Security Best Practice

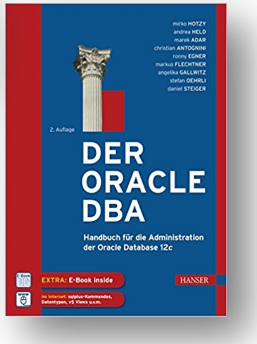
Oracle passwords, but secure!

Stefan Oehrli

Stefan Oehrli – Data Platforms

ORACLE ACE

Terraform ASSOCIATE HashiCorp CERTIFIED



stefan.oehrli@accenture.com

Tech Architecture Manager

- Since 1997 active in various IT areas
- More than 25 years of experience in Oracle databases
- Focus: Protecting data and operating databases securely
  - Security assessments and reviews
  - Database security concepts and their implementation
  - Oracle Backup & Recovery concepts and troubleshooting
  - Oracle Enterprise User and Advanced Security, DB Vault, …
  - Oracle Directory Services
- Co-author of the book The Oracle DBA (Hanser, 2016/07)

oradba.ch

@stefanoehrli

DATA PLATFORMS

WHY? We are the game changer for our client's data platform projects

HOW? Maximum automation, maximum efficiency, maximum quality!

WHAT? We build innovative data platforms based on our blueprints, assets and tools.

3 key benefits

1 Architecture expertise from hands-on projects

2 Delivery of tailor-made data platforms

3 Integrated Teams - Like a Rowing team, perfect alignment and interaction.

Tools and Blueprints

Key enabler for the implementation of modern data platforms at a high speed and quality.

Continuous Optimization

Tools and Blueprints are continuously optimized to the customer and project's needs.

Expertise

Expert group for modern data platforms from technical implementation to project management and organization

Agenda

Or how best to burn down time in your spare time…

1. Introduction
2. Oracle Password Hashes
3. Oracle Logon Process
4. Challenges
5. Password Complexity
6. Good Practice
7. Conclusion

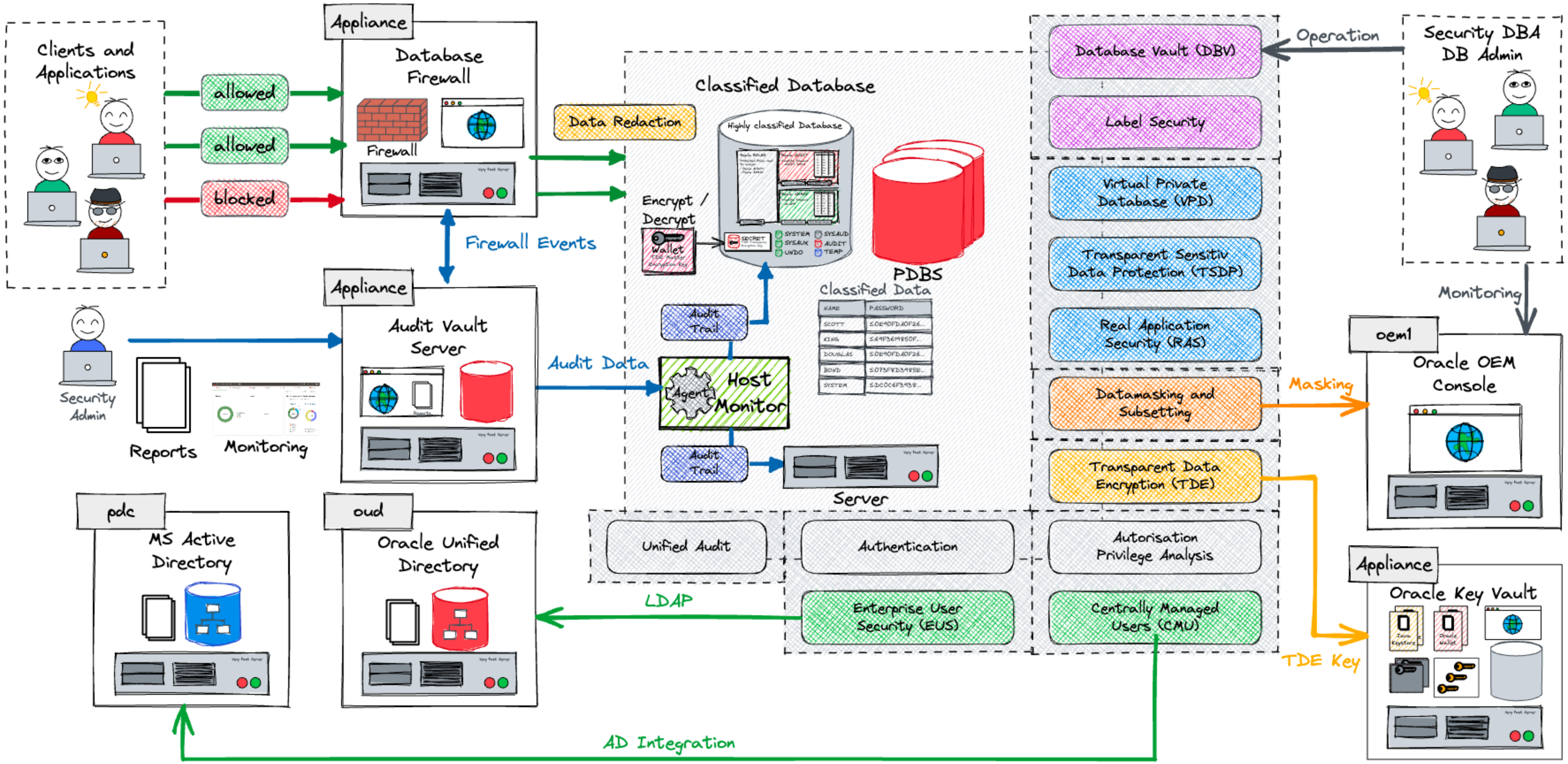
1

Introduction

Why did I start this topic in the first place?

Oracle Database Maximum Security Architecture

How much security do you need?



Password Security

But honestly, are passwords still an issue?

- Password based authentication is still one of the most used methods → Flexibility
- A large number of DB, Clients or Apps require legacy hashes / protocols → Compatibility
- Password Verification Functions do not keep pace with CPU evolvements → Standards
- The standards of the vendors are usually not the securest → Security Hardening
- Software, hashes and protocols reveal security flaws over time

Secure authentication is crucial, otherwise further security measures are questionable

Security is really important, especially as we are the world's number one Router maker
SECURITY FIRST
He's right!
We're going to protect router's settings behind a login/password
OK, but how do we make sure that people change the default password?
We'll put «admin/password» as the default, all the admins will panic and change it as soon as they set up!
Ah yeah, not bad!
Good idea!
SOME TIME LATER
What's the username and password for Router #1481?
admin/password
OK that's what I thought

