AVSAKNADEN AV KONTROLLER

By Martin Jartelius

WWW.OUTPOST24.COM

Outpost24

WHOAMI?

- Distributed systems development studies
- “Tabletop” security consultant
- IT forensics analyst
- Penetration tester
- Lead penetration tester
- CSO

Martin Jartelius

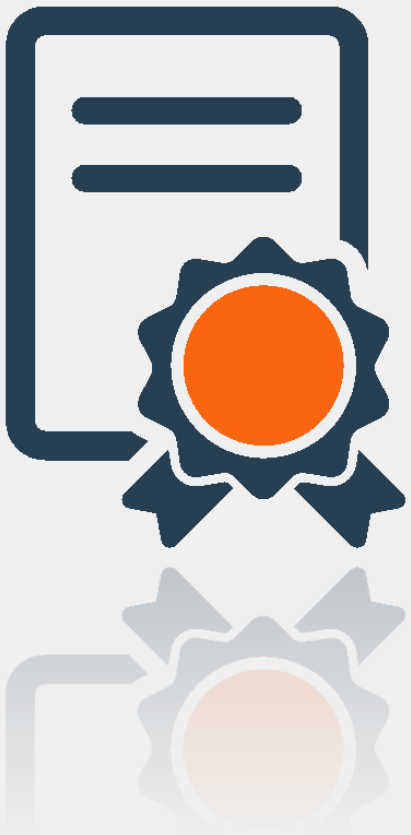
Interesting tasks

- “How do we map WASC to OWASP to CWE and CAPECS?”
- Explaining the difference between a false positive and an accepted risk
- Lots of research, exploitation, security testing and fresh air

ADVANTAGE

- Global company founded in Sweden
- Vulnerability management
- Founded in 2001
- Scans network components, servers and web applications
- Released SWAT in 2014 to target high profile web applications

ABOUT OUTPOST24



- “Best scanning engine” and “Ease of use” by Frost & Sullivan (2014)
- Over 56,000 vulnerability controls
- Most supported CVEs of all vendors
- Dun & Bradstreet AAA credit rating
- 2500+ customers around the globe

SWAT

State-of-the-art
vulnerability management
solution

Outpost24
Security experts

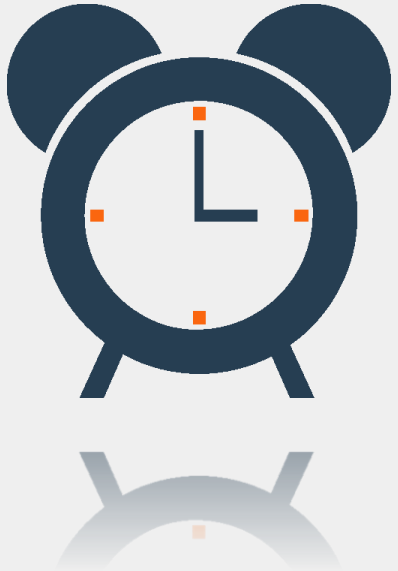
Best web application
security

BENEFITS OF SWAT

- Immediate deployment
- No false-positives
- Continuous monitoring
- Production-safe scanning
- Fully managed security services
- Analysis, verification, testing and false-positive elimination
- 24/7 technical support

40 MINUTE-SESSION

- Insecure direct object references
- Missing function level access control
- Invalidated redirects and forwards

