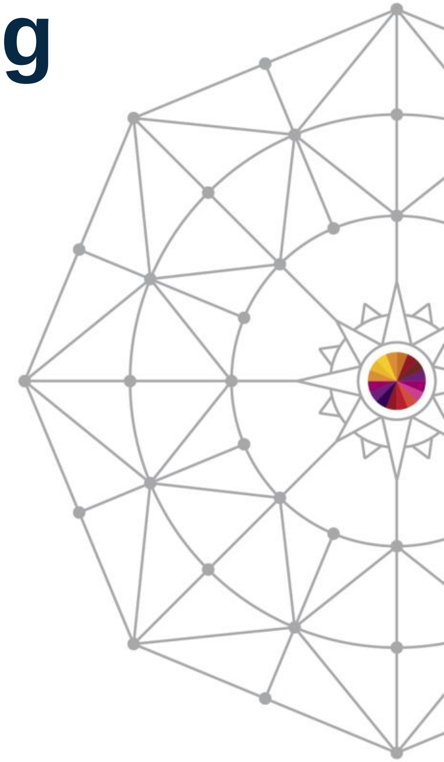
z/OS Hybrid Batch Processing and Big Data

Stephen Goetze
Kirk Wolf
Dovetailed Technologies, LLC

Thursday, August 7, 2014: 1:30 PM-2:30 PM
Session 15496

www.dovetail.com



#SHAREorg

Trademarks

- Co:Z® is a registered trademark of Dovetailed Technologies, LLC
- z/OS®, DB2®, zEnterprise® and zBX® are registered trademarks of IBM Corporation
- Oracle® and Java® are registered trademarks of Oracle and/or its affiliates
- Hadoop®, HDFS™, Hive™, HBase™ and Pig™ are either registered trademarks or trademarks of the Apache Software Foundation
- Linux® is a registered trademark of Linus Torvalds

Agenda

- Define Hybrid Batch Processing
- Hello World Example
- Security Considerations
- Hybrid Batch Processing and Big Data
  - Processing z/OS syslog data with Hive
  - Processing z/OS DB2 data with RHadoop
- Summary / Questions

zEnterprise Hybrid Computing Models

Well Known:

- zBX/zLinux as user-facing edge, web and application servers
  - z/OS provides back-end databases and transaction processing
- zBX as special purpose appliances or optimizers
  - DB2 Analytics Accelerator
  - DataPower

Another Model: **z/OS Hybrid Batch**

- zBX/zLinux/Linux/Windows integrated with z/OS batch

z/OS Hybrid Batch Processing

1. The ability to execute a program or script on a virtual server from a z/OS batch job step
2. The target program may already exist and should require little or no modification
3. The target program's input and output are redirected from/to z/OS spool files or datasets
4. The target program may easily access other z/OS resources: DDs, data sets, POSIX files and programs
5. The target program's exit code is adopted as the z/OS job step condition code

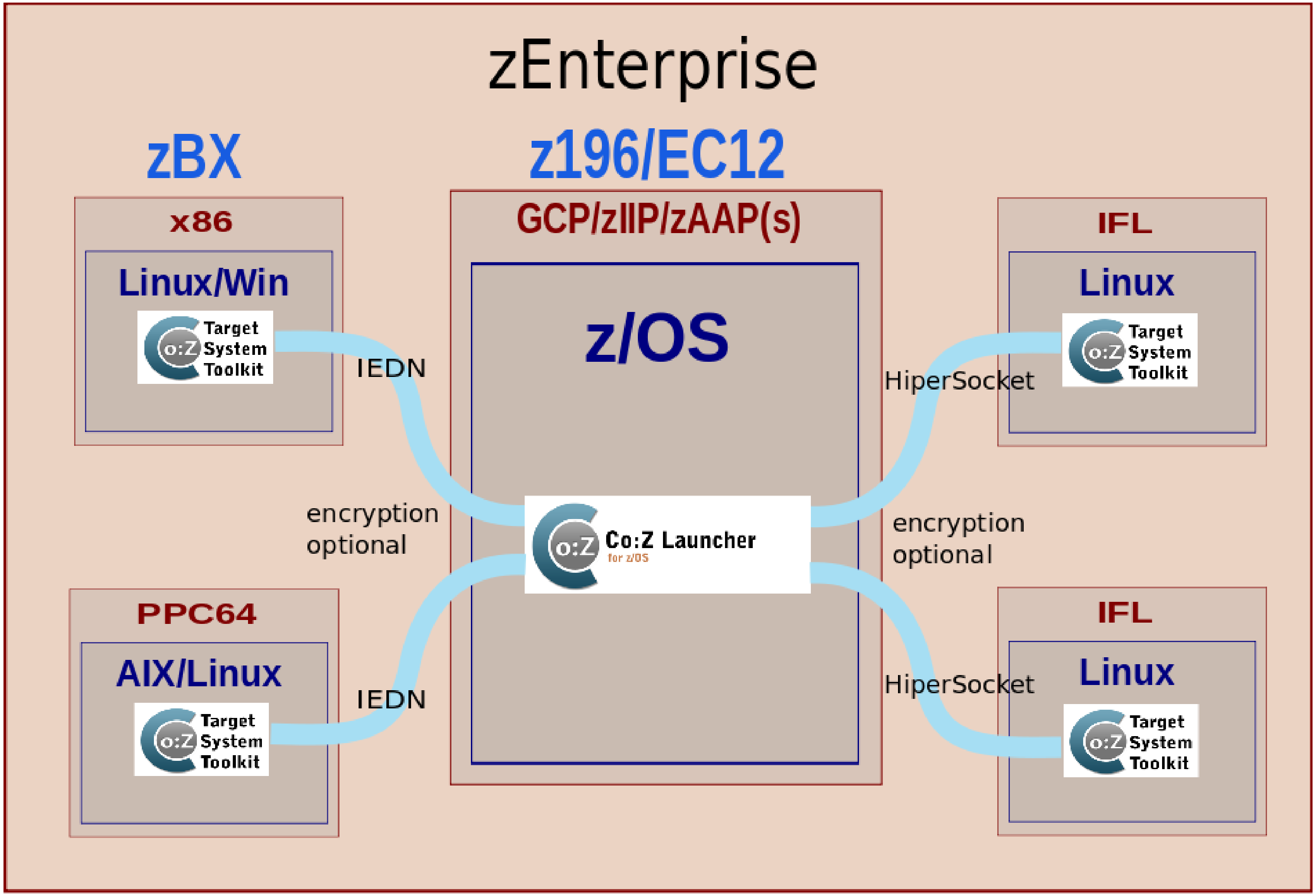
Data security governed by SAF (RACF/ACF2/TSS)

