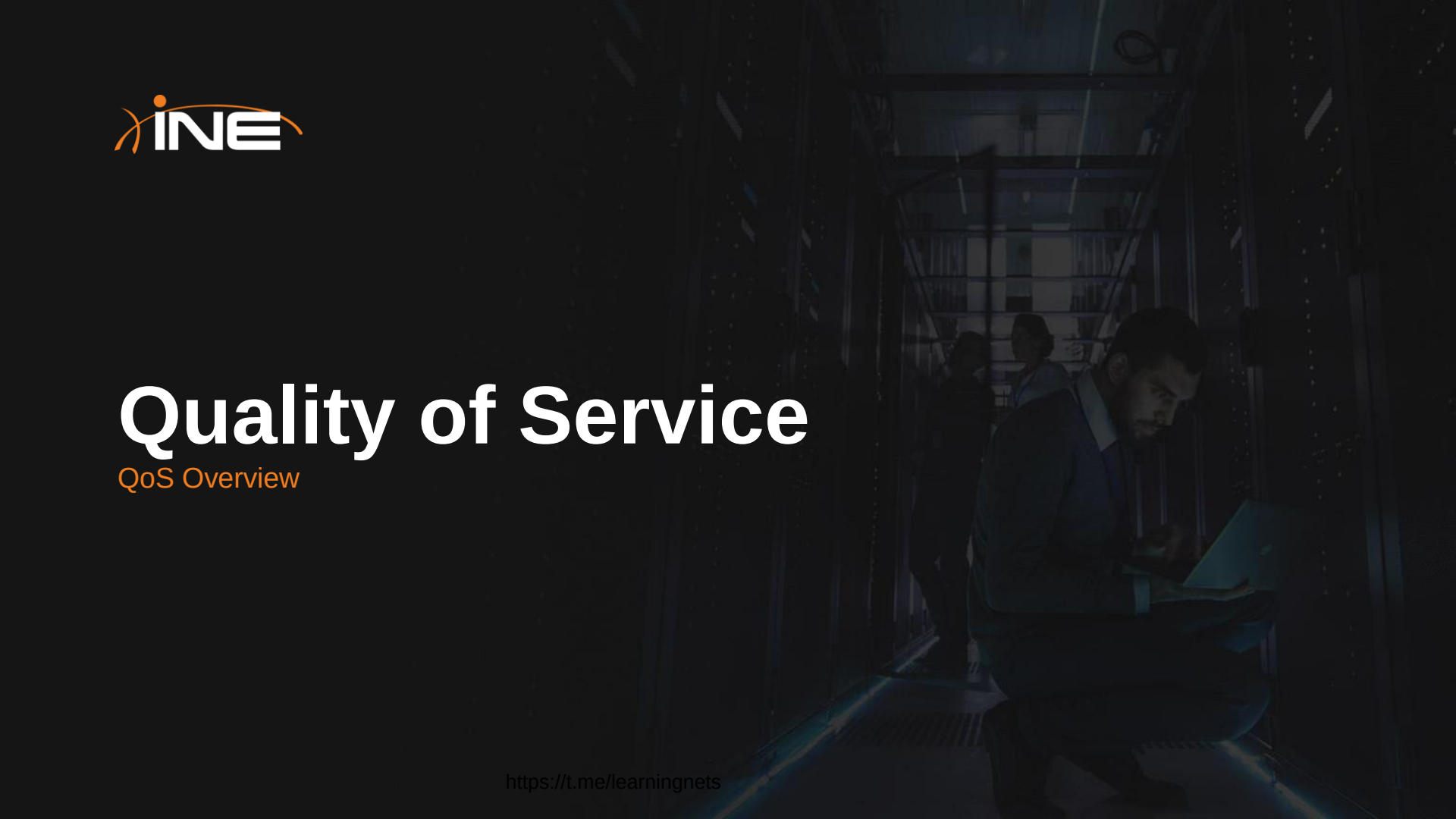


Quality of Service

QoS Overview

In This Section

- Recommended Resources
- What is QoS? Why is QoS Needed?
- QoS Models
- Classification & Marking
- QoS Tools

Recommended Resources

- Books
  - QoS-Enabled Networks: Tools and Foundations
  - End-to-End QoS Network Design
- Online Resources
  - BRKCRS-2501 - Campus QoS Design-Simplified
  - Enterprise Medianet Quality of Service Design 4.0

What is QoS?

- Quality of Service (QoS)
  - *Quality of service is the ability to provide different priority to different applications, users, or data flows, or to guarantee a certain level of performance to a data flow.*
- E.g. different service levels for different types or “classes” of traffic flows

Why is QoS Needed?

- Root Cause: Resource Contention
  - Multiple flows sharing the same link
  - Same or multiple applications
  - Each application has its own requirements
- Contention results in Queueing
  - Packets may be delayed or dropped
  - Effective flow throughput decreases
  - Delay or Jitter may exceed thresholds

Possible Solutions

- Best Solution: Avoid Contention
  - Don’t over-provision
  - Not always possible
- Next Best Solution: QoS
  - Network congestion is controlled
  - Delay/Loss/Jitter/Throughput are controlled
  - Only alleviates temporary congestion

QoS Models

- QoS model defines contention management approach
- Two types
  - Integrated Services
  - Differentiated Services

Integrated Services QoS Model

- RFC 1633 - Integrated Services in the Internet Architecture: an Overview
- What is IntServ?
  - Connection-oriented model
  - Every flow has an explicit reservation end-to-end
  - Does not scale well because network must maintain too much state
- IntServ use case is MPLS TE
  - RFC 2205 - Resource ReSerVation Protocol (RSVP)
  - RFC 3209 - RSVP-TE: Extensions to RSVP for LSP Tunnels

Differentiated Services QoS Model

- RFC 2475 - An Architecture for Differentiated Services
- What is DiffServ?
  - Connectionless model
  - Traffic is grouped into classes
  - QoS behavior is defined by traffic's class
  - Called Per-Hop Behavior (PHB)
- DiffServ is our focus

