



VFILE from VSSI

File Transfer: Shaken, not Stirred

Rick Troth, VSSI

<richard.troth@vsoftsys.com>



Disclaimer

This presentation discusses intended directions and developments. It should not be taken as guaranteed delivery or features. The audience must not expect any eventual result to match what is presented here.



about:Rick



- VM (SP, et al) since 1982, also VMware, Xen, KVM
- MVS (batch) 1981, WYLBUR same time, TSO later (sans ISPF)
- Unix (AIX, UTS, SunOS) 1987
- Linux since 1993
- “Open Edition” 1995

Unix → Linux

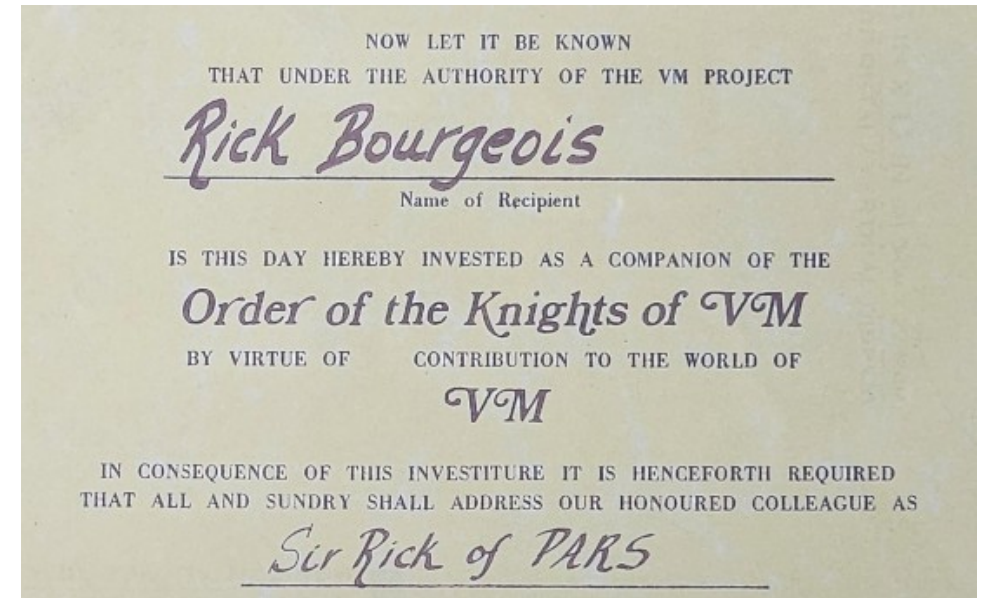
the Open Source thing





about:VSSI

- The VPARS and VTAPE people
- Established in 1982
- Founder is a Knight of VM
- Customer base is typically Fortune 100
- Established solid reputation for product support
- Support cost included in yearly license cost



Virtual Software Systems, Inc.
7715 Browns Bridge Road
Gainesville, Georgia 30506
(770) 781-3200 Fax (770) 781-3210



about:VFILE

- Operational equivalent to RSCS, but ...
- Uses UFT protocol rather than NJE
- Supporting z/VM, Linux, z/Linux, z/OS, Windows, TPF
- Available 1Q26

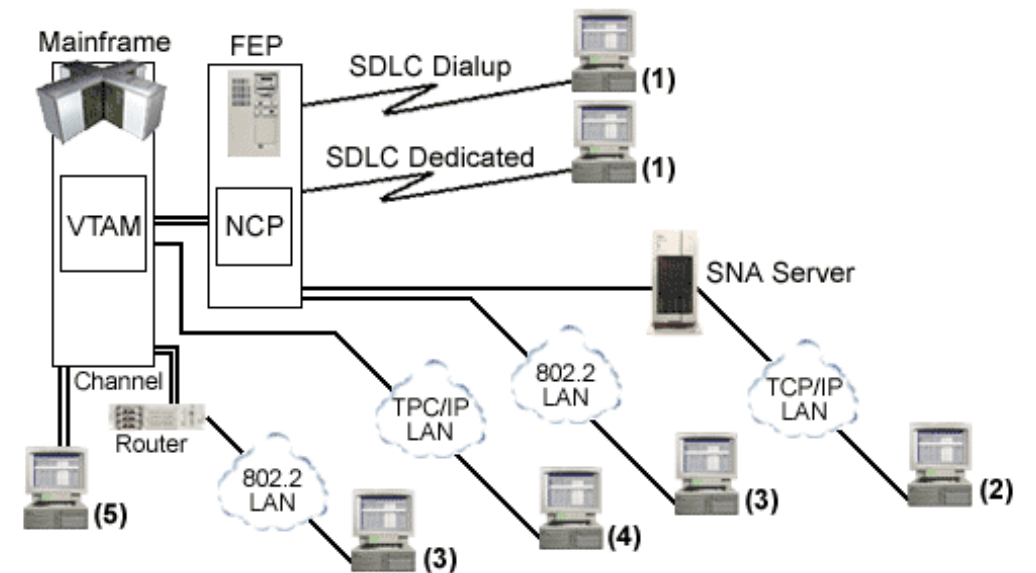
VFILE makes sender-initiated file transfer *easier* to manage and more *broadly* available compared to NJE or other methods.



History: RSCS, NJE, HASP, oh my!

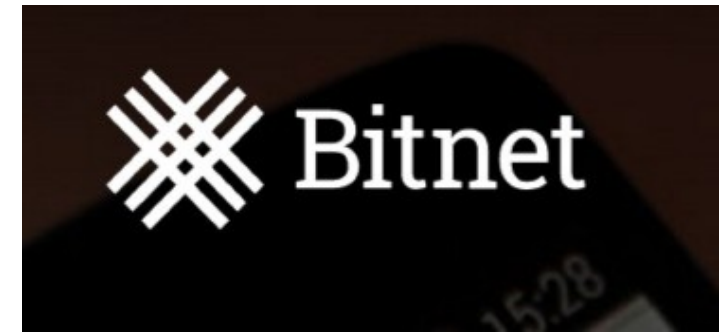
- Houston Automatic Spooling Priority
- HASP became JES2
- ASP became JES3
- RSCS v1

The world was point-to-point
NJE today remains point-to-point





Whither BITNET



- BITNET (including EARN, NETNORTH, Asia)
- Largely Academic and Research (IBM had VNET)
- Multi-Platform (VM, MVS, DEC VMS, Unix, others)

Significant communities ... but ...

