Andreas Wiegenstein

IT Defense, January 30th – February 1st, Berlin

VIRTUALFORGE
we harden your software

Detecting Data Leaks in SAP - The Next Level of Static Code Analysis

My car, my house, my boat, …

VIRTUALFORGE
we harden your software

Andreas Wiegenstein

- Founder of Virtual Forge (Heidelberg), responsible for Research & Development
- SAP Security Researcher, active since 2003
  - Received Credits from SAP for more than 20 reported 0-day Vulnerabilities
- Speaker at international Conferences
  - **SAP TechEd** 2004 (USA & Europe) / 2005 (USA) / 2006 (USA), DSAG 2009
  - **BlackHat** 2011 (Europe), **Hack in the Box** 2011 (Europe)
  - **Troopers** 2011, 2012, 2013, **RSA** 2012 (USA)
- Co-Author of „Sichere ABAP Programmierung" (SAP Press)
- Training Class WDESA3 @ SAP University

CONTENTS

VIRTUALFORGE
we harden your software

What is a Data Leak?

…inspiring to some

http://dataleak.org/

…embarrassing to others

VIRTUALFORGE
we harden your software

"Dude...you have data leakage."

http://www.adexchanger.com/wp-content/uploads/2010/08/data-leakage-gag.jpg

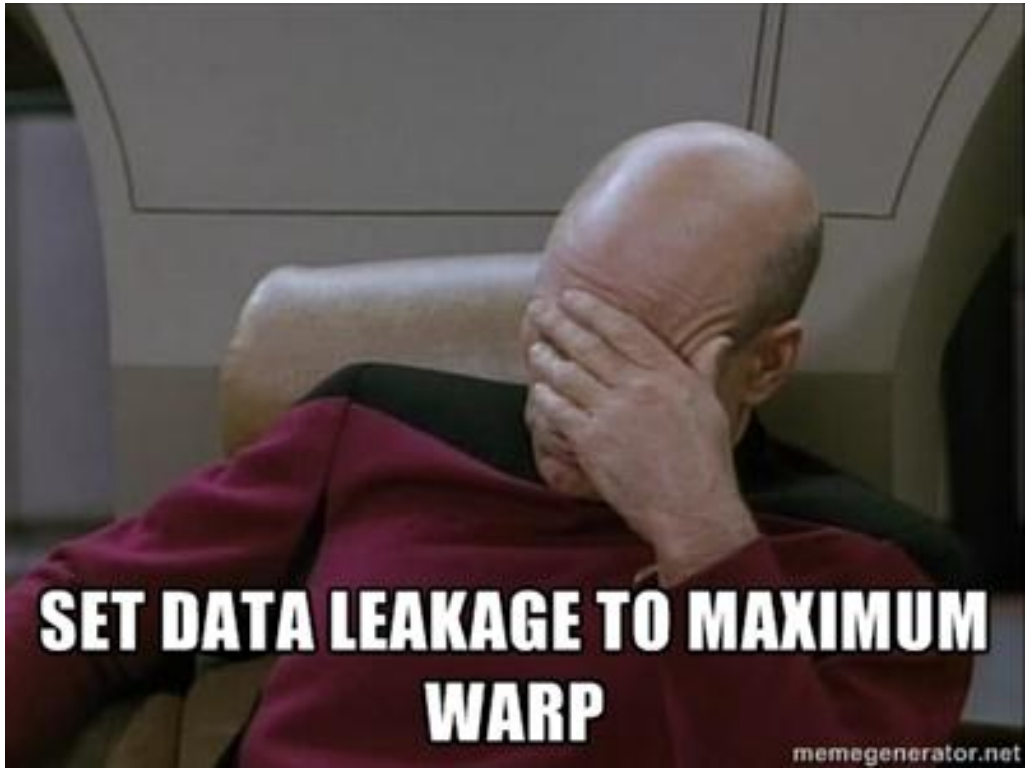
…worse than cat herding

VIRTUALFORGE
we harden your software

The daydreams of cat herders...

http://simonhunt.files.wordpress.com/2010/04/cats3.jpg?w=595

and troublesome, even in the future



http://cdn.memegenerator.net/instances/400x/29782239.jpg

Wikipedia on Data Leaks

VIRTUALFORGE
we harden your software

"... In data leakage incidents, sensitive data is disclosed to unauthorized personnel either by malicious **intent** or inadvertent **mistake**."