Requires new enablement software…

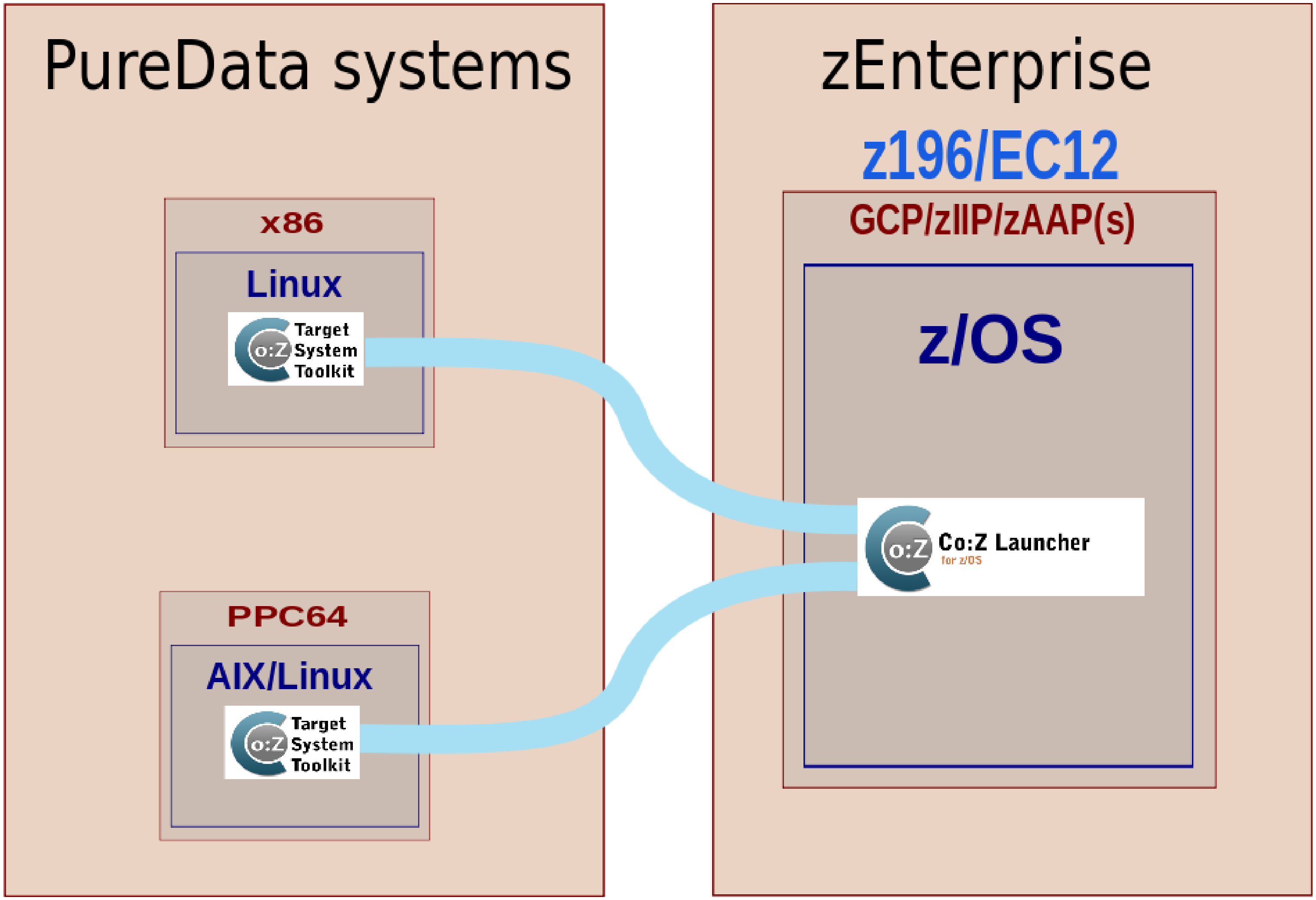
Co:Z Co-Processing Toolkit

- Implements z/OS Hybrid Batch model
- Co:Z Launcher starts a program on a target server and automatically redirects the standard streams back to jobstep DDs
- The target program can use Co:Z DatasetPipes commands to reach back into the active jobstep and access z/OS resources:
  - **`fromdsn/todsn`** – read/write a z/OS DD or data set
  - **`fromfile/tofile`** – read/write a z/OS Unix file
  - **`cozclient`** – run z/OS Unix command
- Free (commercial support licenses are available)
- Visit http://dovetail.com for details

