

The Junior Woodchuck's Guide to Using TLS on z/VM

Or: SSL—ELI5 (Explained Like I'm Five)

A Brian Hugenbruch Talk, Delivered By:

Montana Alexandra Lee, z/VM Networking Developer
montanalee@ibm.com

Agenda

Introduction

Encryption and Certificates – Huh?

Using TLS on z/VM

Use-Cases for Enabling TLS on z/VM

Time for questions (or a nap)

"Junior Woodchuck"?

- "The Guidebook contains information on **lost treasure, a complete survival guide, extensive historical and technical information** [...] However, **it does not contain information that a Junior Woodchuck is already supposed to know** [...] nor does it contain information on allegedly non-existent things"
- "Information is readily available by searching the extensive index;
a key skill of a Junior Woodchuck is being able to retrieve information quickly from the Woodchuck book in the midst of a dangerous situation,

such as **a bear attack,**

an **earthquake,**

falling out of an airplane sans parachute,

or being **swallowed by a crocodile.** "



© 1954, Walt Disney Corporation

Agenda

Introduction

What is TLS?

- **Short answer:** Encrypted data transfer from one machine to another.

- **Long answers:**
 - Transport Layer Security, or TLS (RFC 5246 for TLS 1.2, RFC 8446 for TLS 1.3)
 - Secure Sockets Layer, or SSL (preceded the RFC)
 - Sometimes collectively called SSL, or TLS, or TLS/SSL, or SSL/TLS
 - I only care which one you say if you're talking about an active connection.
 - Uses digital certificates to arrange a combination of asymmetric and symmetric encryption so that two endpoints can agree upon:
 - who the other side is and
 - how to share data back and forth
 - Can get complicated in a hurry

- **We use it because we want our data to be safe**

Why is TLS?

- The internet is a scary place!
- Information should be free--except when it comes to my bank account
- Servers are usually not next to one another—data must pass back and forth
- IBM Z is cool, but how do I trust all the x86 racks or mobile talking to it?
- So **how do I protect my data safely**, without exposing how I encrypt it?

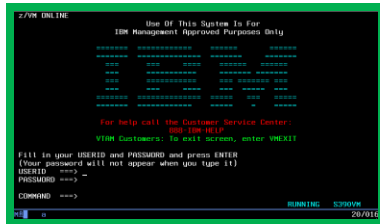
Why TLS on z/VM?

- As a z/VM Systems Programmer, I want my security people to stop pestering me so I can get work done.
- **Your hypervisor is an attack vector**
 - Seriously. I’m sorry, it’s true.
 - I don’t care if it’s just a test system
 - I don’t care if there’s a firewall
 - ...or a VPN
 - If I can see your system, I can try to crack it. TLS stops me.
- **“Because I said so.”**
 - Modern enterprise security (a fancy way of saying “how we do our jobs well”) relies on encryption of both data at rest and data in flight.
 - The people who make rules added TLS to the “you gotta” ruleset
 - World governments, industry regulations, legal standards
 - Even IBM is pushing things closer to “security by default”
- **To use cool new things!**
 - Direct-to-host service requires TLS on z/VM
 - IBM Z MFA already requires TLS on z/VM
 - This trend is likely to continue

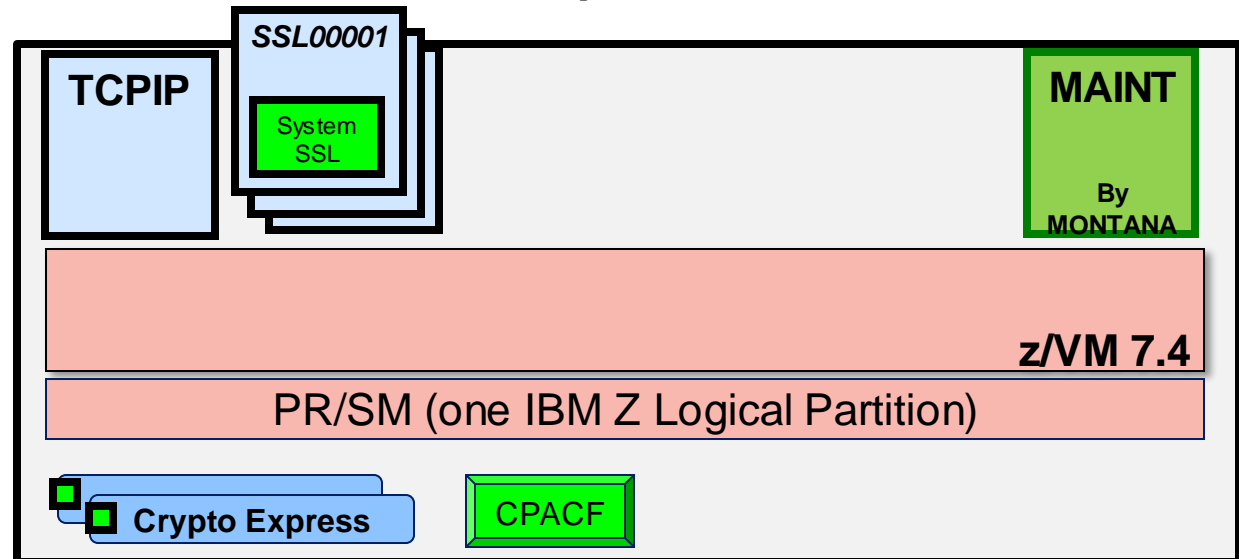
How is TLS? (1 of 5)

- Here's how it works. Two computers want to speak to one another:
 - montana.pok.ibm.com, a z/VM V7.4 system running on an IBM z17.
 - Your laptop. We'll call it "Bob."