With the commercialization of TCP/IP Internet in the early 1990s, BITNET fell from popularity. One could also no longer “just send a file”. Most people turned to attaching files to email.



FTP, SMTP, or SENDFILE

- Rice University and IETF
 - BITNET hub, DECNET hub, Internet/NSFNET hub
- Jon Postel and Joyce Reynolds
 - “Jon and I want to give you port 608”

FTP required credentials at the target end.
SMTP was tuned for correspondence.
(though both are plain text protocols)



UFT was born



UFT for SENDFILE

- RFC 1440 (all day long)
- UFT/1
- TYPE A or I (following FTP, or others) for canonization
- As much metadata as you know (name, date, class, form)

Transaction is *direct* via TCP/IP



UFT for SENDFILE

- RFC *next*
- UFT/2
- DATA statement requires a burst size (number of bytes)
- META statement added to allow arbitrary attributes

... two changes from UFT/1 to UFT/2



UFT Protocol

client

connects

FILE *size from auth*

USER *recipient*

TYPE **A** (or **I** or **N** or other)

DATA *bytecount*

bytes

server

sends herald

200 ACK

200 ACK

200 ACK

300 "send it"

200 ACK

metadata
goes here



UFT Protocol

metadata

NAME *filename*

DATE *date time zone*

CLASS *class [devtype]*

RECFM *recordformat*

PROT *protection*

OWNER *username*

unique to VM

unique to Unix

`github.com/trothtech/uft/
blob/main/doc/protocol.md`



UFT Protocol

metadata

META NAME *filename*

META DATE *date time zone*

META CLASS *class [devtype]*

unique to VM

META RECFM *recordformat*

META PROT *protection*

unique to Unix

META OWNER *username*



UFT Project

modified

- client/server
- VM/CMS: SENDFILE EXEC (client) UFTD EXEC (server)
- Unix, Linux: `uftc.c` ('sf', client), `uftd.c` (server)

<https://github.com/trothtech/uft/>



UFT at VSSI

- Just a tool
- Primarily VM-to-VM
- Initially controversial, but then ...
- “Hey, what if we ...”
 - Make it work like RSCS
 - Make it into a product ... and we did and called it VFILE



VFILE contrast with RSCS

- No secondary topology (no forwarding)
- No multi-hop queuing (no intermediate delays)
- No proprietary protocol (protocol is an open standard)
- Significantly simpler client operation (e.g., laptops)
- Does not require GCS
- No CTCs, no VTAM, ...
- More participating peers

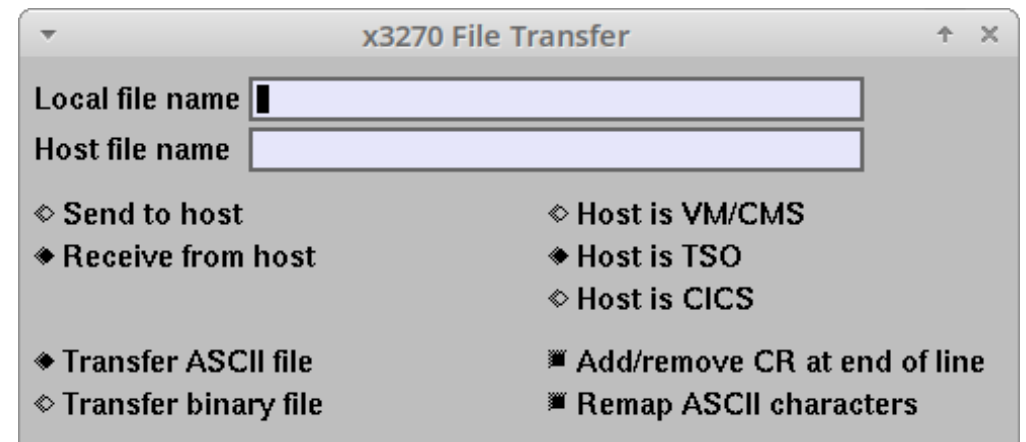


VFILE compared to IND\$FILE

“almost good enough”

- No commandeering your terminal
- No disruption of the transfer (e.g., by message traffic)
- Unattended operation works

```
sf file.txt to user at host  
$ sf -a file.txt user@host
```





VFILE on CMS

TROTHR RDRLIST A0 V 164 Trunc=164 Size=7 Line=1 Col=1 Alt=0

Cmd	Filename	Filetype	Class	User	at	Node	Hold	Records	Date	Time
740TEST	USERBASE	PUN	A	INSTALID		VMA	NONE	4371	2025-02-11	19:28:02
SYSTEM	NETID	PUN	A	MAINT		RMTZVM	NONE	24	2025-03-07	10:46:18
PROFILE	EXEC	PUN	A	VSS71		VMA	NONE	34	2025-03-26	15:10:04
SYSTEM	NETID	PUN	A	MAINT730		VMA	NONE	134	2025-04-02	11:54:43
(none)	(none)	PRT	A	ROOT		HLINUX01	NONE	86	2025-04-04	11:15:10
UFTD	(none)	PUN	A	TROTHR		HLINUX01	NONE	476	2025-04-14	17:11:56
GONE	VMARC	PUN	A	TROTHR		RMTT16	NONE	699	2025-04-25	09:54:59



VFILE on Linux

```
$ rls          remuser  remhost          spid
I----- 1 vss33    vssivma         91136 Jun 13 17:12 0012 hcpmods.doc
A----- 1 rmt     freebsd         635  Mar 07 14:28 0038
N----- 1 maint730 vma            17680 Apr 04 10:00 0044 profile.exec
N----- 1 maint730 vma             2960 Apr 04 10:00 0045 profile.xedit
N----- 1 maint730 vma            10720 Apr 04 10:01 0046 system.netid
I----- 1 rmt     pioneer        29410 Apr 09 12:52 0047
A----- 1 trothr  rmtt16         881  Apr 09 14:53 0049 .profile
I----- 1 trothr  vma           132480 Apr 14 14:53 0054 sendfile.vmarc
```