CommitStrip.com

2

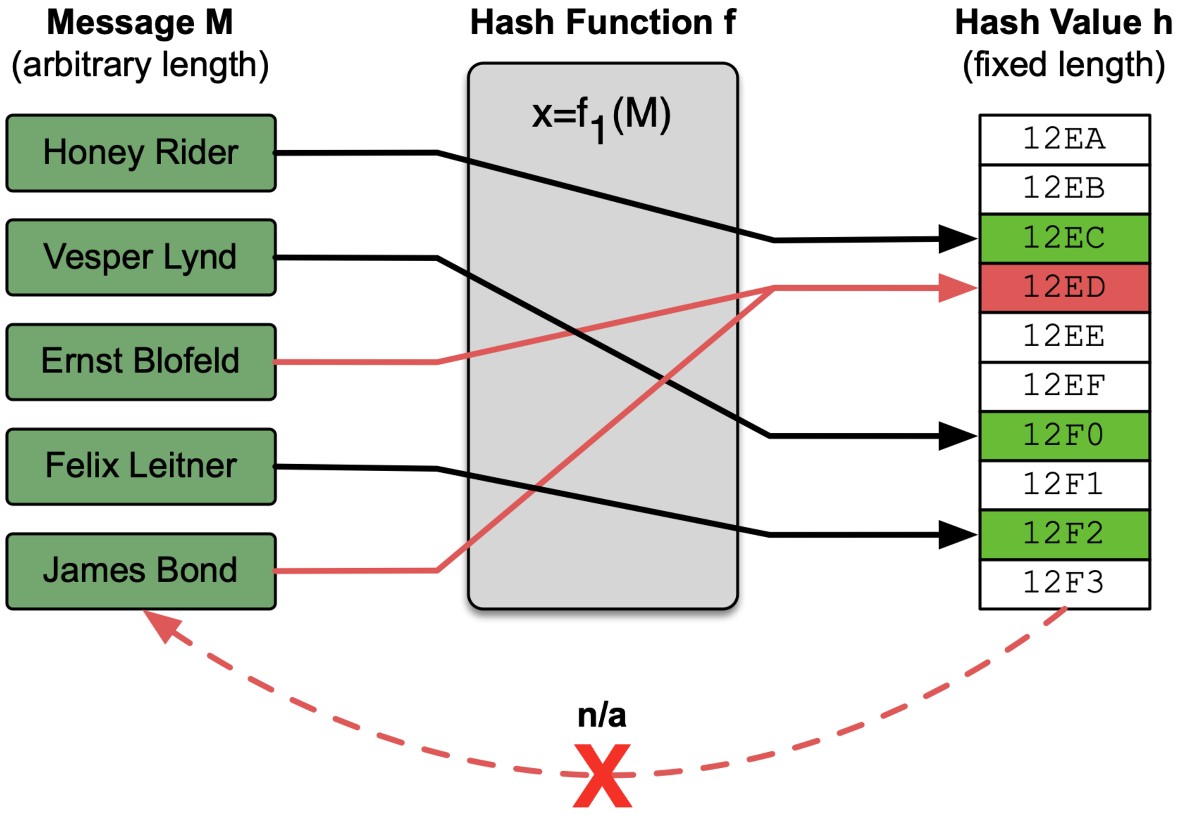
Oracle Password Hashes

Oracle database Authentication under the hood…

Password Hash Function

What is a Hash Function?

- Mathematical algorithm to map data of any size to a bit array of a fixed length
- It is deterministic
- Quick to compute hash for any given message
- One-way function
- Infeasible to generate a message that yields a given hash value
- Infeasible to find two different messages with the same hash value → Collision
- Known Cryptographic Hash Algorithms
  - MD5
  - SHA-1
  - SHA-2 i.e., SHA-256 and SHA-512



Oracle Password Hash Functions

The various algorithms in use for Oracle Database

- Oracle 10g Hash Function
  - Based on DES and an Oracle specific algorithm
  - Case insensitive and weak password Salt => Username
- MD5 based Hash Function
  - used for digest authentication in XDB
- Oracle 11g Hash Function
  - Based on the SHA1 hash algorithm
  - SHA1 is no longer considered safe (since 2005 see Wikipedia SHA-1)
  - Supports case sensitive and multibyte character passwords
- Oracle 12c Hash Function
  - based on a de-optimized algorithm involving PBKDF2 and SHA-512
  - Supports case sensitive and multibyte character passwords
- **Recommendation:** Only use Oracle 12c Hash Function

Oracle 10g Password Verifier

The legacy algorithm

- Passwords of local users are stored as 8-byte password hashes in base table SYS.USER$
- This algorithm has several weaknesses
  1. Weak password salt => user name

```
CREATE USER syste IDENTIFIED BY mmanager;

User created.

ALTER USER system IDENTIFIED BY manager;

User altered.


SELECT name, password FROM sys.user$ WHERE name LIKE 'SYSTE%';

USERNAME                       PASSWORD
------------------------------ ------------------------------
SYSTEM                         D4DF7931AB130E37
SYSTE                          D4DF7931AB130E37
```

Oracle 10g Password Verifier

The legacy algorithm

- This algorithm has several weaknesses
   2. Not case sensitive

```
ALTER USER system IDENTIFIED BY ManAger;

User altered.

SELECT name, password FROM sys.user$ WHERE name LIKE 'SYSTEM';

USERNAME                       PASSWORD
------------------------------ ------------------------------
SYSTEM                         D4DF7931AB130E37
```

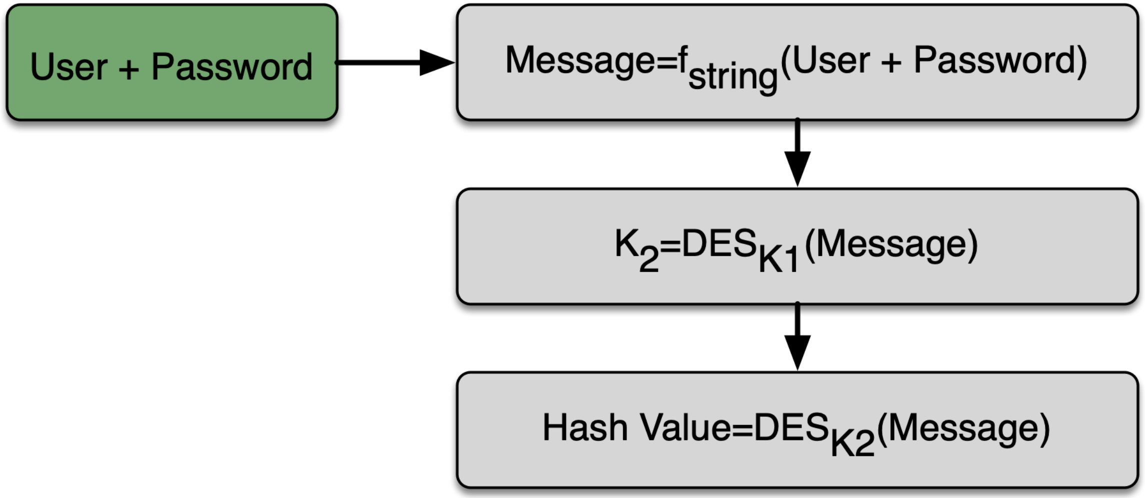
   3. Based on a legacy and proprietary hash function

Oracle 10g Password Algorithm

The legacy algorithm at a glance

Weak Hash Algorithm

1. Associate the user with the password to a clear text string
2. Convert clear text to upper case letters
3. Convert clear text to a Unicode string
4. Encryption of the clear text with DES CBC and a fixe key 0x0123456789ABCDEF If necessary the clear te 0 is padded to the next even block
5. Additional encryption of the clear text with DES CB Here the last block of step 4 is used as the key. The last block is then used as the hash value



User + Password

Message=f_{string}(User + Password)

K_2=DES_{K1}(Message)

Hash Value=DES_{K2}(Message)

Example Oracle 10g Password Algorithm

Simple example to show the vulnerabilities

```
Username : system
Password : manager

- STEP 1 ----------------------------------------------------------
Salted String     : systemmanager

- STEP 2 ----------------------------------------------------------
Upper String      : SYSTEMMANAGER

- STEP 3 ----------------------------------------------------------
Unicode String    : 00530059005300540045004D004D0041004E0041004700450052

- STEP 4 ----------------------------------------------------------
1st Key           : 0123456789ABCDEF
1st Hash value    : 643624EDC5FEA9B402B0B017E7CB7DB713108AC1914E984FE2EDDFE949A0C3C1

- STEP 5 ---------------------------------------------------------
2nd Key           : E2EDDFE949A0C3C1
2nd Hash Value    : A2295A85F9B413C2D2B25971D5199A0BA6C4C6035A4906B2D4DF7931AB130E37
Password Hash     : D4DF7931AB130E37
```