"Bob"

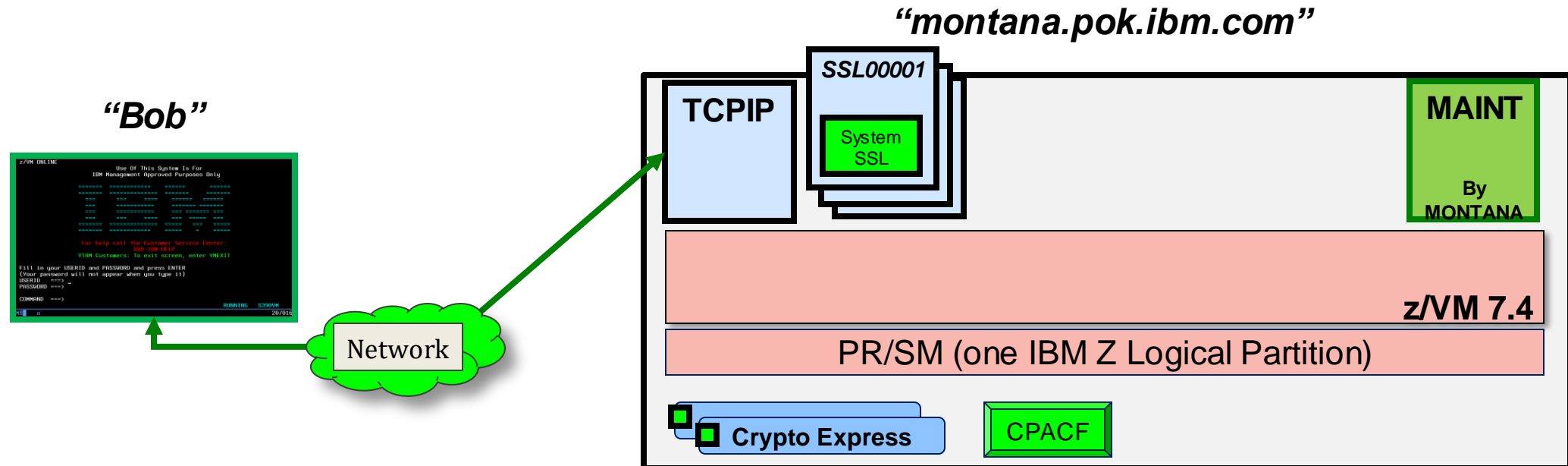


"montana.pok.ibm.com"



How is TLS? (2 of 5)

- Bob reaches out to z/VM and says, “Hi, I’d love to chat. We cool?”
- z/VM says, “Sure, but we **must** move to a secure channel. Here’s my passport, proving I’m z/VM. And I can use any of these five secure channels—is that cool?”



How is TLS? (3 of 5)

- Bob says, “I confirm you’re z/VM, and I can use Channel 05. Let’s do this!”

- or

Bob says, “You’re z/VM, but I can’t use any of those channels. I’m out.”

- or

Bob says, “You say you’re z/VM, but... your passport’s expired? Eww. I’m out.”

- or sometimes

Bob says, “Hi z/VM! I’m Bob! –and here’s my passport to prove it!”

...at which point, z/VM checks its guest list, or calls its party planner, to see if Bob was invited.

- End result: two devices either agree to trust one another and move to a secure channel... or nope out.

How is TLS? (4 of 5)

Now let's break that down a bit further.

- Bob reaches out to z/VM and says, “Hi, I'd love to chat. We cool?”

Your laptop starts a Secure TN3270 connection, using TLS, over to montana.pok.ibm.com ... because you've configured Pcomm or Host on Demand to use TLS.)

- z/VM says, “Sure, but we're moving to a secure channel. Here's my passport, proving I'm z/VM. And I can use any of these five secure channels—is that cool?”

The connection reaches the z/VM TCP/IP stack. z/VM realizes: “I want encryption—and I think Bob wants encryption.”

- z/VM sends its server certificate, along with a list of cipher suites, back to your laptop
- This indicates that (a) z/VM is ready for a secure connection, (b) is willing to play ball, (c) has a server certificate, and (d) can support certain kinds of encryption. This is a long string of codes representing a bunch of different flavors.

How is TLS? (5 of 5)

- Bob says, “I confirm you’re z/VM, and I can use Channel 05. Let’s do this!”
Your laptop can support z/VM’s encryption, and a secure connection is established.

- or

Bob says, “You’re z/VM, but I can only use these other channels. Sorry!”
Your laptop can’t handle z/VM’s awesome encryption; request denied.

- or

Bob says, “You say you’re z/VM, but... this passport’s expired. I’m out.”
z/VM’s **server certificate** has a problem. Your laptop rejects the connection.

- or sometimes

Bob says, “Hi z/VM! I’m Bob! –and here’s my passport to prove it! That cool?”
Your laptop sends back a **client certificate**, identifying itself to z/VM.
z/VM can then evaluate that certificate locally, or use an **OCSP Responder or CDP** to evaluate it centrally

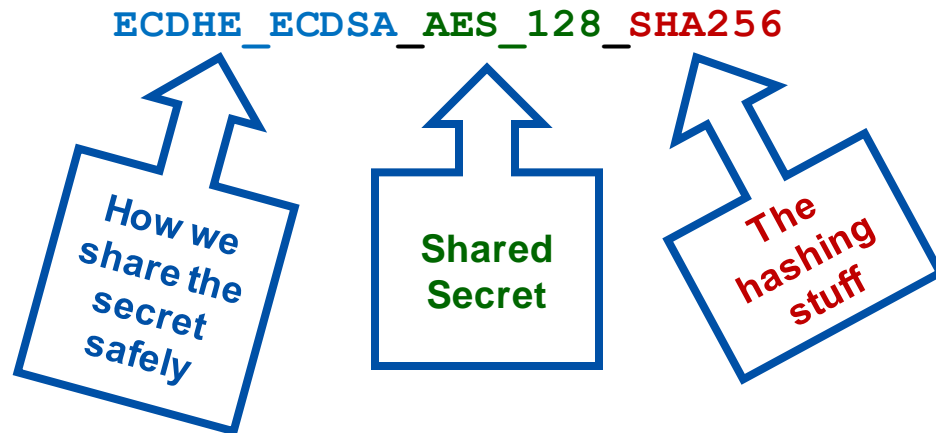
Agenda

Encryption and Certificates – Huh?