VFILE on Linux

```
$ rls          remuser  remhost          spid
I-A----- 1 vss33    vssivma        91136 Jun 13 17:12 0012 hcpmods.doc
A----- 1 rmt      freebsd         635  Mar 07 14:28 0038
N-A----- 1 maint730 vma            17680 Apr 04 10:00 0044 profile.exec
|||||\_____ msg (M) nomsg (-)
|||||\_____ keep (K) consume (-)
|||||\_____ hold (H) nohold (-)
|||\_____ devtype (prT, Con, pUn)
||\_____ class (A, B, C, etc.)
|\_____ CC (Asa, Machine, none)
\_____ type (I or A or N)
```



VFILE on Linux

```
Message from root@rmt-t16 on <no tty> at 13:33 ...
```

```
UFTSRV1004 File 0003 spooled to rmt origin nord@pioneer  
EOF
```

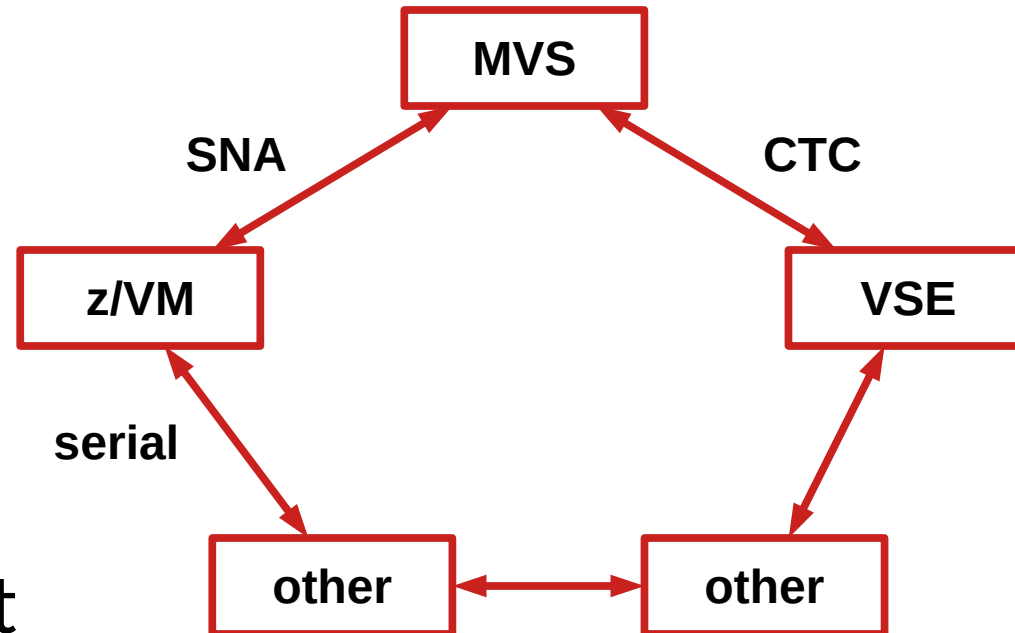
```
Message from root@rmt-t16 on <no tty> at 13:32 ...
```

```
From pioneer(nord): hi  
EOF
```