Classification & Marking

- In order for DiffServ to properly work, traffic must be placed into correct classes
  - I.e. “Classification”
- Traffic classification normally occurs at network ingress edge
  - Typically a manual process we must enforce
- Classification can be encoded inside packet itself
  - Known as packet’s “marking”

Classification Types

- Classification & Marking can happen at multiple places
- Layer 2 Class of Service (CoS)
  - 802.1q Ethernet header
- Layer 3 IP Type of Service (ToS)
  - IP Precedence & Differentiated Services Code Point (DSCP)
- Layer 4
  - TCP & UDP ports
- Upper layers
  - Network Based Application Recognition (NBAR)
  - Deep Packet Inspection (DPI)

QoS Tools

- Used to Implement QoS Models
  - Many tools rely on correct QoS classification & marking
- Different Tools for
  - Network Edge
  - Network Core
- Tools fall into three main categories
  - Admission Control
  - Congestion Management
  - Congestion Avoidance

Admission Control Techniques

- Used to enforce traffic marking or traffic rate
- Two main types
  - Traffic Policing
  - Traffic Shaping

Traffic Policing

- Used to limit inbound and outbound traffic flows
  - Traffic that exceeds the rate can be dropped, marked, or re-marked
  - Typically applied on ingress edge
- Example use case
  - PE connects to CE with GigE port
  - Circuit is provisioned at 250Mbps
  - PE applies inbound policer at port level
    - If traffic <= 250Mbps, transmit
    - If traffic > 250Mbps, drop

Traffic Shaping

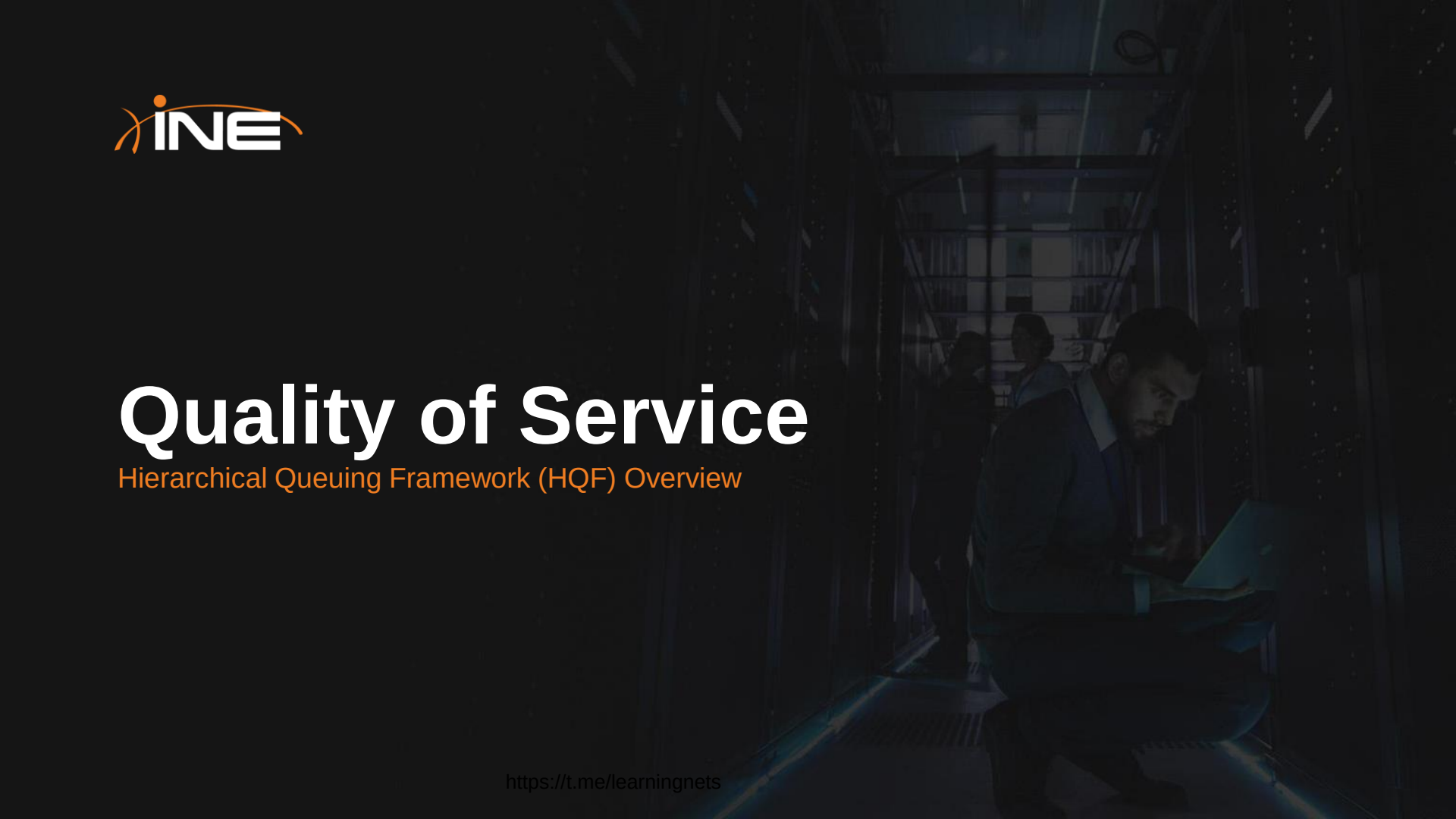
- Used to normalize outbound traffic flows
  - Smooth out traffic bursts
  - Prepares traffic for ingress policing
  - Delay and Queue exceeding traffic
- Example use case
  - PE connects to CE with GigE port
  - Circuit is provisioned at 250Mbps
  - CE applies outbound shaper at port level
    - If traffic <= 250Mbps, transmit
    - If traffic > 250Mbps, queue for later transmission

