Implementation and Performance Analysis of SNMP on a TLS/TCP Base.

Du, Shayman and Rozenblitz

Sarwar S Raza

WPI – CS 577

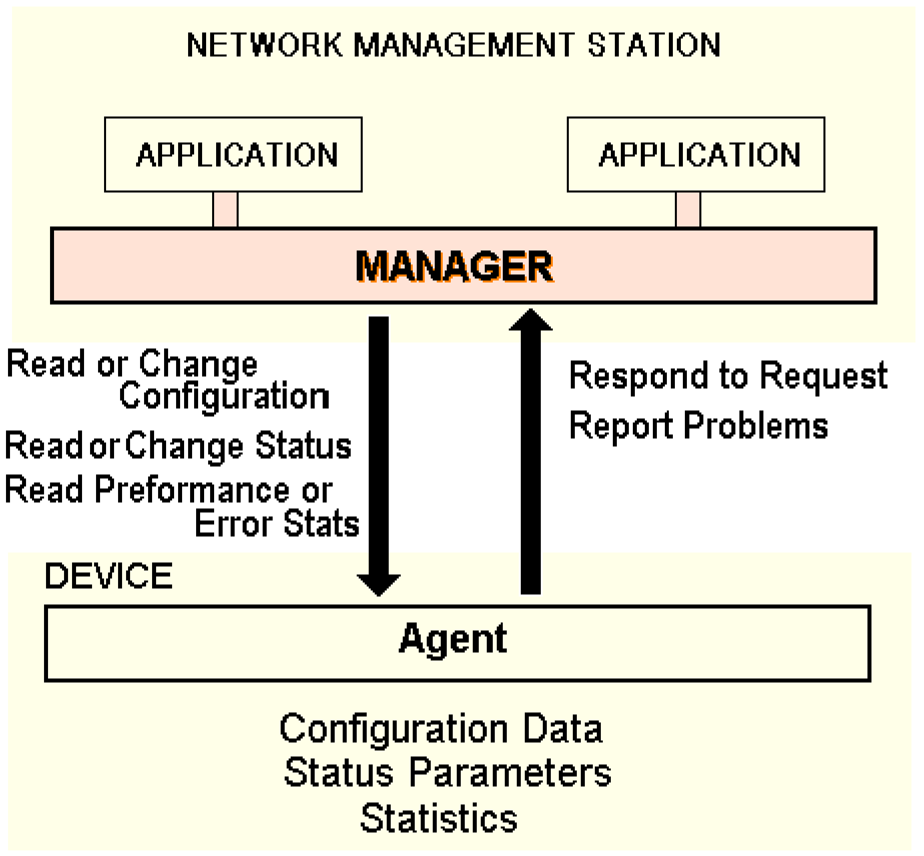
Outline

- Overview of SNMP
- Introduction
- TLS
- Implementation of SNMP/TLS/TCP
- Performance comparisons
- Conclusions

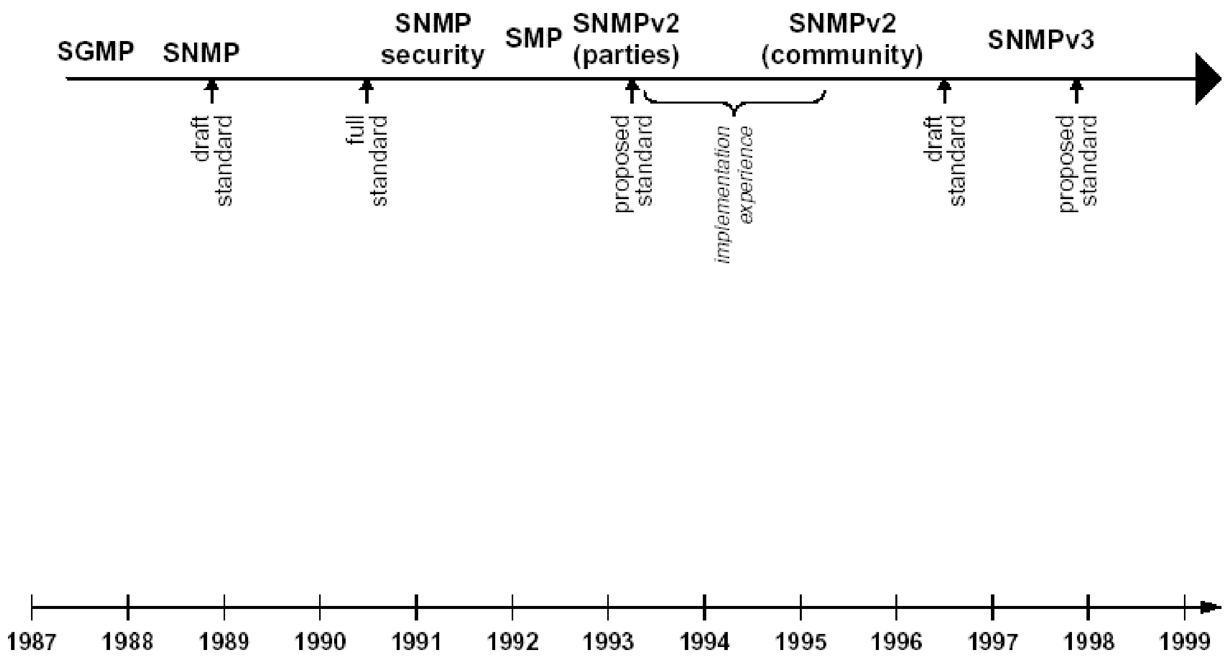
Network Management

- **A <u>Network Management System</u> is composed of:**
- Network elements or nodes containing a processing entity called an agent, responsible for performing the management functions requested by the management station.
- Management applications which monitor, configure and control managed elements.
- A management protocol used to communicate management information between the management stations and the agents in the network elements.
- Management information.