... assuming 'write' finds your TTY



Traditional NJE

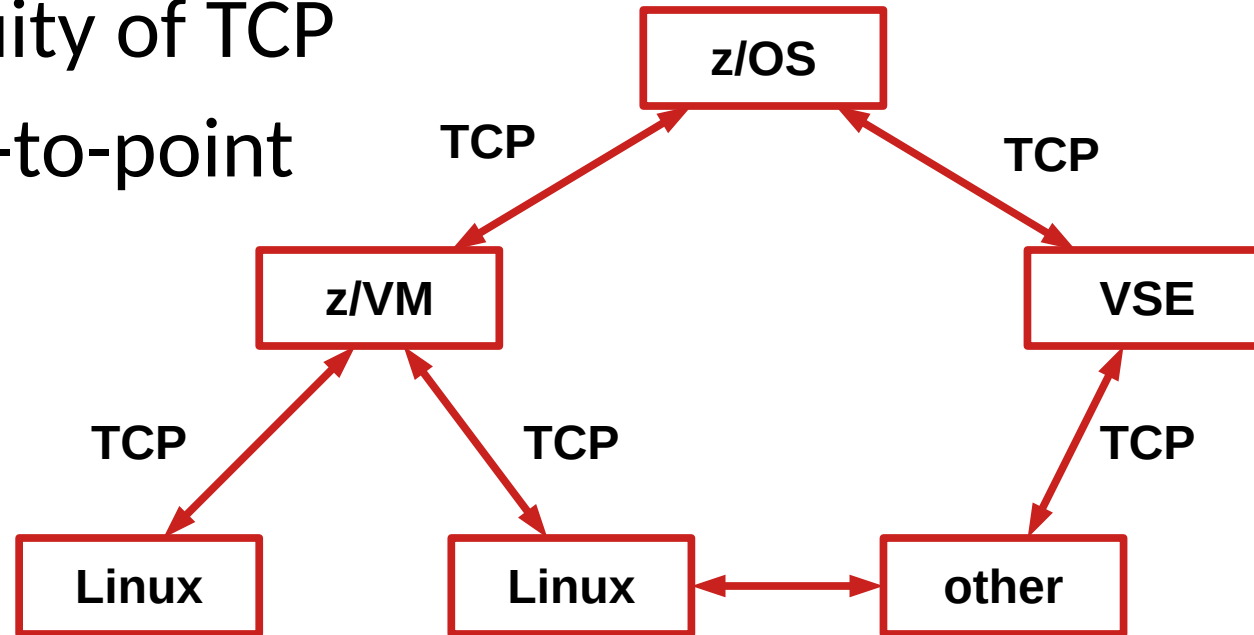


- All point-to-point
- Various link types



Modern NJE

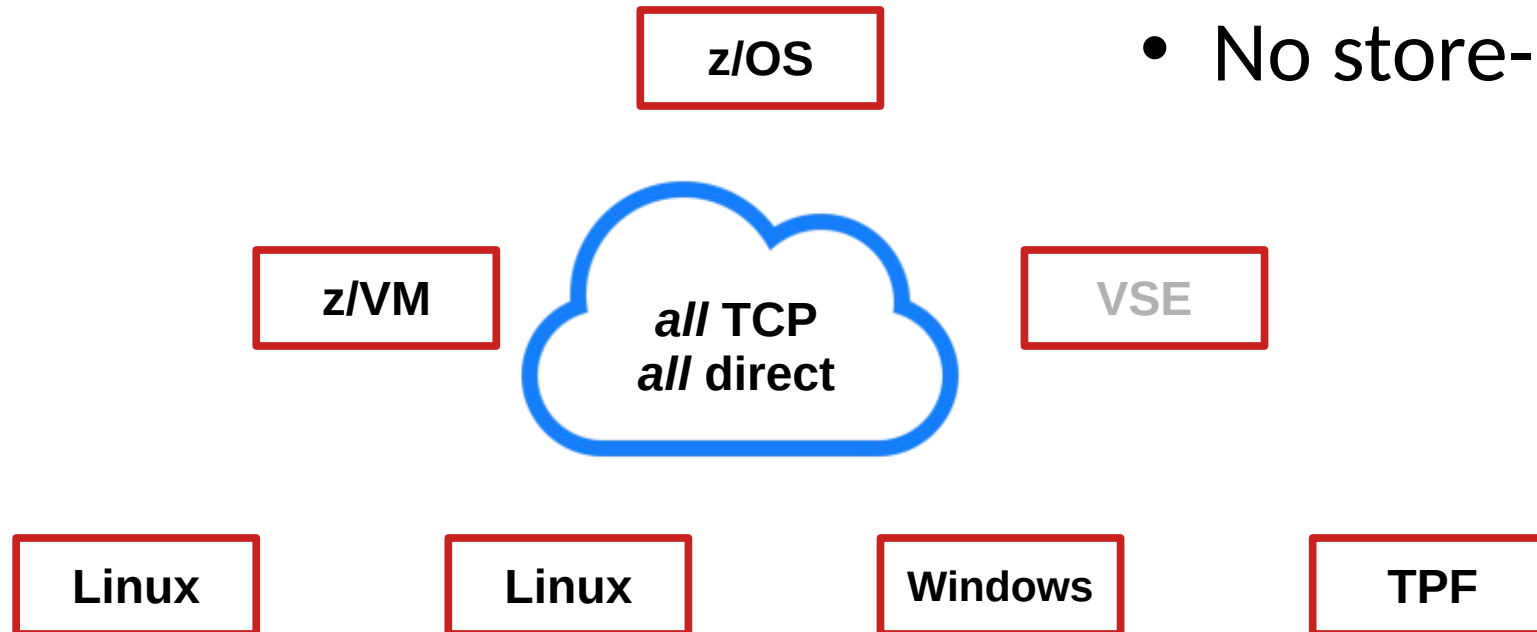
- The ubiquity of TCP
- Still point-to-point