Congestion Management Techniques

- Used to deal with congestion once it occurs
  - I.e. Queueing
- Queueing types
  - First in First Out (FIFO)
  - Weighted Fair Queueing (WFQ)
  - Priority Queueing (PQ) / Low Latency Queuing (LLQ)
- Example use case
  - CE to PE link is experiencing packet loss
  - Apply LLQ to give VoIP low delay
  - Apply WFQ to guarantee 50% BW for SQL
  - All other traffic gets best effort FIFO

Congestion Avoidance Techniques

- Try to prevent congestion before it occurs
  - I.e. packet drop strategy
- Drop strategy types
  - Weighted Random Early Detection (WRED)
  - Tail Drop
- Example use case
  - CE to PE link is experiencing packet loss
  - Apply WRED to selectively drop low priority TCP flows
  - Senders go into TCP Slow Start
  - Congestion management is offloaded to the end host



Quality of Service

Hierarchical Queuing Framework (HQF) Overview

In This Section

+ Queueing Overview
+ MQC/HQF Overview
+ MQC/HQF Configuration

Queueing Overview

- Queueing occurs when packets are delayed by router
- Simplified in Ethernet switches
  - Hardware queues only
- Could be Hierarchical
  - PVC-Queue (Frame-Relay)
    - Interface-Queue (Software Queue)
      - Hardware-Queue (TX-Ring)
- “Fancy” queueing methods apply to software queue
  - How traffic is processed when waiting for TxR

MQC/HQF Overview

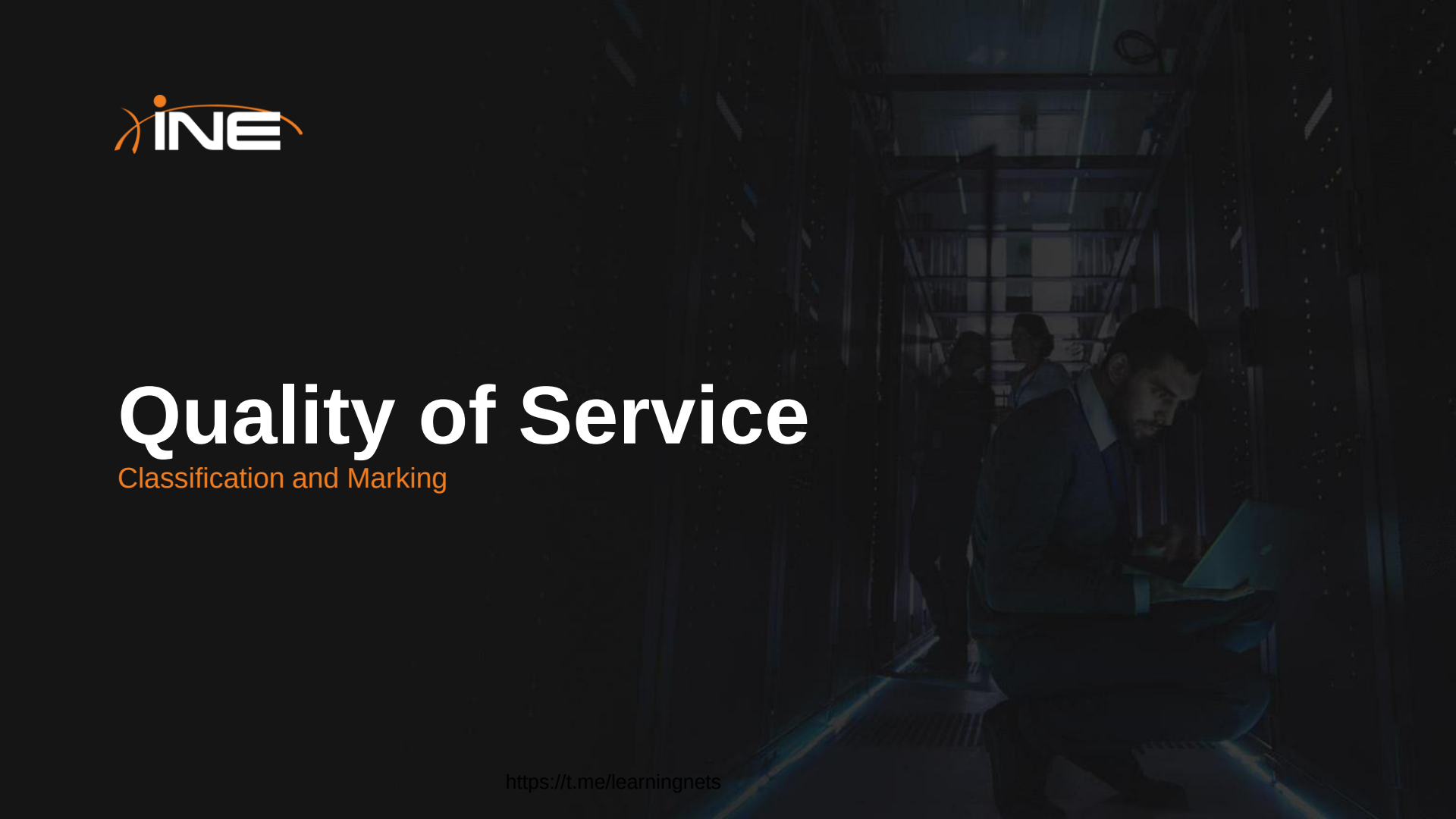
- Modular Quality of Service Command Line Interface
  - Allows multiple QoS methods per interface per direction
  - “Legacy” QoS did not
- Previously called CBWFQ
  - Class Based Weighted Fair Queueing
- Now called HQF
  - Hierarchical Queueing Framework
  - As of IOS 12.4(20)T

MQC/HQF Configuration

- Define traffic classes
  - **class-map**
  - Define traffic match criteria
- Define traffic policy
  - **policy-map**
  - Define actions
- Apply policy
  - **service-policy [in/out]** on interface