Network Management



SNMP: A brief history



Simple Network Management Protocol

- SNMP is the prevailing standard for management of TCP/IP networks. SNMP is layered on top of UDP, the User Datagram Protocol.
- An SNMP management station monitors and controls a managed node by issuing requests directed to the agent residing in the managed node. The agent interprets the request and performs the function accordingly.
- All SNMP transactions take place using PDUs (Protocol Data Units).

Simple Network Management Protocol

- IETF RFCs 1155, 1156, and 1157 define the Simple Network Management Protocol (SNMP). The Internet community developed SNMP to allow diverse network objects to participate in a global network management architecture. Network managing systems can poll network entities implementing SNMP for information relevant to a particular network management implementation. Network management systems learn of problems by receiving traps or change notices from network devices implementing SNMP.

Choice of underlying protocol

- UDP has been chosen and recommended for SNMP transport protocol. This is fine because at the beginning, SNMP was targeted at managing Internet nodes and the predominant Internet protocol suite TCP/IP . The choice of TCP/IP suite continues to make sense because IP became the protocol for commercial backbone networks. And users can count on a TCP/IP implementation available on any type of host and router.

Choice of underlying protocol

- TCP and UDP provide transport services. But UDP was preferred. This is due to TCP characteristics, it is a relatively complicated protocol and consumes more memory and CPU resources, whereas a UDP stack is easy to build and run. Device vendors need only have built simple version of IP and UDP on their devices. Thus the software requirements are kept simple enough, and, can, in most cases, be stored in ROM. UDP is well suited to the brief request / response message used in network management communication.

SNMP PDU types

- There are 4 types of request PDUs:

- **GET** get a specific name or instance
- **GET-NEXT** get the next-object thate follows the given name/instance
- **SET** set variables(s)
- **TRAP** report a trap event that has occurred.

SNMP Message format

Version	Community	PDU

PDU Format

PDU type	Request ID	Error status	Error index	Object 1, value 1	Object 2, value 2	…

The GET and GETNEXT PDUs

- The GET and GET-NEXT PDUs consists of pairs of variables to get, and a value of null for those variables.
- The generate a RESPONSE PDU, where the null value is replaced with the data that was requested, or an error code is sent back.
- The variable/value pairs are called *variable bindings*. Multiple varbinds may be enclosed in a single PDU.

The SET PDU

- The SET PDU consists of pairs of variables to set, and the value that the network operator wants to assign to those variables.
- It generates a RESPONSE PDU from the agent, that contains success/error status and the value of the variable that was just SET.
- Though returning back the same value is redundant, it allows for the SET-REPONSE PDU to be virtually identical to the GET/GET-NEXT RESPONSE PDU.

