Quellcodescans von Webanwendungen in der Praxis

Gängige Fallstricke und Wege zum erfolgreichen Einsatz in Unternehmen

Sebastian Schinzel
Virtual Forge GmbH

Web 1.0:
sebastian.schinzel@virtualforge.com

Web 2.0:
https://twitter.com/seecurity

OWASP
17.11.2011

The OWASP Foundation
http://www.owasp.org

Neuigkeiten zu vergangenen OWASP-Vorträgen

- 2008 (Uni Mannheim)
  - Measuring the Security of Web Applications

- 2009 (Virtual Forge)
  - Sichere Entwicklung und gängige Schwachstellen in eigenentwickelten SAP-Web-Anwendungen

SAP-Security - Sicherheitslöcher in eigenem ABAP-Code stopfen
http://www.virtualforge.com/tl_files/Theme/Artikel/ix.0711.118-122.pdf

Neuigkeiten zu vergangenen OWASP-Vorträgen

- 2010 (Uni Mannheim / Uni Erlangen)
  - Seitenkanalschwachstellen im Web erkennen und verhindern

Detecting Hidden Storage Side Channel Vulnerabilities in Networked Applications
Felix C. Freiling and Sebastian Schinzel
http://sebastian-schinzel.de/_download/ifip-sec2011.pdf

An Efficient Mitigation Method for Timing Side Channels on the Web
Sebastian Schinzel
http://sebastian-schinzel.de/_download/cosade-2011-extended-abstract.pdf

Time is on my Side - Exploiting Timing Side Channel Vulnerabilities on the Web
Sebastian Schinzel
28th Chaos Communication Congress, December 27th to 30th, 2011

- 2011 (Virtual Forge)
  - Quellcodescans von Webanwendungen in der Praxis

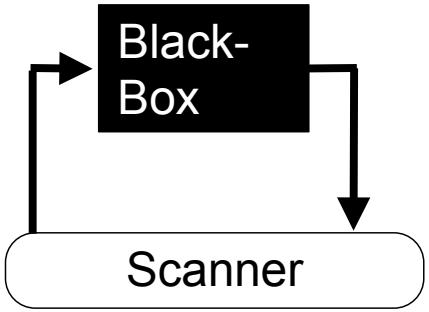
Crash-Kurs: Automatisierte Security-Analyse

Begriffe:

- SCA: Statische (Quell-) Code Analyse
- Scanner: Werkzeug, das Analyse durchführt
- Finding: Resultat das Scanner anhand von Testfällen erzeugt
- Testfall (Test case): Regel, nach der nach Schwachstelle gesucht wird

Scanner mit einem oder mehreren Testfällen führt SCA über Ihren Quellcode durch und erzeugt eine Menge von Findings

Crash-Kurs: Automatisierte Security-Analyse



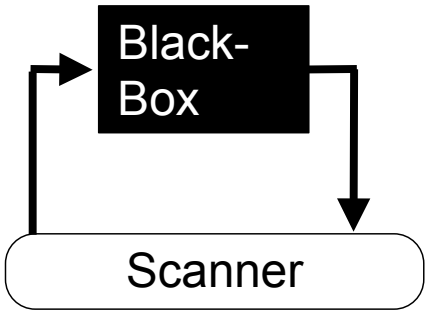
Crash-Kurs: Automatisierte Security-Analyse

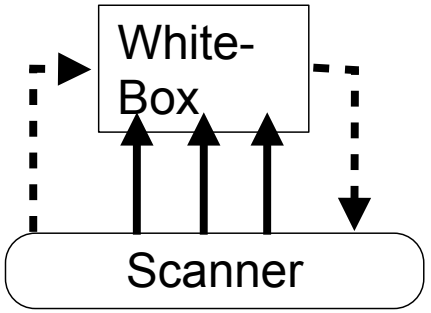
Black-Box:

- Kein Zugriff auf Quellcode

White-Box:

- Zugriff auf Quellcode





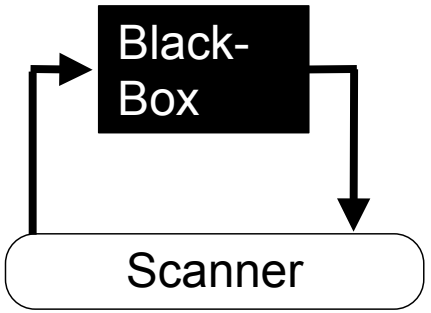
Crash-Kurs: Automatisierte Security-Analyse

Black-Box:

- Kein Zugriff auf Quellcode

White-Box:

- Zugriff auf Quellcode

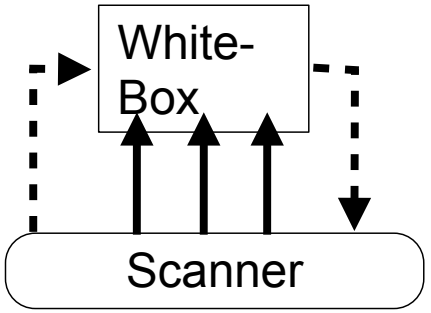


Dynamische Analyse:

- Anwendung wird ausgeführt

Statische Analyse:

- Quellcode wird in abstraktes Modell überführt
- nur "symbolische" Ausführung



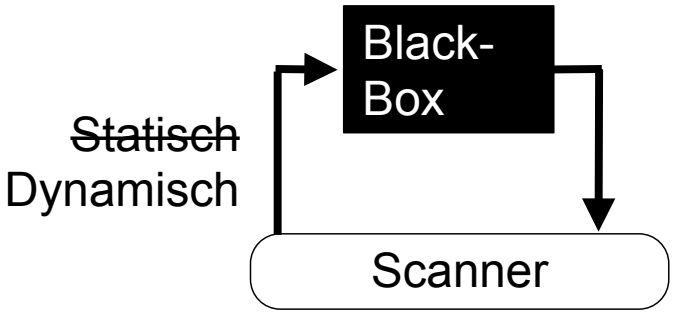
Crash-Kurs: Automatisierte Security-Analyse

Black-Box:

- Kein Zugriff auf Quellcode

White-Box:

- Zugriff auf Quellcode

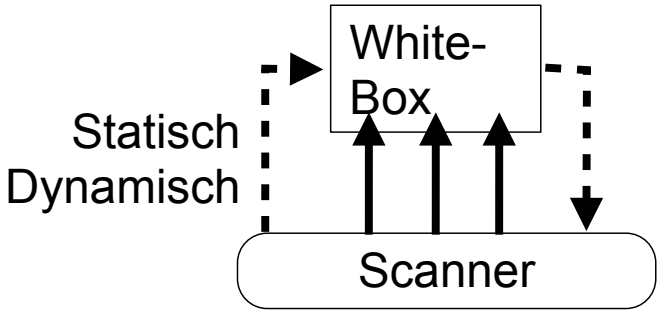


Dynamische Analyse:

- Anwendung wird ausgeführt

Statische Analyse:

- Quellcode wird in abstraktes Modell überführt
- nur "symbolische" Ausführung



Grundlagen - Black-Box vs. White-Box

Black-Box-Analyse findet Schwachstellen, die

- sich in aktiven Code befinden
- sich in Code befinden, der vom Scanner angesprochen wird
- direkt ausnutzbar sind

Grundlagen - Black-Box vs. White-Box

Black-Box-Analyse findet keine Schwachstellen, die

- sich in (aktuell) inaktivem Code befinden
- sich in Code befinden, den der Scanner nicht angesprochen hat (Abdeckung des Scans)
- die zufällig und zu diesem Zeitpunkt nicht ausnutzbar sind

--> “Wie kann ich die Schwachstelle in der Anwendung schließen?”