MQC Verification

- Useful commands
  - **show class-map**
  - **show run class-map**
  - **show policy-map**
  - **show run policy-map**
  - **show policy-map interface**



Quality of Service

Classification and Marking

In This Section

+ DiffServ Classification Overview
+ Types of Markings
+ Classification Configuration
+ Marking Configuration

DiffServ Classification Overview

- RFC 4594 - Configuration Guidelines for DiffServ Service Classes
- BRKCRS-2501 - Campus QoS Design-Simplified
- Medianet QoS Design Strategy – At A Glance

Types of Markings

- Layer 2 Markings
  - Frame-Relay DE bit (1 bit)
  - MPLS EXP bits (3 bits)
  - 802.1q/ISL CoS bits (3 bits)
- IPv4 & IPv6 ToS Byte
  - IP Precedence (3 bits)
  - DSCP (6 bits)

IP Precedence

- Seven classes
  - 7 - network
  - 6 - internet
  - 5 - critical
  - 4 - flash-override
  - 3 - flash
  - 2 - immediate
  - 1 - priority
  - 0 – routine

DSCP Default & EF PHBs

- Default
  - Best Effort
  - DSCP value 0 (000000)
- Expedited Forwarding (EF)
  - Priority Traffic
  - DSCP value 46 (101110)

DSCP AF PHB

- Assured Forwarding (AF)
  - Bandwidth Guaranteed
- Four classes
  - AFxy where x = 1 – 4
  - Higher is more preferred
- Three drop precedences
  - AFxy where y = 1 – 3
  - Higher means higher drop precedence
- DSCP value (xxxyy0)

DSCP CS PHB

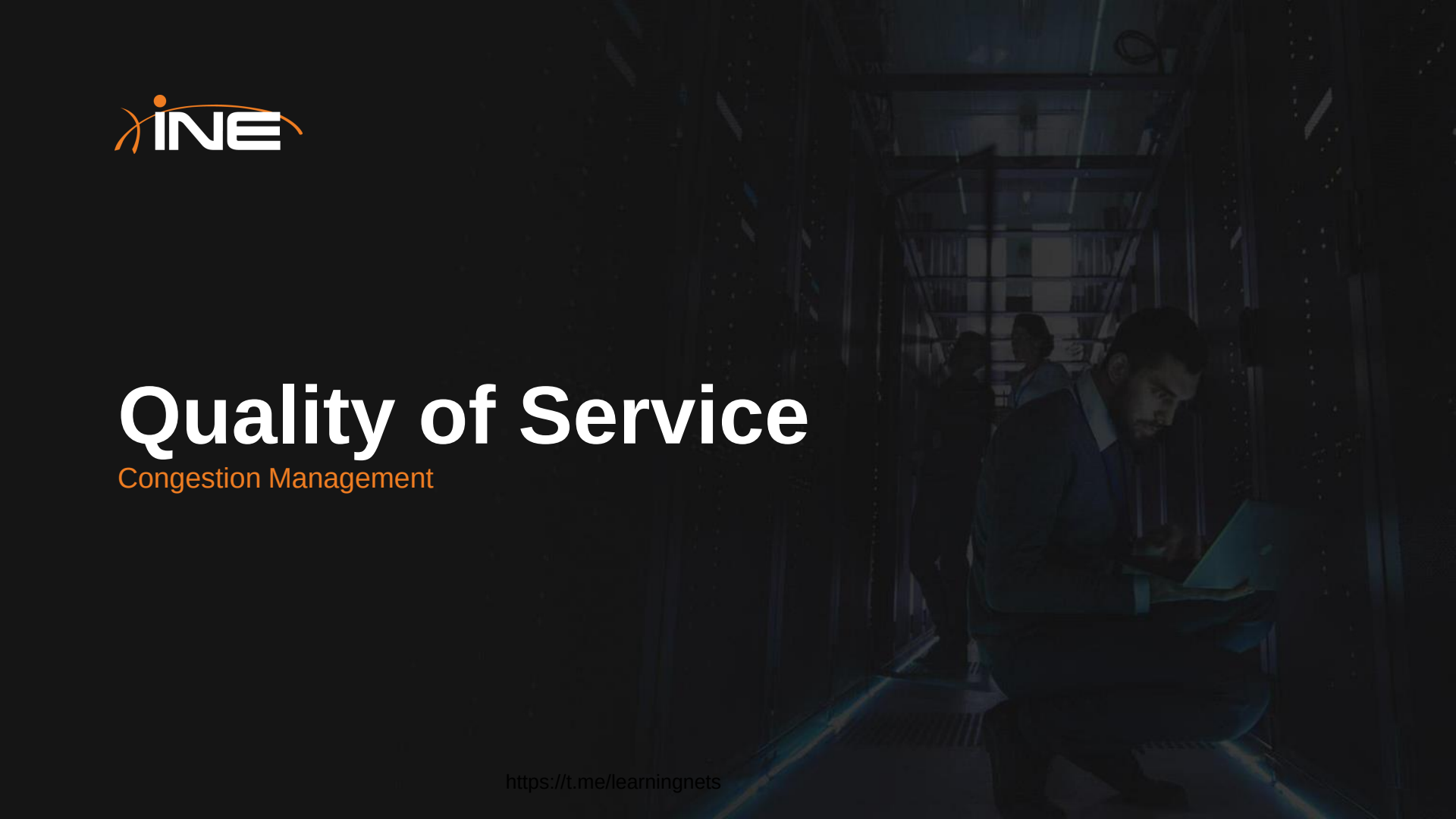
- Class Selector (CS)
  - Backwards compatible with IP Precedence
- Seven classes
  - CSx where x = 1 – 7
  - Higher is more preferred

Configuring Classification

- MQC Classification Options
  - Match-Any vs. Match-All
  - Access-Lists
  - DSCP/IP Precedence
  - NBAR
  - Source Interface
  - Source/Destination MAC address
- Can combine multiple matches in one class