So how does that tie back to encryption?

- The story of Bob and MONTANA didn't seem that complicated, but a lot of things happened there.
 - We picked a version of TLS to use
 - We agreed on how to exchange a common secret
 - We agreed on how big that secret is
 - We agreed on how to share that common secret safely
 - We even did some hashing on it to make sure nothing broke in transit

- When we say "Channel 05," we really mean something like this. It's not scary if you know what the bits are:



Selecting a cipher suite

- If you're programming the server side, [your security people](#) are going to have rules about these
 - E.g., “Do use AES-256 and SHA-256. Please don't use DES and MD5.”
 - These may change over time. Don't panic.
- In z/VM, some cipher suites will be enabled by default. Some will not.
 - You'll have **EXEMPT** and **ENABLE** keywords in your DTCPARMS to toggle these settings
 - There are also **HIGH MEDIUM and LOW** options, which auto-configure based on [IBM's recommendations](#)
 - This may not be the same as what makes your security people happy!
- [Pick a list of cipher suites which will make your security people happy](#)
 - Server policy is always the starting point for a TLS handshake (*we offered a list of Channels, starting with 05*)
 - Your client (Pcomm, Filezilla, another z/VM system, etc) will have a list of ciphers it can use
 - If your client and server aren't in agreement on **anything**, no connections happen
 - **Your security people probably won't say “Disallow channel 0009,” they'll just say “No DES.”**
 - z/VM has a list of cipher suites, with codes, in the **Planning & Customization manual** for cases like that
- The cipher used is dependent upon your **digital certificates**



Brian Hugenbruch

Root certificate authority

Expired: Sunday, April 14, 2019 at 12:05:18 PM Eastern Daylight Time

ⓘ This certificate is marked as trusted for this account

> Trust

∨ Details

Subject Name

Country or Region US
State/Province NY
Locality Endicott
Organization IBM
Organizational Unit zVM Security
Common Name Brian Hugenbruch
Email Address bwhugen@us.ibm.com

Issuer Name

Country or Region US
State/Province NY
Locality Endicott
Organization IBM
Organizational Unit zVM Security
Common Name Brian Hugenbruch
Email Address bwhugen@us.ibm.com

Serial Number 00 A2 AA 0C 80 C5 B5 E2 58

Version 3

Signature Algorithm SHA-1 with RSA Encryption (1.2.840.113549.1.1.5)

Parameters None

Not Valid Before Sunday, April 13, 2014 at 12:05:18 PM Eastern Daylight Time

Not Valid After Sunday, April 14, 2019 at 12:05:18 PM Eastern Daylight Time

Public Key Info

Algorithm RSA Encryption (1.2.840.113549.1.1.1)

Parameters None

Public Key 128 bytes : CD 16 EA 87 01 68 9F 24 ...

Exponent 65537

Key Size 1,024 bits

Certificates are passports for the internet

- Issued by a “Certificate Authority” which serves as a root of trust
- Contains identifying information – name (or server domain name); country; a digital fingerprint and a hash; extra stuff if you need to be fancy
- You can use them for different reasons! Such as:
 - **Server certificate** – certificates for servers.
 - “Hi, I’m montana.pok.ibm.com and here’s my IP address, too.”
 - **Client certificates** – these are for humans.
 - “Hi, I’m Bill.”
 - **Code signing certificates** – for validating download integrity and source.
 - “This code was produced and signed by IBM. We’re here to help!”

#zVM #IBMz

Where do I get them?

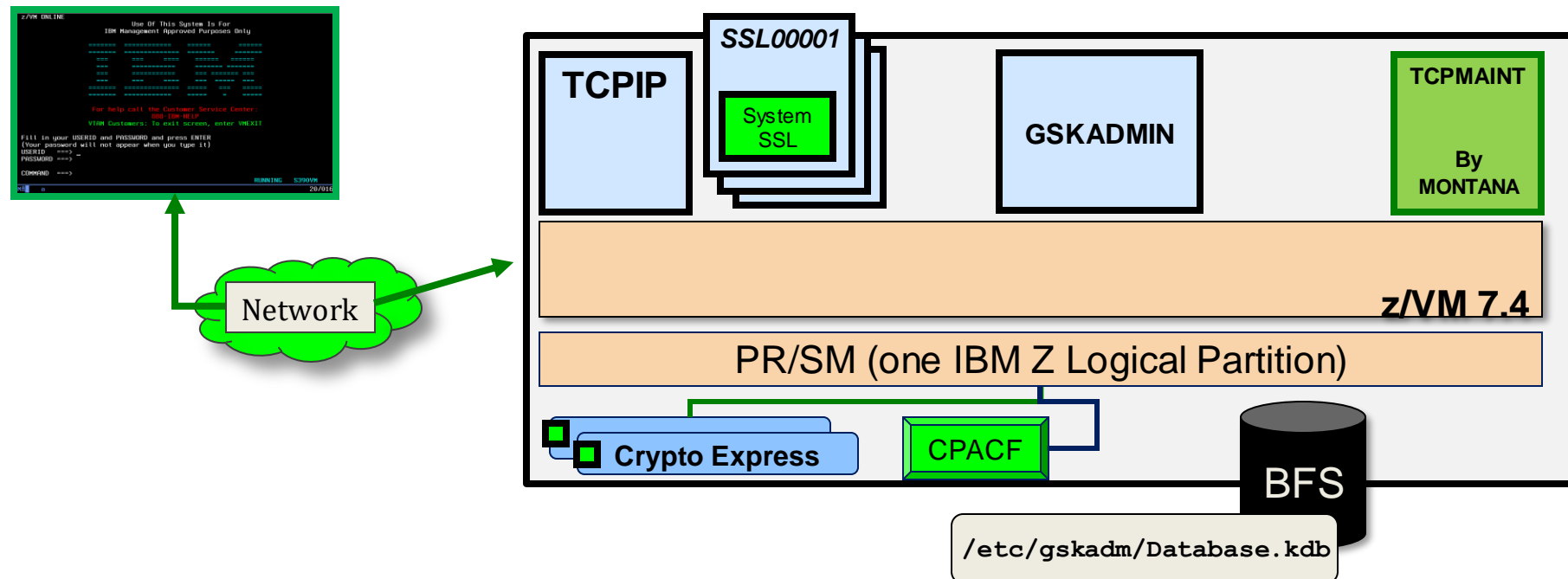
- Either **you make your own** or **you talk to your security people** (who'll **make or buy them** for you)