Grundlagen - Black-Box vs. White-Box

Black-Box-Analyse findet keine Schwachstellen, die

- sich in (aktuell) inaktivem Code befinden
- sich in Code befinden, den der Scanner nicht angesprochen hat (Abdeckung des Scans)
- die zufällig und zu diesem Zeitpunkt nicht ausnutzbar sind

--> "Wie kann ich die Schwachstelle in der Anwendung schließen?"

http://www.cigital.com/justiceleague/2007/03/19/badness-ometers-are-good-do-you-own-one/

Grundlagen - Black-Box vs. White-Box

White-Box-Analyse findet Schwachstellen

- in der gesamten Code-Basis
- die sich in Test-Code befinden
- die zufällig nicht ausgenutzt werden können
- Name: “Non-Exploitables”, nicht “False Positives”

Grundlagen - Black-Box vs. White-Box

White-Box-Analyse findet Schwachstellen

- in der gesamten Code-Basis
- die sich in Test-Code befinden
- die zufällig nicht ausgenutzt werden können
- Name: “Non-Exploitables”, nicht “False Positives”

White-Box-Analyse liefert direkt die Stellen im Code, in denen sich die Schwachstellen befinden

- Fokus auf die Reparatur, nicht nur Entdeckung von Schwachstellen

Grundlagen - Black-Box vs. White-Box

White-Box-Analyse hat es schwer Schwachstellen zu finden, die

- sich in der Geschäftslogik befinden (“Forceful Browsing”, “Cross Site Request Forgery”)
- durch komplexe Frameworks “obfuskiert” werden (aber ausnutzbar sind)
- durch die Konfiguration von Programmier-Frameworks, Betriebssystem, Applicationserver, usw. entstehen

Einführung in statische Quellcodeanalyse

Drei verschiedene Typen von SCA:

- Einfache Mustererkennung (1/3)
  - Durchsucht einzelne Codezeilen
  - Beispiel: Verdächtige Worte in Kommentaren oder Variablennamen
  - password = ‘virtualforge4223’.

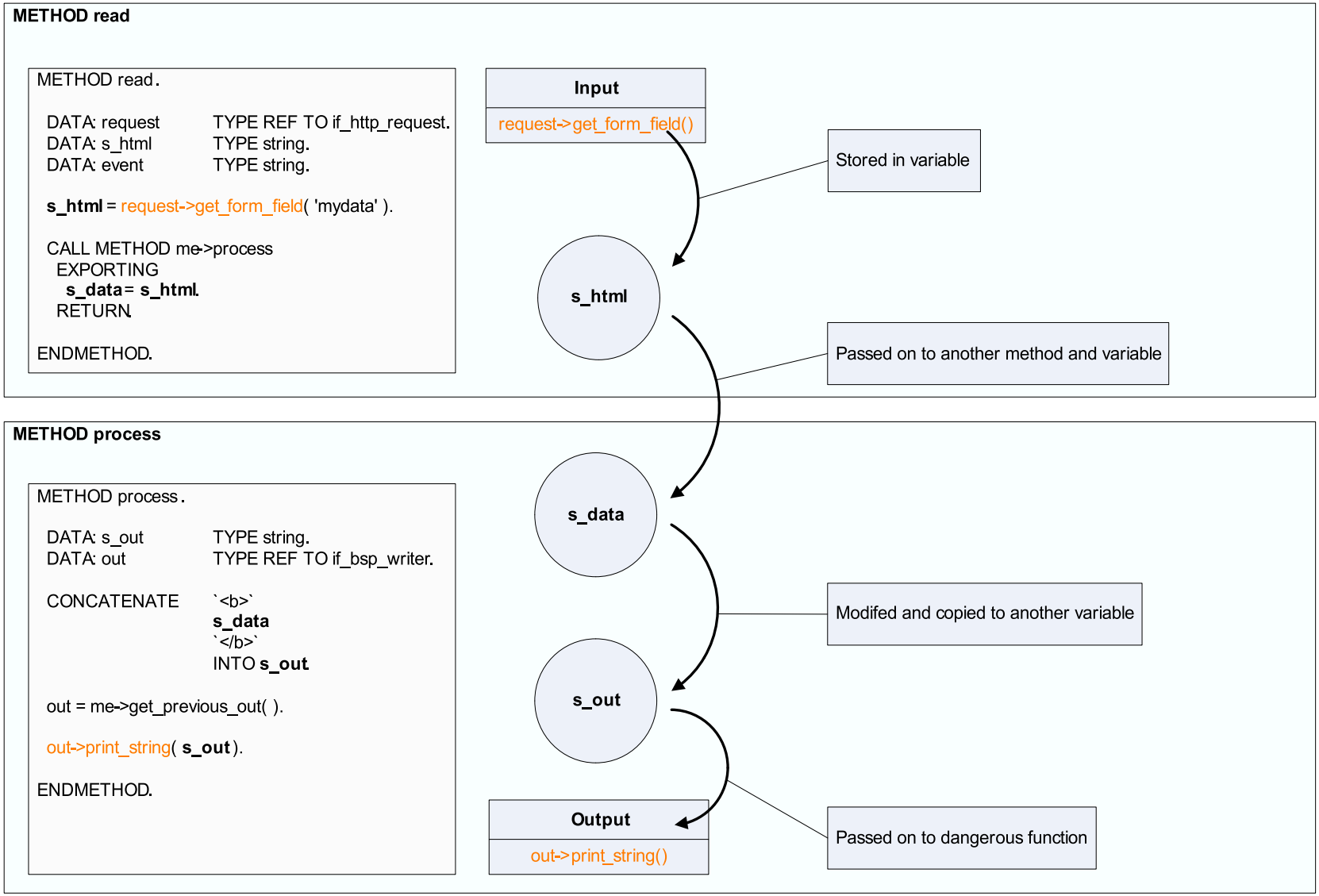
- Kontrollfluss-Analyse (2/3)
  - Durchsucht Blöcke nach Mustern
  - Beispiel: Fehlende Berechtigungsprüfung

Einführung in statische Quellcodeanalyse

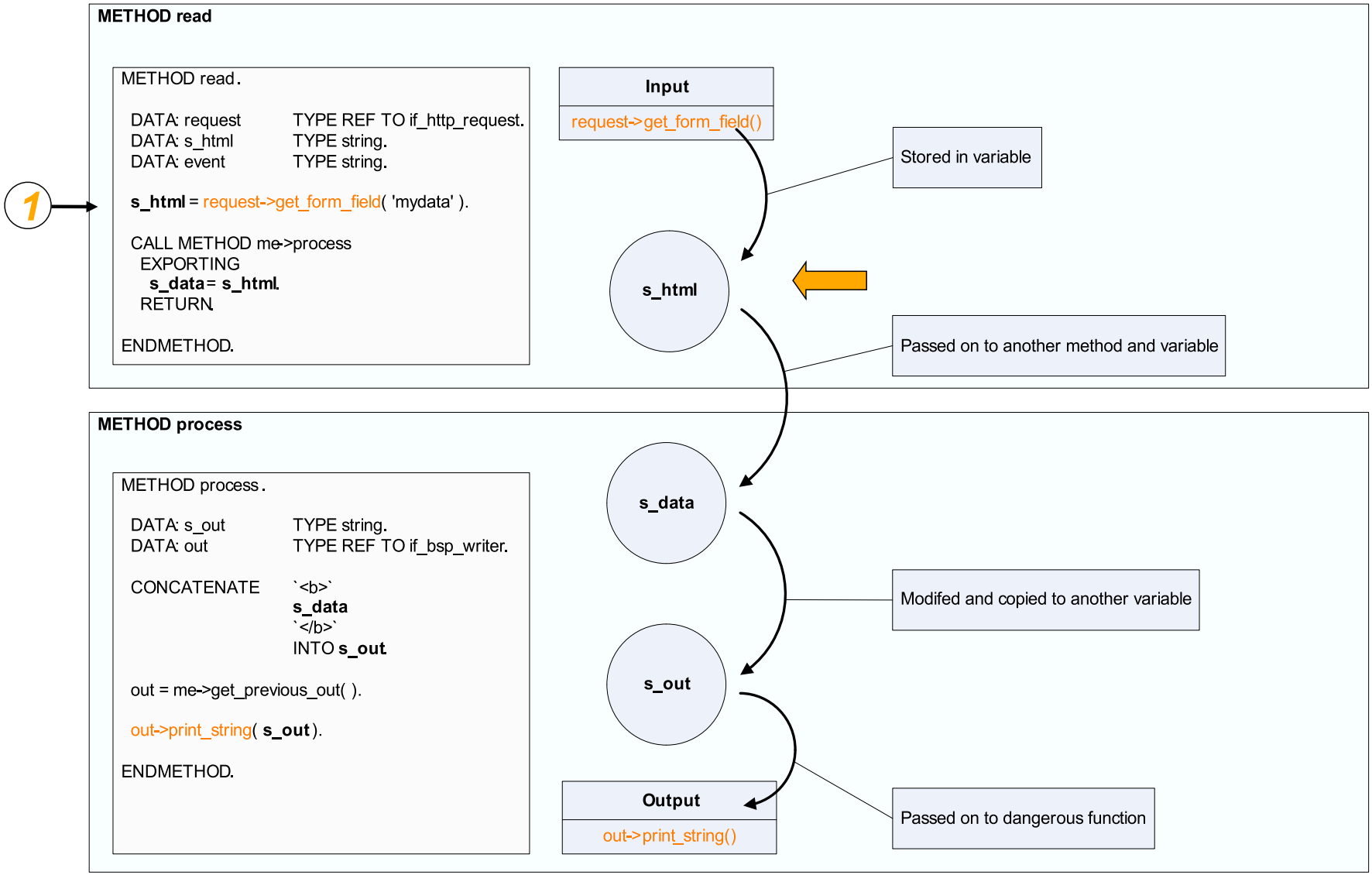
Drei verschiedene Typen von SCA:

- Datenfluss-Analyse (3/3)
  - Folgt dem Pfad, auf dem Daten durch das Programm fliesst
  - Beispiel: Cross Site Scripting
  - "An welchen Stellen werden Benutzereingaben ungefiltert in eine Web-Seite geschrieben?"

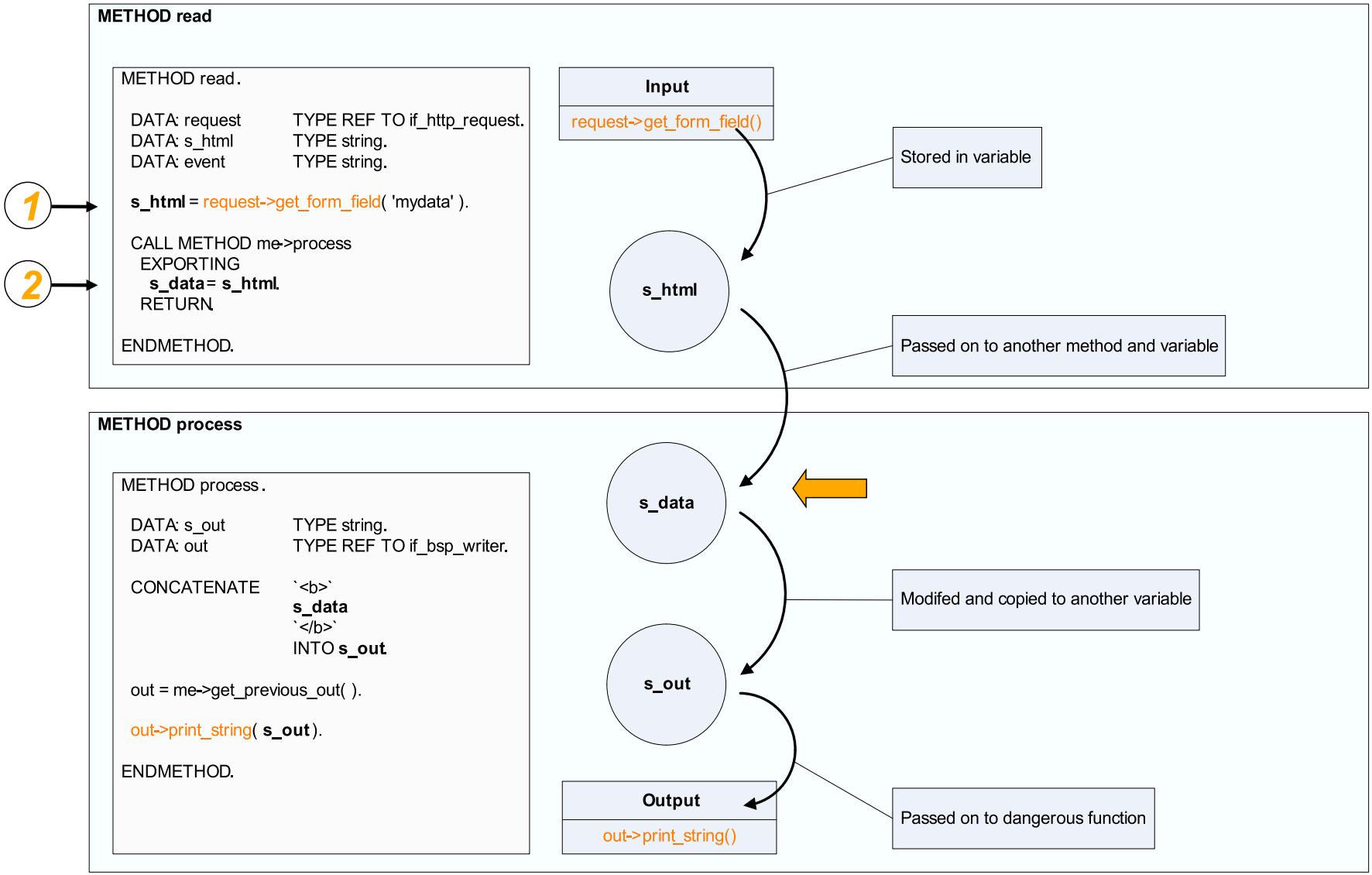
Einführung in statische Quellcodeanalyse