Configuring Marking

- Marking can be configured both input and output
- Specifically implemented with
  - MQC/HQF policy
  - Legacy rate-limit (policer)
  - Policy Based Routing (PBR)



Quality of Service

Congestion Management

In This Section

+ FIFO Queuing
+ Weighted Fair Queueing
+ MQC/HQF Bandwidth
+ MQC/HQF Priority

FIFO Queueing

- Simplest and easiest to implement
  - Only parameter is queue-depth
- Configuration
  - Disable previous queueing strategy
    - I.e. **no fair-queue**
  - Define queue depth
    - **hold-queue out**
- Typically used as part of other solutions
  - E.g. CBWFQ/HQF

Fair Queueing

- Also known as max-min scheduling
- Services multiple requests for a shared resource
  - Step 1: Share resource equally
  - Step 2: Take excessive amounts
  - Step 3: Share excess equally among unsatisfied requests

Weighted Fair Queueing

- Max-min scheduling, but not equal
  - Allocate bandwidth per flow proportional to weight
- Flow is defined dynamically
  - Src/Dst IP + Src/Dst Port + ToS Byte
- Weight is IP Precedence + 1

Weighted Fair Queueing

- Configuration
  - **fair-queue <CDT> <QUEUES>**
  - **hold-queue out <MAX BUFFERS>**
- Congestive Discard Threshold (CDT)
  - Individual queue size threshold
- If number of flows > number of queues…
  - Flow collision occurs and queues are shared

CBWFQ/HQF

- Allows defining of custom flows
  - Class definition using MQC Syntax
  - **bandwidth** keyword defines class's "weight"
- Bandwidth is shared proportionally to weight
  - Relative sharing, not absolute reservation

CBWFQ/HQF (cont.)

- Every Queue in HQF is FIFO
  - Includes **class-default**
  - Buffer-limit with **queue-limit** command
    - Global buffer limit with hold-queue out
  - Can be turned into Fair-Queue
    - Command **fair-queue <FLOWS>**
    - All flows are equal, no weighting
    - Queue limit per flow is 1/4*queue-limit

CBWFQ/HQF (cont.)

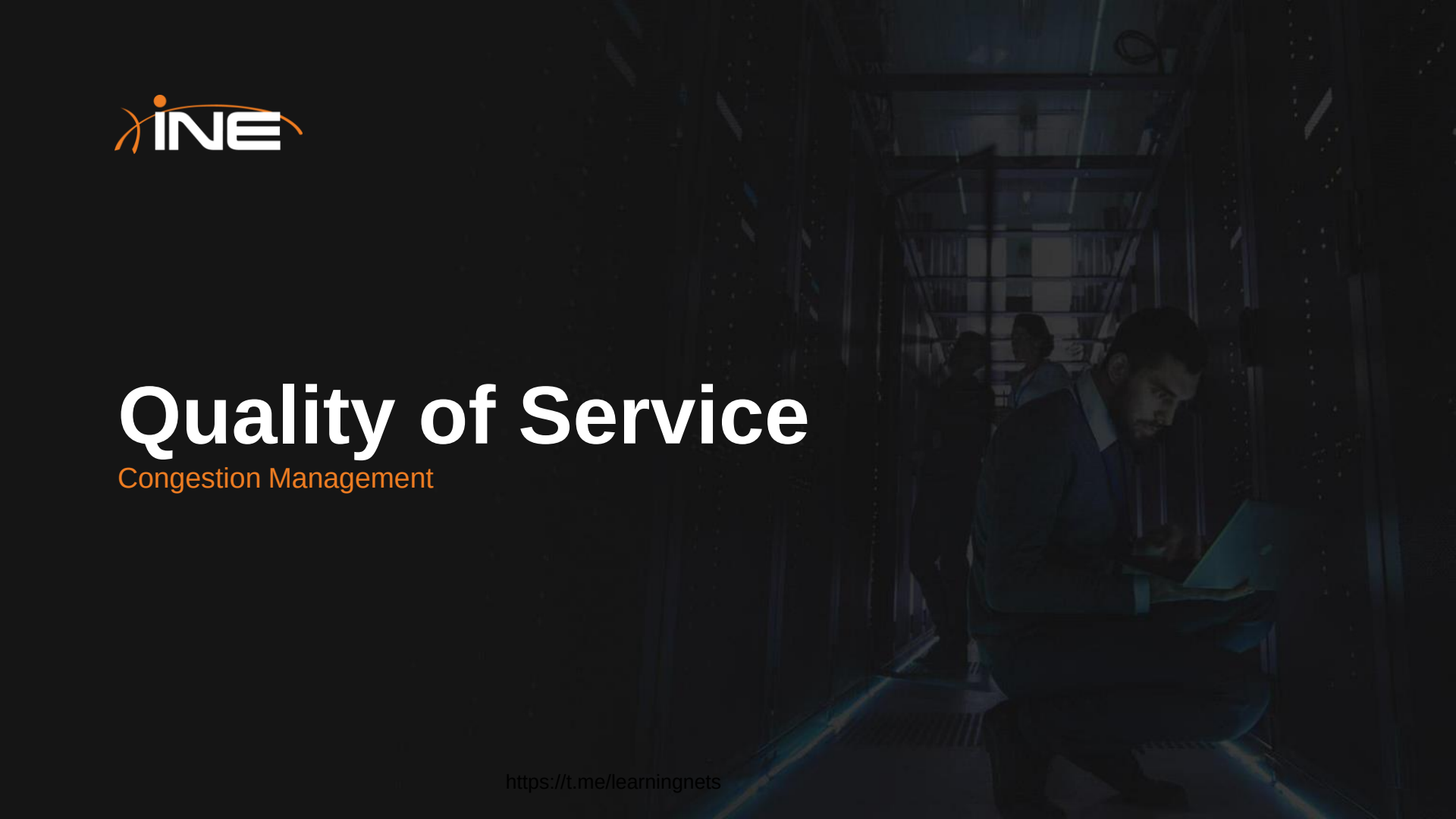
- Reservations
  - Absolute with **bandwidth [Kbps]**
  - Relative with **bandwidth percent [%]**
    - Percent of interface “bandwidth” setting
  - All bandwidths must sum to interface “bandwidth”
- Class-Default
  - Always guaranteed at least 1% of interface BW
    - max-reserved-bandwidth now deprecated

LLQ in HQF

- Priority Queue
  - Only one per HQF configuration
    - Designated with **priority [X]**
    - Always emptied first
  - Optionally policed to X Kbps only in times of congestion
    - Congestion defined as having TX-Ring full
  - Multiple classes can have priority
    - Share single queue but could be policed differently

LLQ in HQF (cont.)