- **Option 1: “Make your own”**
 - Great for test environments or low-impact prod
 - You act as your own “Certificate Authority” (fancy term for “people who issue certificates”)
 - Most certificates aren't “root” certificates—they're chained together and refer back to a common point of trust
 - Saves money, but not necessarily time

- **Option 2: “Talk to your security people”**
 - Your environment may have rules and regulations about what type of certs you use
 - Either your security team is already acting as a CA, or they purchase certificates for your company
 - They'll send them to you, based on input
 - Again, you'll get a series of certificates—the root, your server certificate, and anything in between them
 - Saves time, but not necessarily money

Now that I have them, where do I put them?

- On z/VM, we have a **separate userid called GSKADMIN** that handles certificates
- They're stored in the Byte File System – your TCP/IP stack and SSL worker machines look there for them
- A program called **gskkyman** (same as the one on z/OS) handles the creation of databases and working with certs



Agenda

Using TLS on z/VM

So how do you use TLS on z/VM? (1 of 3)

▪ Planning

- You need to know **which services** offered by the z/VM TCP/IP stack you want to protect.
- Do they need **separate certificates**? Is one okay for the domain name / IP address?
- **How many TCP/IP stacks** are on your z/VM partition? Does that make a difference?
- Do you need to share a database amongst multiple members of a **Single System Image**?

Planning for TLS

- **At a minimum, you'll want a server certificate for your z/VM TCP/IP stack**
 - You may elect to have **different certificates for different applications** (Telnet, FTPS, RSCS...)
 - If you have more than one TCP/IP stack (and thus different domain names), you may elect to have different certificates for those as well
 - You'll have **different TCP/IP stacks on different members of your SSI**
 - ...**but you can share a single certificate database** amongst different TCP/IP stacks

- You may need client certificates for humans and for certain automated applications

- **Chat with your security people about TLS versions and cipher suites**
 - TLS 1.3 has been widely adopted since introduction, but TLS 1.2 is still common
 - The z/VM TLS defaults are pretty aggressive, but we can't guarantee your security people aren't more paranoid than we are*

**which is saying something!*

So how do you use TLS on z/VM? (2 of 3)

Certificate Management

- You need to acquire or make some certificates
 - Certificates last *nnn* days through *nnn* years, so you'll want to plan to manage these over time
 - If you're using client certificates, make sure you can trust those, too
 - That just means “add more stuff to the database”
-
1. **LOGON** (or LOGON..BY) to **GSKADMIN**
 - GSKADMIN is auto-configured to mount all the right filepools for BFS access
 2. Use **OPENVM PUTBFS** to store certificates into the Byte-File System
 3. Use **gskkyman** program (panel-driven) to store certificates and keys in your database
 4. Issue **SSLADMIN REFRESH** if updating a certificate database in use by an active TLS Server

GSKADMIN > Using gskkyman

Key Management Menu

Database: /etc/gskadm/ForThisPresentation.kdb
Expiration: 2015/12/15 15:49:12

- 1 - Manage keys and certificates
- 2 - Manage certificates
- 3 - Manage certificate requests
- 4 - Create new certificate request
- 5 - Receive requested certificate or a renewal certificate
- 6 - Create a self-signed certificate
- 7 - Import a certificate
- 8 - Import a certificate and a private key
- 9 - Show the default key
- 10 - Store database password
- 11 - Show database record length

- 0 - Exit program

Enter option number (press ENTER to return to previous menu):

So how do you use TLS on z/VM? (3 of 3)

Configuration

- Set your security policy in the DTCPARMS— **LOGON** (or LOGON..BY) to **TCPMAINT**
- **For each service, adjust your security settings**
 - [SSL administrator userids go into the :Admin_ID_List](#). In DTCPARMS
 - Telnet, FTPS, SMTP, and RSCS for “dynamic” support
 - Anything on the PORT statement (e.g. SMAPI) has “static” support
 - Either is viable, but don’t use both for the same service!
- **Use ALLOWED when testing, so you don’t lock yourself out**
 - When you’re ready, switch to REQUIRED
- **Try to note who’s using what services when:** **NETSTAT IDENTIFY SSL**
 - Identifies current active secure connections
 - Important for tracking users of older TLS versions
 - Those people will complain if you change TLS settings and things don’t work
 - (Alternately: those automated services might break)

z/VM TLS/SSL Server >> DTCPARMS Options

:Admin_ID_list.	Userids authorized to execute privileged commands – e.g., SSLADMIN commands
:Mixedcaseparms.	Parameters are supported in mixed case
:Mount.	Certificate database location. Default is /etc/gskadm/
:Parms.	See next slide
:Stack.	Associated TCPIP virtual machine One pool per stack. One stack per pool. Non-negotiable. <i>This tag is required; otherwise, the SSL server/ pool cannot be identified during stack initialization!</i>
:Timestamp.	On/Off for timestamps on terminal messages and cmd responses
:Timezone.	Set timezone of server
:Vmlink.	Sets a Pool member's SFS space