Oracle 11g Password Verifier

The newer password algorithm

- Based on SHA-1 and supports Case Sensitive and Multibyte Character Passwords
  - Actually everything that your character set offers
  - But special characters requires quotes e.g. " "
- Password hash is stored in column SPARE4 in base table SYS.USER$
  - Hash value does have the prefix S:

```
SELECT name, regexp_substr(spare4,'((S\:.+);|(S\:.+))',1,1,'i’,1) HASH
FROM user$ WHERE name='TEST';

NAME       HASH
---------- --------------------------------------------------------------
TEST       S:885B3ACB933CCBEF42DA4455BC4F1597E823F144A37F22B76F48F0CFFC52
```

- The hash function is a simple SHA-1 function

```
sys.user$spare4 = SHA1(pwd concat with salt) concat with salt
```

Example Oracle 11g Password Algorithm

Simple example to show the salt

```
ALTER USER test IDENTIFIED BY Welcome1;

SELECT name,
substr(regexp_substr(spare4,'((S\:.+);|(S\:.+));',1,1,'i',1), 1,40 ) HASH,
substr(regexp_substr(spare4,'((S\:.+);|(S\:.+));',1,1,'i',1), 41) SALT
FROM user$ WHERE name='TEST’;

NAME       HASH                                     SALT
---------- ---------------------------------------- --------------------
TEST       885B3ACB933CCBEF42DA4455BC4F1597E823F144 A37F22B76F48F0CFFC52

SELECT sys.dbms_crypto.hash(utl_raw.cast_to_raw('Welcome1')||
hextoraw('A37F22B76F48F0CFFC52'),3) HASH FROM dual;

HASH
----------------------------------------
885B3ACB933CCBEF42DA4455BC4F1597E823F144
```

Oracle 12c Password Verifier

The latest password algorithm

- Based on a de-optimized algorithm involving PBKDF2 and SHA-512
  - See Oracle® Database Security Guide 19c About the 12C Version of the Password Hash
- Supports Case Sensitive and Multibyte Character Passwords
- Password hash is stored in column SPARE4 in base table SYS.USER$
  - Hash value does have the prefix T:
- Oracle 12c Password Hash is supported by Client / Server Oracle Release 11.2.0.3

```
SELECT name, regexp_substr(spare4,'((T\:.+);|(T\:.+))',1,1,'i',1) HASH
FROM user$ WHERE name='TEST';

NAME  HASH
----- ----------------------------------
TEST  T:1902FCD14B0096A5F6E44E2C0B877479118791737 40A0FC8D8D346532731FE46A272123A0C53D79BDF26
        AB4FABAEEEF2964DEAE00B4626696C6CBE2ABEF753006B8D0E3DFA2CB0480115E8457AE954E6
```

Which Password Verifier is available

Verify which used does have which password verifier available

- Query PASSWORD_VERSIONS from DBA_USERS

```
SELECT username,password_versions FROM dba_users
WHERE username LIKE 'USER_%' ORDER BY 1;

USERNAME      PASSWORD_VERSIONS
------------------------- -----------------
USER_10G      10G
USER_11G      11G
USER_12C      12C
USER_ALL      10G 11G 12C
```

- Effective hash values stored in *USER$*
  - Oracle 10g Hash column PASSWORD
  - Oracle 11g Hash column SPARE4 Prefix S:
  - Oracle 12c Hash column SPARE4 Prefix T:

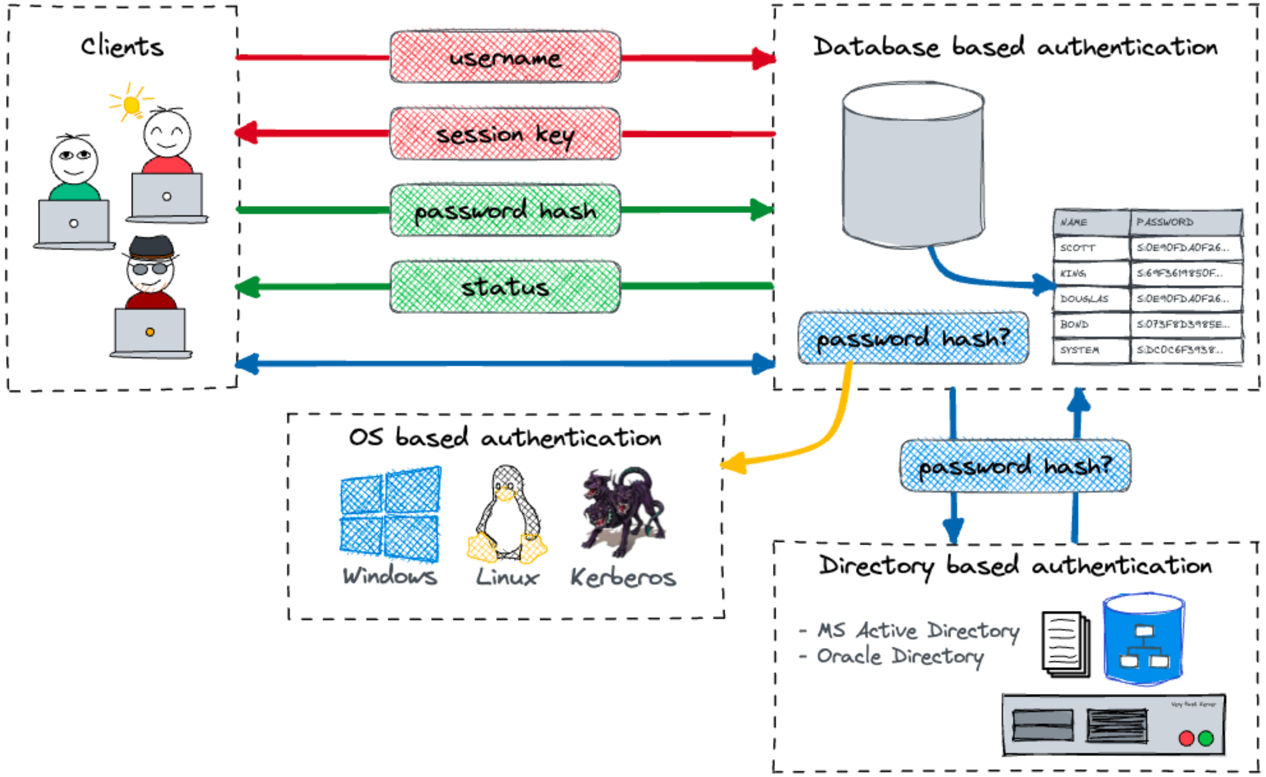
3

Oracle Logon Process

What happens during database login

Oracle Logon Process

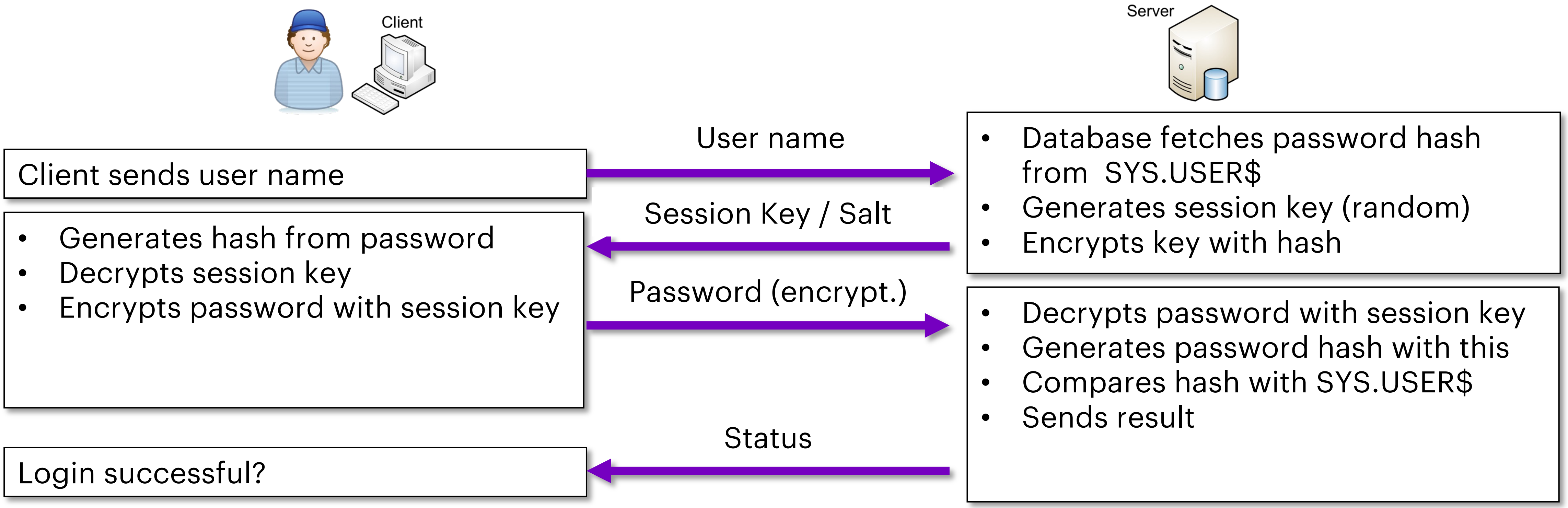
- Establish initial connection i.e. TNS name resolution, connection request to listener, etc.
- Negotiate session- and optional encryption keys
- Initiate authentication either ...
  - Password base for DB, CMU, EUS, Proxy or orapwd file authentication
  - External / OS based for OS, Kerberos, Radius, SSL or admin privileges e.g. SYSDBA
- Password based authentication is always done on the DB i.e. password hashes have to be available to the database
  - *SYS.USER$* or orapwd file
  - EUS/CMU relevant LDAP attributes e.g. *userPassword, orclCommonAttribute*



NAME	PASSWORD
SCOTT	S:0E90FDA0F26...
KING	S:69F3619850F...
DOUGLAS	S:0E90FDA0F26...
BOND	S:073F8D3985E...
SYSTEM	S:DC0C6F3938...

Oracle Login Process O3Logon/O5Logon

How Oracle Negotiates the Password Verifier



Client

Server

Client sends user name

User name

- Database fetches password hash from SYS.USER$
- Generates session key (random)
- Encrypts key with hash

Session Key / Salt

- Generates hash from password
- Decrypts session key
- Encrypts password with session key

Password (encrypt.)

- Decrypts password with session key
- Generates password hash with this
- Compares hash with SYS.USER$
- Sends result

Status

Login successful?

Authentication Protocol

How Oracle Negotiates the Password Verifier

- Login protocol is defined by the **sqlnet.ora** configuration
  - SQLNET.ALLOWED_LOGON_VERSION_SERVER (default 12)
  - SQLNET.ALLOWED_LOGON_VERSION_CLIENT (default 11)
- Here "version" refers to the version of the login protocol, not the database version
- Appropriate password versions / hashes must be available
  - See DBA_USERS.PASSWORD_VERSIONS
- Default value of ALLOWED_LOGON_VERSION_SERVER
  - Up to Oracle 12.1.0.2 => **8** all hashes are created
  - From Oracle 12.2.0.1 => **12** only 11c and 12c hashes are created
- **Recommended** setting for ALLOWED_LOGON_VERSION_SERVER is 12a
  - Only the 12c Password Verifier is used

Overview Authentication Protocol

Authentication Registration protocols version and the limitations / capabilities

ALV	Password Version	Clientability	Meaning
12a	12c	O7L_MR	Only Oracle 12.1.0x clients
12	11g, 12c	O7L_NP	Only clients with CPUOct 2012
11	10g, 11g, 12c	O5L	Oracle 10g and later, DBs older than 11.2.0.3 or without CPUOct 2012 must use 10g passwords
10	10g, 11g, 12c	O5L	
9	10g, 11g, 12c	O4L	Oracle 9i and newer
8	10g, 11g, 12c	O3L	Oracle 8i and older

4

Challenges

What challenges may arise

Protocol and Password Hashes

What can get wrong...

- Corresponding password versions / hashes must be available
  - See DBA_USERS.PASSWORD_VERSIONS
- If the version is not greater/equal, the connection is terminated
  - `ORA-28040: No matching authentication protocol`
- If the corresponding hash is missing, the connection is terminated
  - `ORA-01017: invalid username/password; logon denied`
- By setting/deleting the corresponding hashes, you can indirectly control which logon protocol is used

```
SQL> ALTER USER scott IDENTIFIED BY values
'S:22D8239017006EBDE054108BF367F225B5E731D12C91A3BEB31FA28D4A38';
```

Weaknesses in the password system

What can get wrong...

- Password hashes are all over the place
  - Not everywhere, but in enough places
  - Miscellaneous base tables in the data dictionary
  - **orapwd** file used for remote login as administrative user
- If the hashes are known, dictionary, rule or brute force based attacks are possible
- Limitation and vulnerabilities of password hash functions
  - E.g. known hash collisions
- Character restriction (no upper/lower case up to and including Oracle 10g, in principle no special characters allowed)
  - Partial compatibility problems with different tools

Risks of the Oracle Login Process

General assessment of Oracle passwords

- Is the login process secure?
- User name passes through the network unencrypted
- But no password, no password hash
- Password is automatically encrypted between client and server via AES
- If password hash known, session key could be decrypted
- Vulnerability found for password verifier using SHA-1 in October 2012
  - Security vulnerability in login process CVE-2012-3137
  - Clients and servers need to be patched and password reset
  - Information in MOS Note 1492721.1 and 1493990.1
  - **Hint:** Every Client which is not patched or using legacy logon process is still affected from this vulnerability

Configuration – ORA-01017 or ORA-28040

When all goes south...

- False Configurations can lead to issues, mostly to ORA-01017 or ORA-28040
  - E.g. set SEC_CASE_SENSITIVE_LOGON=FALSE and ALLOWED_LOGON_VERSION_SERVER>=12
- Database Migrations using expdp/impdp import users as they are
  - Can lead to wrong / missing password verifiers
  - Source DB has only 10g hashes but target requires 11g or 12c password verifiers
  - MOS Note 2289453.1 ORA-39384 Warning: User <USERNAME> Has been locked …
  - Post by Mike Dietrich What happens to PASSWORD_VERSIONS during an upgrade to Oracle 12.2?

Configuration – ORA-01017 or ORA-28040

When all goes south...