zEnterprise Hybrid Batch Processing

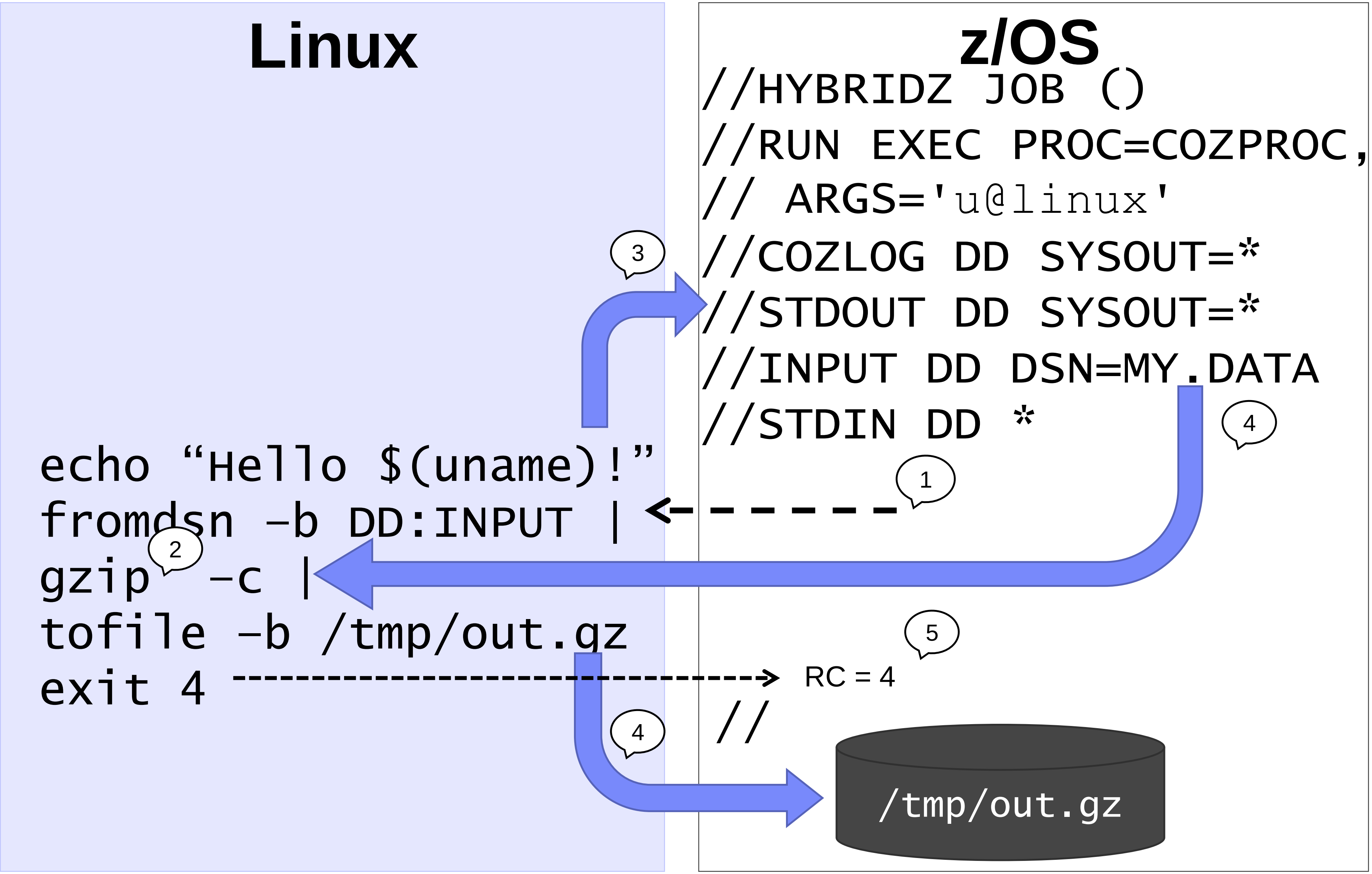


PureData Hybrid Batch Processing



Hybrid Batch – Hello World

- Simple example illustrating the principles of Hybrid Batch Processing
- Launch a process on a remote Linux server
  - Write a message to stdout
  - In a pipeline:
    - Read the contents of a dataset from a jobstep DD
    - Compress the contents using the Linux gzip command
    - Write the compressed data to the z/OS Unix file system
  - Exit with a return code that sets the jobstep CC



Linux

```
echo “Hello $(uname)!”
fromdsn -b DD:INPUT |
gzip -c |
tofile -b /tmp/out.gz
exit 4
```

z/OS

```
//HYBRIDZ JOB ()
//RUN EXEC PROC=COZPROC,
// ARGS='u@linux'
//COZLOG DD SYSOUT=*
//STDOUT DD SYSOUT=*
//INPUT DD DSN=MY.DATA
//STDIN DD *
```

```
//
```

Copyright © 2014 Dovetailed Technologies, LLC

Hello World: Hybrid Batch

1. A script is executed on a virtual server from a z/OS batch job step
2. The script uses a program that already exists -- **`gzip`**
3. Script output is redirected to z/OS spool
4. z/OS resources are easily accessed using **`fromdsn`**, **`tofile`**, etc…
5. The script exit code is adopted as the z/OS job step CC

Hello World – DD:STDOUT

```
Hello Linux!
```

Hello World – DD:COZLOG

```
CoZLauncher[N]: version: 2.2.0 2012-09-01
cozagent[N]: version: 1.1.0 2012-03-16
fromdsn(DD:STDIN)[N]: 5 records/400 bytes read…
fromdsn(DD:INPUT)[N]: 78 records/6240 bytes read…
tofile(/tmp/out.gz)[N]: … 1419 bytes written
todsn(DD:STDOUT)[N]: … 13 bytes written
todsn(DD:STDERR)[N]: … 0 bytes written
CoZLauncher[E]: u@linux target … ended with RC=4
```

Hello World – DD:JESMSGLG

```
JOB01515 ---- FRIDAY,  7 SEPT 2012 ----
JOB01515  IRR010I  USERID GOETZE   IS ASSIG…
JOB01515  ICH70001I GOETZE   LAST ACCESS AT…
JOB01515  $HASP373 HYBRIDZ  STARTED – INIT…
JOB01515  -
JOB01515  -STEPNAME PROCSTEP    RC   EXCP…
JOB01515  -RUN      COZLNCH     04   1345…
JOB01515  -HYBRIDZ  ENDED.  NAME-
JOB01515  $HASP395 HYBRIDZ  ENDED
```

Co:Z Hybrid Batch Network Security is Trusted

- OpenSSH is used for network security
  - – IBM Ported Tools OpenSSH client on z/OS
  - – OpenSSH sshd on target system
- By default, data transfer is tunneled (encrypted) over the ssh connection
  - – Optionally, data can be transferred over raw sockets (option: ssh-tunnel=false)
    - This offers very high performance without encryption costs
    - Ideal for a secure network, such as zEnterprise HiperSockets or IEDN

Co:Z Hybrid Batch Data Security is z/OS Centric

- All z/OS resource access is through the job step:
  - Controlled by SAF (RACF/ACF2/TSS)
  - Normal user privileges
- Storing remote user credentials in SAF digital certificates can extend the reach of the z/OS security envelope to the target system
  - Shared certificate access enables multiple authorized z/OS users to use a single target system id
- Dataset Pipes streaming technology can be used to reduce “data at rest”

Bash Process Substitution

- Make a command (or pipeline) appear as a file:

  `<(cmd) – "cmd"` appears as a readable /dev/fd/nn

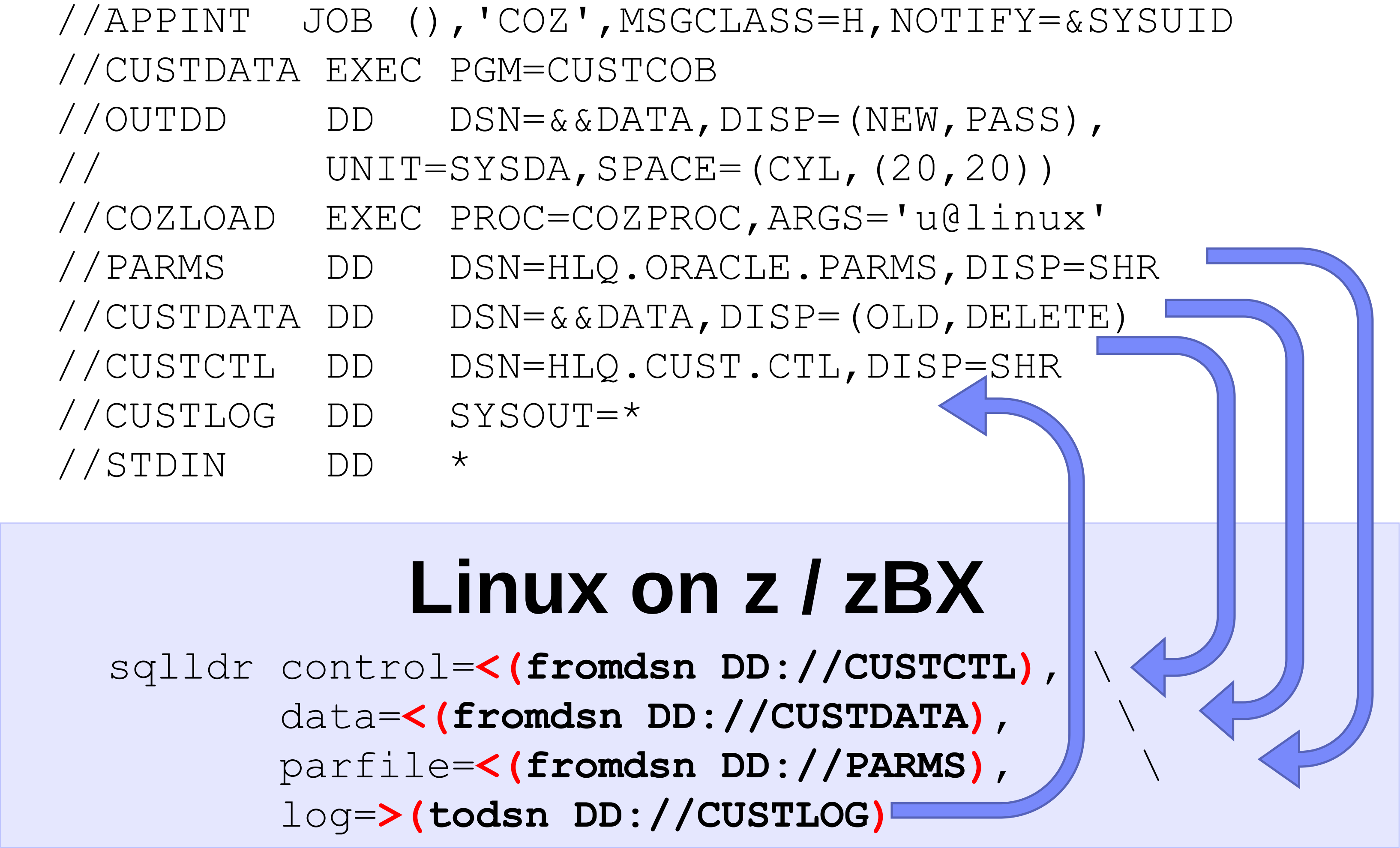
  `>(cmd) – "cmd"` appears as a writable /dev/fd/nn

- Example: `cat <(ls -al)` behaves like this:

```
ls -al > /dev/fd/63
cat /dev/fd/63
rm /dev/fd/63
```

- Very handy for enabling “data in flight” in hybrid batch processing…

z/OS



```
//APPINT   JOB (),'COZ',MSGCLASS=H,NOTIFY=&SYSUID
//CUSTDATA EXEC PGM=CUSTCOB
//OUTDD     DD   DSN=&&DATA,DISP=(NEW,PASS),
//               UNIT=SYSDA,SPACE=(CYL,(20,20))
//COZLOAD  EXEC PROC=COZPROC,ARGS='u@linux'
//PARMS     DD   DSN=HLQ.ORACLE.PARMS,DISP=SHR
//CUSTDATA DD    DSN=&&DATA,DISP=(OLD,DELETE)
//CUSTCTL  DD    DSN=HLQ.CUST.CTL,DISP=SHR
//CUSTLOG  DD    SYSOUT=*
//STDIN     DD   *
```

Linux on z / zBX

```
sqlldr control=<(fromdsn DD://CUSTCTL),  \
       data=<(fromdsn DD://CUSTDATA),     \
       parfile=<(fromdsn DD://PARMS),      \
       log=>(todsn DD://CUSTLOG)
```

Process Substitution Summary

- File centric utilities like **`sqlldr`** can be used without modification
- Facilitates concurrent transfer and loading:
  - *No data at rest!*
  - High performance
- Operations can observe real-time job output in the JES spool
- DatasetPipes commands combined with process substitution allow the SAF security envelope to be extended to the remote system

Big Data and z/OS

- z/OS systems often have the Big Data we want to analyze
  - Very large DB2 instances
  - Very large Data sets
- But, the Hadoop ecosystem is not well suited to z/OS
  - Designed for a cluster of many small relatively inexpensive computers
  - Although Hadoop is Java centric, several tools (e.g. R) don't run on z/OS
  - z/OS compute and storage costs are high
- Hybrid Batch Processing offers a solution
  - Single SAF profile for a security envelope extending to the BigData environment
  - Exploitation of high speed network links (HiperSockets, IEDN)
  - z/OS centric operational control

Co:Z Toolkit and Big Data

The Co:Z Launcher and Dataset Pipes utilities facilitate:

- Loading HDFS with z/OS data
  - DB2
  - VSAM, Sequential Data sets
  - Unix System Services POSIX files
- Map Reduce Analysis
  - Drive Hive, Pig, RHadoop, etc… with scripts maintained on z/OS
  - Monitor progress in the job log
- Move results to z/OS
  - Job spool
  - DB2
  - Data sets
  - POSIX files

Processing z/OS syslog data with Hive

- Connect z/OS Unix System Services file system syslog data and Hadoop

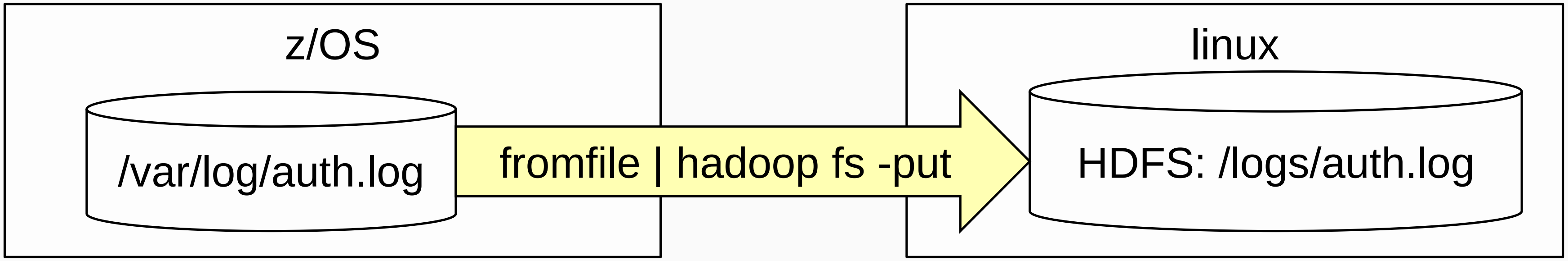
- Illustrate hybrid batch use of common Big Data tools:
  - **hadoop fs** – load Hadoop HDFS
  - **Hive** – run Map/Reduce with an SQL like table definition and query

Processing z/OS syslog data with Hive

- z/OS OpenSSH server logs authorization activity in a syslog Unix System Services file:
  - `/var/log/auth.log`
- Included in these messages are records of failed password authorization attempts for a userid:
  - `Failed password for invalid user <userid>`
- We wish to analyze this data to determine which userids are most commonly associated with failed password attempts

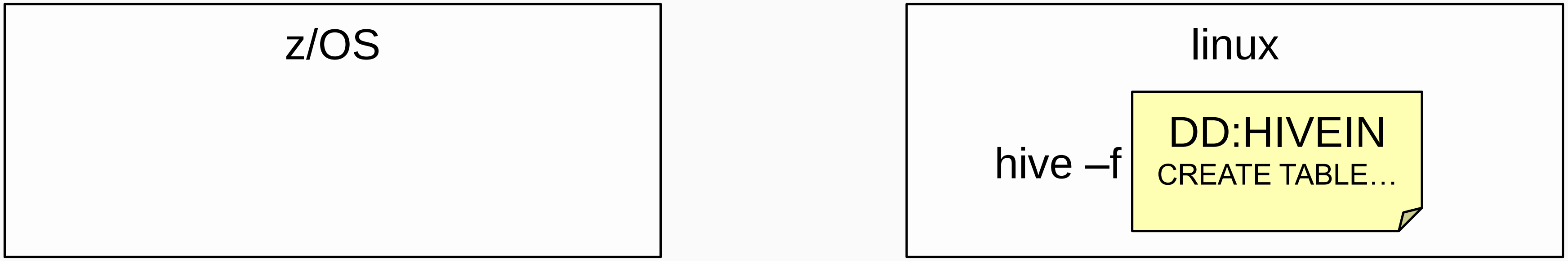
Processing z/OS syslog data with Hive

```
//COZUSERH JOB (),‘COZ',MSGCLASS=H,NOTIFY=&SYSUID
//RUNCOZ EXEC PROC=COZPROC,ARGS='-LI user@linux'
//COZCFG DD *
saf-cert=SSH-RING:RSA-CERT
ssh-tunnel=false
//HIVEIN DD DISP=SHR,DSN=COZUSER.HIVE.SCRIPTS(SYSLOG)
//STDIN  DD *
fromfile /var/log/auth.log | hadoop fs -put - /logs/auth.log
hive -f <(fromdsn DD:HIVEIN)
```



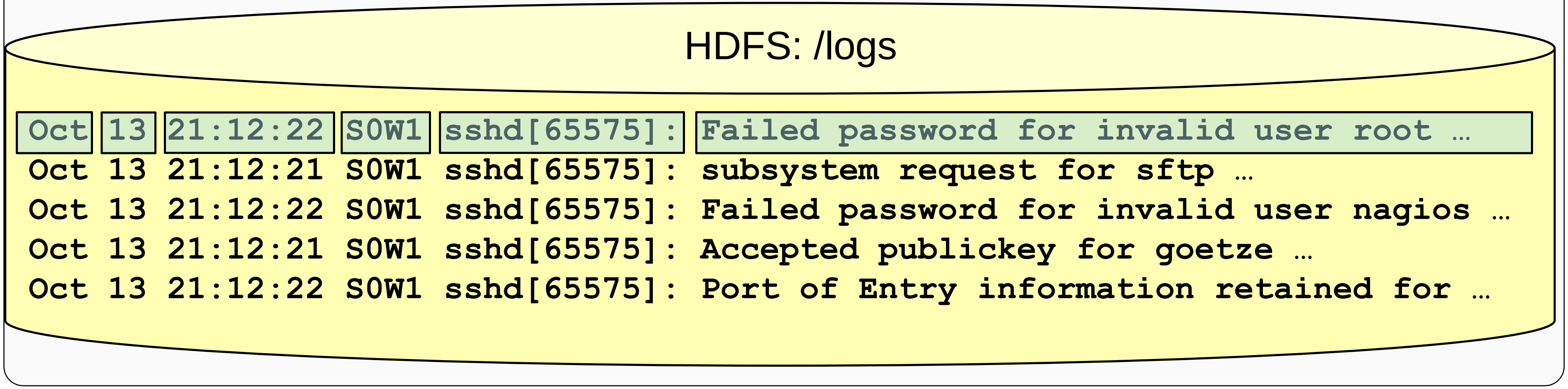
Processing z/OS syslog data with Hive

```
//COZUSERH JOB (),‘COZ',MSGCLASS=H,NOTIFY=&SYSUID
//RUNCOZ EXEC PROC=COZPROC,ARGS='-LI user@linux'
//COZCFG DD *
saf-cert=SSH-RING:RSA-CERT
ssh-tunnel=false
//HIVEIN DD DISP=SHR,DSN=COZUSER.HIVE.SCRIPTS(SYSLOG)
//STDIN  DD *
fromfile /var/log/auth.log | hadoop fs -put - /logs/auth.log
hive -f <(fromdsn DD:HIVEIN)
```