TLS Server :Parms. for security policy

- Persists for the run-time for a server or server pool. Must be consistent for all members of a server pool
- Options:
 - **KEYFILE** - BFS location of the certificate database
 - **CACHELIFE** - for secure connections, in hours, minutes, seconds
 - **CACHECLEANUP** - processed every n connections
 - **MODE** - sets a cryptographic compliance mode
 - **MODE FIPS-140-2**
 - **MODE NIST-800-131A**
 - **PROTOCOL** - enable or disable SSL/TLS levels.
 - **TLS 1.2 only enabled by default** << As of z/VM 7.3
 - **Available protocols change based on MODE**
 - **EXEMPT|ENABLE** - disable or enable particular cipher suites
 - **GSKTRACE** - enable System SSL tracing (service only)
 - **TRACE/NOTRACE** - enable SSL Server tracing
 - Can be dynamically manipulated via authorized SSLADMIN commands

SSLADMIN command

- **CLEAR** remove userid(s) set by SET
- **CLOSECON / LOG** retrieves console log
- **HELP** displays help information
- **QUERY**
 - Status Summary returns general server data
 - Status Details returns specific server data
 - Settings returns current command defaults
 - Cache returns cache data
 - Sessions returns data on active secure sessions
 - Trace returns trace settings
- **RESTART** quiesces and re-IPL's SSL server
- **REFRESH** reaccess certificate database
- **SET** sets default targets for SSLADMIN commands
- **START / STOP** starts / stops an SSL server
- **SYSTEM** used to issue CP or CMS commands
- **TRACE / NOTRACE** enables / disables tracing

Agenda

Use-Cases for Enabling TLS on z/VM

1. “I want to connect securely via TN3270 emulator.”

1. **First, get a server certificate** << let's say your security people have these for you

2. **Install them into your database** << by default, this is `/etc/gskadm/Database.kdb`
 1. Send to z/VM
 2. Transfer from minidisk to BFS using **OPENVM PUTBFS**
 3. Use gskkyman to open a database, then “Import Certificates With Keys”
 4. Make sure your server certificate has a label that's 8 characters long and uppercase, e.g. BWHSERVE

3. **Configure your TCP/IP environment for TLS**
 1. In PROFILE TCPIP, make sure SSLSERVERID points to the same pool prefix you have in DTCPARMS (Default: **SSL**)
 2. In PROFILE TCPIP, update your **INTERNALCLIENTPARMS** settings
 - **SECURECONNECTION** should be **ALLOWED** to start, then **REQUIRED** after you test it
 - **TLSLABEL** should be BWHSERVE
 3. In DTCPARMS, configure the :Parms. Parameter of your TLS server with your Mode, Protocol, and cipher suites

4. **IPL your TCP/IP stack with the TLS server.**
(Telnet is built into the TCPIP virtual machine; you don't need to do anything else)

2. “I want to connect securely via FTPS.”

1. **First, get a server certificate for FTP** << let’s say your security people have these for you

2. **Install them into your database** << by default, this is `/etc/gskadm/Database.kdb`
 1. Send to z/VM
 2. Transfer from minidisk to BFS using OPENVM PUTBFS
 3. Use gskkyman to open a database, then “Import Certificates With Keys”
 4. Make sure your server certificate has a label that’s 8 characters long and uppercase, e.g. BWHFTPS1

3. **Configure your TCP/IP environment for TLS**
 1. In PROFILE TCPIP, make sure SSLSERVERID points to the same pool prefix you have in DTCPARMS (Default: **SSL**)
 2. In **SRVRFTP CONFIG**, update your TLS/SSL settings
 - **SECURECONTROL** should be ALLOWED to start, then REQUIRED after you test it
 - **SECUREDATA** should be ALLOWED to start, then REQUIRED after you test it
 - **TLSLABEL** should be BWHFTPS1
 3. In DTCPARMS, configure the :Parms. Parameter of your TLS server with your Mode, Protocol, and cipher suites

4. **IPL your TCP/IP stack with the TLS server and your FTP machines**

3. “I want to add client certificates.”

1. **First, get client certificates for whoever is connecting to this IP address**
2. **Install them into your database** << by default, this is `/etc/gskadm/Database.kdb`
 1. Send to z/VM
 2. Transfer from minidisk to BFS using **OPENVM PUTBFS**
 3. Use ***gskkyman*** to open a database, then “Import Certificates With Keys”
 4. For client certificates, these can be from the same chain as your server certificates. This isn’t required. **HOWEVER.** The full list of certificates for your client certs do need to be installed to enable trust!
3. **Configure your TCP/IP environment so your servers will ask for (and process) client certificates**
 - For Telnet and FTPS, this is CLIENTCERTCHECK (REQUIRED / ALLOWED)
 - There is a PORT statement option for this, for other services
4. **Did you (or your security people) need Online Certificate Status Protocol (OCSP) or Certificate Revocation List (CRL) Distribution Points (CDP) instead of a local database?**
 - If so: this is now available for z/VM V7.2!
 - **Insert the OCSPEnable. Tag into your DTCPARMS along with the address of that other database.**

4. “I want to expand or contract my SSL pool size”

- z/VM kicks off five (5) TLS servers by default, which are set to handle up to 600 secure connections each
 - Maybe you don’t need that many. Or maybe you need more!
1. Make sure the SSL **POOL statement** in the z/VM user directory has the right number of machines
 2. In PROFILE TCPIP, update your **SLLIMITS** option to reflect how many connections per server you wish
 3. Re-IPL your TCP/IP environment

5. “I need to check my configuration. How do I...”

“Turn off algorithm [x]? My security scan is complaining!”

1. **EXEMPT** the cipher suite by name.
2. **STOP/START** all your TLS server machines to pick up changes, then confirm output of **SSLADMIN QUERY STATUS DETAILS >>>**

“Meet my company’s policies on key sizes?”