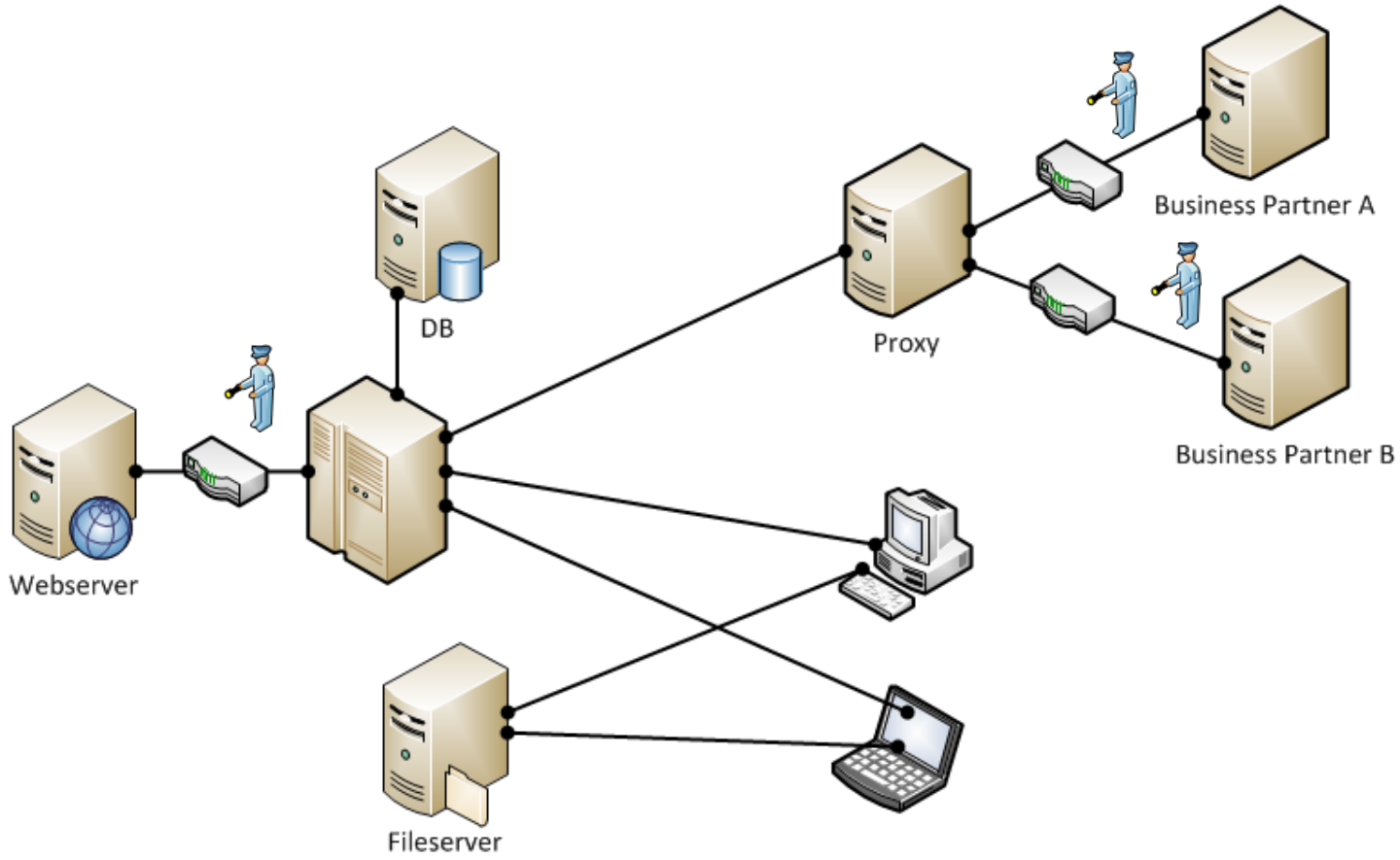
(December 2012)

http://en.wikipedia.org/wiki/Data_loss_prevention_software

Conventional Data Leak Prevention (DLP)

(1) Network: “Data in Motion”