//HIVEIN DD DISP=SHR,DSN=COZUSER.HIVE.SCRIPTS(SYSLOG)

```
CREATE TABLE IF NOT EXISTS syslogdata (
 month STRING,
 day STRING,
 time STRING,
 host STRING,
 event STRING,
 msg STRING)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'
WITH SERDEPROPERTIES ("input.regex" =
 "(\\w+)\\s+(\\d+)\\s+(\\d+:\\d+:\\d+)\\s+(\\w+\\W*\\w*)\\s+(.*?\\:)\\s+(.*$)“
) STORED AS TEXTFILE LOCATION '/logs';
```



```
Oct 13 21:12:22 S0W1 sshd[65575]: Failed password for invalid user root …
Oct 13 21:12:21 S0W1 sshd[65575]: subsystem request for sftp …
Oct 13 21:12:22 S0W1 sshd[65575]: Failed password for invalid user nagios …
Oct 13 21:12:21 S0W1 sshd[65575]: Accepted publickey for goetze …
Oct 13 21:12:22 S0W1 sshd[65575]: Port of Entry information retained for …
```

//HIVEIN DD DISP=SHR,DSN=COZUSER.HIVE.SCRIPTS(SYSLOG)

```
CREATE TABLE IF NOT EXISTS syslogdata (
 month STRING,
 day STRING,
 time STRING,
 host STRING,
 event STRING,
 msg STRING)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'
WITH SERDEPROPERTIES ("input.regex" =
 "(\\w+)\\s+(\\d+)\\s+(\\d+:\\d+:\\d+)\\s+(\\w+\\W*\\w*)\\s+(.*?\\:)\\s+(.*$)“
) STORED AS TEXTFILE LOCATION '/logs';
SELECT split(msg, ' ')[5] username, count(*) num
FROM syslogdata
WHERE msg LIKE 'Failed password for invalid user%'
GROUP BY split(msg, ' ')[5]
ORDER BY num desc,username;
```

```
 Failed password for invalid user root ...
 Failed password for invalid user nagios ...
 ...
```

Hive – Log Output

- By default, Hive writes its log to the **`stderr`** file descriptor on the target system
- Co:Z *automatically* redirects back to the job spool
- DD:STDERR

```
Time taken: 4.283 seconds
Total MapReduce jobs = 2
Launching Job 1 out of 2
…
Hadoop job information for Stage-1: number of mappers:
 1; number of reducers: 1
2014-04-24 08:33:55,847 Stage-1 map = 0%,  reduce = 0%
…
2014-04-24 08:36:49,447 Stage-2 map = 100%,  reduce =
 100%, Cumulative CPU 6.89 sec
```

Hive – Query Output

- By default, Hive writes its output to the **`stdout`** file descriptor on the target system
- Co:Z *automatically* redirects back to the job spool
- DD:STDOUT

```
root      68215
admin     1511
www       315
nagios    240
test      226
oracle    191
…
```

- Easily expands to process large numbers/types of log files incrementally stored in HDFS

Processing z/OS DB2 data with RHadoop

- z/OS DB2 High Performance Unload (HPU)
  - – Provides (among other things) rapid unload of table spaces
  - – Table space data can be accessed from target system with Co:Z
    - **cozclient** dataset pipes command
    - **inzutilb** HPU wrapper
    - Enable “data in flight” from z/OS DB2 to Big Data environments
- R and Hadoop have a natural affinity
- RHadoop developed by RevolutionAnalytics
  - – Apache 2 License
  - – Packages include rmr, rhdfs, rhbase

Processing z/OS DB2 data with RHadoop

- z/OS DB2 table DOVET.CLICKS contains information about each visitor to a website:
  - Timestamp
  - IP Address
  - URL
  - ID
  - City
  - Country
  - State
- We want to analyze this data using R to predict the likelihood of “next day” visits by country

Processing z/OS DB2 data with RHadoop

```
//CZUSERR JOB (),‘COZ',MSGCLASS=H,NOTIFY=&SYSUID,CLASS=A
//RUNCOZ EXEC PROC=COZPROC,ARGS=‘u@linux’
//COZCFG DD *
saf-cert=SSH-RING:RSA-CERT
ssh-tunnel=false
//STDIN DD *
hadoop fs -rmr   /user/rhadoop
hadoop fs -mkdir /user/rhadoop/in
hadoop fs -mkdir /user/rhadoop/out
fromdsn //DD:HPUIN | cozclient -ib inzutilb.sh 'DBAG,HPU' |
 hadoop fs -put - /user/rhadoop/in/clicks.csv
Rscript <(fromdsn DD:RSCRIPT)
hadoop fs -cat /user/rhadoop/out/* | todsn DD:RRESULT
/*
//RSCRIPT  DD DISP=SHR,DSN=COZUSER.RHADOOP(CLICKS)
//RRESULT  DD SYSOUT=*
//HPUIN    DD *
```

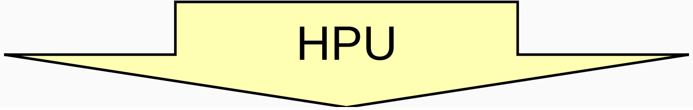
Dataset Pipes cozclient command and INZUTILB

