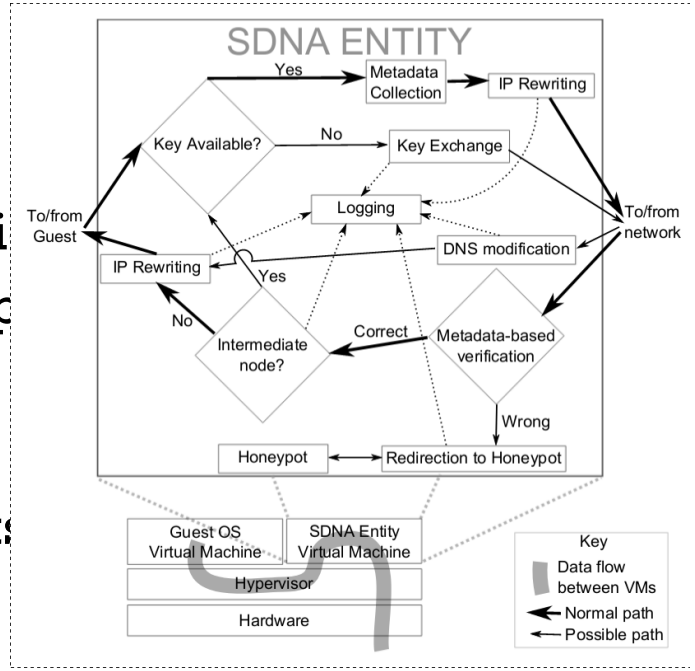
- Active Repositioning in Cyberspace for SYNchronized Evasion.
- Rotate IP addresses for all hosts on network simultaneously.
  - Hop time is the period after which addresses are reassigned.
  - Hop time Varies from fractions of a second to minutes.
- Shorter hop time means the network is harder to map but adds significant overhead.

Table 8: ARCSYNE CPU Usage vs Scale and Hop Delay

		Hop Delay		
		0.1	**1.0**	**10**
Scale	**10**	≈10%	≈3%	≈1%
	20	≈30%	≈5%	≈1%

SDNA

- Self-shielding Dynamic Network Architecture
- Multiple layers of security
  - Hypervisor between OS and network.
  - Masks OS of hosts through encapsulati
  - Rewrites IP addresses to prevent mapp
  - Randomizes packet routing.
  - Credentials hidden from hosts.
  - Redirecting of unauthenticated packets to a honeypot.
- Level of security determined by number of whitelisted protocols.



RESULTS (ARCSYNE)

- Obfuscated hosts → hard to identify attacks from noise.
- Halting attacks quickly enables more tries per period, allowing attackers to try more attempts faster.
- More security operations → less resources for mission.

Table 4: Attack Metrics for MTD Configurations

Configuration	Confidentiality	Success	Productivity	Integrity
No ARCSYNE	0.7	1.0	0.5	1.0
ARCSYNE	1.0	0.2	0.7	0.0

Table 6: Mission Metrics for MTD Configurations

Configuration	Confidentiality	Success	Productivity	Integrity
No ARCSYNE	0.2	1.0	1.0	0.7
ARCSYNE	1.0	0.9	0.9	0.6

RESULTS (ARCSYNE)

- Increasing hop delay reduces Mission Integrity, Mission Productivity, and Mission Success.

Table 7: ARCSYNE Mission Metrics vs. Hop Delay

Hop Delay	Confidentiality	Integrity	Productivity	Success
0.1s	1.0	0.75	1.0	1.0
1.0s	1.0	0.75	1.0	1.0
10.0s	1.0	0.45	0.8	0.75

RESULTS (SDNA)

- Honeypot masks attack outcome, reducing attack productivity.
- Unclear why Mission Confidentiality dropped down for External case.