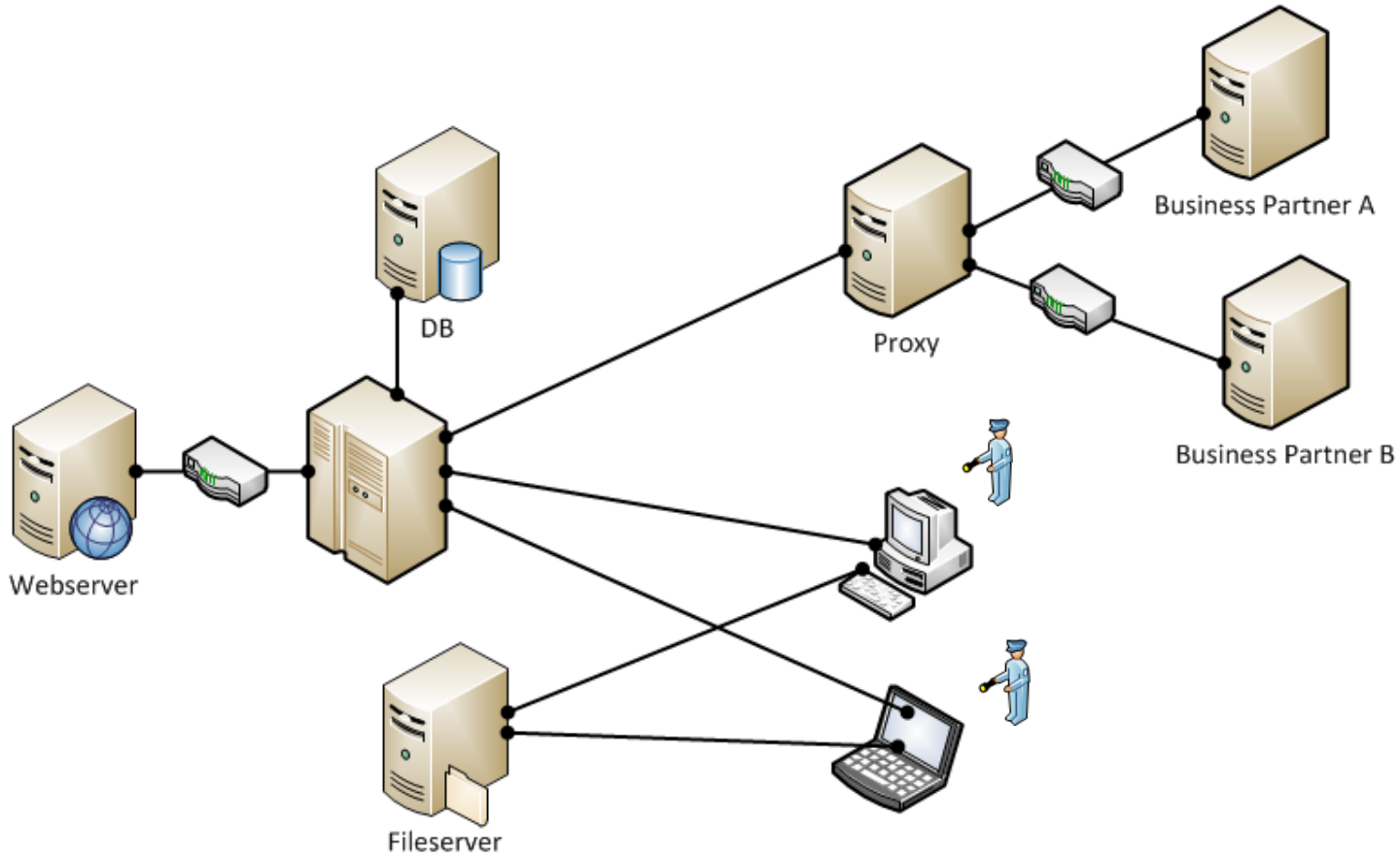
VIRTUALFORGE
we harden your software



- Monitor network egress/exit points
  - Multiple sensors required
  - Sensors must understand communication protocols

(2) Endpoint: “Data in use”

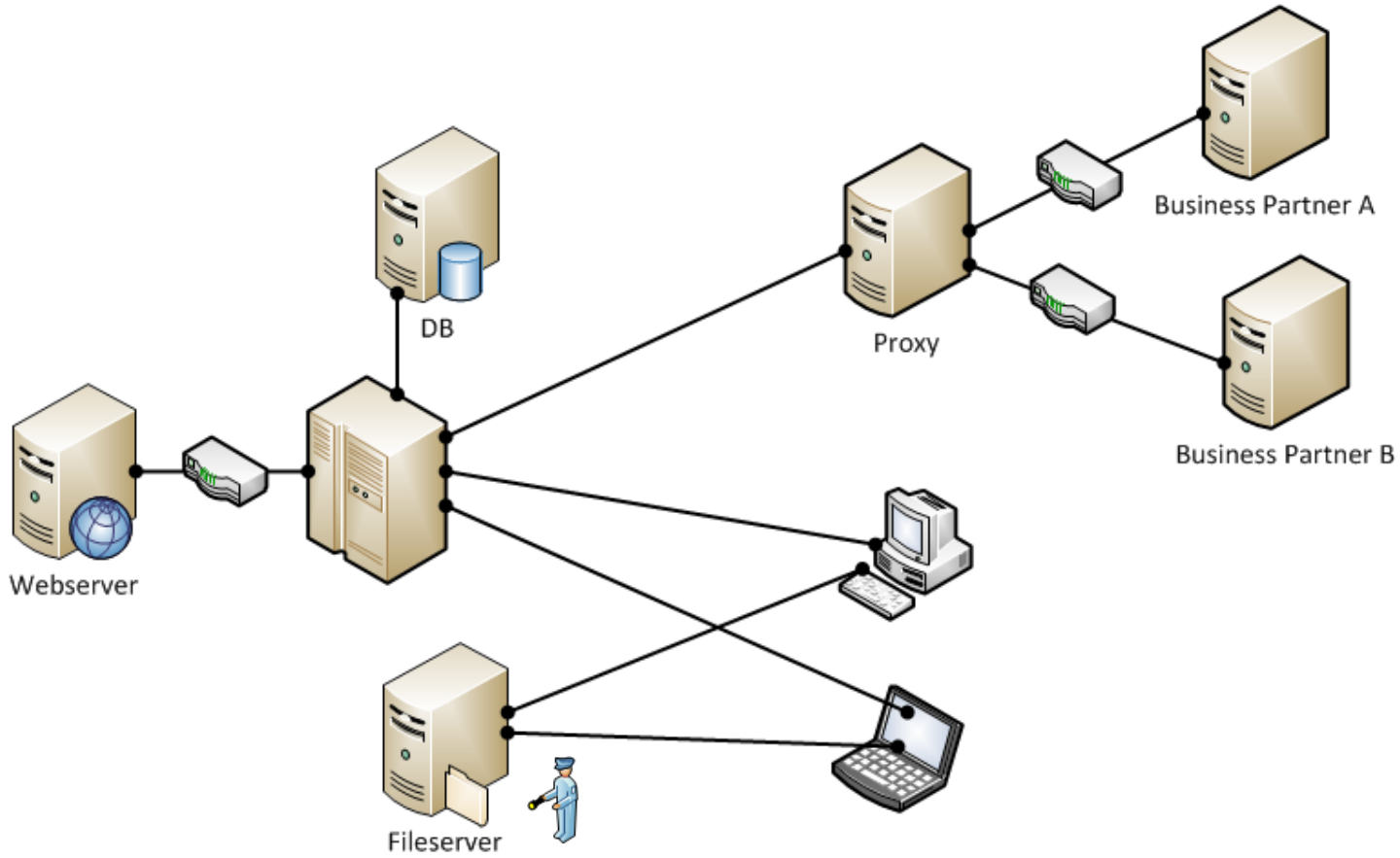
VIRTUALFORGE
we harden your software



- Monitor Endpoints
  - Sensor required for every endpoint
  - Many endpoints can‘t be monitored

(3) Endpoint: “Data at Rest”

VIRTUALFORGE
we harden your software



- Monitor critical Files
  - Apply tags to files
  - Does not work on DB content

DLP Solution Characteristics

VIRTUALFORGE
we harden your software

- DLP Solutions observe **Data**
- Their strength depends on
  - Data Analysis Capabilities (Heuristics)
    - What is „13.12.1982“ ?
      - The Date of an Order?
      - The Date of a Log Entry?
      - Someone‘s Birthday?
  - Pervasiveness
    - All relevant Network Points must be monitored
    - All relevant Protocols must be covered
    - We already learned: Not all Devices can be covered