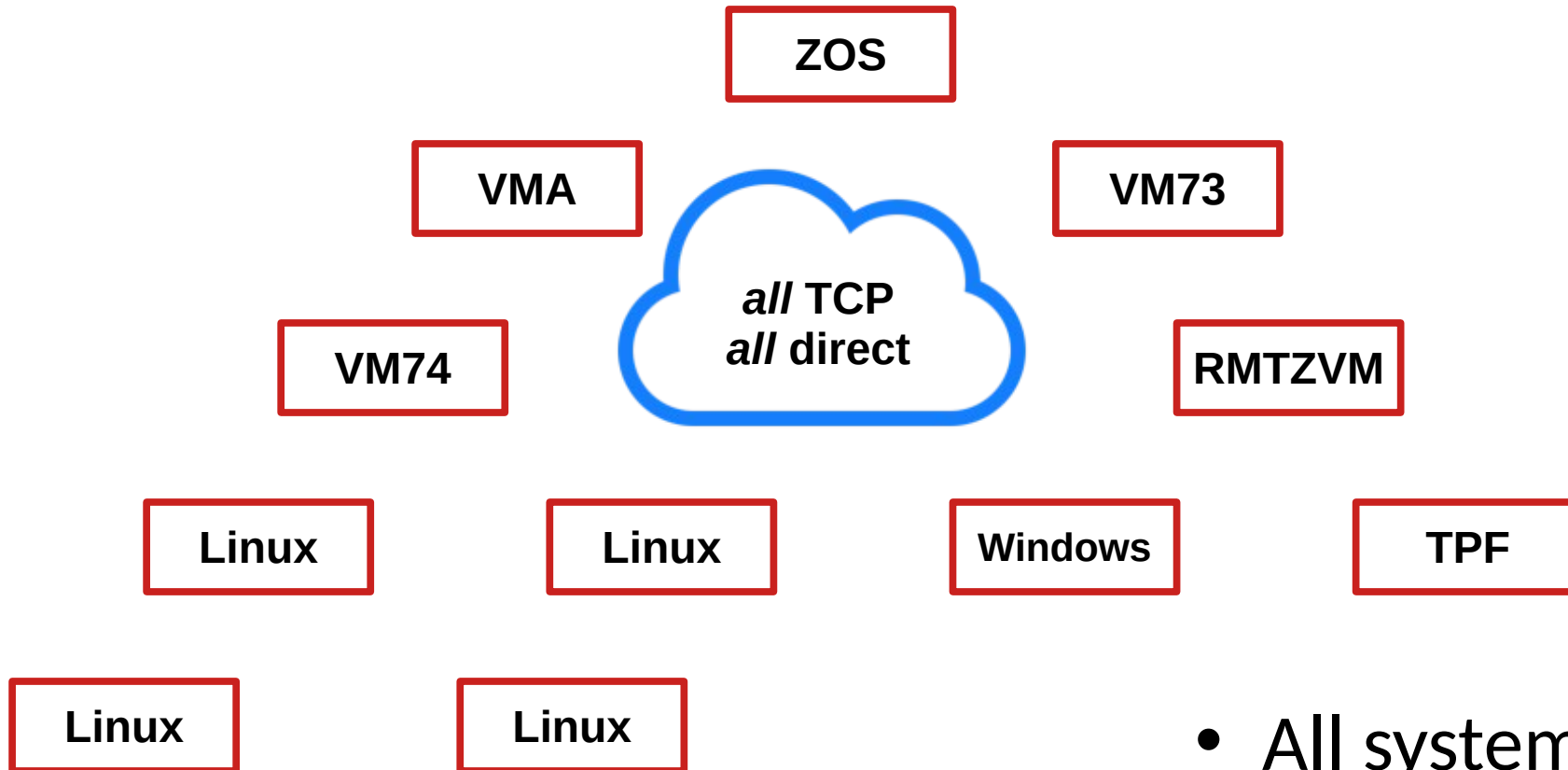
VFILE Network

- No fixed links
- No store-and-forward





VFILE at VSSI



- All systems
- Any platform



VFILE Deliverables

`vf1s == cp query rdr`
`vfmon`
`vfrcv`
`vfrm == cp purge rdr`
`vfmv == cp transfer rdr`
`vfcp`
`vfr1`
`vfpeek`

Product consistency
with

Platform conformity

(Linux commands
look like Linux;
Windows commands
look like Windows)



this page intentionally left blank
(break here for demos)



VFILE Summary

- File transfer between systems just like RSCS gave you,
- Without the hassle store-and-forward topology maint
- Alternative to passwords into your FTP jobs
- Alternative to IND\$FILE
- Lower complexity and increased connectivity
- Does not preclude continued use of RSCS



VFILE Summary

Now, your Linux, Windows, z/OS, z/VM and other platforms can participate in sender-initiated file transfers without having to know your entire network layout or use proprietary protocols. VFILE uses open internet protocols to do its work, without having to maintain BITNET-style tables of next hosts for file and message transfer.



VFILE Summary

With VFILE, you can accept files from across the world – *or not* - it's your decision to make! With flexible ACCEPT and DENY list processing, you can quickly configure to accept files from only pre-approved hosts, dynamically altered and easily implemented.



thank you!

Rick Troth, VSSI

`<richard.troth@vsoftsys.com>`



supplemental

UFT server, apart from VFILE,
requires `'inetd'` or `'xinetd'` with limited controls.

VFILE provides explicit allow/deny for specific peers.



supplemental

UFT 's£' client, even on CMS, is a true TCP/IP client.

This is okay for Unix/Linux but not so much on z/VM.

VFILE provides a service virtual machine that works like RSCS, accepting files from users and then sending the files on behalf of the users. With VFILE, users are not relegated to TCP/IP services. (They don't have to access the TCP/IP product disk to send files.)



supplemental - SIFT

SIFT = “Sender-Initiated File Transfer”
and is sometimes used conversationally in place of UFT.

Within the UFT project, SIFT is an offline form of UFT protocol.
Stages in the VM implementation can read a “SIFT job”.



supplemental - mobile coding

- Reaction to Internet consumer explosion
- Developed while working at Rice University
- Linux 0.99 riding Houston Metro bus
- On a Compaq Concerto pen tablet
- Blew away “Windows for Pen Computing”
and put Linux on it (of course)





Fun with UFT

Found some long-lived TCP connections on a development host

Noted their IP addresses and checked reverse DNS

- `vulnscan1.cyhy.ncats.cyber.dhs.gov`
- `vulnscan2.cyhy.ncats.cyber.dhs.gov`
- `vulnscan3.cyhy.ncats.cyber.dhs.gov ...`

very interesting

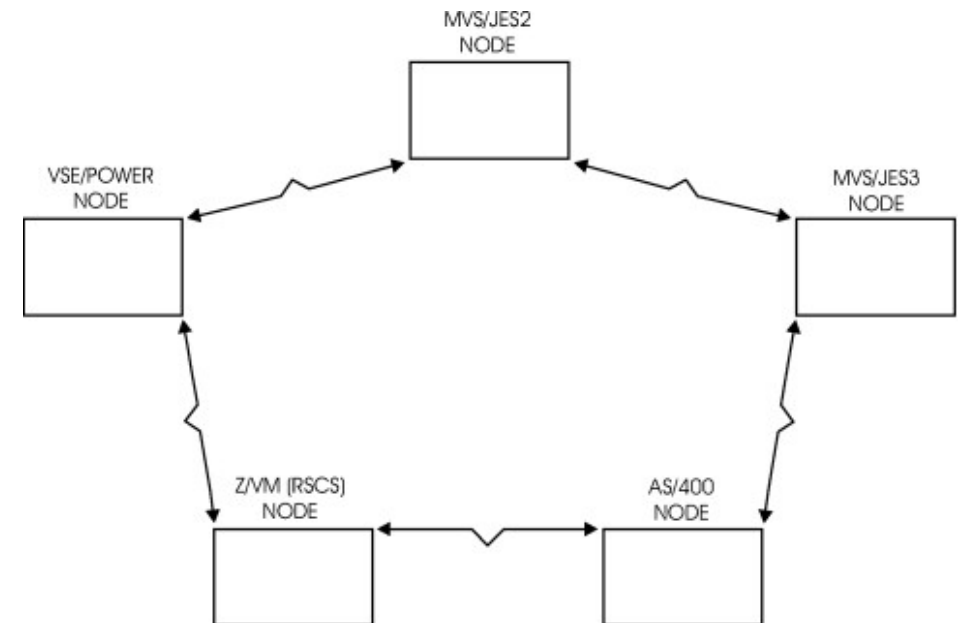




supplemental - HNET

- Modern BITNET, call it HNET
- Largely Hobbyist
- Multi-Platform (VM, MVS, DEC VMS, Unix, others)

A community, but small





VFILE from VSSI

File Transfer: Shaken, not Stirred

Rick Troth, VSSI

<richard.troth@vsoftsys.com>



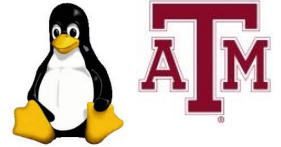
Disclaimer

This presentation discusses intended directions and developments. It should not be taken as guaranteed delivery or features. The audience must not expect any eventual result to match what is presented here.