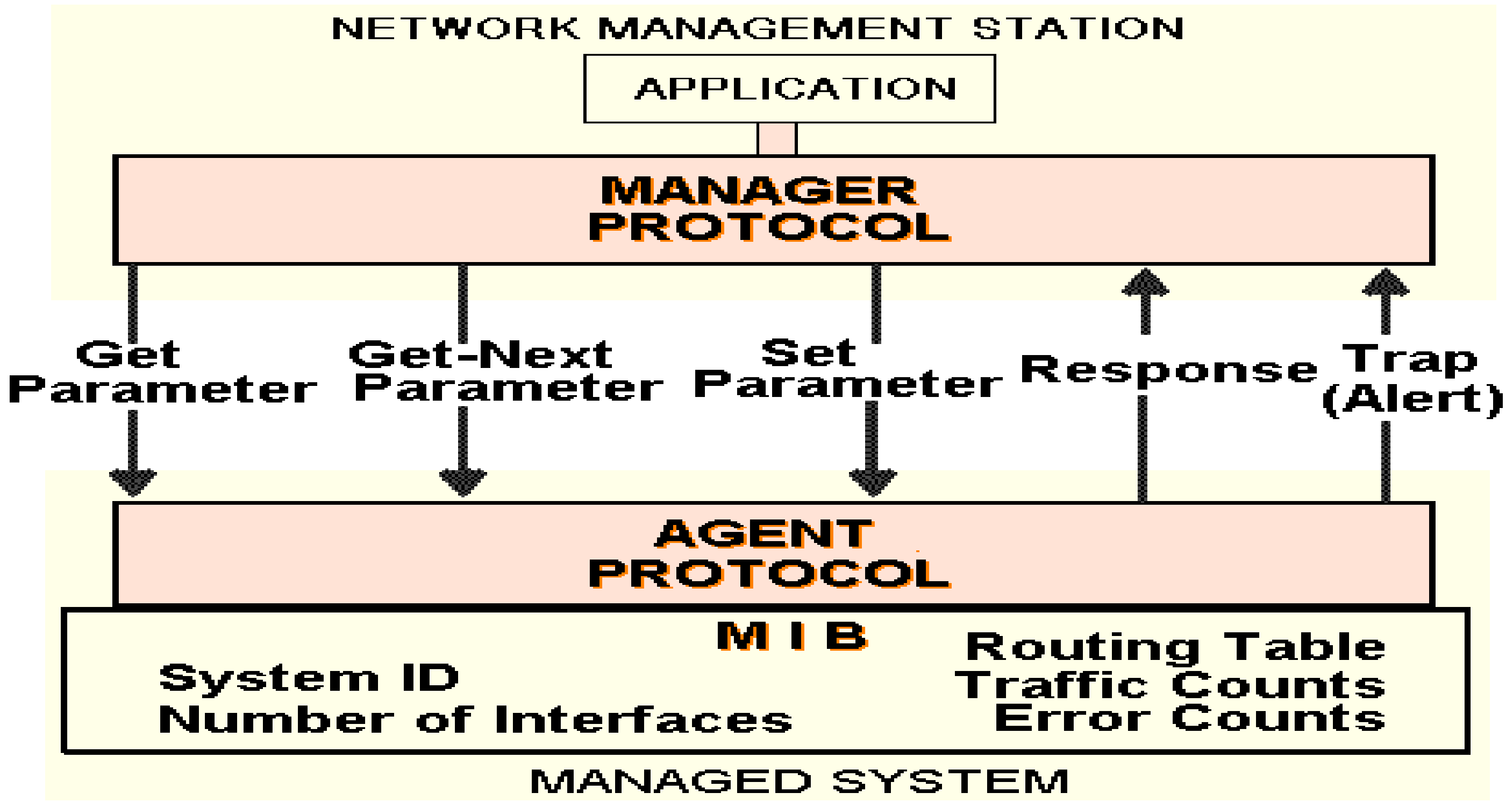
The TRAP PDU

- An agent can *asynchronously* send an unsolicited TRAP to the management application to signal an extraordinary event.

Management Information Base

- Defines the management information that is exchanged between the managed node and the management application.
- A unit of managed information, referred to previously as a variable, is called a *Managed Object.*
- A MIB is a collection of Managed Objects.
- MIBs are specified in ASN.1, a somewhat primitive data declaration language.

The complete picture



WPI

Example of use

- You want to bring a switch port from a state of *Down* to a state of *Up*.
- The Switch port state is identified by a variable ‘SwPortState’ where:

<u>Down</u> = 0

<u>Up</u> = 1.

Example

- We can first look at the <u>current</u> status of the variable using an SNMP get-request.

- GET: variable = SwPortState, value = null
- GET Response: variable = SwPortState, value = 0, error = no error

Example

- To change the value of the variable, use an SNMP set request.

- SET: variable = SwPortState, value = 1
- SET Response: variable = SwPortState, value =1, error = no error.

Example

- After the set has been completed, the AGENT will run handlers on the device that will change the port state from down(0) to up(1).

Security in SNMP

- SNMP v1 – very limited security
- Security in SNMP is commonly referred to as *trivial authentication.*
- You must know the device’s IP address in order to talk to it.
- Your must also know the *community string*, a “password” that is sent in **clear text** as part of the SNMP message.

Security improvements – SNMP V3

- SNMPv3 provides encryption and authentication as part of the core protocol. Specifically, SNMPv3 with USM (User based security model) recognizes three levels of security:

1. Without authentication and without privacy (**noAuthNoPriv**)
2. With authentication but without privacy (**authNoPriv**)
3. With authentication and privacy (**authPriv**)

The authors' premise…

- When large amounts of data need to be transferred, they must be transported using small-sized SNMP over UDP messages which result in excessive latency.
- Transporting SNMP over TCP reduces the latency by removing the limitation on message size and by allowing several segments of data to be in transit at the same time (due to TCP window mechanism).

The authors' premise…

- TCP has the additional advantage of taking care of retransmission. This GREATLY simplifies management applications since retransmission need not be implemented at the application level, but can be relegated to the transport protocol.

Protocol level Security options

- SNMP over UDP → IPSec at layer 3
- SNMP over TCP → TLS

TLS Introduction

- TLS = **T**ransport **L**ayer **S**ecurity
- A protocol that provides communication security over the Internet.
- Is aimed at preventing eavesdropping, tampering and message forgery between client server communications.
- Based on SSLv3, and is an IETF standard.

TLS Layers

- The protocol is composed of two layers:

1. The TLS Record Protocol
2. The TLS Handshake protocol, TLS Change Cipher Specification Protocol and TLS Alert Protocol.

TLS Record Protocol

- Provides connection security with two basic properties:

1. Connection privacy. Symmetric cryptography is used for data encryption (e.g. DES, 3DES,RC4). Encryption can be turned off.
2. The connection is securely reliable. Message transport includes a keyed cryptographic message authentication check (MAC).

TLS Handshake Protocol

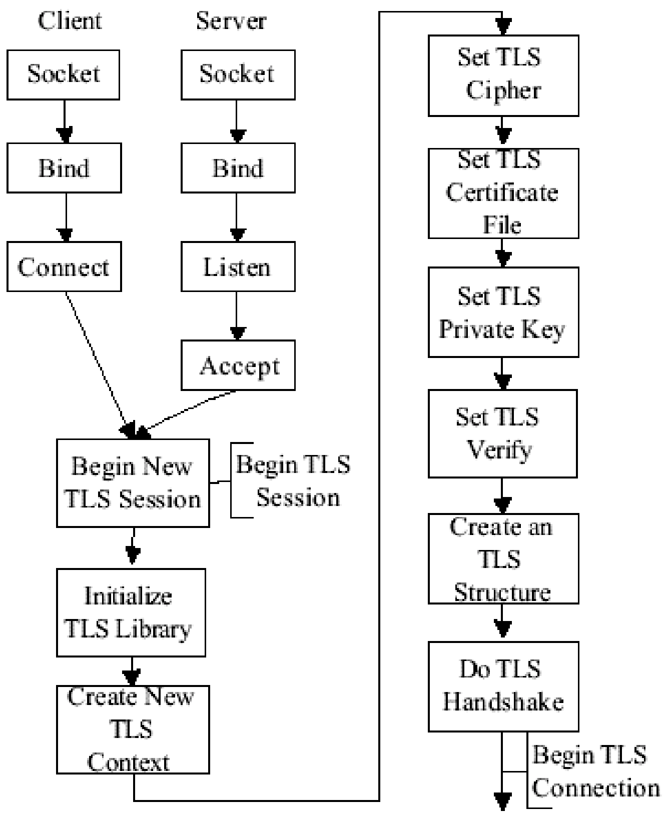
- Allows the server and client to authenticate each other and to negotiate an encryption algorithm and cryptographic keys before the application protocol transmits or receives its first byte of data.