- Applications limiting password character pool
  - Some applications cannot handle certain special characters, umlauts etc.
  - $ " @ # can be challenging to escape properly
- Client Libraries (OCI, JDBC,...) not coping with new hash algorithms
  - Legacy issue from Oracle 10g to 11g transition
  - Client occasionally simply converted the password to uppercase

5

Password Complexity

What happens during database login

Password Profiles

Oracle Feature to control complexity

- Since Oracle8 it is possible to create password profiles and assign them to users
- Password profiles define the criteria for passwords
  - **complexity** with a password check function
  - Number of **incorrect logins**, number, lock and grace time
  - **Validity period** of passwords
  - **Password history**
- Oracle provides a script **utlpwdmg.sql** to configure password profiles and functions
  - The script is updated with every Oracle release
  - The script is not executed depending on the Release / Create method
  - It includes profiles based on CIS and Database STIG recommendations
- Password verification function can be created using Oracle functions:
  - **ora_string_distance** Calculation of the difference between two strings according to the Levenshtein distance
  - **ora_complexity_check** Checking the password complexity of a string

Good idea to specify complexity rules?

The downside of password complexity rules

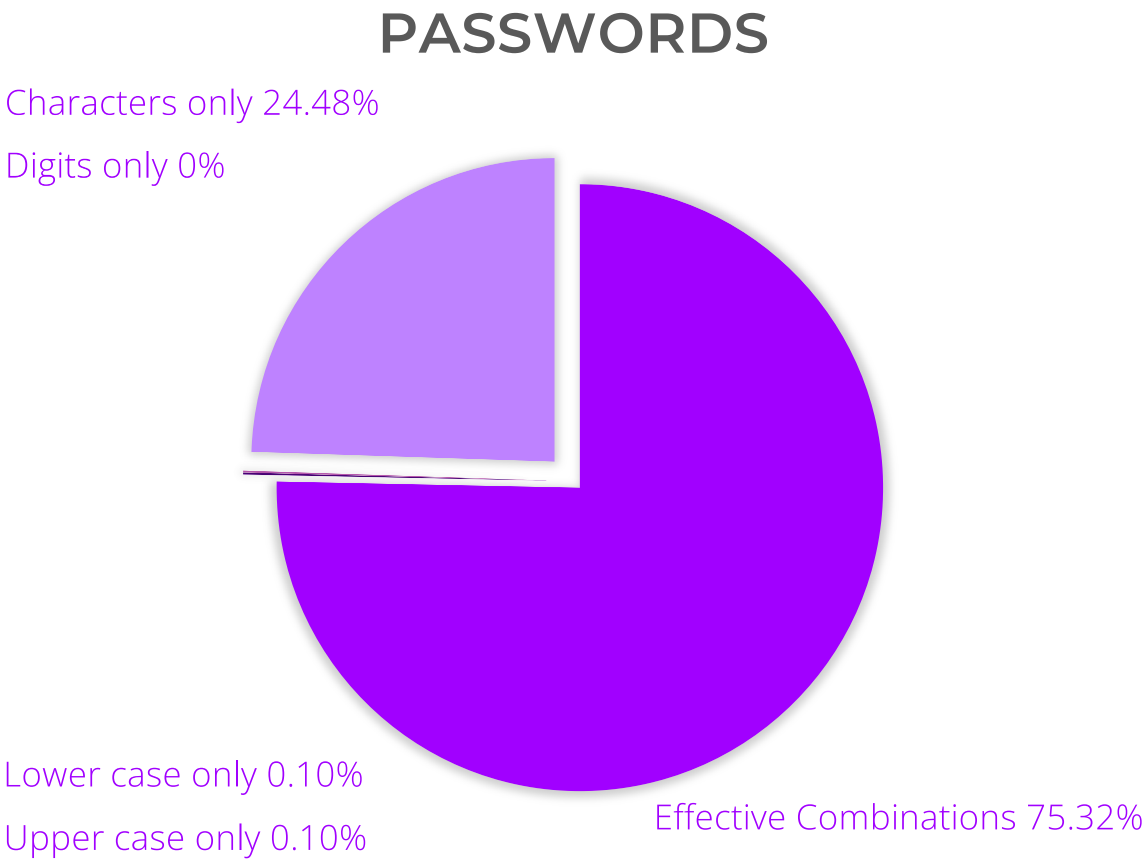
Example Password Rule

- Password with digits, upper and lower case letters
- 8-character password length
- At least 1 capital letter
- At least 1 lower case letter
- At least 1 digit

The Problem

- Number of characters 26+26+10=62
- Combinations for 8-character password 62^8
- Minus the special cases:
  - Digits only 10^8
  - Letters only 52^8
  - Upper and lower case only $26^8 + 26^8$

About a quarter less combinations!

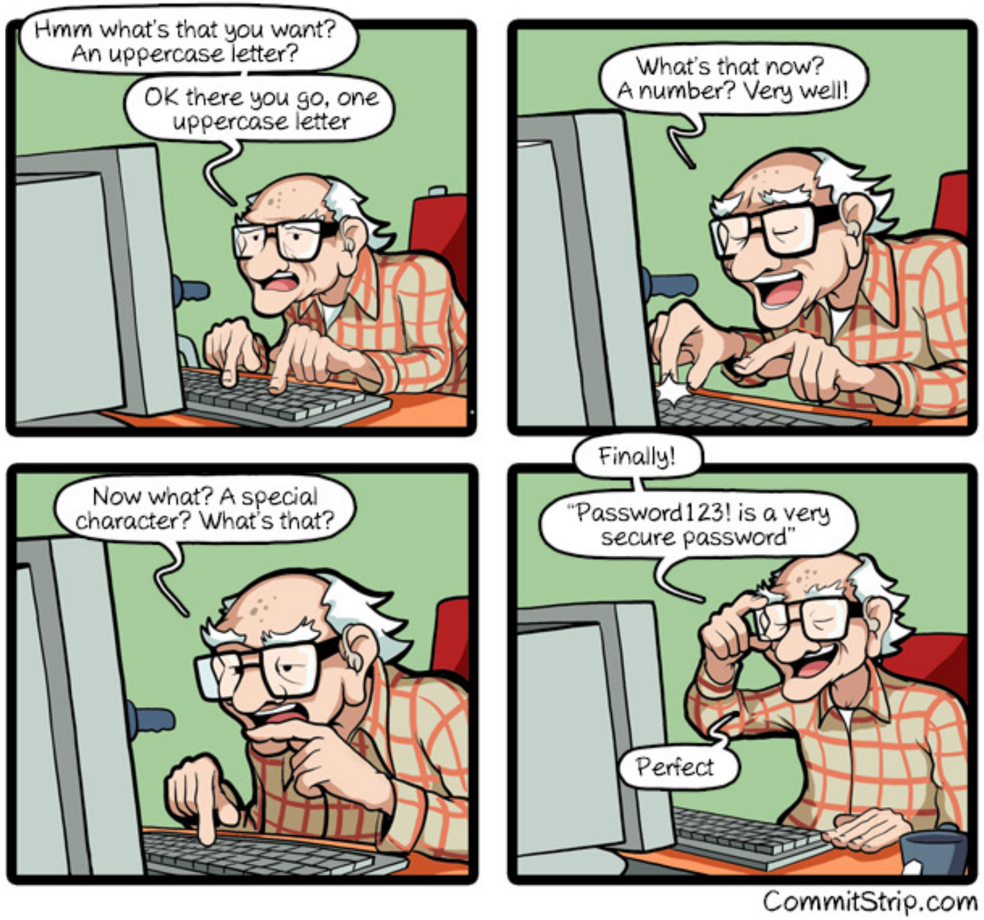


But what are good Passwords?