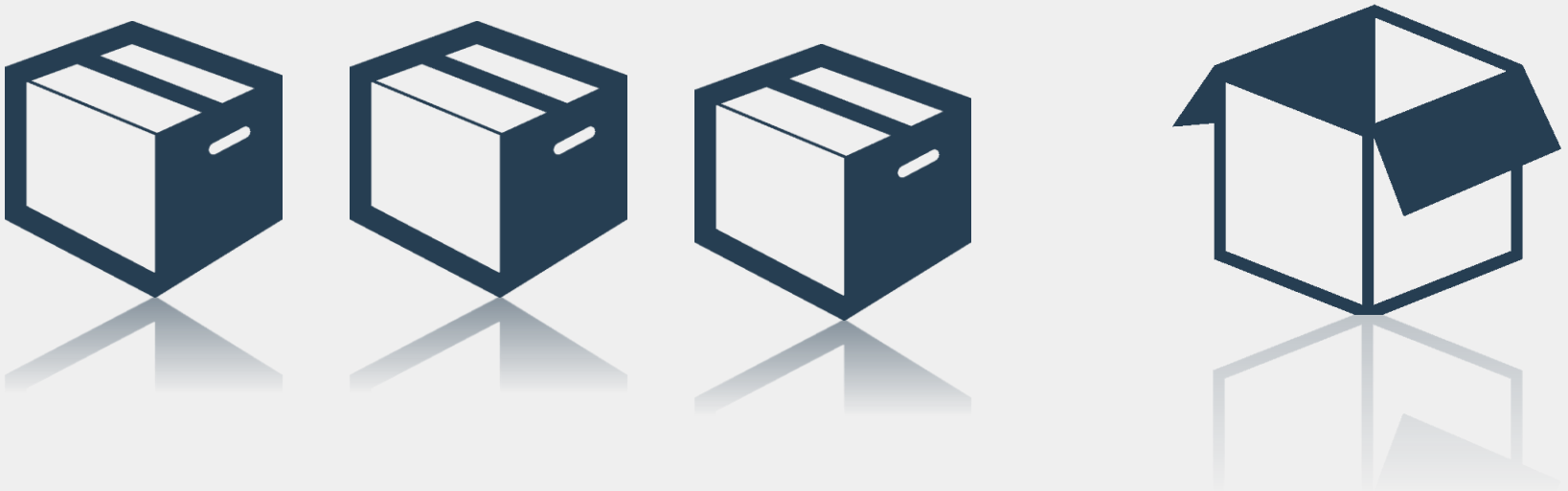


OR

"*Why stupid people ensure my job in IT security*"

INSECURE DIRECT OBJECT REFERENCES

“A direct object reference occurs when a developer exposes a reference to an internal implementation object, such as a file, directory, or database key. Without an access control check or other protection, attackers can manipulate these references to access unauthorized data.” – OWASP TOP 10 2013



INSECURE DIRECT OBJECT REFERENCES

In real life this is rather obvious

"Pick a number between 1 and 10"
"Ok… 11!"

Or

"You can read the first, second and tenth bank statement on my desk"

"Ok, then please give me the fourth"

INSECURE DIRECT OBJECT REFERENCES

- User is intended to be able to access the functionality
- A passed value is used to access objects
- Access restrictions are not applied to ensure the user is authorized to access said objects

http://example.com/readmessage.php?message=15

http://example2.com/edituser.php?id=10

INSECURE DIRECT OBJECT REFERENCES

Ease of detection?

Manual analysis	Easy
Automatic detection	**Almost impossible**

Testing for this problem

Identify any references in the applications to identify sensitive information

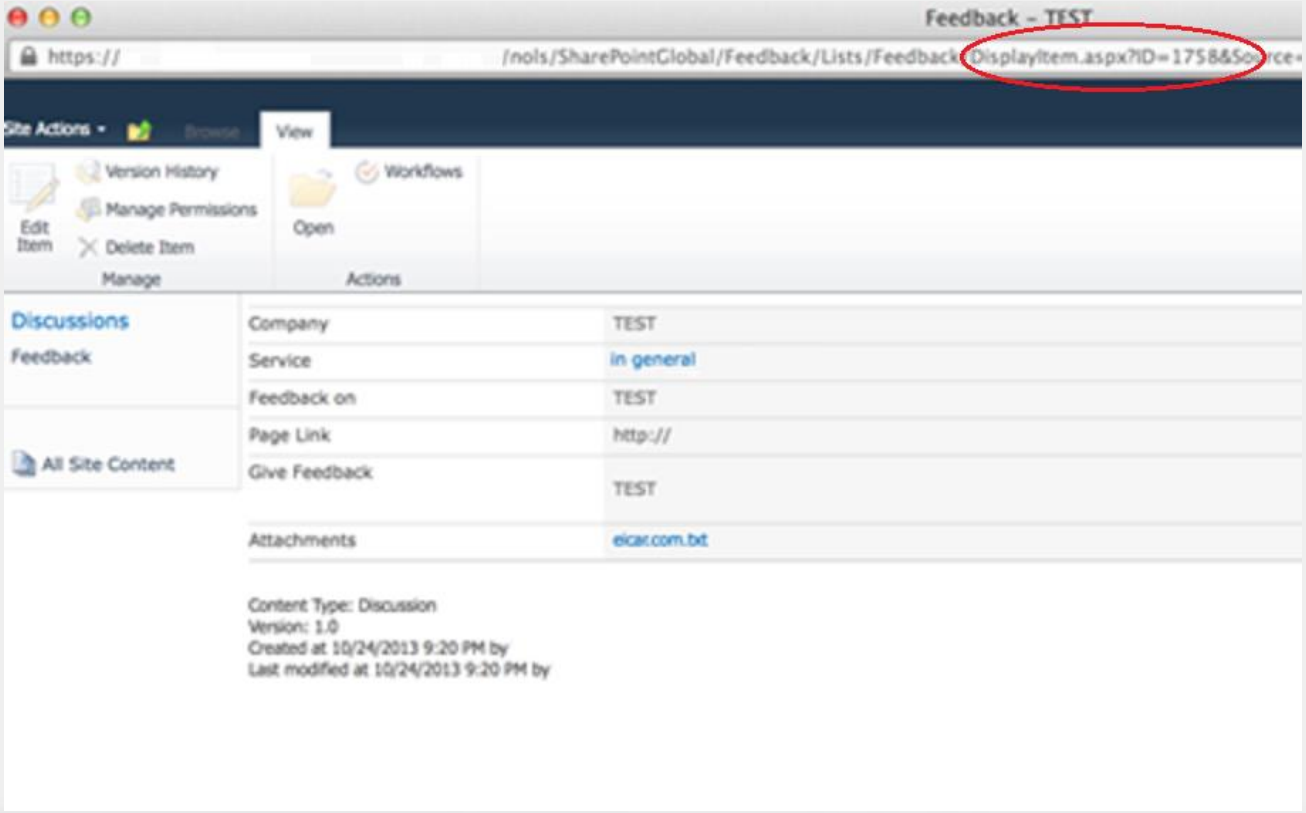
Verify if they are relative or direct references