Table 4: Attack Metrics for MTD Configurations

Configuration	Confidentiality	Success	Productivity	Integrity
No SDNA	0.7	0.9	0.7	0.9
SDNA, External	1.0	0.3	0.7	0.0
SDNA, Internal	1.0	0.5	0.4	0.1

Table 6: Mission Metrics for MTD Configurations

Configuration	Confidentiality	Success	Productivity	Integrity
No SDNA	0.5	1.0	1.0	1.0
SDNA, External	0.2	0.5	0.4	0.5
SDNA, Internal	1.0	0.5	0.4	0.5

RESULTS (SDNA)

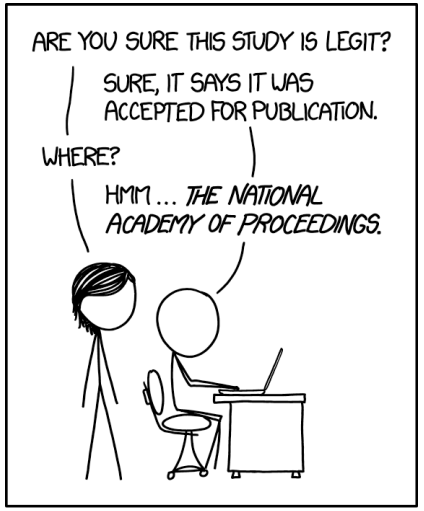
- Increasing SDNA security reduced Attack Success, Attack Productivity, and Attack Integrity.
  - Reducing permitted protocols doesn't affect network performance, but removes avenues of attack.

Table 5: SSHPass Attack Metrics for SDNA Security Levels

Configuration	Confidentiality	Success	Productivity	Integrity
No SDNA	1.0	1.0	1.0	1.0
SDNA, Low	1.0	0.4	0.6	0.4
SDNA, Med.	1.0	0.2	0.6	0.3
SDNA, High	1.0	0.0	0.5	0.0

STUDY ISSUES

- ARCSYNE and SDNA control don't match.
- Unclear if SDNA security levels run with internal or external.
- Unclear why external SDNA lowered Mission Confidentiality.
- Arguably insufficient study to show fidelity of metric system.



QUANTITATIVE METRICS FOR MTD EVALUATION

- Cyber quantitative metrics (CQM) such as Productivity, Success, Confidentiality, and Integrity can be used for MTD evaluation.

Metric	Defender	Attacker	Formula
Productivity	Rate of task completion.	Rate of exploitation.	$Prod(\mu,\nu) = \frac{1}{\lvert T\rvert}\sum_{\{t\in T\}} \nu(t, duration)$
Success	Normalized value of task completion success.	Normalized value of attack success	$Succ(\mu,\nu) = \frac{1}{\lvert T\rvert}\sum_{\{t\in T\}} \nu(t, succ) \in [0,1]$