Or what are definitely bad passwords…

Not easy to answer anyway, if there is an answer at least.
A few principles and good practices:

- Passwords must be easy to “remember” either by you or your password manager
- Pool of unique characters should be as large as possible … and feasible ☺
- Maximum manageable length should be selected
  - The longer, the better ☺
- Password should not be based on common words, names or know passwords i.e. password dictionary
- Do not follow any obvious rules
- Password should have **high entropy**


Hmm what's that you want? An uppercase letter?
OK there you go, one uppercase letter
What's that now? A number? Very well!
Finally!
Now what? A special character? What's that?
"Password123! is a very secure password"
Perfect
CommitStrip.com

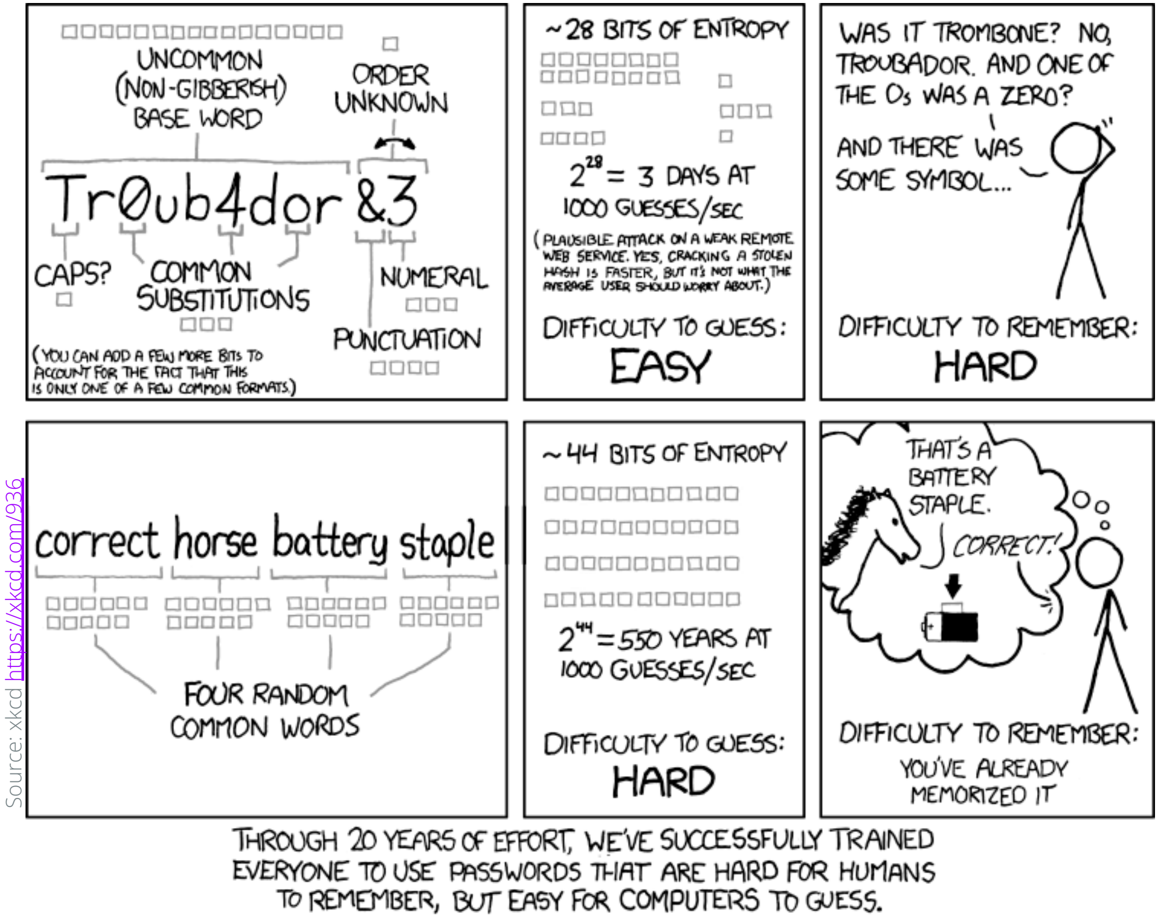
Password Entropy

A bit of math…

- Entropy is a measurement of how unpredictable a password $E = Log_2(R^L)$
  - R^L = number of possible passwords
  - E = password entropy in bits
  - R = pool of unique character
  - L = number of character i.e. password length
- Entropy for the example before $E = Log_2(62^8) = 47.6$ bits
- Today's GPU can calculate several million hashes per second
  - MacBook Pro 2020 400MH/s for Oracle 10g
  - 36 - 59 bits **used to** be reasonable secure
- Safe Password? It depends…
  - … on how the password is generated (random is not always that random)
  - … on a possible attack method e.g. *Welcome1* meets the password rule

Example Strong Passwords

Simple tricks to get complex passwords



UNCOMMON (NON-GIBBERISH) BASE WORD
ORDER UNKNOWN
Tr0ub4dor &3
CAPS?
COMMON SUBSTITUTIONS
NUMERAL
PUNCTUATION
(YOU CAN ADD A FEW MORE BITS TO ACCOUNT FOR THE FACT THAT THIS IS ONLY ONE OF A FEW COMMON FORMATS.)
~28 BITS OF ENTROPY
2^28 = 3 DAYS AT 1000 GUESSES/SEC
(PLAUSIBLE ATTACK ON A WEAK REMOTE WEB SERVICE. YES, CRACKING A STOLEN HASH IS FASTER, BUT IT'S NOT WHAT THE AVERAGE USER SHOULD WORRY ABOUT.)
DIFFICULTY TO GUESS: EASY
WAS IT TROMBONE? NO, TROUBADOR. AND ONE OF THE 0s WAS A ZERO?
AND THERE WAS SOME SYMBOL...
DIFFICULTY TO REMEMBER: HARD
correct horse battery staple
FOUR RANDOM COMMON WORDS
~44 BITS OF ENTROPY
2^44 = 550 YEARS AT 1000 GUESSES/SEC
DIFFICULTY TO GUESS: HARD
THAT'S A BATTERY STAPLE.
CORRECT!
DIFFICULTY TO REMEMBER: YOU'VE ALREADY MEMORIZED IT
THROUGH 20 YEARS OF EFFORT, WE'VE SUCCESSFULLY TRAINED EVERYONE TO USE PASSWORDS THAT ARE HARD FOR HUMANS TO REMEMBER, BUT EASY FOR COMPUTERS TO GUESS.

Source: xkcd https://xkcd.com/936

Check the Passwords!

Where are still default passwords in use…

- The view DBA_USERS_WITH_DEFPWD can be used to easily check whether the default passwords of users created by Oracle have been changed

```
SELECT username FROM dba_users_with_defpwd;

USERNAME
----------
CTXSYS
SCOTT
```

- Alternative checking of the known hash with appropriate tools
  – DBMS_CRYPTO to calculate the hash manually
  – Password Crack Tools like *Hashcat, John the Ripper* and others

Password Verification Using Tools

But be careful when and where to use…

- Tools *Hashcat* and *John the Ripper* do support a wide range of known password hashes
  - Including all hash functions used by Oracle e.g. 10g, 11g, 12c
- GPU power is a crucial factor when calculating hash values
  - Tools do use CPU and GPU to calculate hashes where GPU
  - Whereby GPU are faster by factors
- Different attack methods are possible:
  - **Dictionary based** – testing passwords from wordlist e.g. 5-10 Mio
  - **Rule based** – Extend wordlist by rules e.g. flip chars, add numbers etc.
  - **Brute force** – Calculate every combination out of a character pool
- The tools are basically free and publicly available
  - Relatively well documented and No darknet experience required ☺
- The use might be illegal depending on country and region
  - Depends on the purpose of use