Attempt to alter the intended behavior by passing other references

Risks

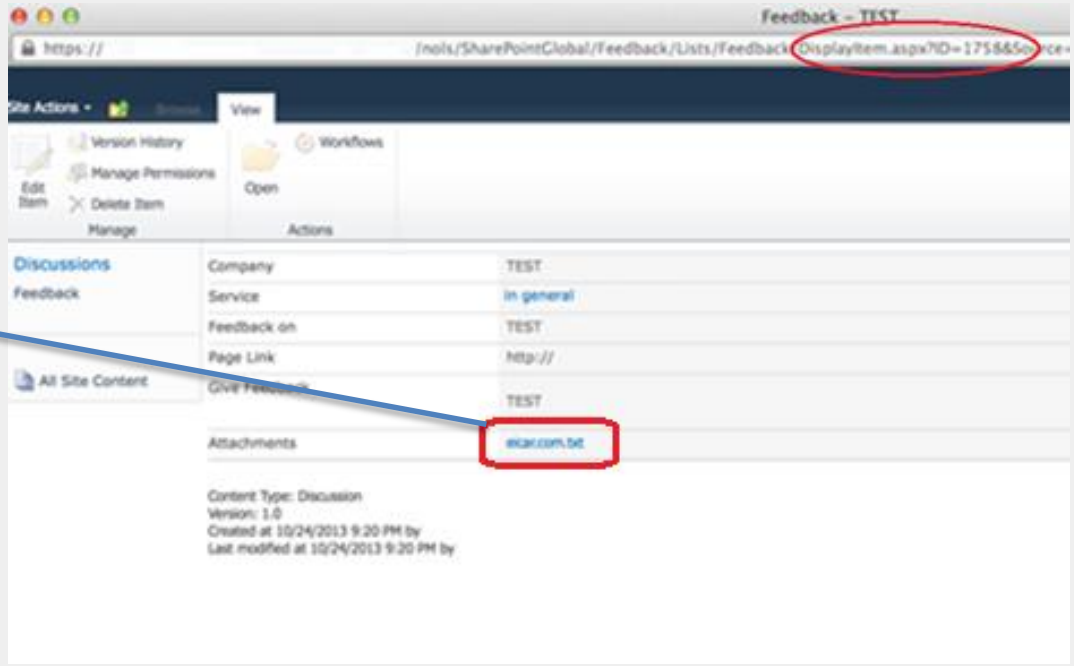
Total loss of confidentiality

Combination of vulnerabilities follow the 1+1=3 logic



INSECURE DIRECT OBJECT REFERENCE

EICAR.COM.TXT



INSECURE DIRECT OBJECT REFERENCE

Persistent XSS possible also

Could only hit the user who uploaded it

Combined with Insecure Direct Object References we get what?