1. Generate an RSA or DSA certificate of appropriate key length (usually 2048 or greater)
2. **Use MODE NIST-800-131A**
(minimum transport key size of 2048)
3. **STOP/START** all your TLS server machines, then confirm output of **SSLADMIN QUERY STATUS DETAILS >>>**

#zVM #IBMz

DTCSSL2430I Cryptographic Mode details:

Server Status Modes

```
-----  
S1000001 Enabled <*None*>  
S1000001 Disabled FIPS-140-2 NIST-800-131A
```

DTCSSL2430I Protocol details:

Server Status Protocols

```
-----  
S1000001 Enabled TLSV1_2 TLSV1_1  
S1000001 Disabled TLSV1_0 SSLV3 SSLV2
```

DTCSSL2430I Cipher details:

Server State Ciphers

```
-----  
S1000001 Included RSA_AES_128_GCM_SHA256 RSA_AES_256_GCM_SHA384  
S1000001 Included DHE_RSA_AES_128_GCM_SHA256  
S1000001 Included DHE_RSA_AES_256_GCM_SHA384  
S1000001 Included DHE_DSS_AES_128_GCM_SHA256  
S1000001 Included DHE_DSS_AES_256_GCM_SHA384 RSA_AES_128_SHA256  
S1000001 Included RSA_AES_256_SHA256 DHE_DSS_AES_128_SHA256  
S1000001 Included DHE_RSA_AES_128_SHA256 DHE_DSS_AES_256_SHA256  
S1000001 Included DHE_RSA_AES_256_SHA256 RSA_AES_256  
DHE_DSS_AES_256  
S1000001 Included DHE_RSA_AES_256 RSA_AES_128 DHE_DSS_AES_128  
S1000001 Included DHE_RSA_AES_128 3DES_168_SHA DHE_RSA_3DES  
S1000001 Included DHE_DSS_3DES DES_56_SHA DHE_RSA_DES  
DHE_DSS_DES  
S1000001 Exempt DH_RSA_AES_128_GCM_SHA256  
S1000001 Exempt DH_RSA_AES_256_GCM_SHA384  
S1000001 Exempt DH_DSS_AES_128_GCM_SHA256  
S1000001 Exempt DH_DSS_AES_256_GCM_SHA384  
DH_DSS_AES_128_SHA256  
S1000001 Exempt DH_RSA_AES_128_SHA256 DH_DSS_AES_256_SHA256  
S1000001 Exempt DH_RSA_AES_256_SHA256 RC4_128_SHA RC4_128_MD5  
S1000001 Exempt DH_DSS_AES_256 DH_RSA_AES_256 DH_DSS_AES_128  
S1000001 Exempt DH_RSA_AES_128 RC2_128_MD5 DH_RSA_3DES DH_DSS_3DES  
S1000001 Exempt DH_RSA_DES DH_DSS_DES RC4_40_MD5 RC2_40_MD5  
S1000001 Exempt NULL_SHA256 NULL_SHA NULL_MD5 NULL
```

6. “My security people want this FIPS mode thing?”

- US Federal Information Processing Standard (FIPS) 140-2 is a set of guidelines related to cryptographic use
- Or, more succinctly: **“FIPS mode == more rules.”**

- 1. Make sure you have a FIPS-mode database** << this may mean transferring certificates. Plan ahead!
 - This may also mean changing your DTCPARMS to point to a new database
- 2. Make sure your certificates are of an appropriate size** << 2048 is preferred nowadays, unless using Elliptic Curve
- 3. Update your DTCPARMS** << **MODE FIPS-140-2 on that :Parm tag!**
 - Auto-configures some settings in your TLS server
 - You may still need to exempt some ciphers
- 4. Restart your TLS Server to use the new database**

7. “My SSL server just shuts right back down. Help!”

- **This is most commonly a database configuration error**
 - Pointing to the wrong database
 - SSLPOOL set-up scripts not run
 - Openvm permit not run against the database and stash file (to grant the TLS server permission to read it)
- **Check your SSL Server console for error messages**
- If necessary, add TRACE FLOW or TRACE DEBUG to your DTCPARMS to trace start-up
 - This will help IBM Service figure out what’s going on

Common data you may need to debug SSL server problems:

- TCPIP DATA (connection to the TCP/IP stack)
- DTCPARMS (server configuration, SSLDCSS configuration)
 - *Most common problems tend to be either a misconfiguration of DTCPARMS or a DTCPARMS / TCPIP mismatch*
- PROFILE TCPIP (stack configuration)
- SSL, TCP/IP and SSL DCSS Management Agent server console messages
- SSLADMIN or NETSTAT command responses
- GSKADMIN console information
- Trace output from SSL or TCP/IP

8. “My security people said go ahead and make your own certs.”

- You could use Linux tooling (openssl) to create certificates, if that’s easy for you
- You can also use gskkyman itself to act as your own Certificate Authority

You’ll need: two databases in your BFS >> one to hold your TLS certificates, and one to do your certificate signing

1. Create a **root key** in your signer database
2. Create **1-n certificate requests** in your TLS database. Export the requests to your BFS.
 - Make sure your requests have the right kinds of encryption specified
 - If you need OCSP, make sure you add the domain name and IP address into appropriate extensions

3. **Sign the certificates.** This is done from z/VM command line:

```
GSKADMIN: `gskkyman -g -x 5000 -cr ZVMPCSRV.req -ct ZVMPCSRV.cert -k  
Certificate_Signer_Db_Fips.kdb -l ZVMROOT  
Enter database password (press ENTER to cancel):
```

```
Certificate created.`
```

4. **Import the signed files from your BFS into your TLS database**
5. **Proceed with all other use-cases** as though you’d gotten the certificates from your security nerds

9. “I need to do certificate cleanup. Help!” (1/2)

- Certificates are complicated, and gskkyman (as a menu-driven thing) isn’t great for giving you everything you need
- The CERTMGR command, introduced in APAR PH44044 for z/VM V7.2, makes this a little easier:

```
CERTMGR QUERY
```

```
Database /etc/gskadm/certmgr2.kdb
```

```
Enter database password (press ENTER to cancel):
```

```
<----- Certificate -----> <- Key -> <-Signature-> Self
  Type      Expires      Trust Type Size Type      Hash      Sign      Label
-----
Entity 19 Aug 2022   Yes  ECC   320 RSA   SHA-224 No   Entity_1
Inter  27 Jun 2024   Yes  RSA  2048 RSA   SHA-224 No   Inter_1
Root   04 May 2025   Yes  RSA  2048 RSA   SHA-224 Yes  Root_1
Entity 19 Aug 2027   Yes  ECC   320 RSA   SHA-224 No   Entity_2
Root   10 Jan 2028   Yes  RSA  2048 RSA   SHA-224 Yes  Root_2
```