What is possible

Hashcat on an Intel based MacBook Pro 2018

- Simple Hashcat benchmark for the Oracle 7+ hashes i.e. 10g password verifier

```
hashcat --benchmark --hash-type 3100 -D 1,2,3
hashcat (v6.1.1) starting in benchmark mode...

OpenCL API (OpenCL 1.2 (Oct 29 2020 19:50:08)) - Platform #1 [Apple]
=====================================================================
* Device #1: Intel(R) Core(TM) i9-8950HK CPU @ 2.90GHz, 32704/32768 MB
* Device #2: Intel(R) UHD Graphics 630, 1472/1536 MB (384 MB allocatable), 24MCU
* Device #3: AMD Radeon Pro 560X Compute Engine, 4032/4096 MB (1024 MB allocatable), 16MCU

Hashmode: 3100 - Oracle H: Type (Oracle 7+)

Speed.#1.........: 11719.5 kH/s (66.85ms) @ Accel:128 Loops:512 Thr:1 Vec:4
Speed.#2.........:  4423.3 kH/s (85.02ms) @ Accel:128 Loops:16 Thr:8 Vec:1
Speed.#3.........:   117.8 MH/s (67.33ms) @ Accel:128 Loops:64 Thr:64 Vec:1
Speed.#*.........:   133.9 MH/s
```

What is possible

Hashcat on the latest Intel based MacBook Pro 2020

- Simple Hashcat benchmark for the Oracle 7+ hashes i.e. 10g password verifier

```
hashcat --benchmark --hash-type 3100 -D 1,2,3
hashcat (v6.1.1) starting in benchmark mode...

OpenCL API (OpenCL 1.2 (Jun  8 2020 17:36:15)) - Platform #1 [Apple]
====================================================================
* Device #1: Intel(R) Core(TM) i9-9980HK CPU @ 2.40GHz, 65472/65536 MB
* Device #2: Intel(R) UHD Graphics 630, 1472/1536 MB (384 MB allocatable), 24MCU
* Device #3: AMD Radeon Pro 5500M Compute Engine, 8112/8176 MB (2044 MB allocatable), 24MCU

Hashmode: 3100 - Oracle H: Type (Oracle 7+)

Speed.#1.........:  8891.4 kH/s (58.73ms) @ Accel:32 Loops:1024 Thr:1 Vec:4
Speed.#2.........:  4653.3 kH/s (78.22ms) @ Accel:4 Loops:512 Thr:8 Vec:1
Speed.#3.........:   400.4 MH/s (61.61ms) @ Accel:256 Loops:64 Thr:64 Vec:1
Speed.#*.........:   414.0 MH/s
```

What is generally possible?

A supercomputer is not necessarily required

Performance for other hash values differs

Hash Type	MB Pro 2018	MB Pro 2020	Nvidia GTX 1080 TI
MD5	4’921.4 MH/s	11’240.0 MH/s	31’103.4 MH/s
SHA-1	1’783.2 MH/s	4’296.9 MH/s	11’374.1 MH/s
Oracle 7+	133.9 MH/s	414.0 MH/s	1’320.0 MH/s
Oracle 11+	1’766.6 MH/s	4’283.2 MH/s	11’222.5 MH/s
Oracle 12+	4390 H/s	3698 H/s	150.2 kH/s

Power of my MacBook pro not enough?

- No need to rent a Cray-2
- Just buy a decent graphic card or two i.e., for game not office usage ☺
- Set up a compute instance in a cloud
  - All cloud vendors have options for GPU support

6

Good Practice

What happens during database login

Good Practice

Things that should be considered…

Keep your Oracle Clients **and** Server up to date

- Stay updated by following Critical Patch Updates, Security Alerts and Bulletins
- Install security fixes in a **reasonable** time frame

Consider using strong Authentication

- Kerberos and SSL based Authentication

Don't use legacy password verifier

- Use Oracle password file version 12.2
- Explicitly configure ALLOWED_LOGON_VERSION_SERVER to 12a and exclusively use 12c hash values
- Start using **PBKDF2 SHA-512** for directory-based password authentication with EUS and CMU
- **Art. 32 GDPR Security of processing**
  MD5, SHA-1 and Oracle 10g password verifiers are definitely not state of the art any more

Good Practice

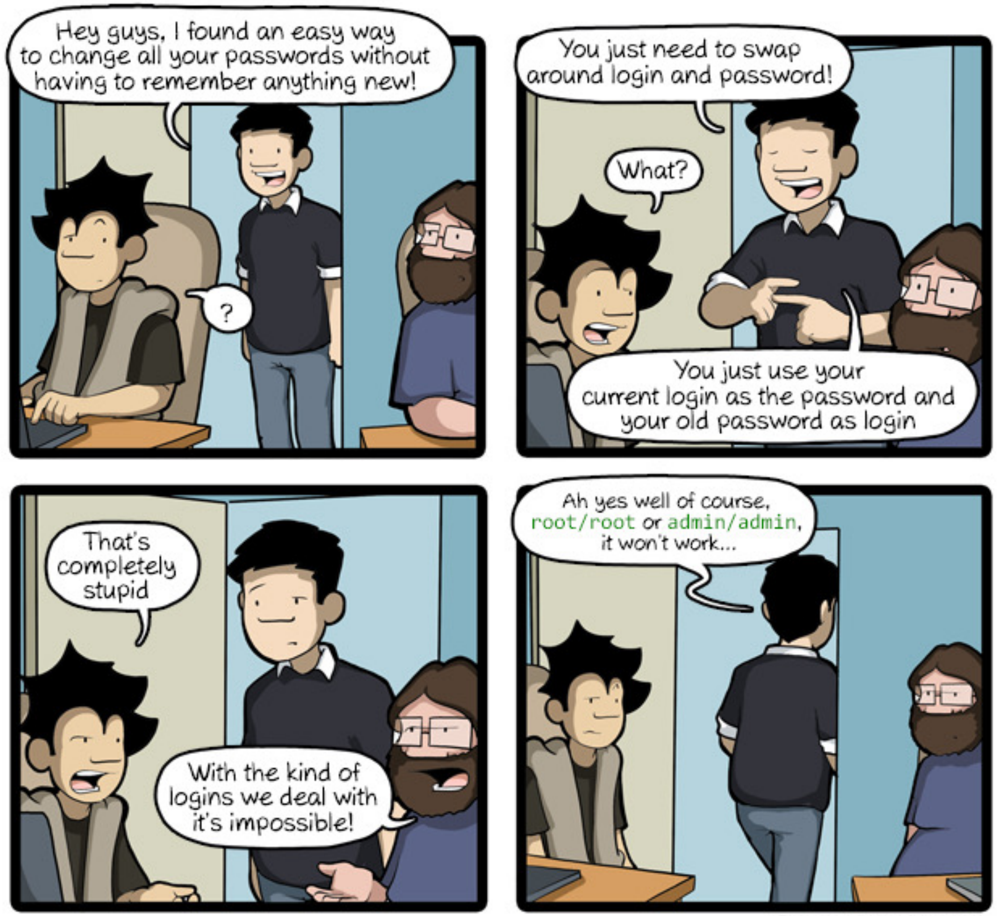
What about internal standards and trainings?

Revise your password policies

- NIST, CIS, STIG and other standards are continuously adjusted
- Does the complexity rule still make sense or does it just reduce the amount of possibilities

User awareness training

- Make sure your user know the principle of good and bad
- Use of phase phrase rather than password


Hey guys, I found an easy way to change all your passwords without having to remember anything new!
?
You just need to swap around login and password!
What?
You just use your current login as the password and your old password as login
That's completely stupid
With the kind of logins we deal with it's impossible!
Ah yes well of course, root/root or admin/admin, it won't work...