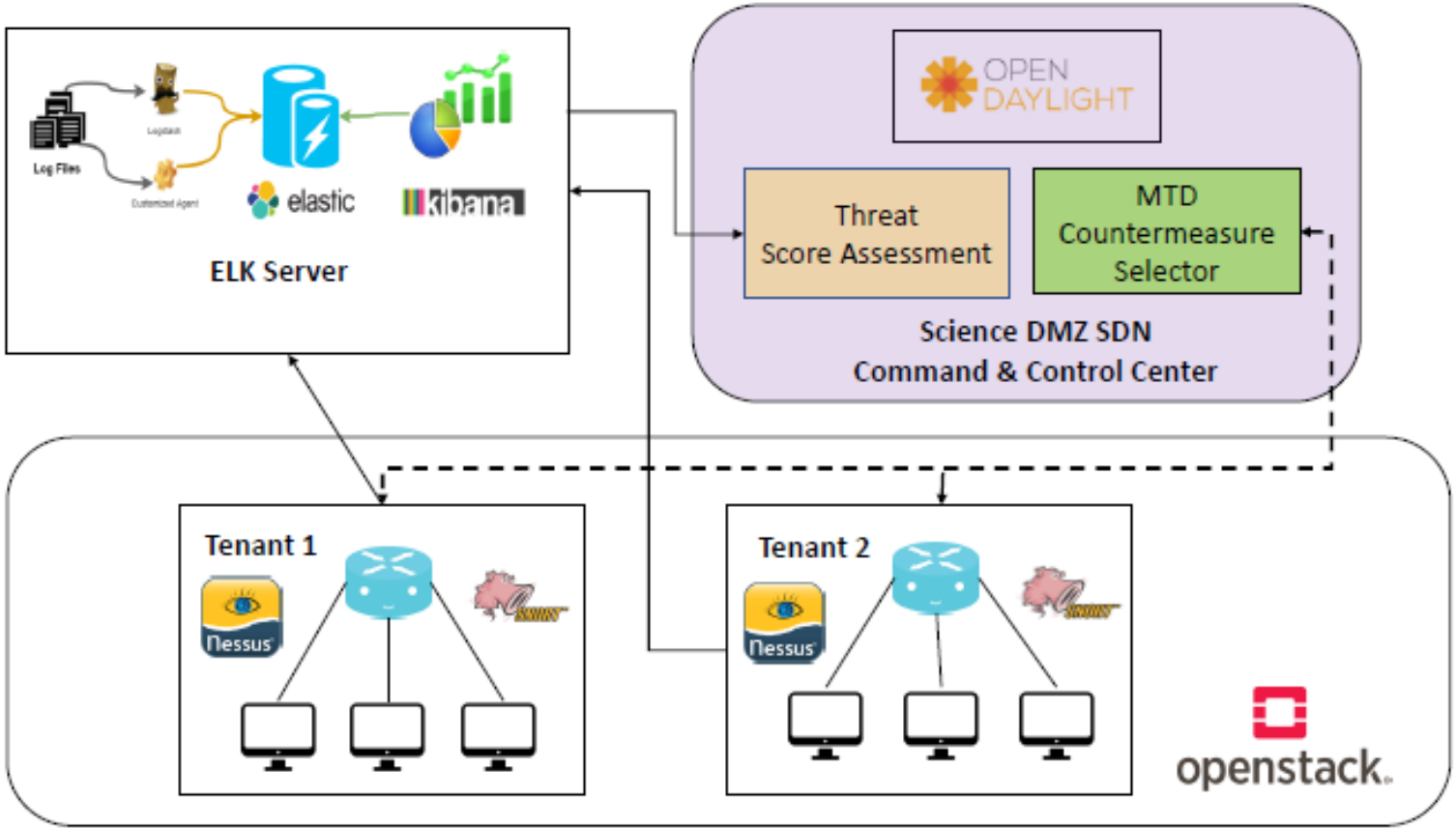
QUANTITATIVE METRICS FOR MTD EVALUATION

Metric	Defender	Attacker	Formula
Confidentiality	How much information is exposed by task.	Fraction of attack activity that can be detected.	$Con(\mu,\nu) = \frac{1}{\lvert T\rvert}\sum_{\{t\in \mathrm{T}\}} \nu(t, unexp)$
Integrity	Fraction of information produced by task that needs to be preserved.	Accuracy of the information viewed by the attacker.	$Int(\mu,\nu) = \frac{1}{\lvert T\rvert}\sum_{\{t\in \mathrm{T}\}} \nu(t, intact)$

MTD BASED ANALYSIS AND EVALUATION FRAMEWORK (MASON)

- Detects and prevents multi-stage attacks in multi-tenant cloud network managed by SDN.
- Security budget considerations for MTD deployment.
- Identification of most-critical services in the network, based on centrality in network and vulnerabilities on the service.
- Threat score calculated based on Intrusion Detection System (IDS) alerts and vulnerabilities.