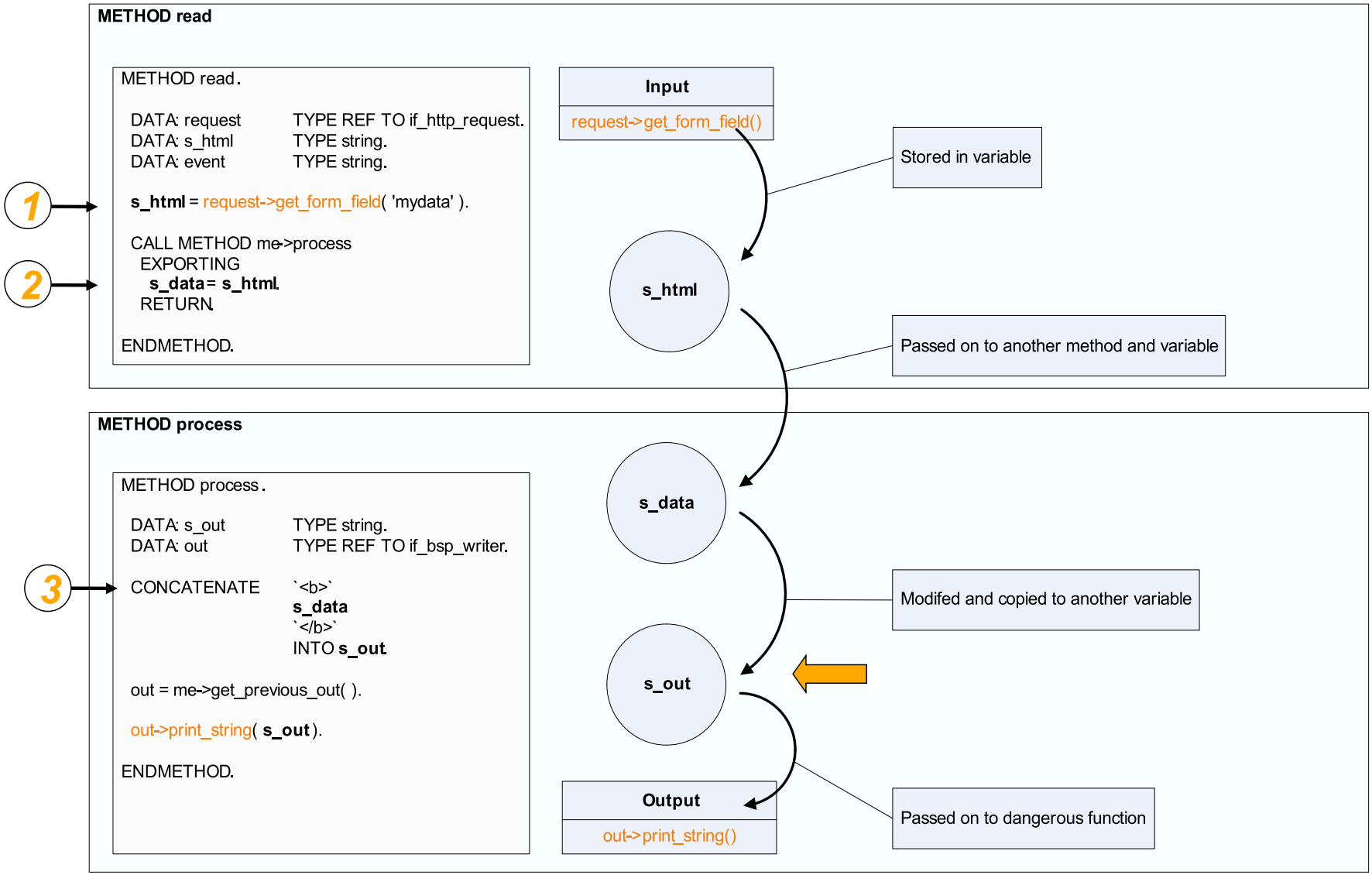
Einführung in statische Quellcodeanalyse



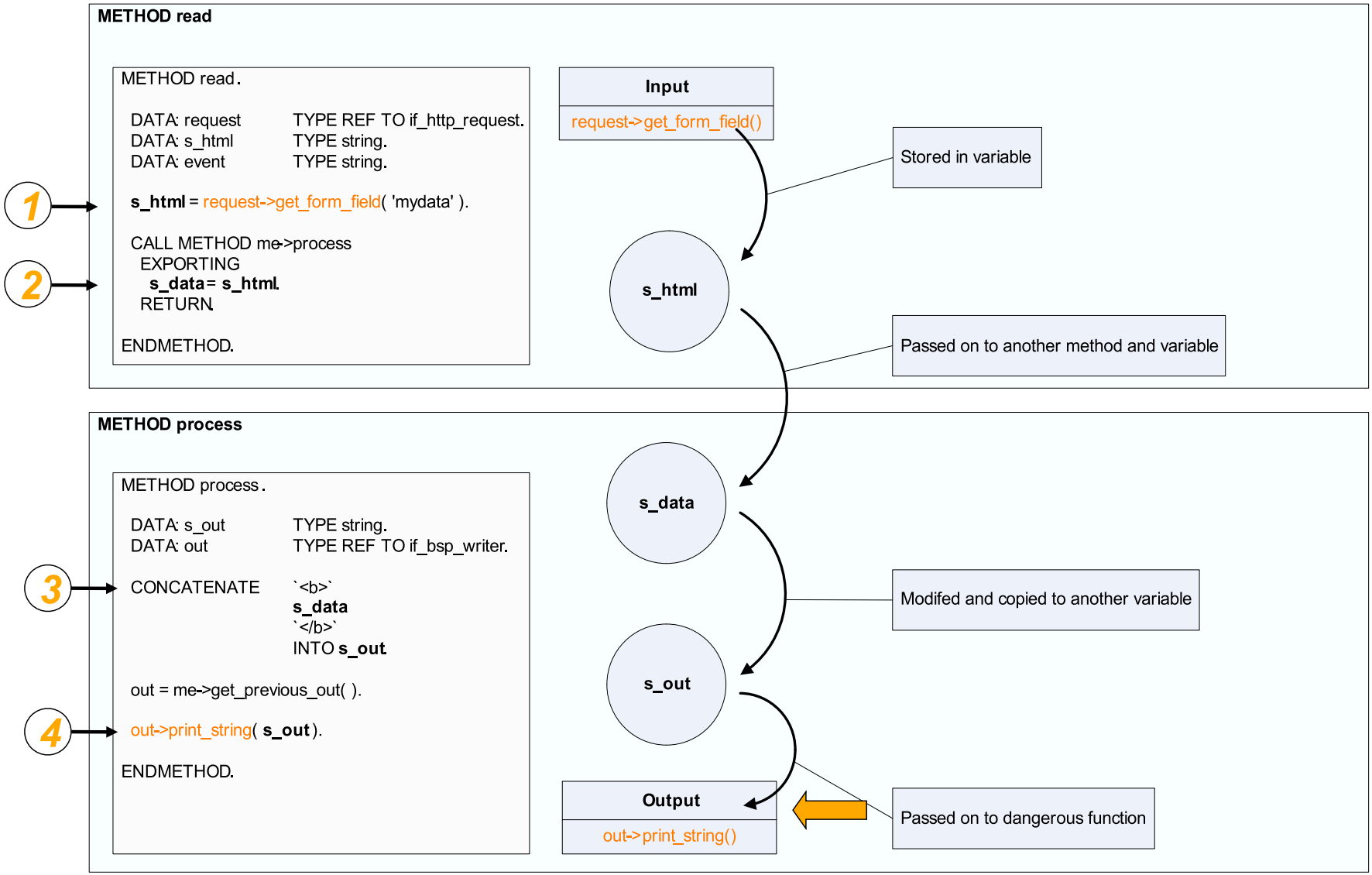
Einführung in statische Quellcodeanalyse



Einführung in statische Quellcodeanalyse



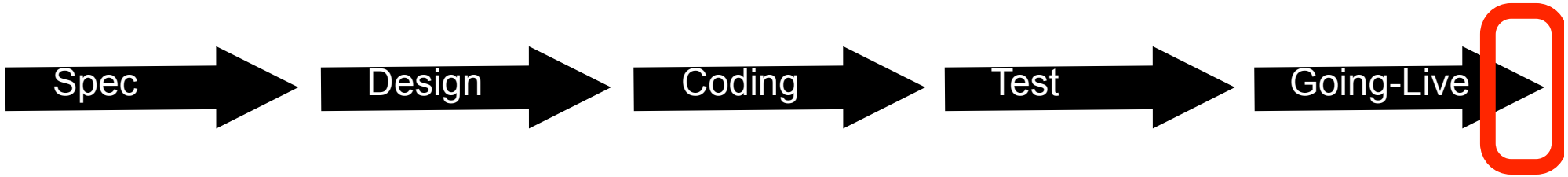
Einführung in statische Quellcodeanalyse



Roadmap

- Integration von SCA in den Softwareentwicklungsprozess (SDL)
- Kommunikation innerhalb des Unternehmens
- Umgang mit Findings
- Umgang mit Testfällen

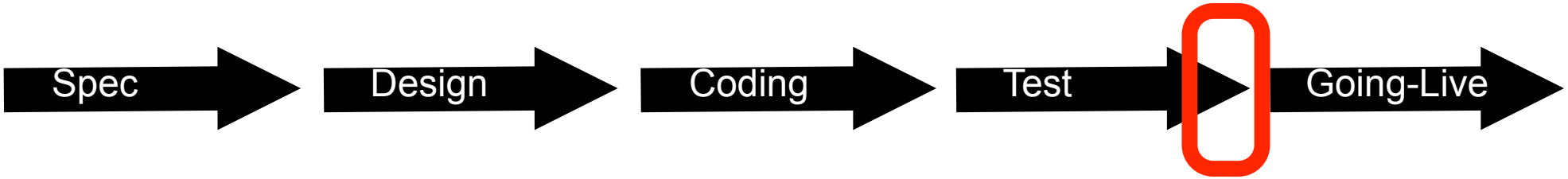
Integration von SCA in den SDL



Einsatz nach Going-Live:

- Erneute Einarbeitung von Entwicklern, Redundande Entwicklung, Retest, neuer Roll-out

Integration von SCA in den SDL



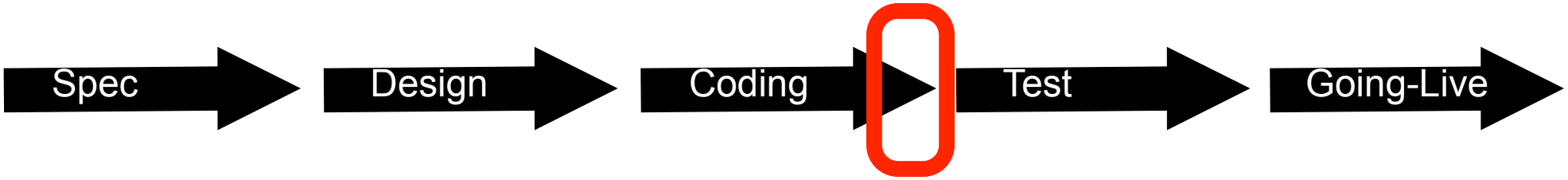
Einsatz nach Going-Live:

- Erneute Einarbeitung von Entwicklern, Redundande Entwicklung, Retest, neuer Roll-out

Einsatz nach Test

- Erneute Einarbeitung von Entwicklern, Redundande Entwicklung, Retest

Integration von SCA in den SDL



Einsatz nach Going-Live:

- Erneute Einarbeitung von Entwicklern, Redundande Entwicklung, Retest, neuer Roll-out

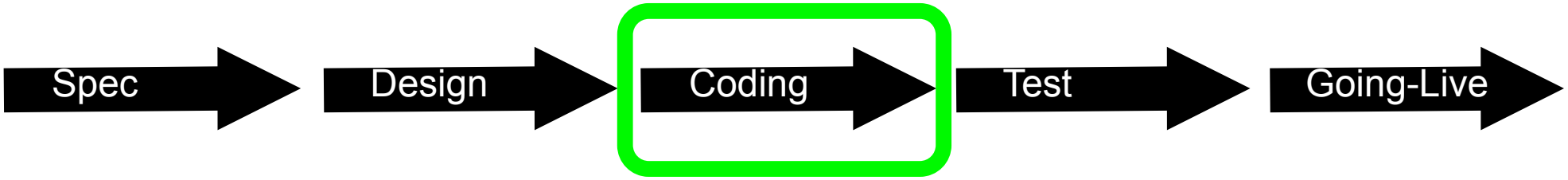
Einsatz nach Test

- Erneute Einarbeitung von Entwicklern, Redundande Entwicklung, Retest

Einsatz nach Coding

- Redundende Entwicklung

Integration von SCA in den SDL



Einsatz nach Going-Live:

- Erneute Einarbeitung von Entwicklern, Redundande Entwicklung, Retest, neuer Roll-out

Einsatz nach Test

- Erneute Einarbeitung von Entwicklern, Redundande Entwicklung, Retest

Einsatz nach Coding

- Redundende Entwicklung

Einsatz während Coding

- --> Training on the job?

Roadmap

- Integration von SCA in den Softwareentwicklungsprozess (SDL)
- Kommunikation innerhalb des Unternehmens
- Umgang mit Findings
- Umgang mit Testfällen

Kommunikation innerhalb des Unternehmens

Naives Vorgehen:

- Entwicklern und Management die ungefilterten Zahlen aus SCA Report vortragen

Kommunikation innerhalb des Unternehmens

Naives Vorgehen:

- Entwicklern und Management die ungefilterten Zahlen aus SCA Report vortragen

Ihre Anwendungen haben **2342** unternehmenskritische Schwachstellen

Kommunikation innerhalb des Unternehmens

Entwickler sind sehr stolz auf ihren Code...

```
FUNCTION GET_CC_DATA.
    DATA: cc_no TYPE string.

    CALL FUNCTION check_authority.

    IF sy-subrc = 0.
*       Everything looks ok.
       CALL FUNCTION do_something.
    ENDIF.
```

=

Kommunikation innerhalb des Unternehmens

Entwickler sind sehr stolz auf ihren Code...

```
FUNCTION GET_CC_DATA.
    DATA: cc_no TYPE string.

    CALL FUNCTION check_authority.

    IF sy-subrc = 0.
*       Everything looks ok.
      CALL FUNCTION do_something.
    ENDIF.
```

=

Not cute!

Kommunikation innerhalb des Unternehmens

Entwickler sind sehr stolz auf ihren Code...

```
FUNCTION GET_CC_DATA.
    DATA: cc_no TYPE string.

    CALL FUNCTION check_authority.

    IF sy-subrc = 0.
*       Everything looks ok.
      CALL FUNCTION do_something.
    ENDIF.
```

=

Not cute!

← Your code!!

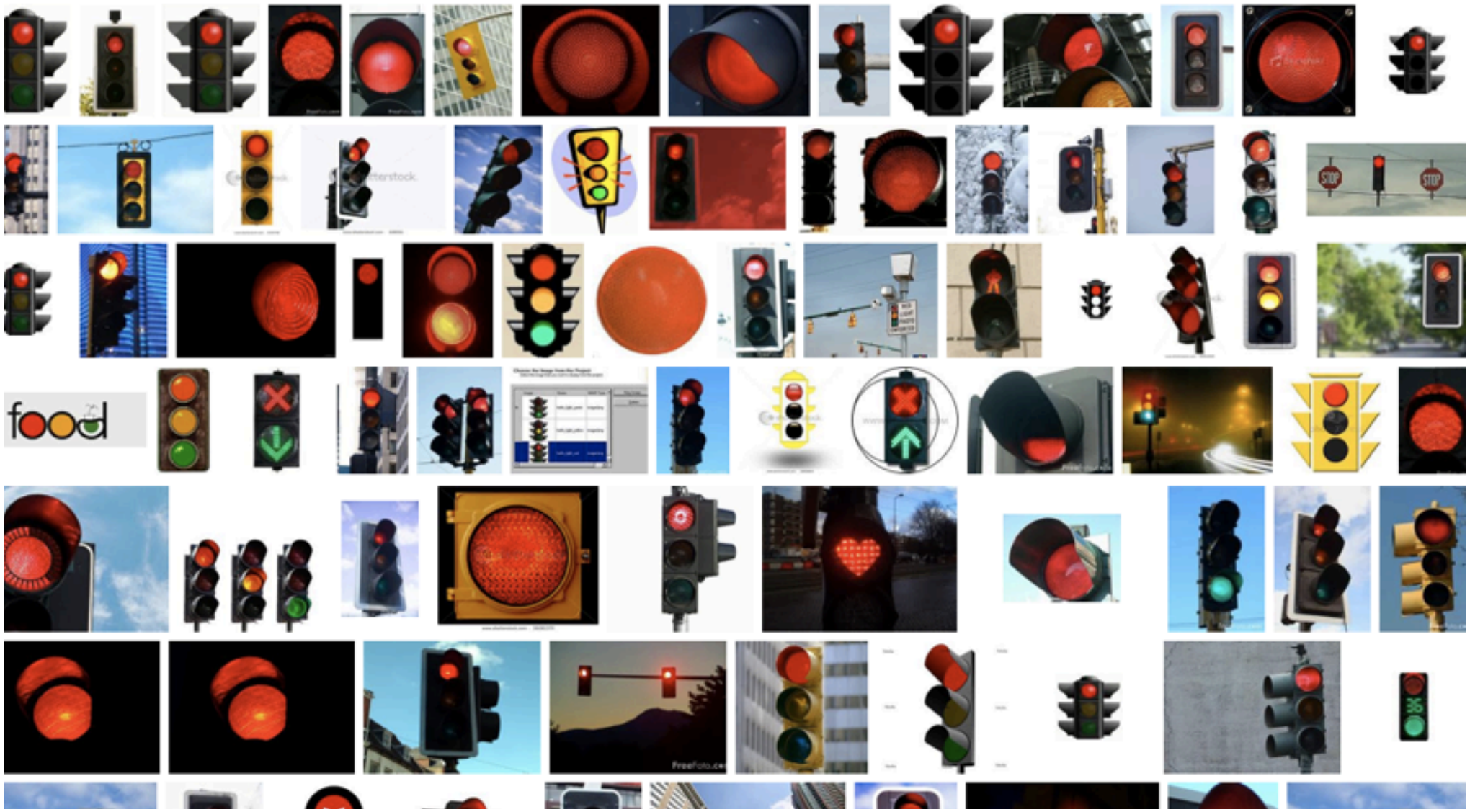
← Really ugly!!