A
B
C



about:Rick



- VM (SP, et al) since 1982, also VMware, Xen, KVM
- MVS (batch) 1981, WYLBUR same time, TSO later (sans ISPF)
- Unix (AIX, UTS, SunOS) 1987
- Linux since 1993
- “Open Edition” 1995

Unix → Linux

the Open Source thing



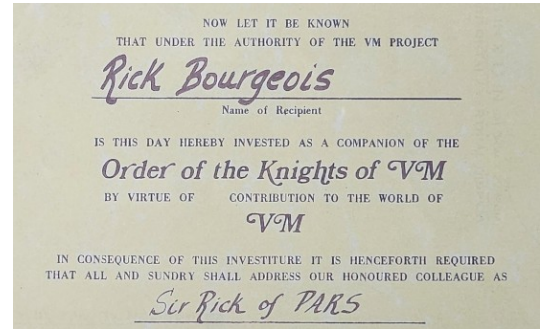
VM Workshop

One honor that I'm greatly thankful for:
Knights of VM: Sir Santa



about:VSSI

- The VPARS and VTAPE people
- Established in 1982
- Founder is a Knight of VM
- Customer base is typically Fortune 100
- Established solid reputation for product support
- Support cost included in yearly license cost



Virtual Software Systems, Inc.
7715 Browns Bridge Road
Gainesville, Georgia 30506
(770) 781-3200 Fax (770) 781-3210

Rick Bourgeois founder.
Richard Bourgeois (grandson) owner and CEO.



about:VFILE

- Operational equivalent to RSCS, but ...
- Uses UFT protocol rather than NJE
- Supporting z/VM, Linux, z/Linux, z/OS, Windows, TPF
- Available 1Q26

VFILE makes sender-initiated file transfer *easier* to manage and more *broadly* available compared to NJE or other methods.

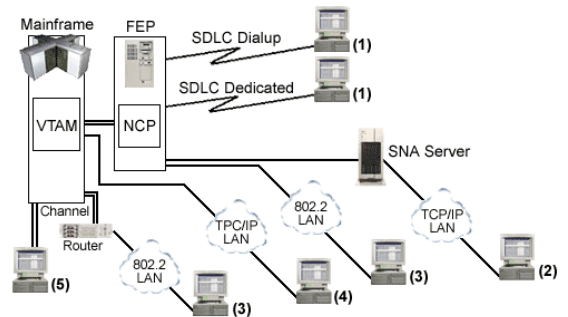
The problem we're solving is the complexity with existing sender-initiated methods like NJE.



History: RSCS, NJE, HASP, oh my!

- Houston Automatic Spooling Priority
- HASP became JES2
- ASP became JES3
- RSCS v1

The world was point-to-point
NJE today remains point-to-point



https://en.wikipedia.org/wiki/Houston_Automatic_Spooling_Priority



Whither BITNET



- BITNET (including EARN, NETNORTH, Asia)
- Largely Academic and Research (IBM had VNET)
- Multi-Platform (VM, MVS, DEC VMS, Unix, others)

Significant communities ... but ...

With the commercialization of TCP/IP Internet in the early 1990s, BITNET fell from popularity. One could also no longer “just send a file”. Most people turned to attaching files to email.

This was my world in those days.

Both Texas A&M (alma mater) and Rice University (employer) were on BITNET. Most of my own network activity was via BITNET.

BITNET was multi-platform, never the less there were significant platforms which were excluded.



FTP, SMTP, or SENDFILE

- Rice University and IETF
 - BITNET hub, DECNET hub, Internet/NSFNET hub
- Jon Postel and Joyce Reynolds
 - “Jon and I want to give you port 608”

FTP required credentials at the target end.
SMTP was tuned for correspondence.
(though both are plain text protocols)



UFT was born

Reaction to Internet consumer explosion
Missing the ability to “just send a file”

So now, not only an alternative protocol,
but also more platforms than NJE covered.



UFT for SENDFILE

- RFC 1440 (all day long)
- UFT/1
- TYPE A or I (following FTP, or others) for canonization
- As much metadata as you know (name, date, class, form)

Transaction is *direct* via TCP/IP



UFT for SENDFILE

- RFC *next*
- UFT/2
- DATA statement requires a burst size (number of bytes)
- META statement added to allow arbitrary attributes

... two changes from UFT/1 to UFT/2



UFT Protocol

<i>client</i>	<i>server</i>	
connects	sends herald	
FILE <i>size from auth</i>	200 ACK	
USER <i>recipient</i>	200 ACK	
TYPE A (or I or N or other)	200 ACK	
<hr/>		
DATA <i>bytecount</i>	300 "send it"	metadata goes here
<i>bytes</i>	200 ACK	



UFT Protocol

metadata

NAME *filename*

DATE *date time zone*

CLASS *class [devtype]*

unique to VM

RECFM *recordformat*

PROT *protection*

unique to Unix

OWNER *username*

[github.com/trothtech/uft/
blob/main/doc/protocol.md](https://github.com/trothtech/uft/blob/main/doc/protocol.md)



UFT Protocol

metadata

META NAME *filename*

META DATE *date time zone*

META CLASS *class [devtype]* unique to VM

META RECFM *recordformat*

META PROT *protection* unique to Unix

META OWNER *username*



UFT Project

modified

- client/server
- VM/CMS: SENDFILE EXEC (client) UFTD EXEC (server)
- Unix, Linux: `uftc.c` ('sf', client), `uftd.c` (server)

<https://github.com/trothtech/uft/>

Contrast VFILE with UFT:
VFILE requires no changes to your CMS environment.



UFT at VSSI

- Just a tool
- Primarily VM-to-VM
- Initially controversial, but then ...
- “Hey, what if we ...”
 - Make it work like RSCS
 - Make it into a product ... and we did and called it VFILE

Our security guy did not grow up in a world where sending a file without credentials or authentication was a normal thing to do. He kinda freaked out. (We also did not employ SSL/TLS at that stage, which made his stress even worse.)