DLP Solutions Summary

VIRTUALFORGE
we harden your software

Pro‘s

- They provide results
- Independent of Software in use

Con‘s

- High Complexity in Deployment
  - Many Sensors, many Protocols to cover, continuous Operation required
- DLP is a „last Line of Defense“ Approach
  - If the Solution fails, the Data is gone
- Heuristic produces false positives and negatives
  - Scrambled Data can‘t be analyzed
- Users operating the DLP Solution may see critical Data
  - Raises additional Issues

Conventional Static Code Analysis (SCA)

SCA Characteristics

VIRTUALFORGE
we harden your software

- SCA Solutions analyze Code for Defects without running it
- Various Issue-detection Algorithms
  - Pattern Tests (simple)
  - Semantic Tests
  - Business Logic Tests
  - Control Flow Tests
  - Data Flow Tests
  - …
- Can detect
  - Injection Flaws (SQL Inj, OS Cmd Inj, …)
  - XSS, CSRF
  - Authorization Flaws
  - …

Pattern Tests (Example)

VIRTUALFORGE
we harden your software

```
REPORT ZFT.

DATA lv_bname TYPE xubname.

SELECT bname FROM usr02 INTO lv_bname
    WHERE aname = sy-uname CLIENT SPECIFIED.
  WRITE lv_bname.
ENDSELECT.
```

Detect if Database Access is cross-Client

Semantic Tests (Example)

VIRTUALFORGE
we harden your software

```
REPORT ZFT.
CALL FUNCTION 'SECURITY_CHECK_INPUT' EXPORTING
                                       code    = lv_input
                                     EXCEPTIONS
                                       empty   = 1
                                       tainted = 0
                                       OTHERS  = 2.
IF sy-subrc NE 0.
  EXIT.
ENDIF.
```

Detect broken exception handling

Business Logic Tests (Example)

VIRTUALFORGE
we harden your software

```
REPORT ZFT.

SELECT SINGLE VKORG FROM T001W INTO T001W-VKORG
       WHERE WERKS = '2342'.
```

Detect hard-coded Plant

Control Flow (Example #1)

VIRTUALFORGE
we harden your software

```
REPORT ZFT.

EXPORT (fields) TO SHARED MEMORY indx(xy) ID co_bufid.

...

DELETE FROM SHARED MEMORY indx(xy) ID co_bufid.
```

Check if SAP Shared Memory is released **after** use

Control Flow (Example #2)

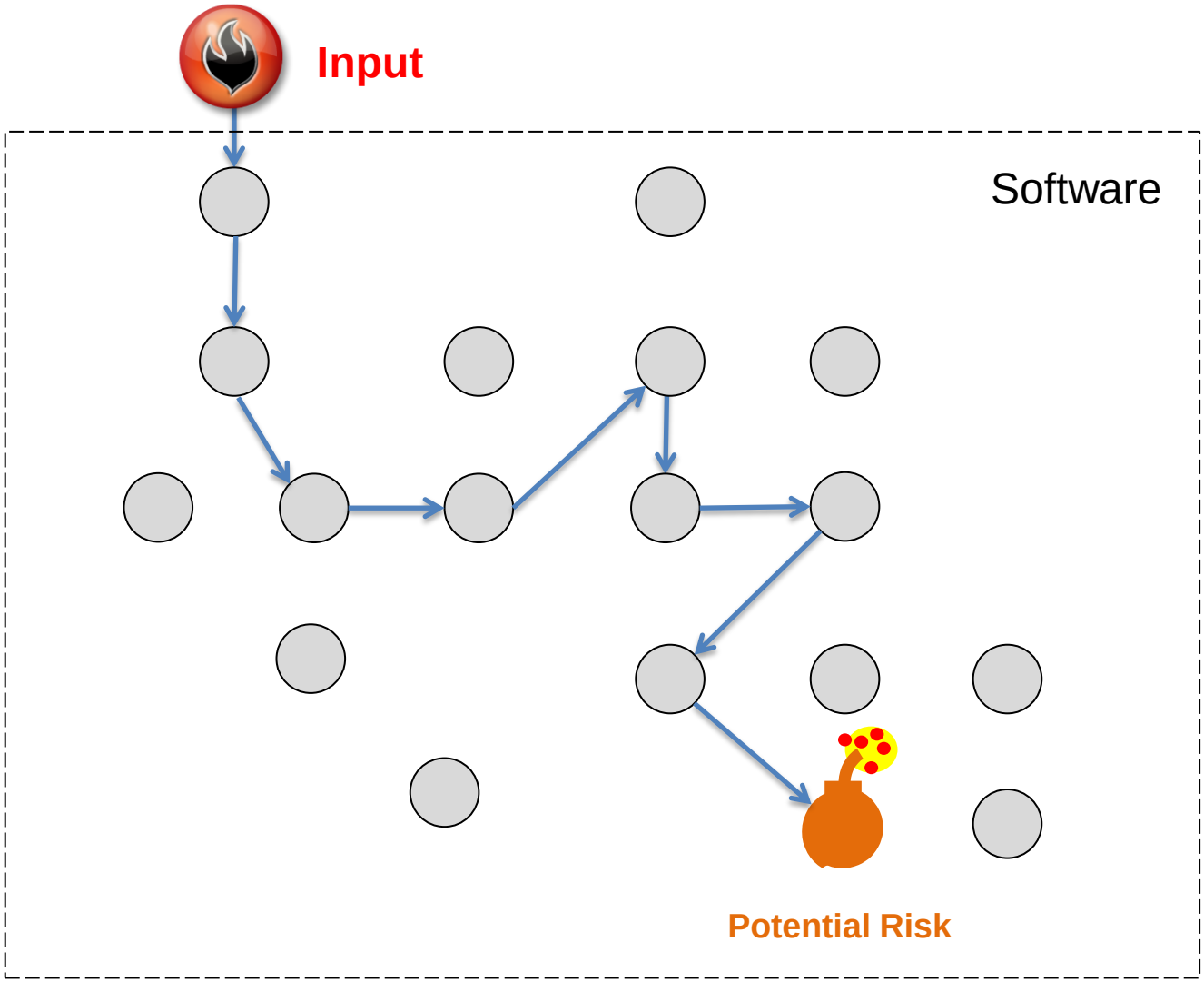
VIRTUALFORGE
we harden your software

```
REPORT ZFT.

AUTHORITY-CHECK OBJECT 'S_TCODE'
                ID     'TCD'
                FIELD  'SE38'.

IF sy-subrc IS INITIAL.
  CALL TRANSACTION 'SE38'.
ENDIF.
```