A way better, higher impact Cross Site Scripting!

We also get to read everyone's support cases, but well…

INSECURE DIRECT OBJECT REFERENCES

Ok, that's a support module.

Lets up the stakes – SCADA

Yes.

Power plants, traffic lights, baby monitors and more…

Time to get to look at HoneyWell Falcon XL WEB

CVE-2014-2717

Well suited for todays talk - it is an almost "Top Ten Complete" device

INSECURE DIRECT OBJECT REFERENCES

Ok, that's a support module.

Let's up the stakes – SCADA

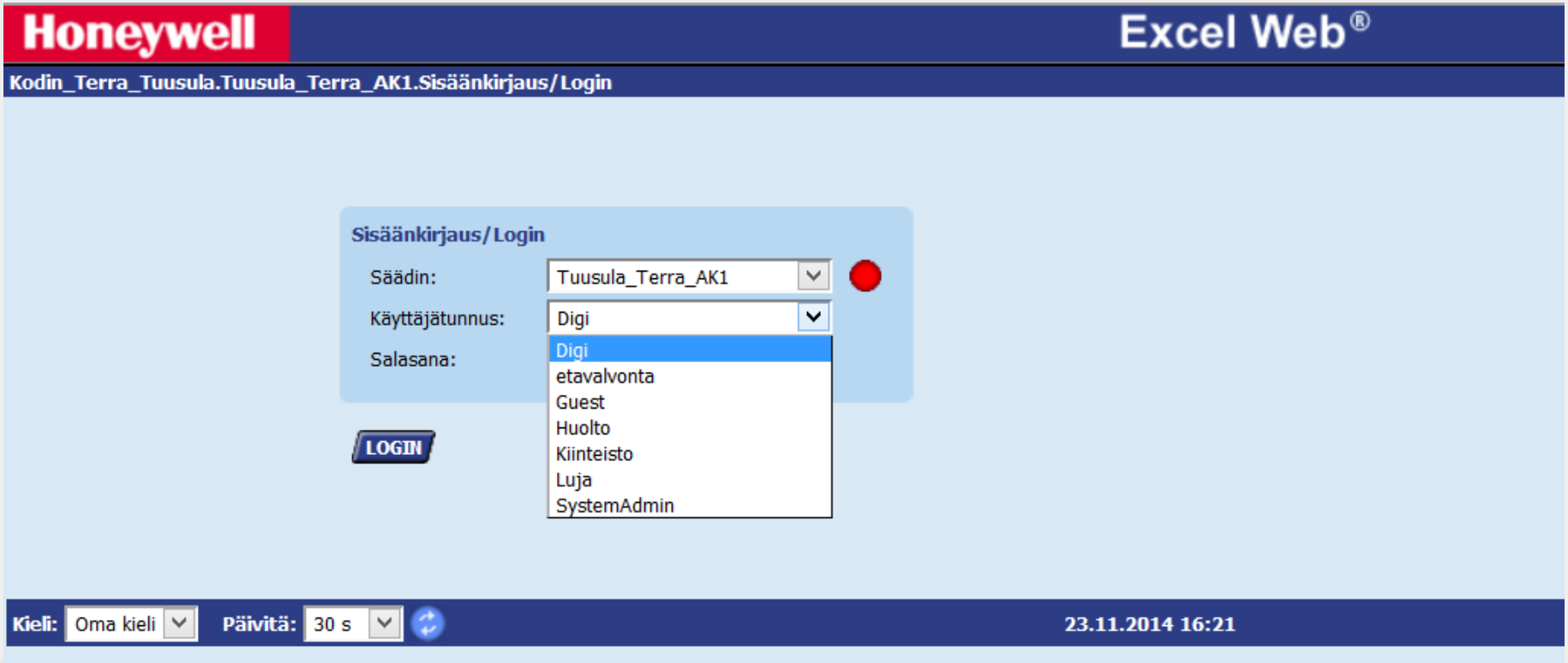
Yes.