Kommunikation innerhalb des Unternehmens

Was der “Management Overview” dem Management sagt:

Kommunikation innerhalb des Unternehmens

Was der “Management Overview” dem Management sagt:



Kommunikation innerhalb des Unternehmens

"Wie verbessere ich die Softwaresicherheit in meinem Unternehmen?"

Kommunikation innerhalb des Unternehmens

"Wie verbessere ich die Softwaresicherheit in meinem Unternehmen?"

(...ohne dass mich alle dafür hassen!)

Roadmap

- Integration von SCA in den Softwareentwicklungsprozess (SDL)
- Kommunikation innerhalb des Unternehmens
- Umgang mit Findings
- Umgang mit Testfällen

Eine Methode für SCA-Rollouts in Organisationen

- 1. Unterstützung vom Management holen
- 2. Rahmen des initialen Scans bestimmen
- 3. Produkt auswählen
- 4. Durchführung und Analyse des Scans
- 5. Nachbereitung

Eine Methode für SCA-Rollouts in Organisationen

1. Unterstützung vom Management holen

- Das Management muss sich offen dazu bekennen
  - dass Softwaresicherheit jetzt Priorität hat
  - dass der Softwareentwicklungsprozess verbessert werden muss
  - dass für das Vorhaben zusätzliche Ressourcen zur Verfügung gestellt werden

Eine Methode für SCA-Rollouts in Organisationen

2. Rahmen des initialen Scans bestimmen

- Eine besonders kritische Anwendung auswählen
  - Anwendung prozessiert kritische Daten
  - Anwendung wird von vielen Benutzern verwendet
  - Anwendung wird von nicht-vertrauenswürdigen Benutzern verwendet
- Starte mit überschaubarer Code-Menge
  - Nicht zu viel Code (--> überschaubare Anzahl von Findings)
  - Nicht zu wenig Code (--> wahrscheinlich keine repräsentative Aussage möglich)
  - 50.000 - 100.000 Lines of Code

Eine Methode für SCA-Rollouts in Organisationen

2. Rahmen des initialen Scans bestimmen

- Anwendung sollte möglichst viele Technologien beinhalten, z.B.
  - verschiedene Web Frameworks
  - Frontend Coding und Backend Coding
  - Datenbank-Abfragen (möglichst read, write und delete)
  - Dateien-Verwaltung (upload, download, copy/move, creation, deletion)
  - Web Services
  - B2B, B2C
  - ...

Eine Methode für SCA-Rollouts in Organisationen

3. Produkt auswählen

- Welche Technologien werden gescannt?
  - Programmiersprachen (Java, ABAP, PHP, .NET, ...)
  - Programmierbibliotheken (Struts, ASPX, SAP, ...)
- Kann das Produkt in vorhandene SDL integriert werden?
  - Unterstützung für vorhandenen Bug Tracker
  - Finding-Unterdrückung
  - "Code-Firewall"-Vorgehen
    - Scans sind verpflichtend bevor Code produktiv geht
    - Code mit kritischen Findings geht nicht produktiv
- Für den Anfang: "Managed Service" von SCA-Produkt

Eine Methode für SCA-Rollouts in Organisationen

3. Produkt auswählen

- Für den Anfang: “Managed Service” von SCA-Produkt
  - Produkt-Lizenz für begrenzte Anzahl von Codezeilen
  - Produkt-Support
  - Erfahrener Berater macht Roll-out (wichtig)

Eine Methode für SCA-Rollouts in Organisationen

4. Durchführung und Analyse des Scans

- Workshop durchführen
  - Teilnehmer
    - Berater mit viel Erfahrung mit dem ausgewählten SCA-Produkt
    - Mindestens einen interessierten Senior-Entwickler
    - Person von Quality-Assurance
    - Projektleiter des ausgewählten Projekts
    - Optional: Mitarbeiter der Fachabteilung, die das ausgewählte Projekt nutzen (werden)
  - Dauer des Workshops: 2-3 Tage

Eine Methode für SCA-Rollouts in Organisationen

4. Durchführung und Analyse des Scans

- Im Workshop scannt der Berater den Quellcode live
  - Alle Testfälle anschalten
  - Scan live am Beamber durchführen und gemeinsam in der Gruppe durch einzelne Findings laufen
    - Entscheidung in Gruppe: “Was wird mit einzelnen Findings gemacht”?

Eine Methode für SCA-Rollouts in Organisationen

4. Durchführung und Analyse des Scans

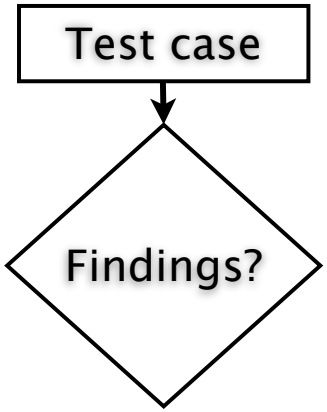
- Vier mögliche Entscheidungen pro Finding:
  1. Risikovermeidung (Code permanent löschen oder Schwachstelle schließen)
     - Als Action-Item an Entwickler geben
  2. Risikoreduktion (Schwachstelle abmildern)
     - Firewall-Regeln erstellen, so dass Angriff schwieriger wird
  3. Risikoverteilung (Vertraglich regeln oder Versicherung)
  4. Selbstbehalt
     - Findings in künftigen Scans unterdrücken
     - Dokumentieren, warum diese Entscheidung getroffen wurde

Eine Methode für SCA-Rollouts in Organisationen

4. Durchführung und Analyse des Scans

- Welche Testfälle sollen künftig verwendet werden? Hängt ab ob ein Testfall
  - überhaupt Findings hatte
  - Findings im produktiven Quellcode hatte
  - Findings ausgibt, die mehrheitlich auf tatsächliche Schwachstellen hingewiesen hat

Eine Methode für SCA-Rollouts in Organisationen



4. Durchführung und Analyse des Scans

- Welche Testfälle sollen künftig verwendet werden? Hängt ab ob ein Testfall
  - überhaupt Findings hatte
  - Findings im produktiven Quellcode hatte
  - Findings ausgibt, die mehrheitlich auf tatsächliche Schwachstellen hingewiesen hat