TLS Handshake protocol

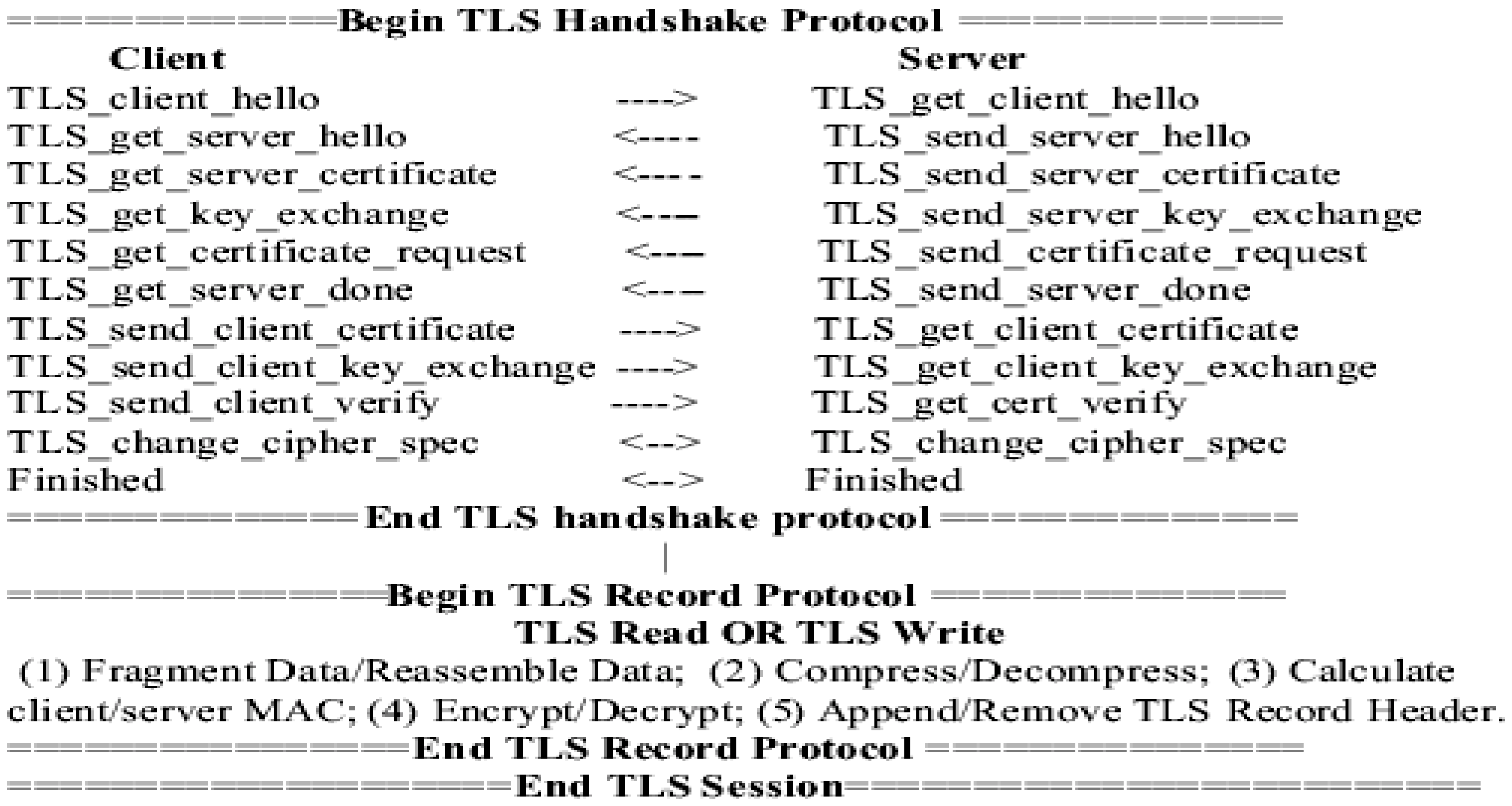
- Allows peers to authenticate their identities, using asymmetric, or public key cryptography.
- Man in the middle attacks can be thwarted – the negotiation of a shared secret is secure.
- The negotiation is reliable. It is not possible to modify the traffic being communicated without one or both parties being alerted.

Implementation of SNMP/TLS/TCP

1. Setup TCP connection
2. Setup TLS over TCP
3. Begin client server communication
4. TLS Change Ciper Spec Protocol causes the pending ceipher state to be copied into current cipher state
5. TLS Alert protocol is used to convey TLS related alerts to peer



TLS Handshake and Record Protocols



============**Begin TLS Handshake Protocol** ============

Client		Server
TLS_client_hello	---->	TLS_get_client_hello
TLS_get_server_hello	<----	TLS_send_server_hello
TLS_get_server_certificate	<----	TLS_send_server_certificate
TLS_get_key_exchange	<---	TLS_send_server_key_exchange
TLS_get_certificate_request	<---	TLS_send_certificate_request
TLS_get_server_done	<---	TLS_send_server_done
TLS_send_client_certificate	---->	TLS_get_client_certificate
TLS_send_client_key_exchange	---->	TLS_get_client_key_exchange
TLS_send_client_verify	---->	TLS_get_cert_verify
TLS_change_cipher_spec	<-->	TLS_change_cipher_spec
Finished	<-->	Finished

=============**End TLS handshake protocol** =============

|

==============**Begin TLS Record Protocol** ==============

TLS Read OR TLS Write

(1) Fragment Data/Reassemble Data; (2) Compress/Decompress; (3) Calculate client/server MAC; (4) Encrypt/Decrypt; (5) Append/Remove TLS Record Header.