Assure that authorization is checked **before** a transaction is called

Data Flow (Methodology)

VIRTUALFORGE
we harden your software



Data Flow (Example)

VIRTUALFORGE
we harden your software

```
REPORT ZFT.

DATA lv_cmd TYPE string.
PARAMETERS os_option TYPE string.

CONCATENATE 'ls -' os_option INTO lv_cmd.

CALL 'SYSTEM' ID 'COMMAND'   FIELD lv_cmd.
```

Check, if Input is passed to a dangerous command

SCA Summary

VIRTUALFORGE
we harden your software

Pro‘s

- White Box Approach: All Functionality can be checked
- Proactive: Analysis takes place **before** productive Use
- SCA finds (inadvertent) Security and Quality Defects

Con‘s

- Certain Behavior is hard to emulate statically
- Compiled Code might behave differently
- Data Leaks not in Scope

Data Leaks in SAP Systems

Case #1 – SAP Standard (1)

VIRTUALFORGE
we harden your software

```
FUNCTION search_bank_address_extended .
*"----------------------------------------------------------------------
*"*"Lokale Schnittstelle:
*"  IMPORTING
*"     VALUE(I_BANKS) LIKE  BNKA-BANKS
*"     VALUE(I_BANKL) LIKE  BNKA-BANKL OPTIONAL
*"     VALUE(I_BANKA) LIKE  BNKA-BANKA OPTIONAL
*"     VALUE(I_ORT01) LIKE  BNKA-ORT01 OPTIONAL
*"     VALUE(I_BNKLZ) LIKE  BNKA-BNKLZ OPTIONAL
*"     VALUE(I_SWIFT) LIKE  BNKA-SWIFT OPTIONAL
*"     VALUE(I_BRNCH) LIKE  BNKA-BRNCH OPTIONAL
*"     VALUE(I_STRAS) LIKE  BNKA-STRAS OPTIONAL
*"  TABLES
*"      ET_BNKA STRUCTURE  BNKA
*"      ET_RETURN STRUCTURE  BAPIRET2 OPTIONAL

...
```

INPUT

OUTPUT

Case #1 – SAP Standard (2)

VIRTUALFORGE
we harden your software

```
FUNCTION search_bank_address_extended .

...

  SELECT * FROM bnka INTO TABLE et_bnka
    WHERE banks EQ   i_banks
      AND bankl LIKE i_bankl
      AND banka LIKE i_banka
      AND stras LIKE i_stras
      AND ort01 LIKE i_ort01
      AND swift LIKE i_swift
      AND bnklz LIKE i_bnklz
      AND brnch LIKE i_brnch.
```

OUTPUT

INPUT

- Table **bnka** is the Bank Master Record of an SAP System
- User controls entire WHERE condition (based on LIKE)
- The FUNCTION can be invoked remotely (e.g. via SOAP)

Case #1 – Side Notes

VF Advisory: **SAP-LOSS-01**

SAP Note: 1684539

Fixed: June 2012

CVSS Base Score: 3.5

CVSS Base Vector: AV:N/AC:M/AU:S/C:P/I:N/A:N

Case #2 – Customer Code (1)

VIRTUALFORGE
we harden your software

```
FUNCTION z_get_PA0002_data .
*"----------------------------------------------------------------------
*"*"Lokale Schnittstelle:
*"  IMPORTING
*"     VALUE(I_PERNR)  LIKE  PA0002-PERNR
*"     VALUE(I_PERID)  LIKE  PA0002-PERID
*"  EXPORTING
*"     VALUE(E_ERRMSG) TYPE  CHAR200
*"  TABLES
*"      ET_PA0002      STRUCTURE  PA0002


...
```

INPUT

OUTPUT

Case #2 – Customer Code (2)