“Having someone send me a file is no different than having someone send me a letter via USPS. I don’t want the postman to have a key to my house just to deliver the mail.”



VFILE contrast with RSCS

- No secondary topology (no forwarding)
- No multi-hop queuing (no intermediate delays)
- No proprietary protocol (protocol is an open standard)
- Significantly simpler client operation (e.g., laptops)
- Does not require GCS
- No CTCs, no VTAM, ...
- More participating peers

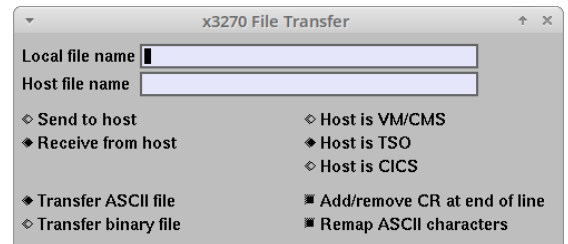


VFILE compared to IND\$FILE

“almost good enough”

- No commandeering your terminal
- No disruption of the transfer (e.g., by message traffic)
- Unattended operation works

```
sf file.txt to user at host  
$ sf -a file.txt user@host
```



I did not know that there was so much “IND\$FILE hate” in the world, though I avoid it myself.

An alternative to IND\$FILE to/from workstations was one of the motivators of the VFILE product.



VFILE on CMS

TROTHR RDRLIST A0 V 164 Trunc=164 Size=7 Line=1 Col=1 Alt=0

Cmd	Filename	Filetype	Class	User	at Node	Hold	Records	Date	Time
740TEST	USERBASE	PUN	A	INSTALID	VMA	NONE	4371	2025-02-11	19:28:02
SYSTEM	NETID	PUN	A	MAINT	RMTZVM	NONE	24	2025-03-07	10:46:18
PROFILE	EXEC	PUN	A	VSS71	VMA	NONE	34	2025-03-26	15:10:04
SYSTEM	NETID	PUN	A	MAINT730	VMA	NONE	134	2025-04-02	11:54:43
(none)	(none)	PRT	A	ROOT	HLINUX01	NONE	86	2025-04-04	11:15:10
UFTD	(none)	PUN	A	TROTHR	HLINUX01	NONE	476	2025-04-14	17:11:56
GONE	VMARC	PUN	A	TROTHR	RMTT16	NONE	699	2025-04-25	09:54:59



VFILE on Linux

```
$ rls          remuser remhost          spid
I----- 1 vss33    vssivma    91136 Jun 13 17:12 0012 hcpmods.doc
A----- 1 rmt      freebsd    635 Mar 07 14:28 0038
N----- 1 maint730 vma        17680 Apr 04 10:00 0044 profile.exec
N----- 1 maint730 vma         2960 Apr 04 10:00 0045 profile.xedit
N----- 1 maint730 vma        10720 Apr 04 10:01 0046 system.netid
I----- 1 rmt      pioneer    29410 Apr 09 12:52 0047
A----- 1 trothr   rmtt16     881 Apr 09 14:53 0049 .profile
I----- 1 trothr   vma       132480 Apr 14 14:53 0054 sendfile.vmarc
```



VFILE on Linux

```
$ rls          remuser  remhost          spid
I-A-----  1 vss33    vssivma         91136 Jun 13 17:12 0012 hcpmods.doc
A-----  1 rmt      freebsd          635  Mar 07 14:28 0038
N-A-----  1 maint730 vma             17680 Apr 04 10:00 0044 profile.exec
| | | | | \___ msg (M) nomsg (-)
| | | | \___ keep (K) consume (-)
| | | \___ hold (H) nohold (-)
| | \___ devtype (prT, Con, pUn)
| \___ class (A, B, C, etc.)
| \___ CC (Asa, Machine, none)
\___ type (I or A or N)
```



VFILE on Linux

Message from root@rmt-t16 on <no tty> at 13:33 ...

```
UFTSRV1004  File 0003 spooled to rmt origin nord@pioneer
EOF
```

Message from root@rmt-t16 on <no tty> at 13:32 ...

```
From pioneer(nord): hi
EOF
```

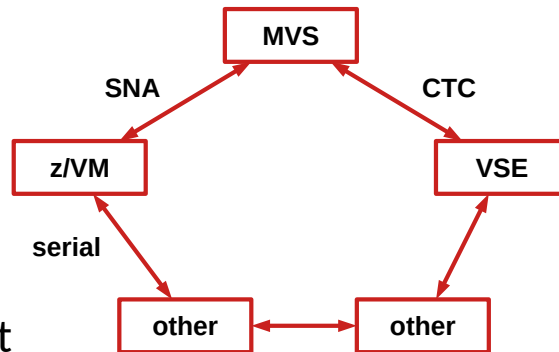
... assuming `write` finds your TTY

Use the `mesg` command on most Unix/Linux systems to turn off `write` reception for a given TTY.

Better to use `vfmon` to catch all such traffic in a controlled manner.



Traditional NJE



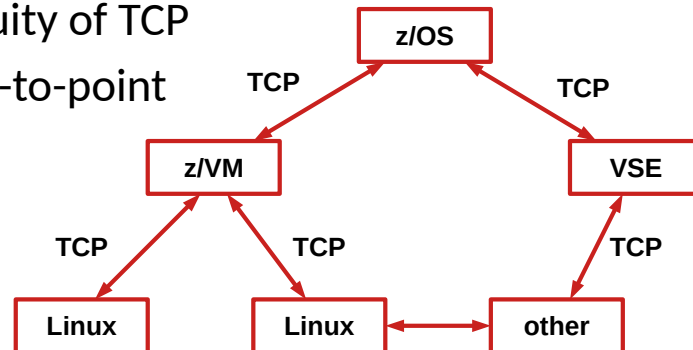
- All point-to-point
- Various link types

This is how sender-initiated file transfer was at first.