===============**End TLS Record Protocol** ===============

===================**End TLS Session**=======================

Performance Tests and Results

- Measurement environment:

Network: Ethernet 10Mbit

Hardware: One Sun Sparc 10 workstation (manager), 128 MB RAM; One Sun Sparc 5 workstation (agent), 128 MB RAM.

Software: Solaris 2.6, UCD-SNMP agent

SNMP/TLS/TCP implementation

Overhead of TLS Security

Measuring overhead of TLS security

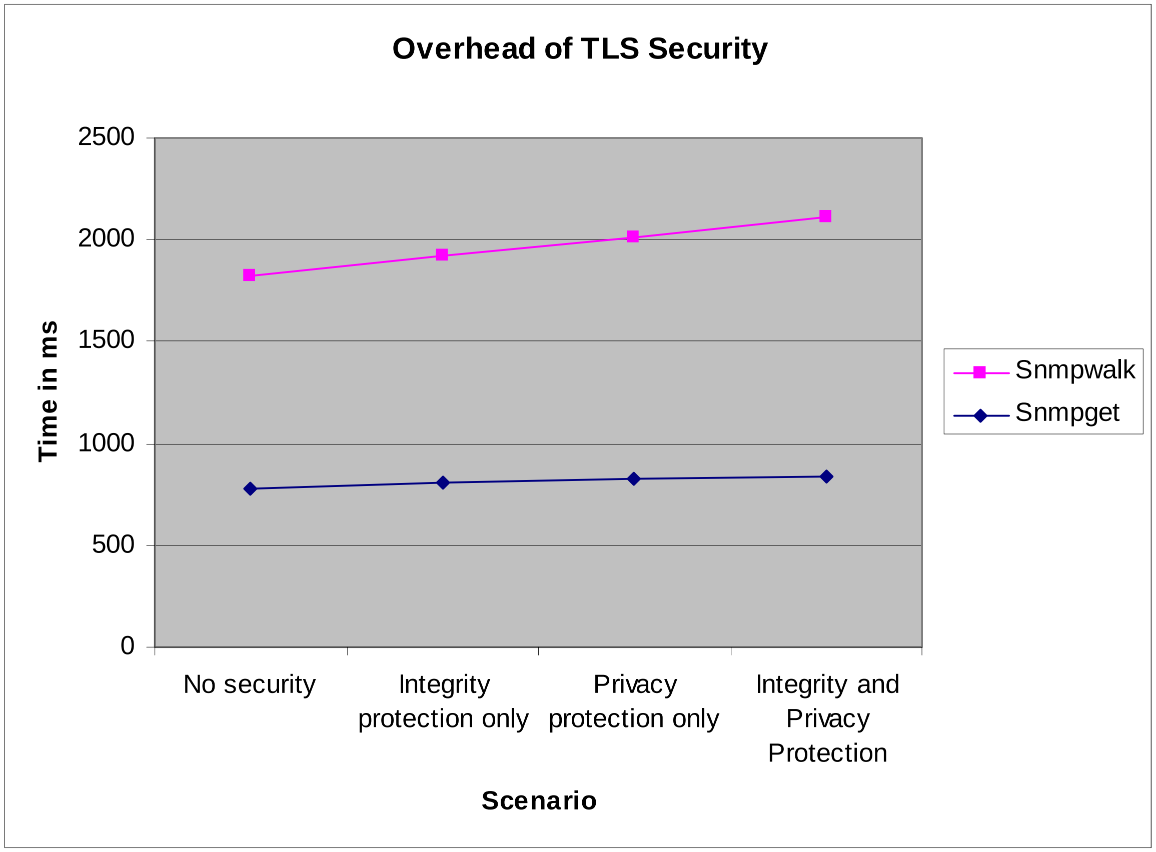
4 scenarios:

1. No security, no compression, no MAC, no encryption
2. Integrity protection only:no compression, has MAC, no encryption
3. Privacy protection only: has compression, no MAC, has encryption
4. Integrity and Privacy Protection: has compression, has MAC, has encryption.

Overhead of TLS Security

- Tests were performed for short sessions (single message, single SNMP variable queried) and for longer sessions, where GET-NEXT was used to walk across an object group in a MIB (34 objects grouped under System).

Overhead of TLS Security



Analysis

- For the short session, Integrity protection takes 4.01% of the session time, provacy protection about 4.52%. Total security takes 8.53% of the session time.
- For the long session, integrity protection takes 7.28%, privacy protection 14.66% and total security 21.94% of the session time.

Analysis

- Longer latency times for long session can be explained as follows:
- The setup times for SNMP, TCP and TLS are incurred only once per session. However, MAC and encryption overheads are incurred for each message in the session.
- NOTE: actual latency per message actually decreases for longer messages.