Power plants, traffic lights, waste water, baby monitors and more

Time to get a look at The HoneyWell Falcon XL WEB

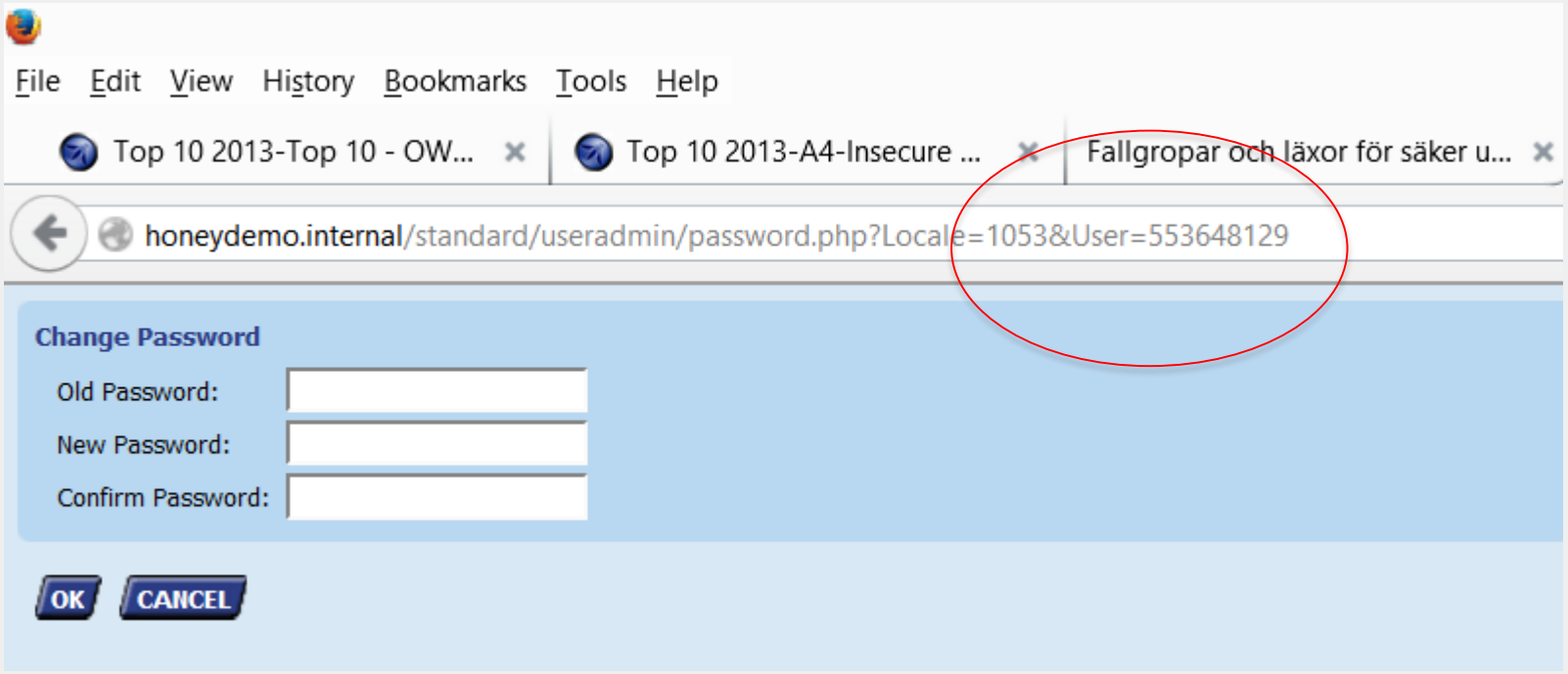
CVE-2014-2717

Well suited for today's talk - it is almost a "Top 10 complete" device