- Remaining Bandwidth
  - Commonly used with LLQ
  - Bandwidth remaining after LLQ allocations
  - Command bandwidth remaining X
  - Calculated as Interface_BW - LLQ_BW



Quality of Service

Congestion Management

In This Section

+ Tail Drop
+ Random Early Detection

Tail Drop

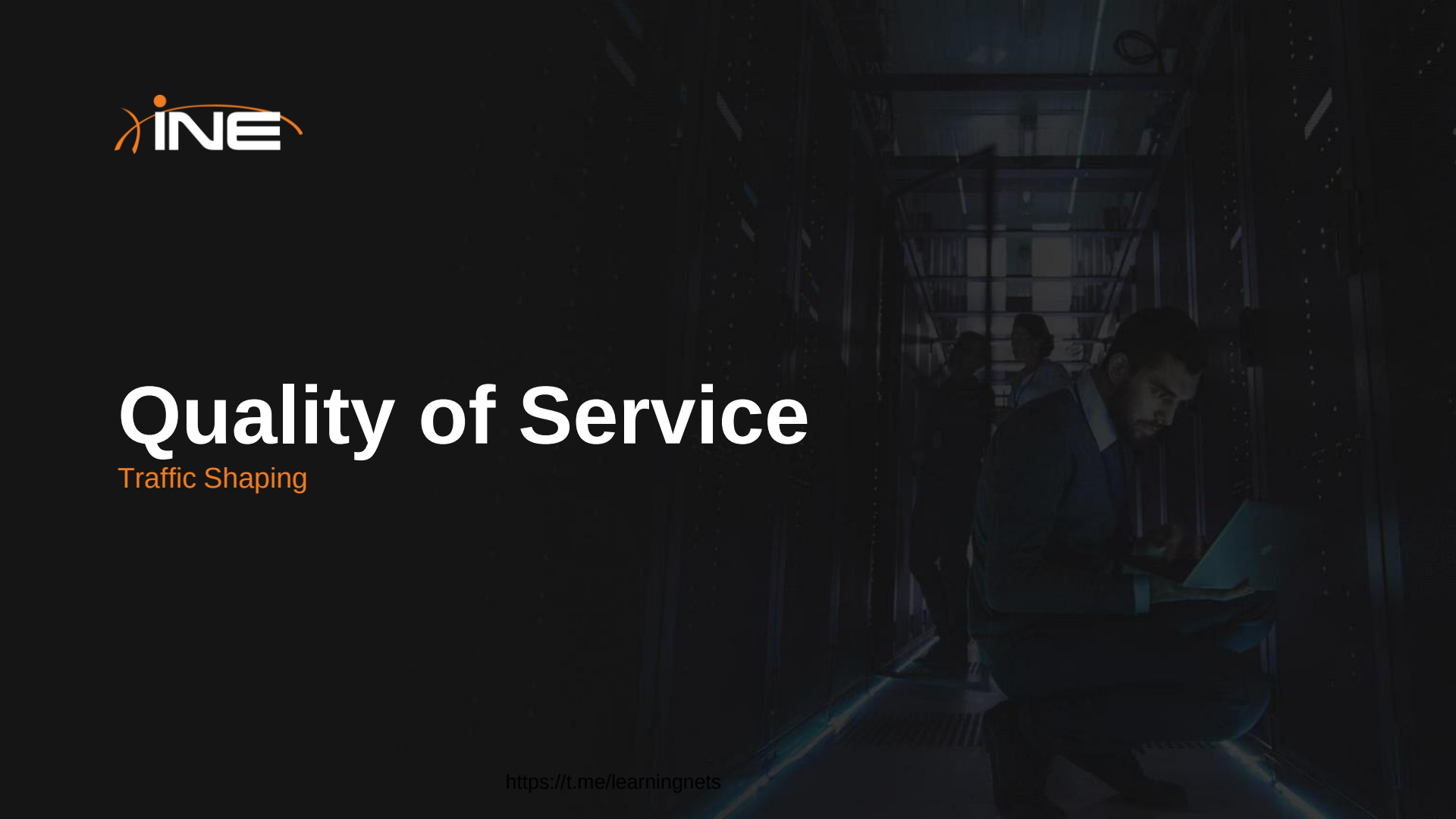
- By default, all queues use tail drop
  - When the queue is full, new packets trying to enter the tail of the queue are denied admission
- Tail drop treats all packets equally
  - No classification is performed
- Tail drop can result in global TCP synchronization
  - Segments from lots of TCP flows are simultaneously dropped
  - Senders all go into TCP slow start at the same time
  - Result is saw-tooth traffic pattern

Random Early Detection

- RED is a congestion avoidance technique
  - Selectively drop flows from the queue before the buffer is 100% full
  - Goal is to send individual senders into slow start
  - Result is more even traffic patterns
- WRED adds weighting to drop algorithm
  - Packets with higher weight are less likely to be dropped

How WRED Works

- WRED tracks average queue depth
  - Smoothened based on weight factor
  - avg=(old_avg*(1-1/2^n))+(q_size*1/2^n)
- Drops packets based on Mark Probability Denominator
  - Probability = 1/Mark_Probability_Denominator
  - Drop probability increases as queue depth increases
  - If queue depth exceeds maximum, tail drop occurs
- Configured as **random-detect**
  - Can be combined with other queueing mechanisms