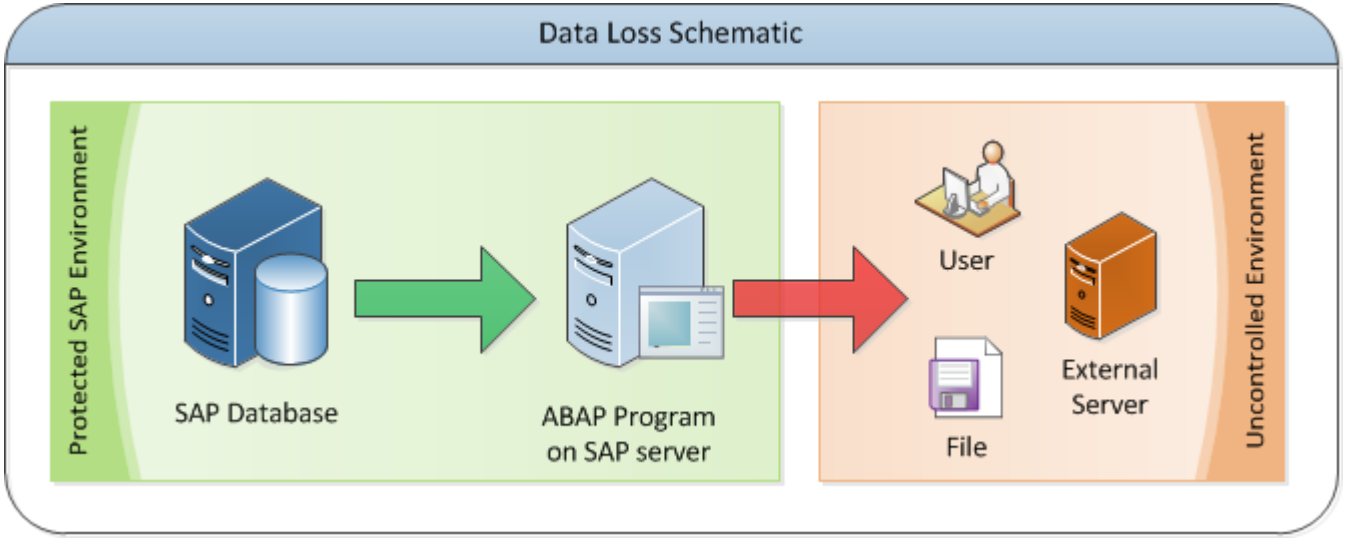
VIRTUALFORGE
we harden your software

```
FUNCTION z_get_PA0002_data .
...
SELECT * FROM pa0002 INTO lt_pa0002 WHERE PERNR = i_pernr.
" Plausibility check
IF lt_pa0002-perid <> i_perid.
  CONCATENATE `PERNR and PERID don't match. PERNR ` i_pernr
              ` corresponds to PERID ` lt_pa0002-perid
              `, not to PERID ` i_perid
    INTO e_errmsg.
  EXIT.
ENDIF.
```

- Column **perid** in table **pa0002** contains the Social Security Number
- User controls entire WHERE condition (based on input parameter)
- The FUNCTION can be invoked remotely (e.g. via SOAP)

Static Data Leak Prevention (S-DLP)

S-DLP Challenges



What would it take, to statically detect Data Leaks in (SAP) Software?

1. Need to reliably (!) identify critical Data (“Assets”)
2. Need to identify Data Transfers to the Outside (“Exits”)
3. Need to identify Data Flow from Assets to Exits

Asset Identification

VIRTUALFORGE
we harden your software

SAP

- All Business Data is stored in the SAP Database

ABAP

- Integrated platform-independent SQL Standard: Open SQL
  - Access to DB Tables is easy to detect / parse
- Thousands of well-known database tables in SAP Standard
  - All critical Tables / Columns are well known

Challenge 1 taken:

Need to reliably identify “Assets”

Exit Discovery

VIRTUALFORGE
we harden your software

ABAP

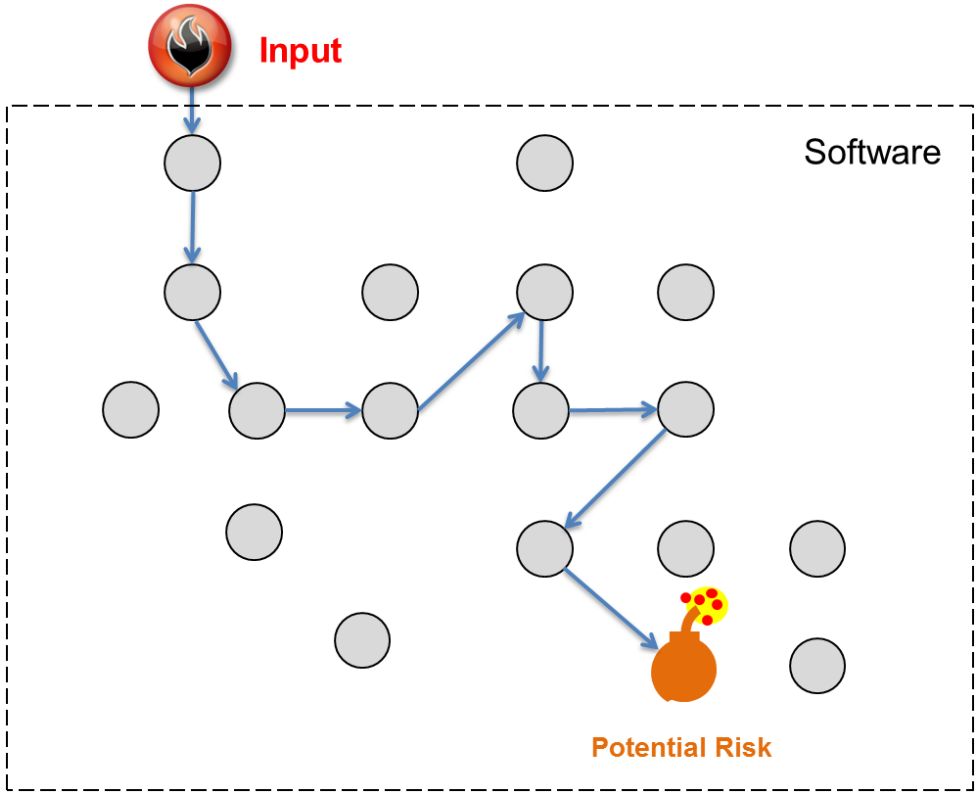
- A limited Number of Commands cause Data Exits
  - Easy to resolve
- A large Number of API functions cause Data Exits
  - A diligent but routine piece of work (ongoing)

Challenge 2 taken:

Need to identify Data Transfers to “Exits”

Inverse Data Flow Analysis ?