9. “I need to do certificate cleanup. Help!” (2/2)

- Certificates are complicated, and gskkyman (as a menu-driven thing) isn't great for giving you everything you need
- The CERTMGR command, introduced in APAR PH44044 for z/VM V7.2, makes this a little easier:

```
CERTMGR QUERY ( CHAIN
```

```
Enter database password (press ENTER to cancel):
```

Expires	Label
04 May 2025	Root_1
27 Jun 2024	Inter_1
19 Aug 2022	Entity_1
10 Jan 2028	Root_2
19 Aug 2027	Entity_2

Agenda

Time for questions (or a nap)

Summary

- **TLS is important for protecting your data – yes, even for z/VM!**
 - It builds trust between server and client
 - It protects data in flight
 - It's vital for participating in the modern world
 - Indeed, it's an industry standard

- **There may come a time when you need to configure TLS**
 - Just because no one's said anything isn't a guarantee you may not need it
 - No one writes security policies that say, “—except for the z/VM folks, we know they're cool.”
 - Yes, that may mean test systems, too

- **It's not as scary as it sounds**
 - There are presentations, manuals, and cookbooks available to help
 - [We're working on more detailed “use-case” documents](#)
 - (We'll post these either to the z/VM webpage and/or Github—if the latter, post comments or contribute!)
 - “Streamlined SSL” item in progress on the z/VM New Function Page
 - *Query GSKKMAN Certificates* was the first step

For More Information ...

- <https://www.vm.ibm.com/related/tcpip/vmsslinf.html>
VM Webpage: TLS Configuration Information (has some use-case and cookbook information already)
- <https://www.vm.ibm.com/newfunction/#qgskkyman>
CERTMGR / Query GSKKYMAN on the New Function webpage (help us make this easier!)
- <https://www.redbooks.ibm.com/redbooks/pdfs/sg248447.pdf>
Securing Your Cloud with z/VM Redbook
- <https://www.redbooks.ibm.com/redpieces/pdfs/sg248147.pdf>
Virtualization Cookbook for z/VM V7.2 (see Ch. 3.5)
- <https://www.vm.ibm.com/devpages/hugenbru/TLSVM710.PDF>
“Security z/VM Connectivity” presentation (for the version where Brian uses *all* the acronyms)

Presenter Information:

Montana Lee
z/VM Networking Developer – VSWITCH and EQDIO
montanalee@ibm.com

Appendix: Debugging the TLS Server on z/VM

TLS/SSL Server: Debugging

Common data you may need to debug SSL server problems:

- TCPIP DATA (connection to the TCP/IP stack)
- DTCPARMS (server configuration, SSLDCSS configuration)
 - *Most common problems tend to be either a misconfiguration of DTCPARMS or a DTCPARMS / TCPIP mismatch*
- PROFILE TCPIP (stack configuration)
- SSL, TCP/IP and SSL DCSS Management Agent server console messages
- SSLADMIN or NETSTAT command responses
- GSKADMIN console information
- Trace output from SSL or TCP/IP

TLS/SSL Server: Debugging

Problem: The SSL server does not initialize and run SSLSERV MODULE

Symptoms:

- TCPIP starts, but SSL server and protected services do not
- Console messages for the SSL server which resemble:

```
DTCRUN1028E :Stack.TCPIP11 specified in GDLRCT2 DTCPARMS D1 does not match  
           "TcpipUserid TCPIP10" in the TCPIP DATA file  
DTCRUN1099E Server not started - correct problem and retry
```

TLS/SSL Server: Debugging

Problem: The SSL server does not initialize and run SSLSERV MODULE

Analysis:

- Check the SSL server console for messages
- Verify that the TCPIPUSERID statement in TCPIP DATA lists the correct TCPIP virtual machine for your SSL server
- Confirm DTCPARMS settings for **:stack.** tags and **:vmlink.** tag
- For an SSL pool server, confirm that the server has been enrolled in the appropriate SFS file pool, and that an alias to the (common-use) PROFILE EXEC is in place
- For an SSL pool server (and, the case of having attempted a restart of the subject server) confirm that DTCPARMS configuration has not been changed, while one or more other pool servers remain in operation

TLS/SSL Server: Debugging

Problem: the server cannot use the key database

Symptoms:

- SSL server does not start
- Console messages for the SSL server which resemble:

```
DMSOVZ2113E Object does not exist: '/../VMSYSU:GSKADMIN/etc/gskadl'  
DTCRUN1001E "OPENVM MOUNT '/../VMSYSU:GSKADMIN/etc/gskadl /" failed with  
           return code 28  
DTCRUN1099E Server not started - correct problem and retry
```

TLS/SSL Server: Debugging

Problem: the server cannot use the key database

Analysis:

- Verify that the Byte File System (BFS) parameters for the DTCPARMS :**mount**.tag
- Confirm that the necessary file permissions have been established
 - Database.kdb, Database.rdb, Database.sth
- Confirm that the file pool server for the BFS user space (**VMSYSU**, by default) is operational
- Use the GSKKMAN utility to confirm that the key database has been properly created, and that the correct database has been identified via the VMSSL command KEYFILE operand

TLS/SSL Server: Debugging

Problem: a server cannot use the session cache

Symptoms:

- TCPIP and SSL pool initialize properly
- Connections suddenly cannot be **re**-established
- SSLADMIN messages which resemble the following:

```
DTCSSL2421E SSL00001: Communication error: Connection timed out
```

TLS/SSL Server: Debugging

Problem: a server cannot use the session cache

Analysis:

- Verify that the SSL DCSS Management Agent is operational
 - QUERY <userid> should indicate that the machine is running disconnected:

```
query ssldcssm
SSLDCSSM - DSC
Ready;
```
- Verify that SSLDCSSM has been configured properly
 - Check DTCPPARMS and configuration files
 - Issue CP QUERY NSS commands
 - **Class E privilege** required for the issuing userid
 - User count should match pool size plus one (SSL* and DCSSM) if servers are running

Output should look similar to the following:

```
--> CP QUERY NSS NAME TCPIP MAP
FILE FILENAME FILETYPE MINSIZE BEGPAG ENDPAG TYPE CL #USERS PARMREGS VMGROUP
9539 TCPIP DCSS N/A 10000 100FF SN R 00006 N/A N/A

--> CP QUERY NSS USERS TCPIP
FILE FILENAME FILETYPE CLASS
9539 TCPIP DCSS R

SSL00005 SSL00004 SSL00002 SSL00003 SSL00001 SSLDCSSM
```

(continued ...)