Quality of Service

Traffic Shaping

In This Section

+ Traffic Shaping Overview
+ Traffic Shaping Configuration

Traffic Shaping Overview

- Goal is to normalize traffic flow
  - Smooth out traffic bursts
  - Prepares traffic for ingress policing
  - Delay and Queue exceeding traffic

Shaping Terminology

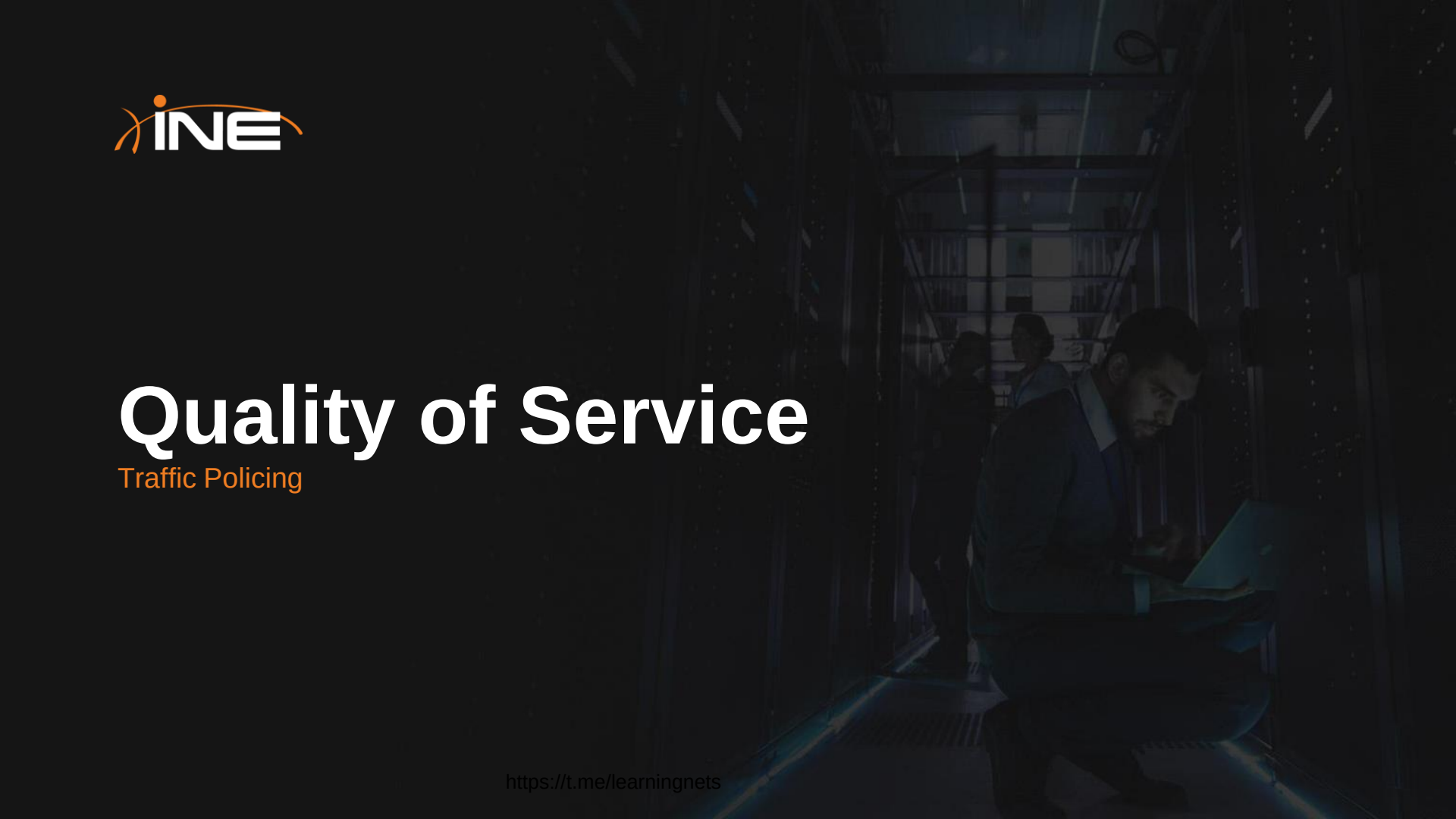
- Access Rate - AR
  - Physical port speed
- Committed Information Rate - CIR
  - Average rate the shaper is targetting
- Time Committed - Tc
  - Time interval in ms to emit traffic bursts
  - Bursts always emitted at Access Rate (AR)
- Burst Committed - Bc
  - Amount of bits that could be sent every Tc
- Burst Excessive - Be
  - Amount of bits over Bc that could be sent during Tc
  - Must be accumulated by idle periods

Shaping Formulas

- Single-Rate Shaper (sub-rate)
  - AIR = interface (port) speed
  - CIR = Bc/Tc = average speed (shaping rate)
  - EIR = (AIR-CIR) = excessive rate (sporadic)
  - Be = EIR*Tc = excessive burst
    - May be prohibited by setting Be=0

MQC Generic Traffic Shaping

- Configured using MQC syntax
  - shape average <CIR> [Bc] [Be]
  - Tc is found implicitly as Bc/CIR
- Default shaper queue is FIFO
  - Can be turned into HQF by associating a child policy-map with shaped class
  - Specify HQF settings in the child policy-map
  - I.e. nested policies



Quality of Service

Traffic Policing

In This Section

+ Traffic Policing Overview
+ Traffic Policing Configuration

Traffic Policing

- Used to meter a packet flow rate
  - Marks packets that exceed metered rate
  - Drop is a mark action
- Normally an ingress operation
  - E.g. PE ingress from CE
- Policing has two parameters
  - Metering rate – CIR
  - Averaging interval – Tc