INSECURE DIRECT OBJECT REFERENCE

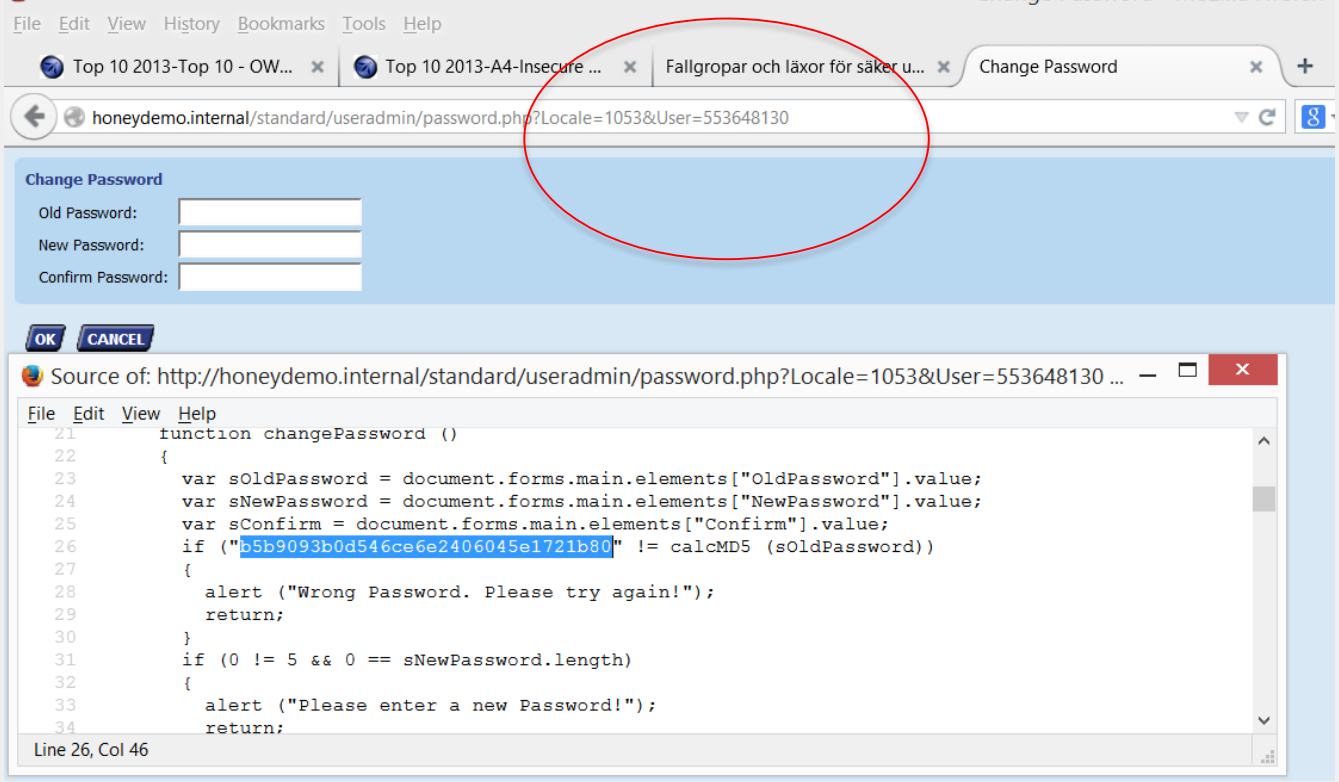
Guest – guest anyone?



INSECURE DIRECT OBJECT REFERENCE

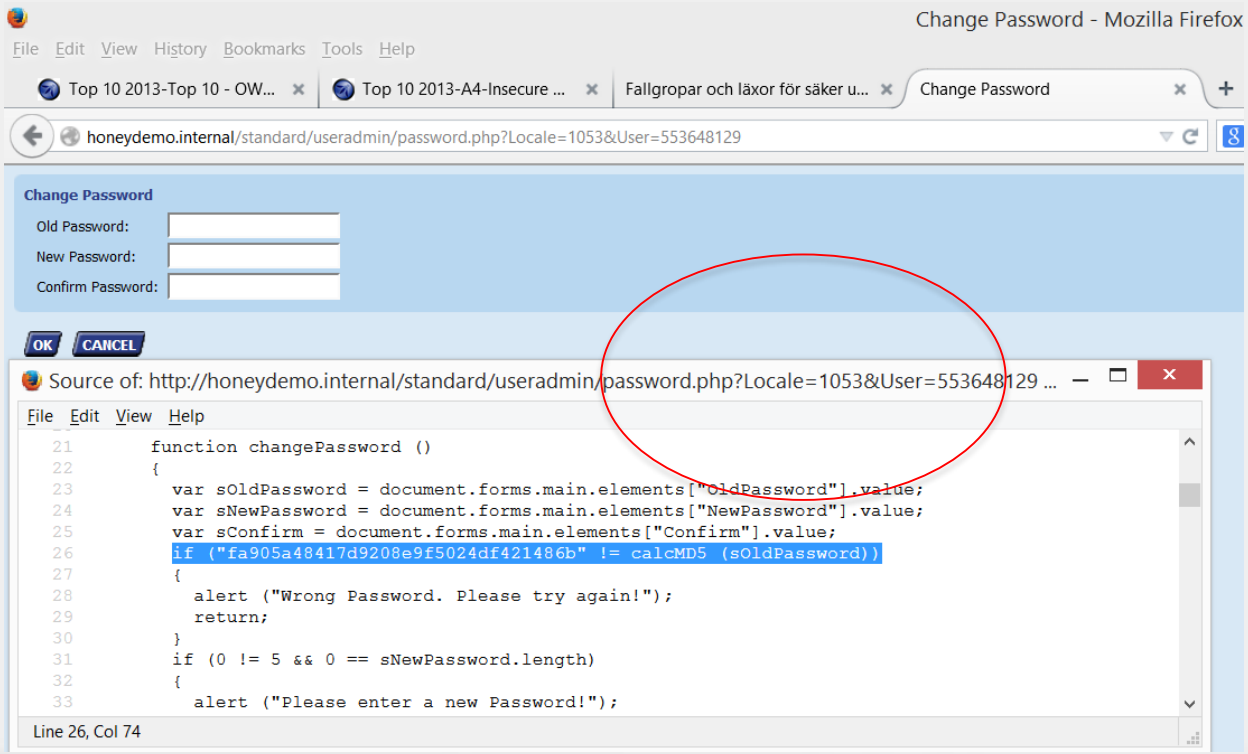
Good start – We have a user ID in the URL