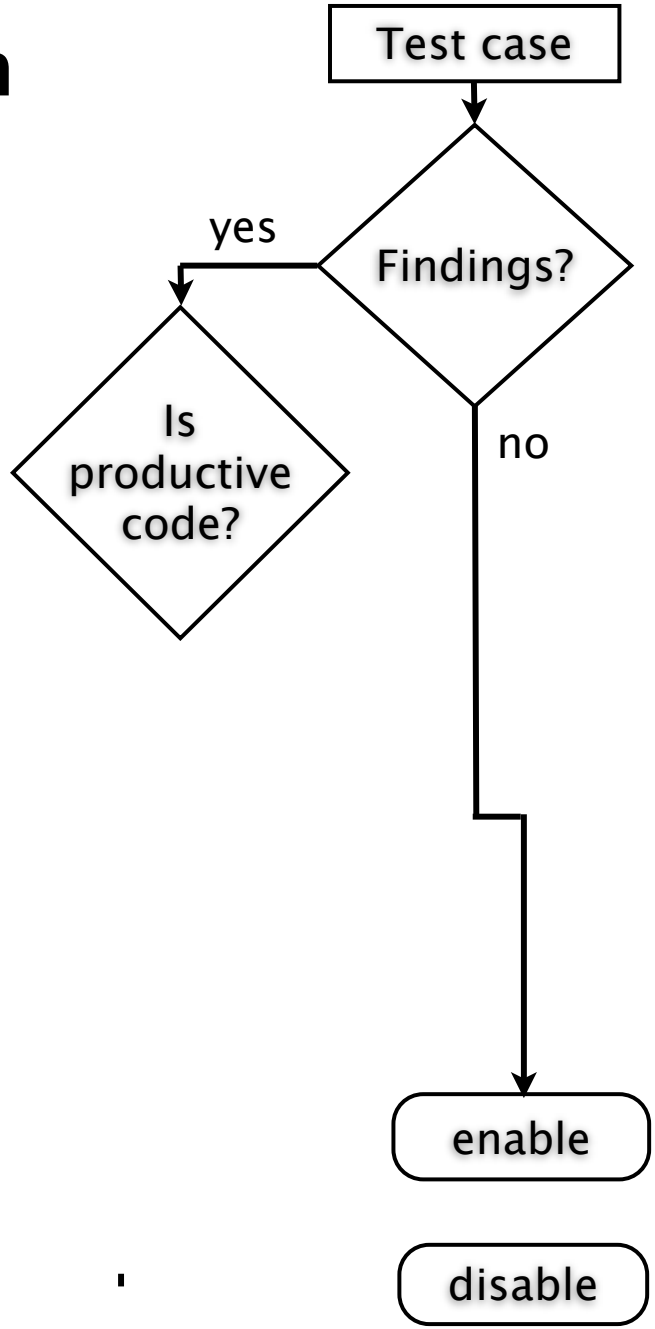
enable

disable

Eine Methode für SCA-Rollouts in Organisationen

4. Durchführung und Analyse des Scans

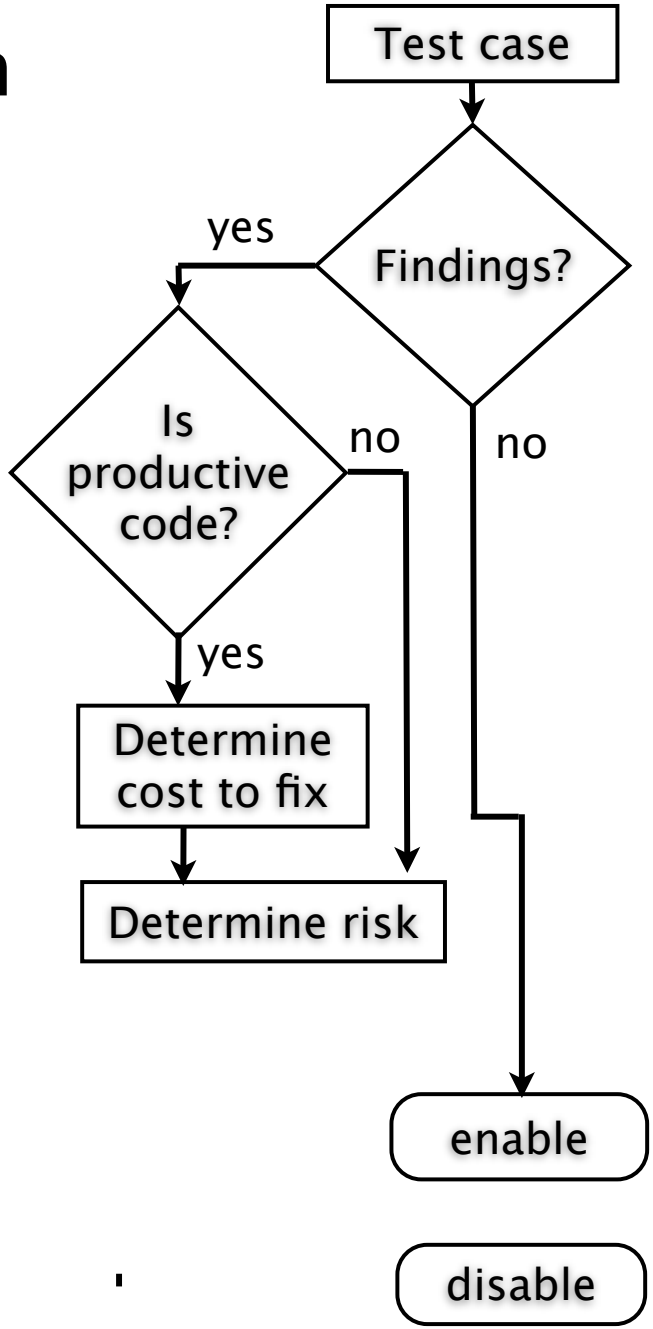
- Welche Testfälle sollen künftig verwendet werden? Hängt ab ob ein Testfall
  - überhaupt Findings hatte
  - Findings im produktiven Quellcode hatte
  - Findings ausgibt, die mehrheitlich auf tatsächliche Schwachstellen hingewiesen hat



Eine Methode für SCA-Rollouts in Organisationen

4. Durchführung und Analyse des Scans

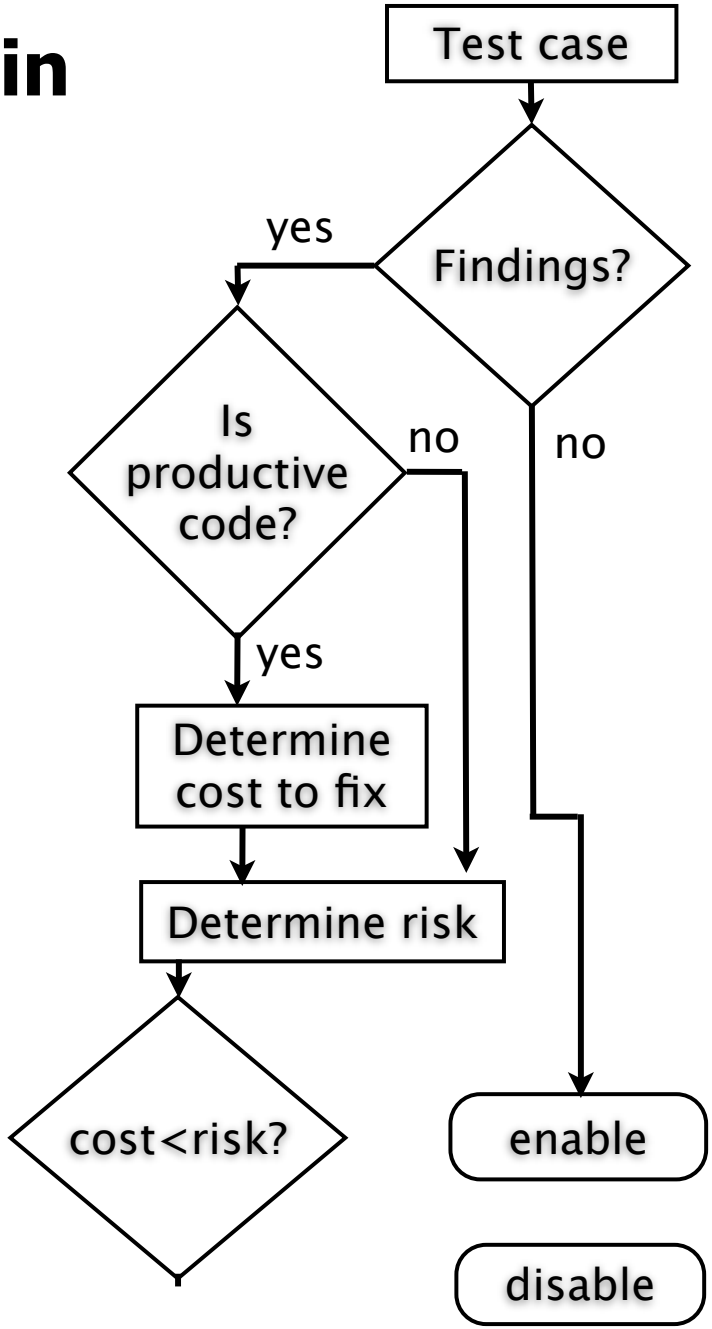
- Welche Testfälle sollen künftig verwendet werden? Hängt ab ob ein Testfall
  - überhaupt Findings hatte
  - Findings im produktiven Quellcode hatte
  - Findings ausgibt, die mehrheitlich auf tatsächliche Schwachstellen hingewiesen hat