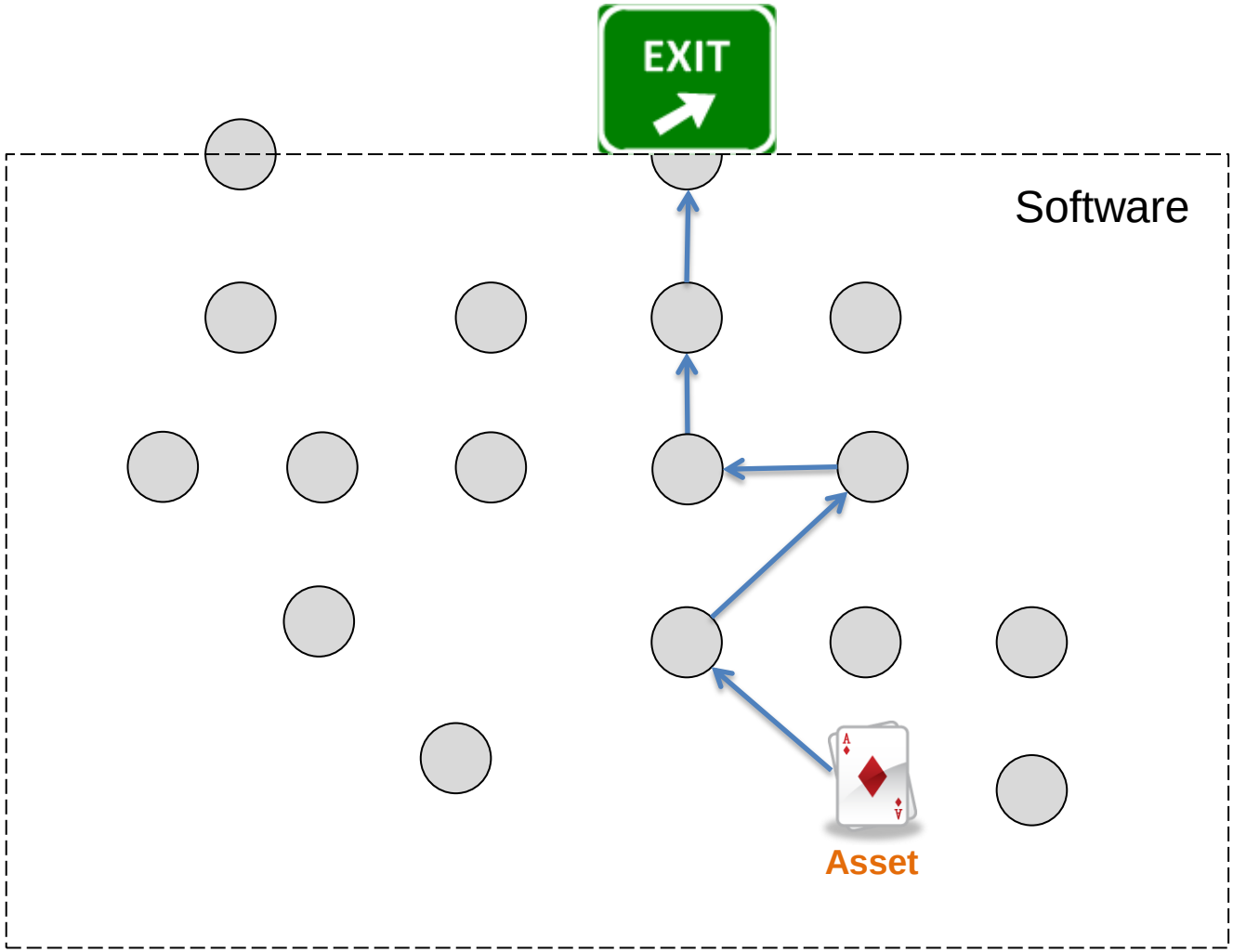
VIRTUALFORGE
we harden your software



- Classic Data Flow Analysis is an “Outside-in” Approach
- We need an „Inside-out“ Approach

We need Asset Flow Analysis

VIRTUALFORGE
we harden your software



Identify Data Flow from Assets to Exits

VIRTUALFORGE
we harden your software

Data Flow Analysis

- Modifications to Input alter Impact of Findings
  - E.g. Escaping, Pre-/Postfixing

Asset Flow Analysis

- Modifications to Assets alter Impact of Findings
  - E.g. Substrings

Challenge 3 taken:

Need to provide Asset Flow Analysis

S-DLP Summary

VIRTUALFORGE
we harden your software

Pro‘s

- S-DLP Concept works
- Proactive: Analysis takes place **before** productive Use
- White Box Approach: All Functionality can be checked
- Almost no false Positives, since Data Source and Meaning is known
- No Sensors required (on Network or Clients)
- All Exit-points of an Application are covered, regardless of Protocol
- If Leaks are detected, critical Data is not disclosed to an Auditor
- Even scrambled Data can be analyzed
- Analysis only required, if Application Code changes