- The **cozclient** command can be used by the target script to run a z/OS Unix System Services command
- Output is piped back the target script
- `fromdsn //DD:HPUIN | cozclient -ib inzutilb.sh 'DBAG,HPU'`
  - `cozclient` reads its input from stdin (piped from DD:/HPUIN)
  - `inzutilb.sh` is a wrapper for the DB2 HPU utility (INZUTILB)
    - Runs authorized on z/OS
    - Dynamically allocates HPU DDs
      - *SYSIN : stdin*
      - *SYSREC1 : stdout*
      - *SYSPRINT : stderr*

DB2 HPU

```
…
fromdsn //DD:HPUIN | cozclient -ib inzutilb.sh 'DBAG,HPU' |
 hadoop fs -put - /user/rhadoop/in/clicks.csv
…
//HPUIN    DD *
UNLOAD TABLESPACE
DB2 FORCE
LOCK NO
SELECT COUNTRY,TS,COUNT(*) FROM DOVET.CLICKS GROUP BY COUNTRY,TS
OUTDDN SYSREC1
FORMAT DELIMITED SEP ',' DELIM '"'
EBCDIC
```

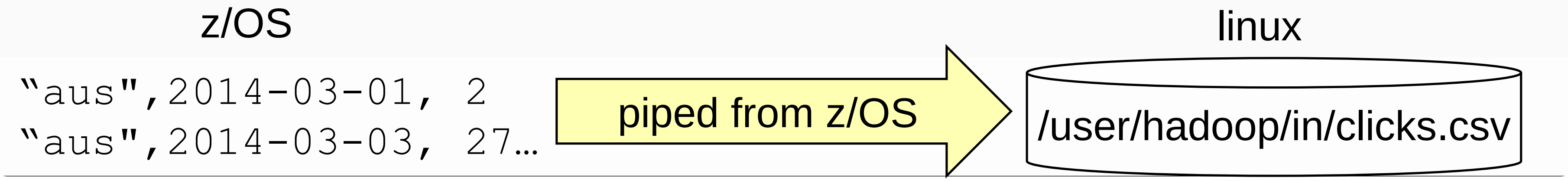
ts	ip	url	swid	city	country	state
2014…	99.122…	http://acme.com…	{7A…	homestead	usa	fl
2014…	203.19…	http://acme.com…	{6E…	perth	aus	wa
2014…	67.230…	http://acme.com…	{92…	guaynabo	pri	na



```
"aus",2014-03-01, 2
"aus",2014-03-03, 27…
```

DB2 HPU

```
…
fromdsn //DD:HPUIN | cozclient -ib inzutilb.sh 'DBAG,HPU' |
 hadoop fs -put - /user/rhadoop/in/clicks.csv
…
//HPUIN    DD *
UNLOAD TABLESPACE
DB2 FORCE
LOCK NO
SELECT COUNTRY,TS,COUNT(*) FROM DOVET.CLICKS GROUP BY COUNTRY,TS
OUTDDN SYSREC1
FORMAT DELIMITED SEP ',' DELIM '"'
EBCDIC
```



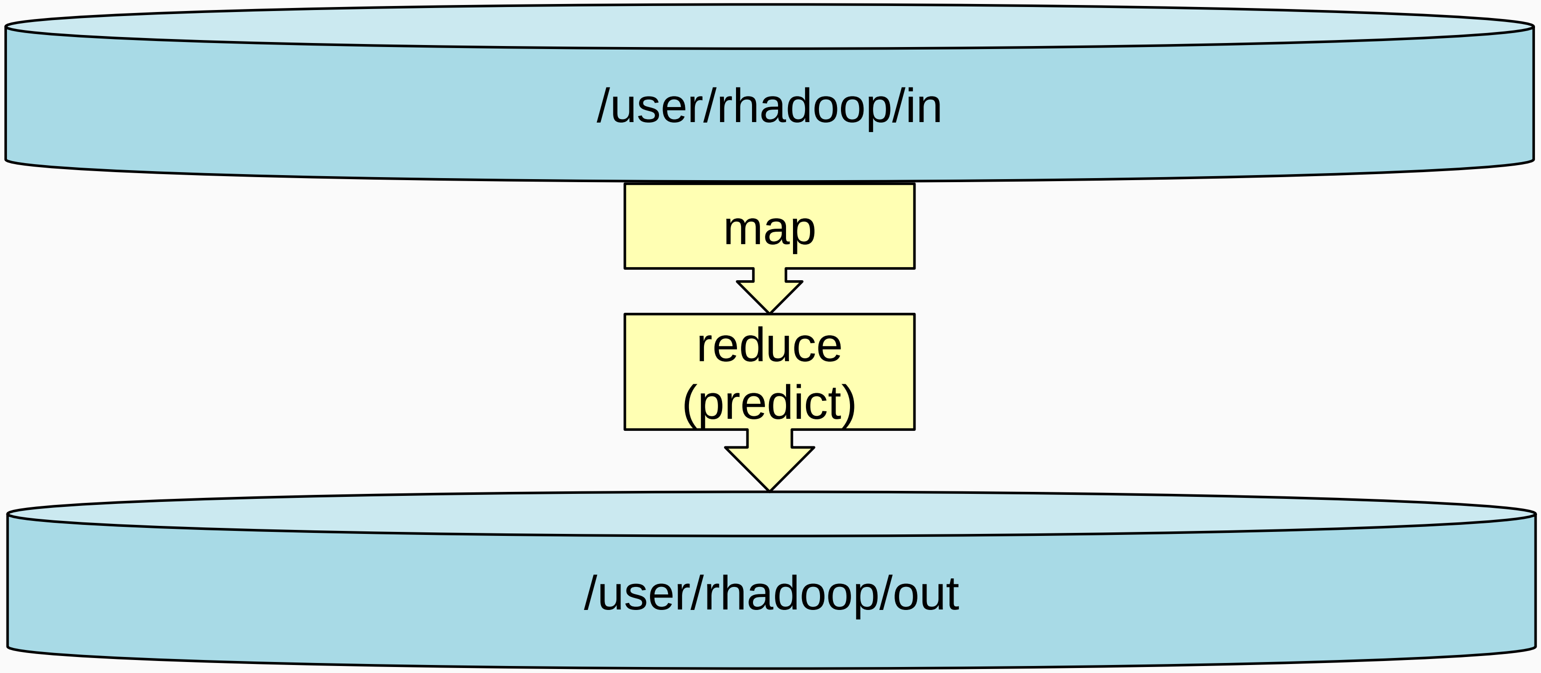
```
“aus",2014-03-01, 2
“aus",2014-03-03, 27…
```