TLS/SSL Server: Debugging

Problem: a server cannot use the session cache

Analysis:

- Verify that SSLDCSSM has been initialized prior to the SSL server
 - `DTCRUN1043I Initiating XAUTOLOG of server SSLDCSSM`
This message should appear in the TCPIP stack's console log prior to any SSL configuration / initialization messages
- Confirm that the necessary NAMESAVE statements are present in the CP directories for the SSL server and its DCSS Management Agent

TLS/SSL Server: Debugging

Problem: The server cannot connect to the TCP/IP virtual machine

Analysis:

- Verify the TCPIPUSERID statement in TCPIP DATA file
 - should cite the correct TCP/IP server virtual machine
- Confirm that the correct TCP/IP server is identified by a DTCPARMS **:stack.** tag defined for the subject SSL server
- Verify that the TCP/IP server is started
- Check the TCP/IP server console for messages that indicate a problem. (*z/VM: TCP/IP Messages and Codes*)
- Use the FLOW or DEBUG traces to gather additional information. Update the DTCPARMS **:parms.** tag for the SSL server to include the TRACE FLOW or TRACE DEBUG operand, then start the server. This will provide debug information during the server start up.

TLS/SSL Server: Debugging

Problem: Incorrect parameters are passed to SSL server

Symptom: SSL server is running, but not behaving as expected

Analysis:

- Use SSLADMIN QUERY STATUS to determine which options are in effect
- Check that all parameters are correctly specified in the DTCPARMS :parms. Tag
- Compare parameters against message DTCRUN1011I in the server console

TLS/SSL Server: Debugging

Problem: Protected application server (e.g., FTP) shuts down at start-up

Symptoms:

- Console files received from application userids on autolog of TCPIP virtual machine
- Application server cannot be autologged, will not respond to commands

TLS/SSL Server: Debugging

Problem: Protected application server (e.g., FTP) shuts down at start-up

Analysis:

- Confirm SSL server is running (NETSTAT CONFIG SSL)
- Confirm SSL server is listening (NETSTAT CONN or NETSTAT ALLCONN)
- Verify the SSLSERVERID statement in PROFILE TCPIP reflects the correct SSL server configuration
- Check the application server console for indications of problems. (*z/VM: TCP/IP Messages and Codes*) For example:

```
12:30:46 DTCFTS8467E Error verifying TLS label NOTTHERE: Label is not recognized
```

(continued ...)

TLS/SSL Server: Debugging

Problem: Protected application server (e.g., FTP) shuts down at start-up

Analysis:

- Using the GSKKYMAN utility, verify that the TLSLABEL specified is present in the certificate database, and conforms to naming requirements
 - On GSKADMIN or other authorized userid, invoke *gskkyman*
 - Open the appropriate certificate database `<filename>.kdb`
 - Choose option 1, “Manage keys and certificates”
 - The certificate with key with the correct TLSLABEL should appear on this list

- Verify the TLSLABEL statement and the correct value have been specified in the application server configuration file:
 - PROFILE TCPIP (or its equivalent) for TELNET
 - SMTP CONFIG (or its equivalent) for SMTP
 - SRVRFTP CONFIG (or its equivalent) for FTP

- An incorrect or misspelled TLSLABEL value in an application server configuration file can prevent such a server from initializing

TLS/SSL Server: Debugging

Problem: Connection to protected application cannot be established

Symptom, z/VM FTP:

```
220 Connection will close if idle for more than 5 minutes.
```

```
>>>AUTH TLS
```

```
421 Temporarily unable to process security
```

```
Command:
```

Symptom, z/VM Telnet:

```
VM TCP/IP Telnet Level 610
```

```
SSL Server is not available on local system.
```

```
Quitting...bye
```

TLS/SSL Server: Debugging

Problem: Connection to protected application cannot be established

Analysis:

- Confirm SSL server is running (NETSTAT CONFIG SSL)
- Confirm SSL server is listening (NETSTAT CONN or NETSTAT ALLCONN)
- Use SSLADMIN QUERY STATUS or NETSTAT CONFIG SSL to determine the current number and maximum number of active sessions
- Check SSL server console log for messages
- Issue SSLADMIN TRACE CONN
- Activate TCPIP tracing (SSL, TCPUP, TCPDOWN) to gather more data

TLS/SSL Server: Debugging

Problem: Connection closes due to errors

Analysis:

- Verify the certificate label is correct:
 - *gskkyman* certificate label, in the appropriate database
 - TLSLABEL on PORT statement or in application server configuration
- Verify that the certificate has not expired
 - View certificate information in *gskkyman*
- Verify that the SSL server is accessing the most recent certificate updates (SSLADMIN REFRESH)
- Check SSL Server console for messages
- Issue SSLADMIN TRACE CONNECTIONS to gather more data

TLS/SSL Server: Debugging

Problem: Incorrect input or output inside a secure connection

Analysis:

- Verify that the subject connection has been established
 - SSLADMIN QUERY SESSIONS
- Check messages from the SSL server for any problems
 - SSLADMIN CLOSECON
- Verify that data is flowing correctly through the server
 - SSLADMIN TRACE CONNECTIONS DATA
 - Try connection again after Trace has been configured
 - Consider limiting the trace to a specific IP address / port