Traffic Policing Parameters

- The larger the Tc the more bursting is allowed
  - Bc = CIR*Tc is maximum burst size allowed momentarily (in bytes)
- Be – excessive burst
  - Max amount of bytes allowed above Bc during Tc
  - Only allowed if Bc was not fully utilized before

Single-Rate Policing Syntax

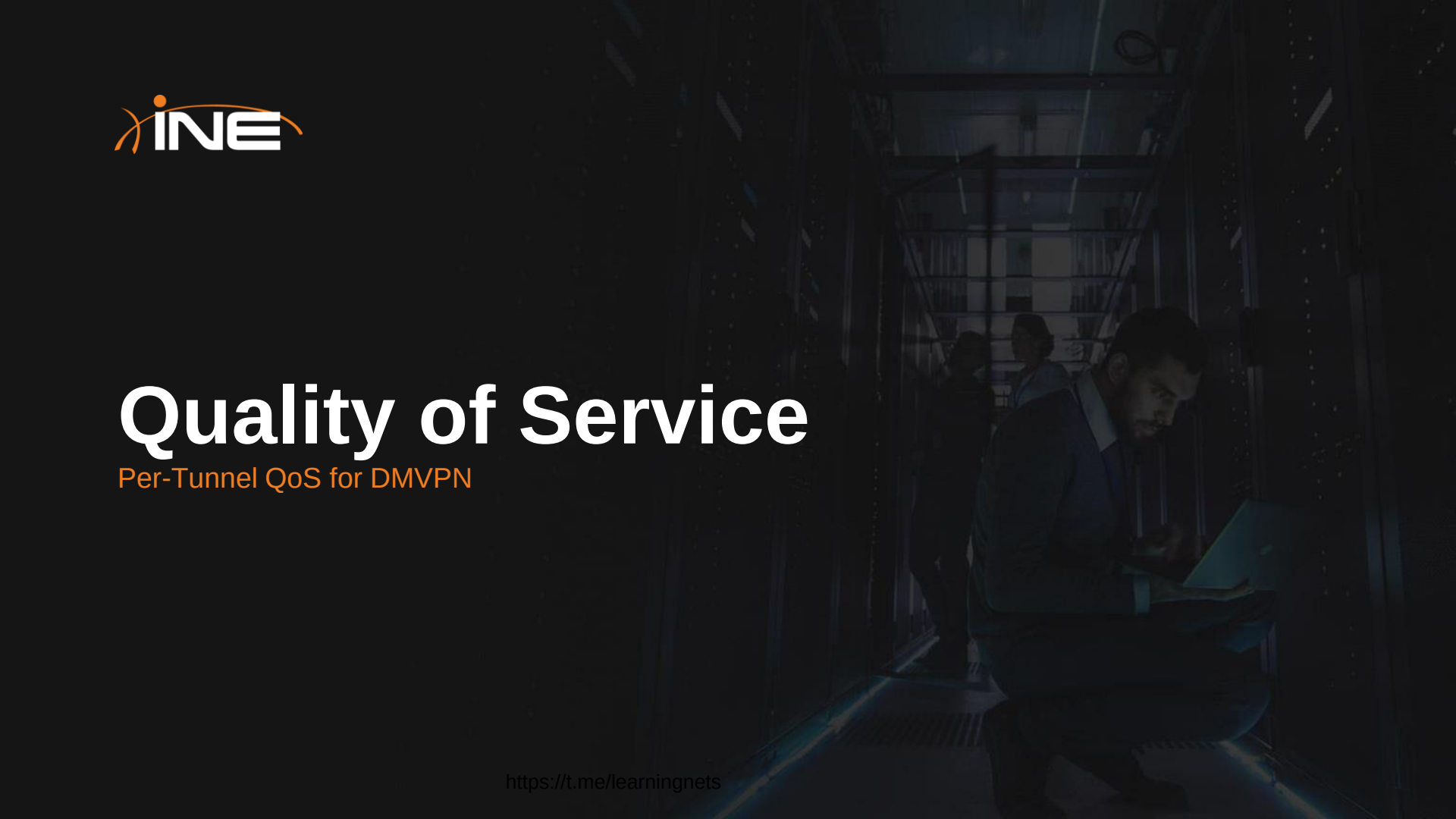
- Configuration
  - police [cir] [<CIR>] [<Bc>] [<Be>]
  - CIR in bps while bursts are in bytes
- Applied to an MQC class
  - Three actions (colors): conform, exceed, violate
  - Exceed: flow exceeds Bc but under Bc+Be
  - Violate: burst size exceeds Bc+Be

Dual-Rate Policing Syntax

- Configuration
  - police cir [<CIR>] bc [<Bc>] pir [<PIR>] be [<Be>]
- Normally used to implement two-rate access
  - Customer is guaranteed CIR
  - Allowed to send up to PIR
  - Traffic between PIR and CIR remarked
    - E.g. lower DSCP

Shaping and Policing Together

- Operations are Complimentary
  - Shaping is done egress
  - Policing is done ingress
- Parameters should match
  - Shaping is set to match policing
  - Same CIR, Bc and Be
    - Policing values could be greater actually



Quality of Service

Per-Tunnel QoS for DMVPN

In This Section

+ DMVPN QoS Design Issues
+ Per-Tunnel QoS for DMVPN

DMVPN QoS Design Issues

- With legacy hub-and-spoke, QoS was applied per-VC
  - E.g. per DLCI policy on Frame Relay
  - Result is that hub can queue separately to each spoke
- With DMVPN, all spokes exist on the same Tunnel and underlay interface
  - How can be apply different rates and fancy queues per-spoke?
- Solution – Per-tunnel QoS for DMVPN

Per-Tunnel QoS for DMVPN

- Spoke signals QoS group to hub through NHRP
  - **ip nhrp attribute group [group]**
  - Group name must match between hub and spoke(s)
- QoS group name maps to QoS template on hub
  - **nhrp map group [group] service-policy output [policy]**
  - Result is that each spoke has a separate QoS policy
- Verified as **show policy-map multipoint** on the hub