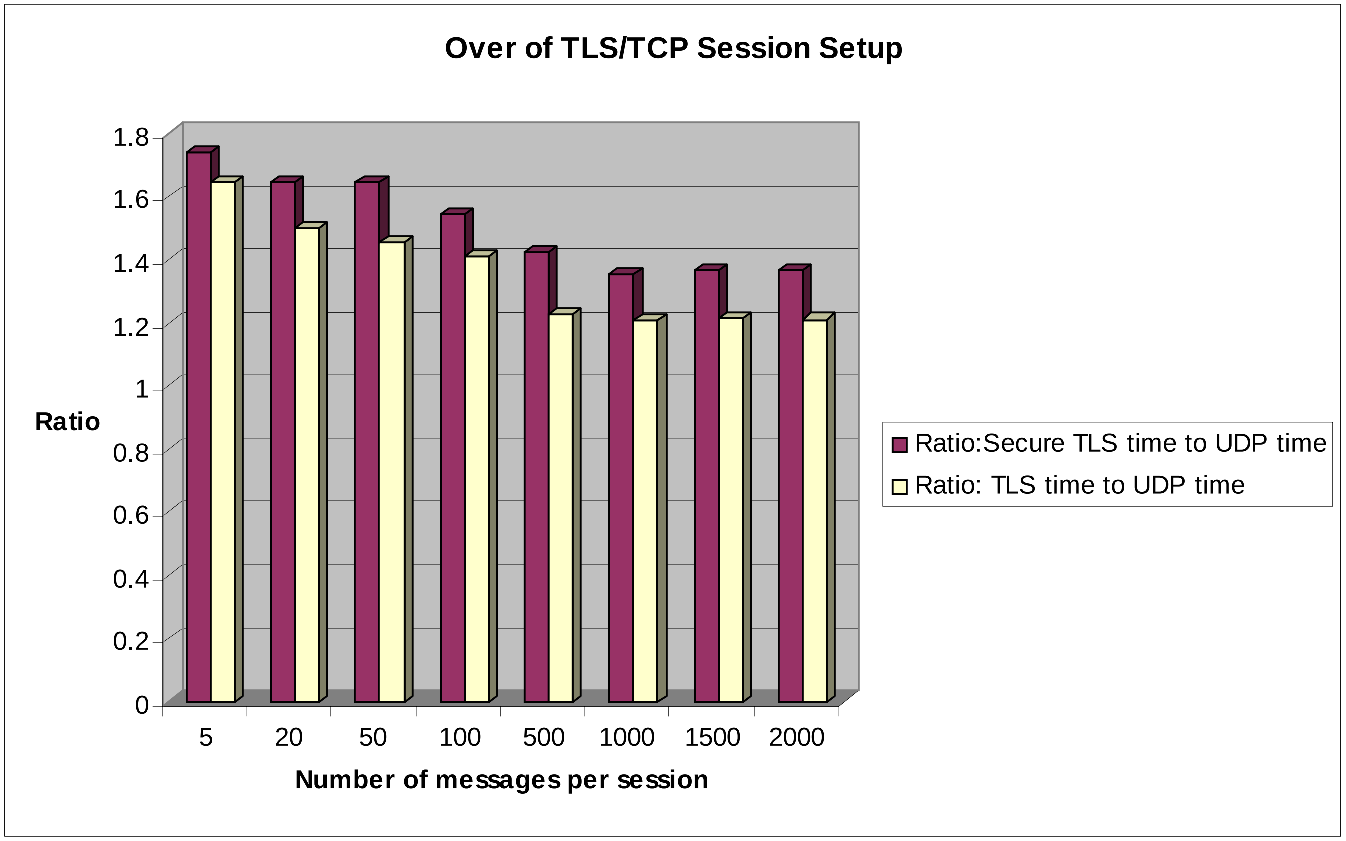
Overhead of TLS/TCP Session Setup

- SNMP/UDP does not incur a setup penalty analogous to the setup of TLS/TCP, where the TLS handshake protocol is used for the client and server to authenticate each other.
- For long sessions, however, this ONE TIME setup cost gets amortized over a large number of messages and only amounts to a low amount of overhead per message.

Test setup

- Since SNMPv1/UPD has no security, for testing purposes, it was first only to SNMPv1/TLS/TCP with peer authentication, but no integrity or privacy protection.
- TLS with full security also used in separate comparison.
- TLS setup time found to be constant: 300 ms

Analysis



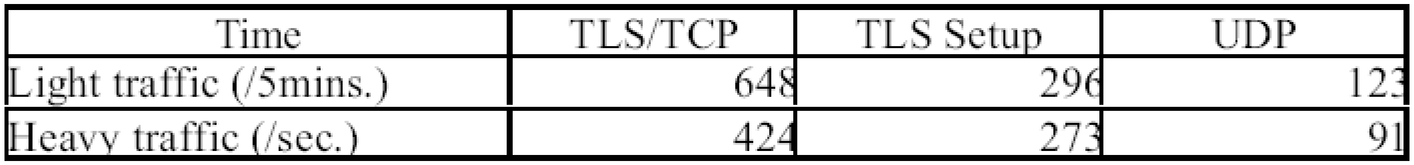
Over of TLS/TCP Session Setup
Ratio
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
5
20
50
100
500
1000
1500
2000
Number of messages per session
Ratio:Secure TLS time to UDP time
Ratio: TLS time to UDP time

Analysis

- When the number of messages in one session is small, the TLS message time is about 1.4~1.6 times the UDP message time. As the number of messages increases, the ration declines to approximately 1.2 for sessions containing at least 500 messages.
- For cases where number of messages > 500, the ratio of the per message time for TCS and UDP becomes essentially constant.