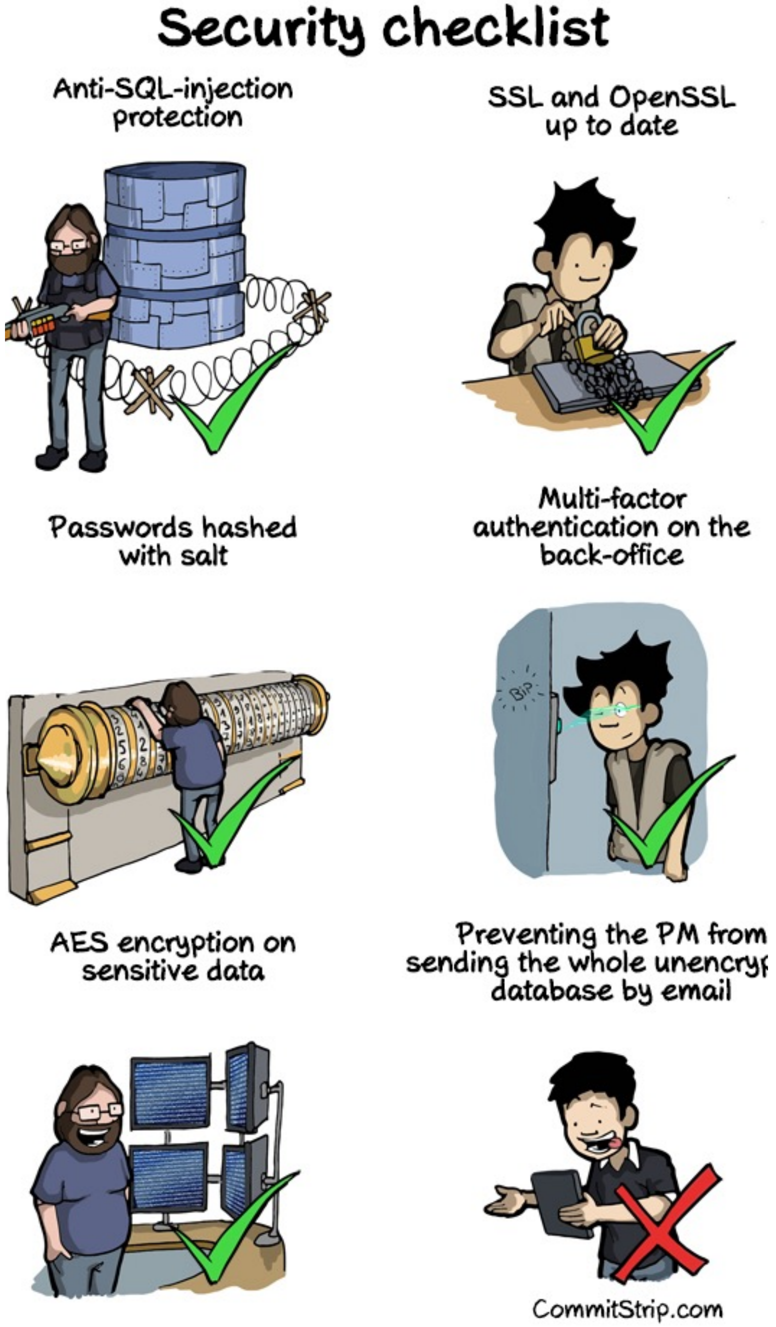
CommitStrip.com

Good Practice

Keep a low profile….

Reduce the attack vector

- Limit access to password hash values
  - e.g., password files, SYS.USER$ and other base tables
- Know where you have password hash values
  - e.g., in application tables
- Implement general database hardening
  - Oracle Database Lockdown
  - Oracle® Database Security Guide 19c
  - CIS Oracle Database Benchmark 19c
  - DoD Oracle Database 12c STIG - Ver 1, Rel 18
- Once again training of security awareness…


Security checklist
Anti-SQL-injection protection
SSL and OpenSSL up to date
Passwords hashed with salt
Multi-factor authentication on the back-office
BIP
AES encryption on sensitive data
Preventing the PM from sending the whole unencryp database by email
CommitStrip.com

7

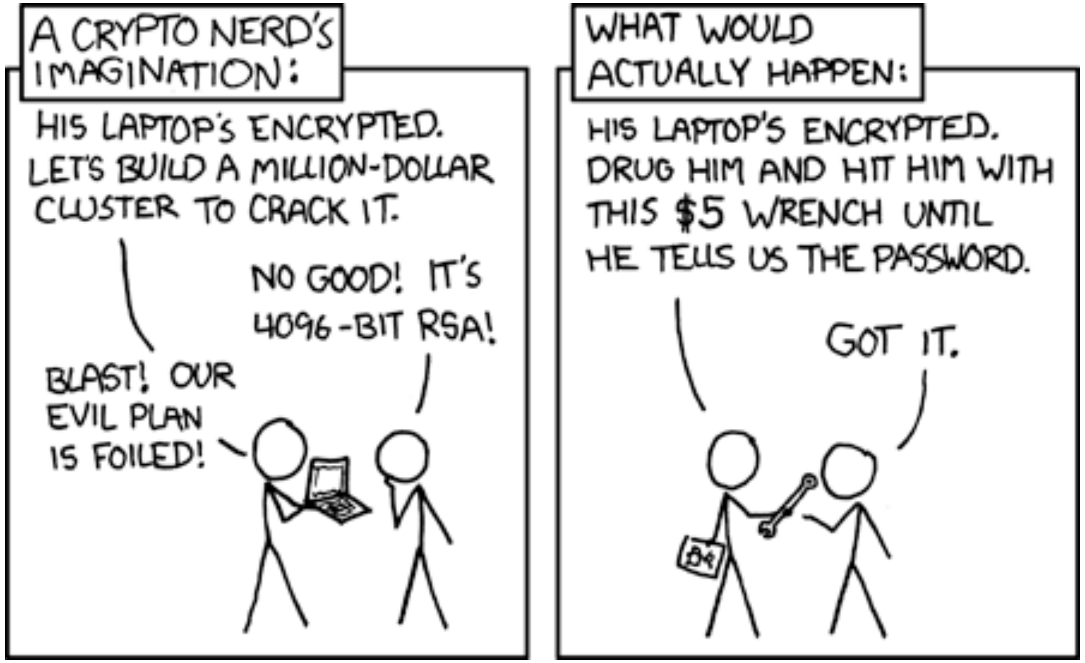
Conclusion

Now, what about the performance of security features?

Conclusion

Is there a performance formular for security features / options?

- There is no absolute security nor secure passwords
  - Computing power evolves
- Revise your password rule
- Keep software up to date
  - That means server **and** clients
- **Don't use** legacy configuration
  - 10g/11g hashes
  - SEC_CASE_SENSITIVE_LOGON
- Consider using strong authentication
  - Kerberos or SSL


A CRYPTO NERD'S IMAGINATION:
HIS LAPTOP'S ENCRYPTED. LET'S BUILD A MILLION-DOLLAR CLUSTER TO CRACK IT.
NO GOOD! IT'S 4096-BIT RSA!
BLAST! OUR EVIL PLAN IS FOILED!
WHAT WOULD ACTUALLY HAPPEN:
HIS LAPTOP'S ENCRYPTED. DRUG HIM AND HIT HIM WITH THIS $5 WRENCH UNTIL HE TELLS US THE PASSWORD.
GOT IT.
$4

Source: xkcd https://xkcd.com/538

The best algorithm is only as good as the chosen password...

Thank You