/standard/useradmin/password.php?Locale=1053&User=[USER ID]



INSECURE DIRECT OBJECT REFERENCE

AND we have an information disclosure – the MD5 of the password (30 is guest)



INSECURE DIRECT OBJECT REFERENCE

AND we have an information disclosure – 29 is an administrator

INSECURE DIRECT OBJECT REFERENCE

Let’s halt for a second and discuss something important

553648130 – That looks fairly large and random

System admin?

553648129

First user added by a customer to the system?

553648131

Session IDs are also non-random

And for a system event returning “Success” they return “4194561”

INSECURE DIRECT OBJECT REFERENCE

Sorry all – I'll derail and explain how to privilege escalate. Just to show how serious those things are.

(Image shows the system as used on Stuttgart Airport, credits to Honeywell CZ)

INSECURE DIRECT OBJECT REFERENCES

```
function onSessionCreated (sResult, sSessionID)
[code to check the PLC is ready (sResult), if so, continue]
…
var sUserName = document.forms.main.elements["LoginUserName"].value;
var sPassword = calcMD5 (document.forms.main.elements["LoginPassword"].value);
sPassword = calcMD5 (sSessionID + sUserName + sPassword);
sUserName = calcMD5 (sUserName);
document.forms.main.elements["LoginSessionID"].value = sSessionID;
document.forms.main.elements["LoginUserNameMD5"].value = sUserName;
document.forms.main.elements["LoginPasswordMD5"].value = sPassword;
submitCommand ("Login");
```