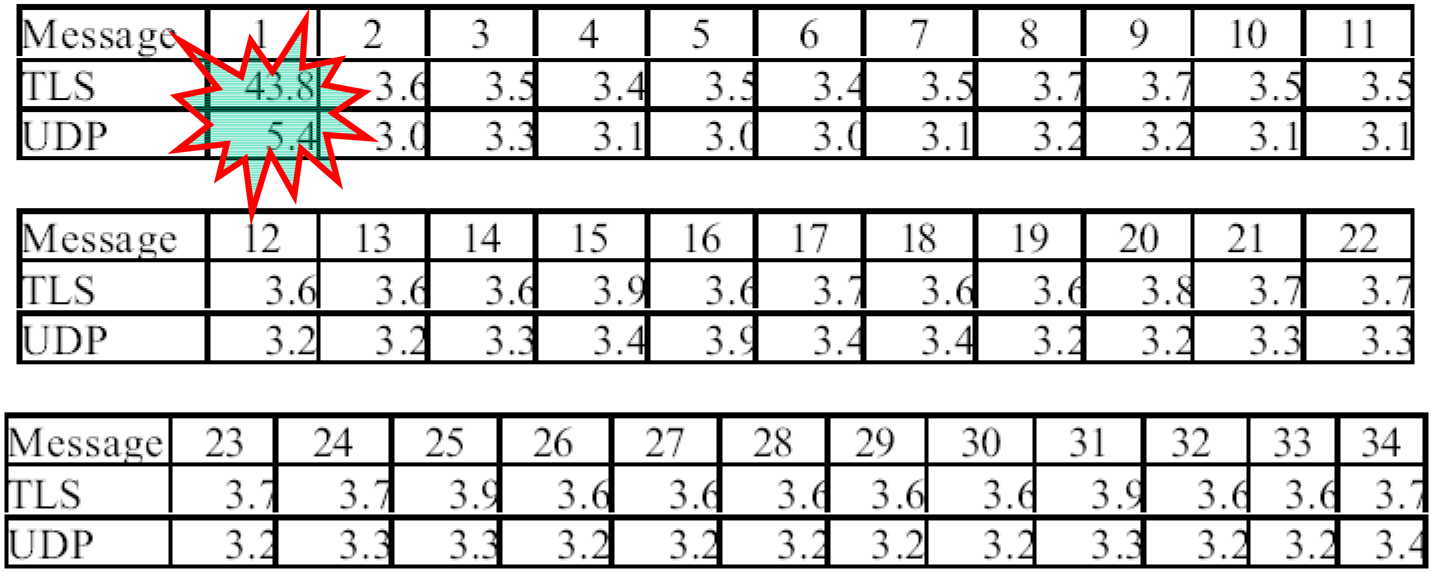
Light vs. Heavy traffic

- Light traffic – one SNMP transaction every 5 minutes
- Heavy traffic – one SNMP transaction per second.
- NOTE: for TLS/TCP, each transaction requires new TLS session.



Time	TLS/TCP	TLS Setup	UDP
Light traffic (/5mins.)	648	296	123
Heavy traffic (/sec.)	424	273	91

Packet by Packet Timing analysis



Message	1	2	3	4	5	6	7	8	9	10	11
TLS	43.8	3.6	3.5	3.4	3.5	3.4	3.5	3.7	3.7	3.5	3.5
UDP	5.4	3.0	3.3	3.1	3.0	3.0	3.1	3.2	3.2	3.1	3.1

Message	12	13	14	15	16	17	18	19	20	21	22
TLS	3.6	3.6	3.6	3.9	3.6	3.7	3.6	3.6	3.8	3.7	3.7
UDP	3.2	3.2	3.3	3.4	3.9	3.4	3.4	3.2	3.2	3.3	3.3

Message	23	24	25	26	27	28	29	30	31	32	33	34
TLS	3.7	3.7	3.9	3.6	3.6	3.6	3.6	3.6	3.9	3.6	3.6	3.7
UDP	3.2	3.3	3.3	3.2	3.2	3.2	3.2	3.2	3.3	3.2	3.2	3.4