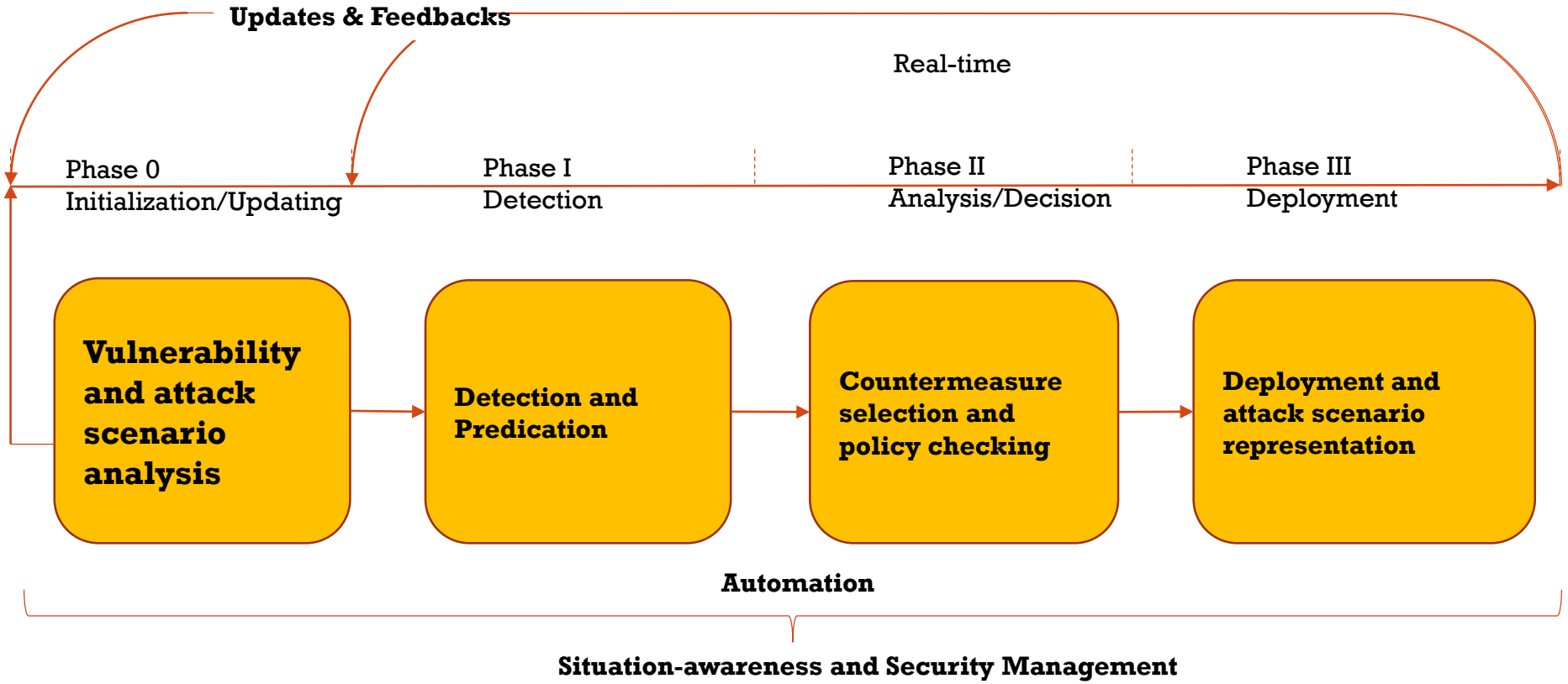
MTD BASED ANALYSIS AND EVALUATION FRAMEWORK (MASON)



MTD BASED ANALYSIS AND EVALUATION FRAMEWORK (MASON)

- Snort IDS used for intrusion detection, Nessus used for vulnerability scanning on each tenant.
- Page rank based threat scoring algorithm used for combining information from static vulnerabilities and dynamic alerts.
- ODL controller used to deploy MTD countermeasure port-hopping based on threat score.
- MTD countermeasure for top 40-50% services can reduce cumulative threat score for network by 97%.

CASE STUDY: AN INTELLIGENT SOFTWARE DEFINED ARCHITECTURE



SUMMARY

- Cyber threats pose a threat to critical infrastructure. MTD based proactive defense can help in detection, analysis and mitigation of cyber-threats.
- State of the art MTD defense mechanism can utilize techniques for information obfuscation at network level, application level, and host level.
- SDN serves as an enabling technology for deployment of MTD on a large scale cloud network. SDN controller can centrally implement MTD countermeasures such as topology reconfiguration, port hopping, IP randomization.

SUMMARY

- Game-Theoretic modeling of MTD helps in formulating attack-defense as a game, where intelligent decision models like Markov Modeling can be used for implementing MTD countermeasures.
- Cyber quantification metrics (CQM) such as confidentiality, integrity, availability and qualitative metrics such as cumulative threat score can help in assessing the effectiveness of MTD framework.
- Repo of our past MTD works (paper's and presentations): https://github.com/ankur8931/mtd-research

CITE THIS WORK

```
@book{huang2018software,
title={Software-Defined Networking and Security: From Theory to Practice},
author={Huang, Dijiang and Chowdhary, Ankur and Pisharody, Sandeep},
year={2018},
publisher={CRC Press}}
```