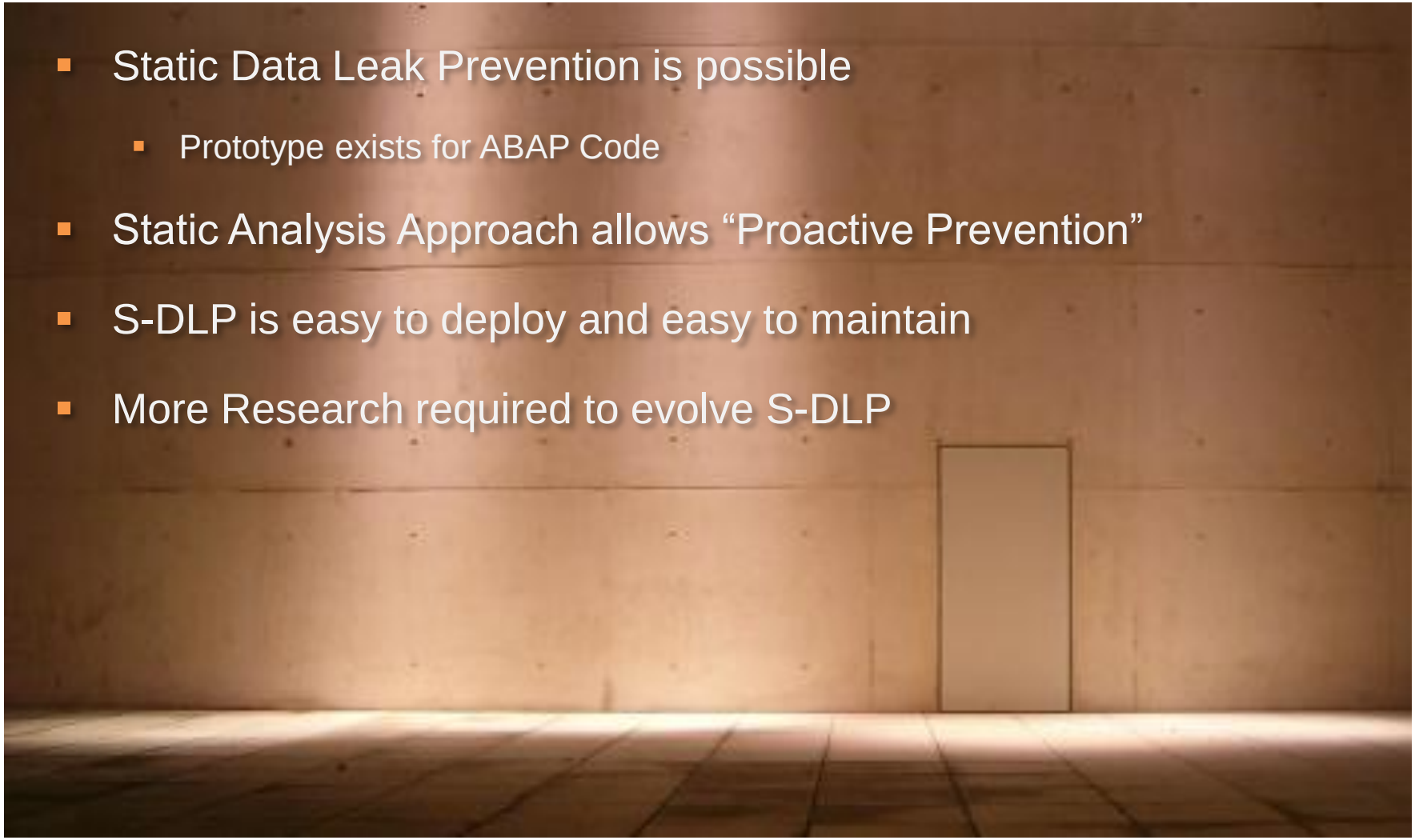
Con‘s

- Currently restricted to SAP (ABAP) Systems
- Currently only monitors Database Content

Summary

S-DLP Summary

VIRTUALFORGE
we harden your software



- Static Data Leak Prevention is possible
  - Prototype exists for ABAP Code
- Static Analysis Approach allows “Proactive Prevention”
- S-DLP is easy to deploy and easy to maintain
- More Research required to evolve S-DLP

ABAP Security Resources

VIRTUALFORGE
we harden your software

Links

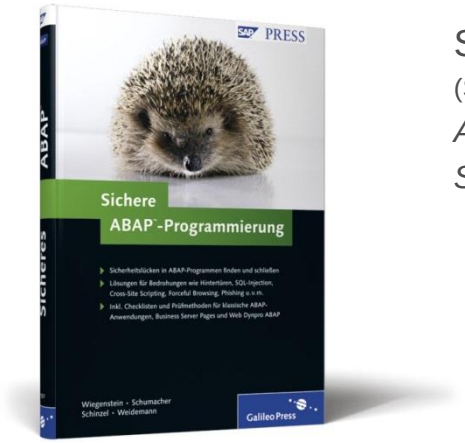
SAP Security Advisories researched by Virtual Forge
http://www.codeprofilers.com/index.php/advisories.html

Organizations

BIZEC
BUSINESS SECURITY

BIZEC – Business Security Initiative
http://www.bizec.org

Literature



Sichere ABAP-Programmierung
(SAP PRESS, 372 S., 2009)
Andreas Wiegenstein, Markus Schumacher, Sebastian Schinzel, Frederik Weidemann

VIRTUALFORGE
we harden your software

Questions?

VIRTUALFORGE
we harden your software

Contact Information

Andreas Wiegenstein

VIRTUALFORGE GmbH

Web: http://virtualforge.com

Mail: andreas.wiegenstein@virtualforge.com

Phone: + 49 (0) 6221 86 89 00

Twitter: @codeprofiler

Disclaimer

SAP, R/3, ABAP, SAP GUI, SAP NetWeaver and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and other countries.

All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only.

The authors assume no responsibility for errors or omissions in this document. The authors do not warrant the accuracy or completeness of the information, text, graphics, links, or other items contained within this material. This document is provided without a warranty of any kind, either express or implied, including but not limited to the implied warranties of merchantability, fitness for a particular purpose, or non-infringement.

The authors shall have no liability for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of this document.

No part of this document may be reproduced without the prior written permission of Virtual Forge GmbH. Not even in case of unforeseen data leaks.