Eine Methode für SCA-Rollouts in Organisationen

4. Durchführung und Analyse des Scans

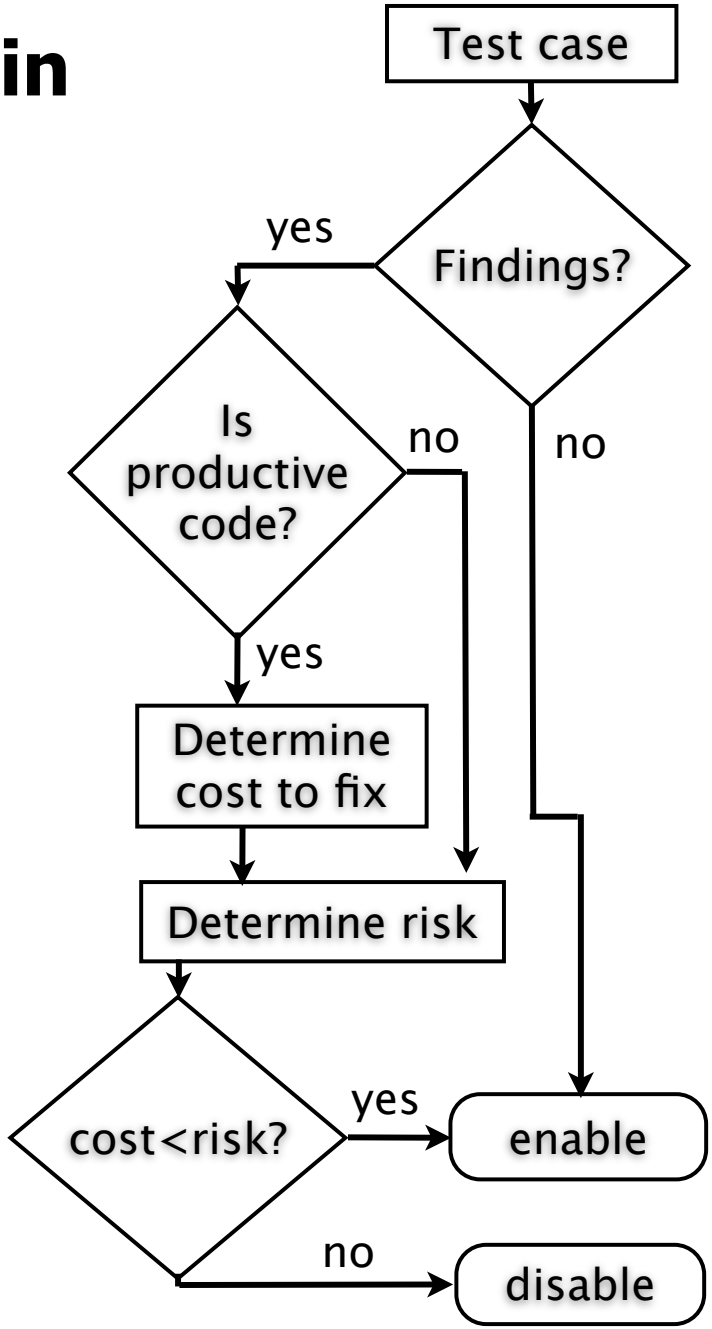
- Welche Testfälle sollen künftig verwendet werden? Hängt ab ob ein Testfall
  - überhaupt Findings hatte
  - Findings im produktiven Quellcode hatte
  - Findings ausgibt, die mehrheitlich auf tatsächliche Schwachstellen hingewiesen hat



Eine Methode für SCA-Rollouts in Organisationen

4. Durchführung und Analyse des Scans

- Welche Testfälle sollen künftig verwendet werden? Hängt ab ob ein Testfall
  - überhaupt Findings hatte
  - Findings im produktiven Quellcode hatte
  - Findings ausgibt, die mehrheitlich auf tatsächliche Schwachstellen hingewiesen hat



Eine Methode für SCA-Rollouts in Organisationen

4. Durchführung und Analyse des Scans

- Weitere Hinweise zum Workshop
  - Vortragende des Workshops muss sehr erfahren sein
  - Teilnehmer werden SCA Produkt kritisch hinterfragen (bestenfalls)
  - Auf kritische Diskussion gefasst sein

Eine Methode für SCA-Rollouts in Organisationen

5. Nachbereitung

- Kurzfristige Aufgaben
  - Die im Workshop verteilten Aufgaben abarbeiten
  - Liste mit Testfällen erstellen, die künftig verwendet werden sollen

- Mittelfristige Aufgaben
  - Aktuell laufende Entwicklungsprojekte identifizieren und dort SCA einführen
  - Automatisierte Sicherheitsprüfungen obligatorisch machen für externe Entwicklungen

Eine Methode für SCA-Rollouts in Organisationen

5. Nachbereitung

- Langfristige Aufgaben
  - SCA fest in Software Development Lifecycle integrieren
    - Sicherheitsprüfungen obligatorisch machen für interne Entwicklungsprojekte
    - Unternehmensspezifische Workshops über sichere Softwareentwicklung durchführen
      - Auf Schwachstellen in eigenen Anwendungen konzentrieren
    - Entwicklungsrichtlinien entsprechend erweitern und mit SCA-Tool prüfen
    - "Qualitygates" einführen, an denen der Code obligatorisch geprüft wird

Eine Methode für SCA-Rollouts in Organisationen

5. Nachbereitung

- Einige Wort über externe Entwicklung:
  - Wenn externe Entwickler eigenen Code scannen und Sie darauf vertrauen, ist "Segregation of Duty" verletzt
  - "Klar! Unser Code hat keine Schwachstellen!" ... weil
    - die wichtigen Testfälle abgeschaltet waren?
    - die Sicherheitslücken so verdreht wurden, dass das SCA-Tool sie nicht mehr entdeckt?
    - die Schwachstellen aus dem SCA-Report heraus manipuliert wurden?
    - ...
- Stichwort Hintertüren...

Zusammenfassung

- Statische Codeanalyse ist ein sehr guter Weg, gefährliche Muster im Quellcode Ihrer Anwendungen zu entdecken

- Beim Roll-Out von SCA-Tools ist Kommunikation wichtig
  - Menschen schreiben Software. Wenn man Software kritisiert, kritisiert man auch Menschen.
  - Die Interpretation von SCA-Reports braucht etwas Fingerspitzengefühl und Übung

- Lerne!
  - Achten Sie darauf, dass einen Wissenstransfer von dem SCA-Tool zu Ihren Entwicklern gibt
  - Werde besser mit jedem neuen Projekt

Danke für die Aufmerksamkeit!

Fragen, Kommentare, Diskussion!

Sebastian Schinzel
sebastian.schinzel@virtualforge.com
https://twitter.com/seecurity