Analysis

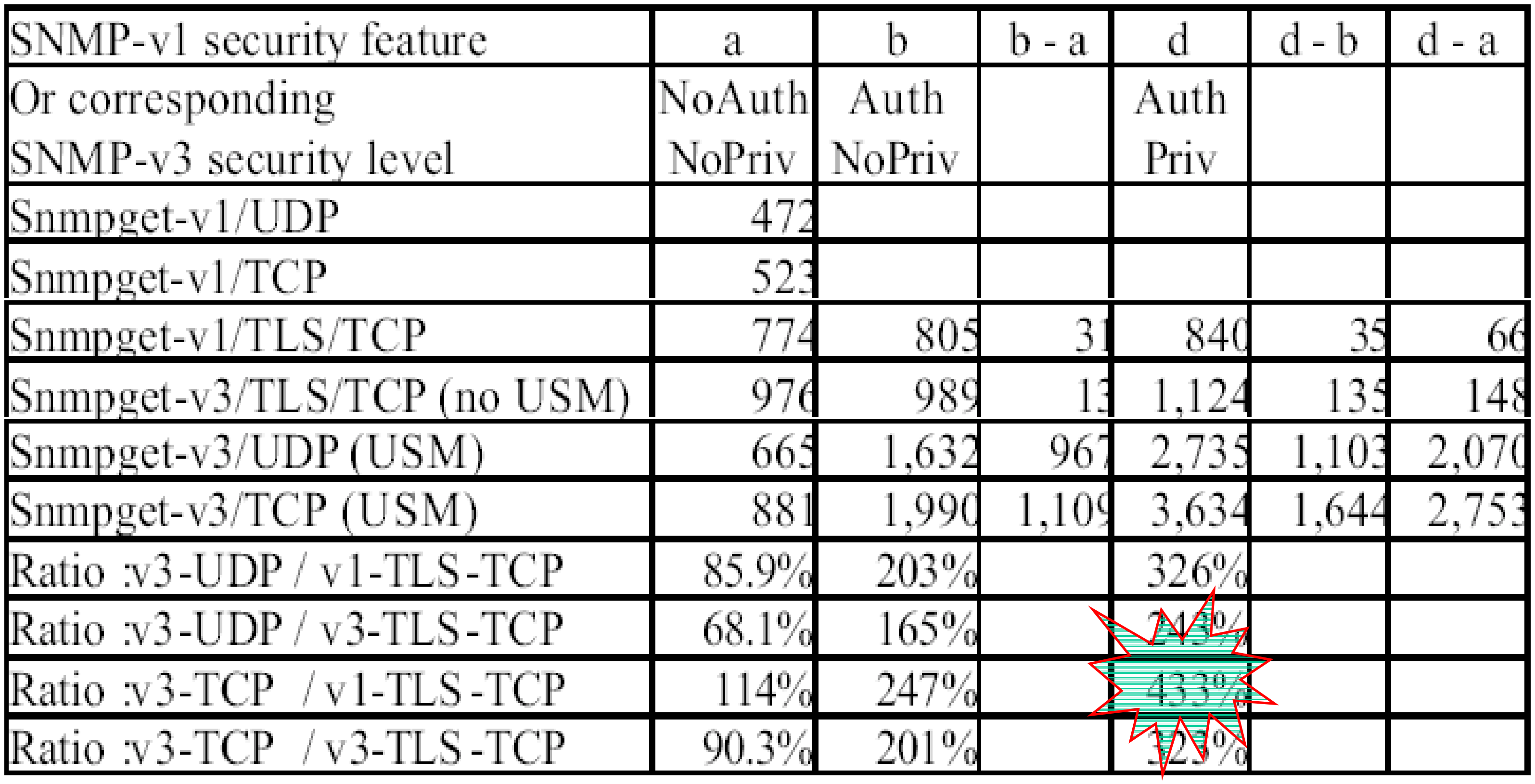
- Individual message exchange times are larger in TLS/TCP than under UDP. This is because of the TLS/TCP session has to implement the TLS record protocol when sending and receiving data. Thus for each message, the following takes place:

1. Fragment data/reassemble data
2. Compress/decompress
3. Calculate client/server MAC
4. Encrypt/Decrypt
5. Append/Remove TLS Record header.

Comparison of SNMP/TLS/TCP w/ SNMPv3/UDP and SNMPv3/TCP

- MD5 is used as the authentication protocol
- DES is used as the encryption algorithm
- All SNMPv3/TLS/TCP and SNMPv1/TLS/TCP are without USM
- SNMPv3/TCP and SNMPv3/UDP have USM.

Snmpget – short sessions



SNMP-v1 security feature	a	b	b - a	d	d - b	d - a
Or corresponding SNMP-v3 security level	NoAuth NoPriv	Auth NoPriv		Auth Priv		
Snmpget-v1/UDP	472					
Snmpget-v1/TCP	523					
Snmpget-v1/TLS/TCP	774	805	31	840	35	66
Snmpget-v3/TLS/TCP (no USM)	976	989	13	1,124	135	148
Snmpget-v3/UDP (USM)	665	1,632	967	2,735	1,103	2,070
Snmpget-v3/TCP (USM)	881	1,990	1,109	3,634	1,644	2,753
Ratio :v3-UDP / v1-TLS-TCP	85.9%	203%		326%		
Ratio :v3-UDP / v3-TLS-TCP	68.1%	165%		243%		
Ratio :v3-TCP / v1-TLS-TCP	114%	247%		433%		
Ratio :v3-TCP / v3-TLS-TCP	90.3%	201%		323%		

WPI

Snmpwalk – long sessions

SNMP-v1 security feature	a	b	b - a	d	d - b	d - a
Or corresponding SNMP-v3 security level	NoAuth NoPriv	Auth NoPriv		Auth Priv		
Snmpwalk-v1/UDP	678					
Snmpwalk-v1/TCP	762					
Snmpwalk-v1 TLS/TCP	1,044	1,120	76	1,273	153	229
Snmpwalk-v3/TLS/TCP (no USM)	1,063	1,135	72	1,323	188	260
Snmpwalk-v3/UDP (USM)	648	1,848	1,200	2,976	1,128	2,328
Snmpwalk-v3/TCP (USM)	947	2,025	1,078	3,305	1,280	2,358
Ratio :v3-UDP / v1-TLS-TCP	62.1%	165%		234%		
Ratio :v3-UDP / v3-TLS-TCP	60.9%	163%		225%		
Ratio :v3-TCP / v1-TLS-TCP	90.7%	181%		260%		
Ratio :v3-TCP / v3-TLS-TCP	89.1%	178%		249%		

Analysis

- With minimal security, SNMPv3/UDP always faster than SNMPv3/TCP
- Addition of security makes a HUGE difference.
- In summary, SNMP/TLS/TCP without USM far more efficient than using SNMPv3 (with USM) over TCP and UDP, with **same security features.**

Conclusions

- Authors assert that session setup overhead and per message security overhead **not excessive** (A subjective claim). They see SNMP/TLS/TCP as a viable choice for secure, reliable network management.
- Comparisons with SNMPv3: Authors assert that it is not at all clear to them to what degree the advantages of are ‘structural’, and to what degree they are affected by relative levels of code optimization in their implementation.
- Further experience with different software implementation is required to verify generality of results.