DB2 HPU Status - DD:COZLOG

```
CoZLauncher[N]: version: 2.4.4 2014-03-18
cozagent[N]: version: 1.1.2 2013-03-19
fromdsn(DD:STDIN)[N]: 8 records/640 bytes read; 299 bytes written
fromdsn(DD:HPUIN)[N]: 7 records/560 bytes read; 172 bytes written
1INZU224I IBM DB2 HIGH PERFORMANCE UNLOAD V4.1
 INZU219I PTFLEVEL=PM98396-Z499
 INZI175I PROCESSING SYSIN AS EBCDIC.
        ----+----1----+----2----+----3----+----4----+----5----+---
 000001  UNLOAD TABLESPACE
 000002  DB2 FORCE
 000003  LOCK NO
 000004  SELECT  COUNTRY, TS, COUNT(*) FROM DOVETAIL.CLICKS GROUP BY
 COUNTRY, TS
 000005  OUTDDN SYSREC1
 000006  FORMAT DELIMITED SEP ',' DELIM '"'
 000007  EBCDIC
 INZI020I DB2 SUB SYSTEM          DBAG DATASHARING GROUP DBAG
          DB2 VERSION             1010 NFM
 ...
```

RHadoop

```
//CZUSERR JOB (),‘COZ',MSGCLASS=H,NOTIFY=&SYSUID,CLASS=A
//RUNCOZ EXEC PROC=COZPROC,ARGS=‘u@linux’
//STDIN DD *
…
Rscript <(fromdsn DD:RSCRIPT)
…
//RSCRIPT  DD DISP=SHR,DSN=COZUSER.RHADOOP(CLICKS)
```



DD:RSCRIPT - Mapper

```
#Modified from Hortonworks example

library(rmr2)

insertRow <- function(target.dataframe, new.day) {
  new.row <- c(new.day, 0)
  target.dataframe <- rbind(target.dataframe,new.row)
  target.dataframe <-
 target.dataframe[order(c(1:(nrow(target.dataframe)-1),
  new.day-0.5)),]
  row.names(target.dataframe) <- 1:nrow(target.dataframe)
  return(target.dataframe)
}

mapper = function(null, line) {
  keyval(line[[1]], paste(line[[1]],line[[2]],line[[3]],sep=","))
}
```

DD:RSCRIPT - Reducer

```
reducer = function(key, val.list) {
  if( length(val.list) < 10 ) return()
  list <- list()
  country <- unlist(strsplit(val.list[[1]], ","))[[1]]
  for(line in val.list) {
    l <- unlist(strsplit(line, split=","))
    x <- list(as.POSIXlt(as.Date(l[[2]]))$mday, l[[3]])
    list[[length(list)+1]] <- x
  }
  list <- lapply(list, as.numeric)
  frame <- do.call(rbind, list)
  colnames(frame) <- c("day","clicksCount")
  i = 1
  while(i < 16) {
    if(i <= nrow(frame)) curDay <- frame[i, "day"]
    if( curDay != i ) frame <- insertRow(frame, i)
    i <- i+1
  }
  model <- lm(clicksCount ~ day, data=as.data.frame(frame))
  p <- predict(model, data.frame(day=16))
  keyval(country, p)
 }
```

DD:RSCRIPT - mapreduce

```
mapreduce(
 input="/user/rhadoop/in",
  input.format=make.input.format("csv", sep = ","),
  output="/user/rhadoop/out",
  output.format="csv",
  map=mapper,
  reduce=reducer
)
```

DB2 Rhadoop Status - DD:STDERR

```
14/04/23 13:39:45 INFO mapreduce.Job:  map 100% reduce 100%
14/04/23 13:39:46 INFO mapreduce.Job: Job job_1397667423931_0064
 completed successfully
14/04/23 13:39:46 INFO mapreduce.Job: Counters: 44
 File System Counters
      FILE: Number of bytes read=17168
      ...
 Job Counters
      Launched map tasks=2
      ...
 Map-Reduce Framework
      Map input records=79
      ...
 Shuffle Errors
      BAD_ID=0
      ...
 rmr
      reduce calls=21
14/04/23 13:39:46 INFO streaming.StreamJob: Output directory:
 /user/rhadoop/out
```

Processing z/OS DB2 data with RHadoop

```
//CZUSERR JOB (),‘COZ',MSGCLASS=H,NOTIFY=&SYSUID,CLASS=A
//RUNCOZ EXEC PROC=COZPROC,ARGS=‘u@linux’
hadoop fs -rmr   /user/rhadoop
hadoop fs -mkdir /user/rhadoop/in
hadoop fs -mkdir /user/rhadoop/out
fromdsn //DD:HPUIN | cozclient -ib inzutilb.sh 'DBAG,HPU' |
 hadoop fs -put - /user/rhadoop/in/clicks.csv
Rscript <(fromdsn DD:RSCRIPT)
hadoop fs -cat /user/rhadoop/out/* | todsn DD:RRESULT
//RRESULT  DD SYSOUT=*
"usa" "36323.3142857143"
"pri" "170.956093189964"
```

Processing z/OS DB2 data with RHadoop

Hybrid Batch Principles revisited:

1. R analysis executed on a virtual server from a z/OS batch job step
2. Uses existing programs – **`Rscript,hadoop fs`**
3. Output is redirected to z/OS spool
4. DB2 HPU data easily accessed via **`cozclient`**
5. The script exit code is adopted as the z/OS job step CC

Big Data Opportunities:

- Incremental growth in Hadoop – zBX/PureData systems are relatively inexpensive
- All processing stays within the z/OS security envelope
- Facilitates R analysis of DB2 data over time
- Opens up new analysis insights without affecting production systems

Summary

- zEnterprise / z/OS / Linux
  - Provides hybrid computing environment
- Co:Z Launcher and Target System Toolkit
  - Provides framework for hybrid *batch* processing
- Co:Z Hybrid Batch enables BigData with z/OS
  - High speed data movement
  - SAF security dictates access to z/OS resources *and* can be used to control access to target (BigData) systems
  - z/OS retains operational control

Website: http://dovetail.com

Email: info@dovetail.com