Modern NJE

- The ubiquity of TCP
- Still point-to-point



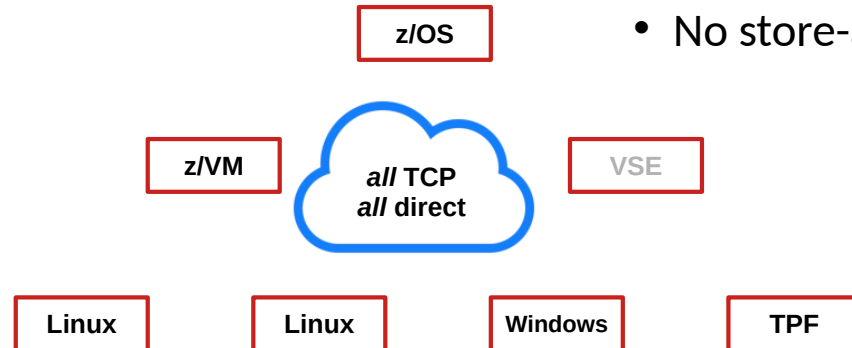
Today, we leverage TCP/IP for connectivity.

NJE networks remain point-to-point and store-and-forward. (Not bad, just complicated.)



VFILE Network

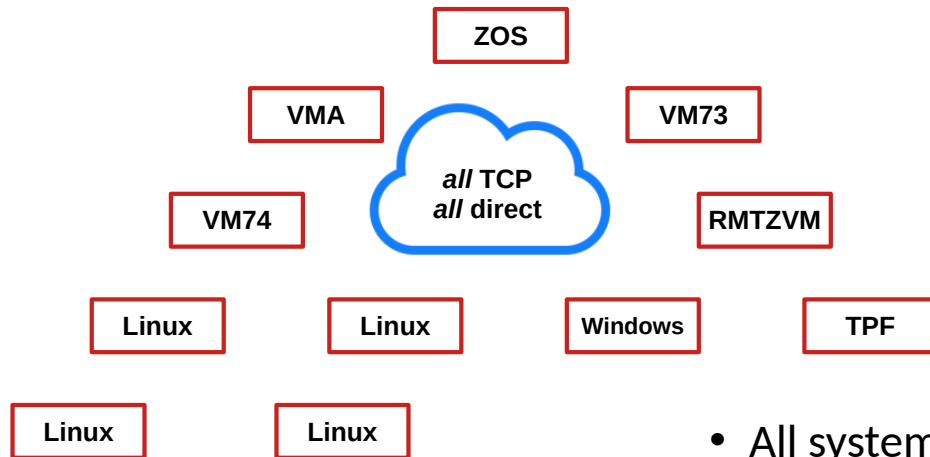
- No fixed links
- No store-and-forward



No fixed links.
No secondary topology.
No store-and-forward or file routing.
All UFT transactions are direct.



VFILE at VSSI



- All systems
- Any platform

Eating our own dogfood, of course!

We at VSSI use VFILE to move files quickly and reliably from system to system and across platforms.



VFILE Deliverables

<code>vfis</code>	<code>== cp query rdr</code>	Product consistency
<code>vfmon</code>		with
<code>vfrcv</code>		Platform conformity
<code>vfrm</code>	<code>== cp purge rdr</code>	
<code>vfmv</code>	<code>== cp transfer rdr</code>	(Linux commands
<code>vfcp</code>		look like Linux;
<code>vfrl</code>		Windows commands
<code>vfpeek</code>		look like Windows)

We can't use 'cp' on Linux because that command verb is already taken. We won't use 'CP' because shift is annoying. We could possibly borrow some verbs from CP and CMS, but that would induce some inconsistencies.

So we prefix "vf" to all product-specific utility names.

VFILE is designed to be user friendly from the get go.



this page intentionally left blank
(break here for demos)



VFILE Summary

- File transfer between systems just like RSCS gave you,
- Without the hassle store-and-forward topology maint
- Alternative to passwords into your FTP jobs
- Alternative to IND\$FILE
- Lower complexity and increased connectivity
- Does not preclude continued use of RSCS

About RSCS:
VFILE is not a “rip and replace”.



VFILE Summary

Now, your Linux, Windows, z/OS, z/VM and other platforms can participate in sender-initiated file transfers without having to know your entire network layout or use proprietary protocols. VFILE uses open internet protocols to do its work, without having to maintain BITNET-style tables of next hosts for file and message transfer.



VFILE Summary

With VFILE, you can accept files from across the world – *or not* - it's your decision to make! With flexible ACCEPT and DENY list processing, you can quickly configure to accept files from only pre-approved hosts, dynamically altered and easily implemented.

Strictly speaking: no Z needed.
VFILE is a multi-platform product based on an open standard Internet protocol.



thank you!

Rick Troth, VSSI

`<richard.troth@vsoftsys.com>`



supplemental

UFT server, apart from VFILE,
requires 'inetd' or 'xinetd' with limited controls.

VFILE provides explicit allow/deny for specific peers.



supplemental

UFT 's£' client, even on CMS, is a true TCP/IP client.

This is okay for Unix/Linux but not so much on z/VM.

VFILE provides a service virtual machine that works like RSCS, accepting files from users and then sending the files on behalf of the users. With VFILE, users are not relegated to TCP/IP services. (They don't have to access the TCP/IP product disk to send files.)



supplemental – SIFT

SIFT = “Sender-Initiated File Transfer”
and is sometimes used conversationally in place of UFT.

Within the UFT project, SIFT is an offline form of UFT protocol.
Stages in the VM implementation can read a “SIFT job”.



supplemental - mobile coding

- Reaction to Internet consumer explosion
- Developed while working at Rice University
- Linux 0.99 riding Houston Metro bus
- On a Compaq Concerto pen tablet
- Blew away “Windows for Pen Computing”
and put Linux on it (of course)





Fun with UFT

Found some long-lived TCP connections on a development host
Noted their IP addresses and checked reverse DNS

- `vulnscan1.cyhy.ncats.cyber.dhs.gov`
- `vulnscan2.cyhy.ncats.cyber.dhs.gov`
- `vulnscan3.cyhy.ncats.cyber.dhs.gov ...`

vevy intewesting





supplemental - HNET

- Modern BITNET, call it HNET
- Largely Hobbyist
- Multi-Platform (VM, MVS, DEC VMS, Unix, others)

A community, but small