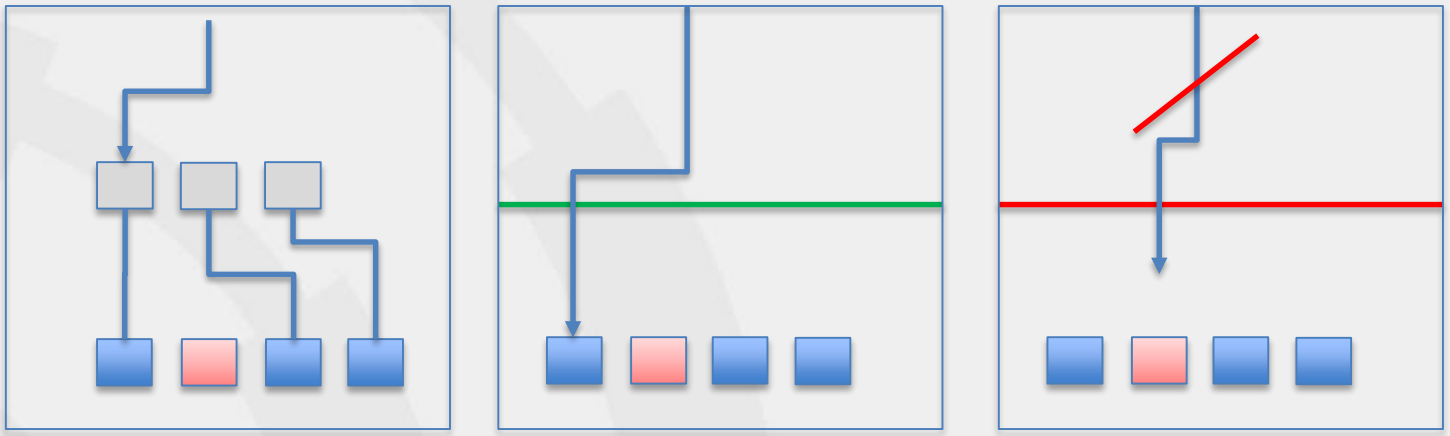
INSECURE DIRECT OBJECT REFERENCES

How to prevent this vulnerability

Preventing insecure direct object references requires selecting an approach for protecting each user accessible accessible object (e.g., object number, filename):

- Use per user or session indirect object references.
- Check access. Each use of a direct object reference from an untrusted source must include an access access control check to ensure the user is authorized for the requested object.

https://www.owasp.org/index.php/ESAPI



INSECURE DIRECT OBJECT REFERENCES

Further reading

OWASP
https://www.owasp.org/index.php/Top_10_2013-A4-Insecure_Direct_Object_References

MISSING FUNCTION LEVEL ACCESS CONTROL

“Applications do not always protect application functions properly. Sometimes, function level protection is managed via configuration, and the system is misconfigured. Developers must include the proper code checks, and sometimes they might forget to do so.

Detecting such flaws is easy. The hardest part is identifying which pages (URLs) or functions exist to attack” – OWASP TOP10 2013

MISSING FUNCTION LEVEL ACCESS CONTROL

This may seem odd in real life, but the best effort is visualized in these two images:

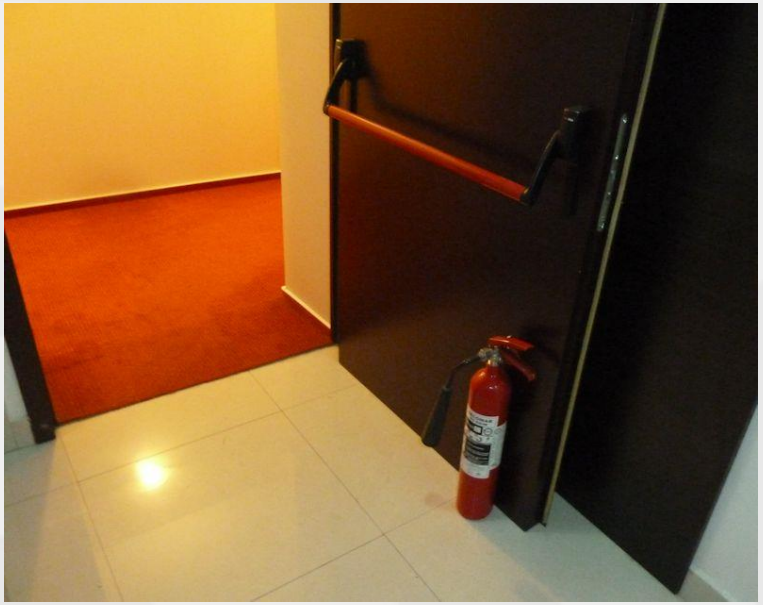
Imagine two stamps available at a bank:

OK to leave out

May require access control

MISSING FUNCTION LEVEL ACCESS CONTROL

This translates a bit strange to real life, but it is best visualized here:



MISSING FUNCTION LEVEL ACCESS CONTROL

“User” is not intended to have access to the functionality, nevertheless he can do it with no authorization or a very low level of authorization. Often, but far from always, this is combined with the last problem area.

Examples from the last week or so of testing systems:

/Shell/Statements/AccountBalance.aspx?account=[direct object reference]

/system/admin/network/diagnostics/ping?ip=[command injection]

/changepassword.php?id=[direct object reference]

/reboot.cgi

/install.php

/editUser.php?id=[direct object reference]

MISSING FUNCTION LEVEL ACCESS CONTROL

Ease of detection?

Manual analysis	Easy
Automatic detection	**Almost impossible**

Testing for this problem

Walk through your application and all calls

Determine which ones are privileged

Attempt to call the function from a lower privilege level

Risks

Total loss of confidentiality, integrity, availability

MISSING FUNCTION LEVEL ACCESS CONTROL

We will start with an example.

Remember the change password?

Yes, that's pre-authentication. That's a perfect example

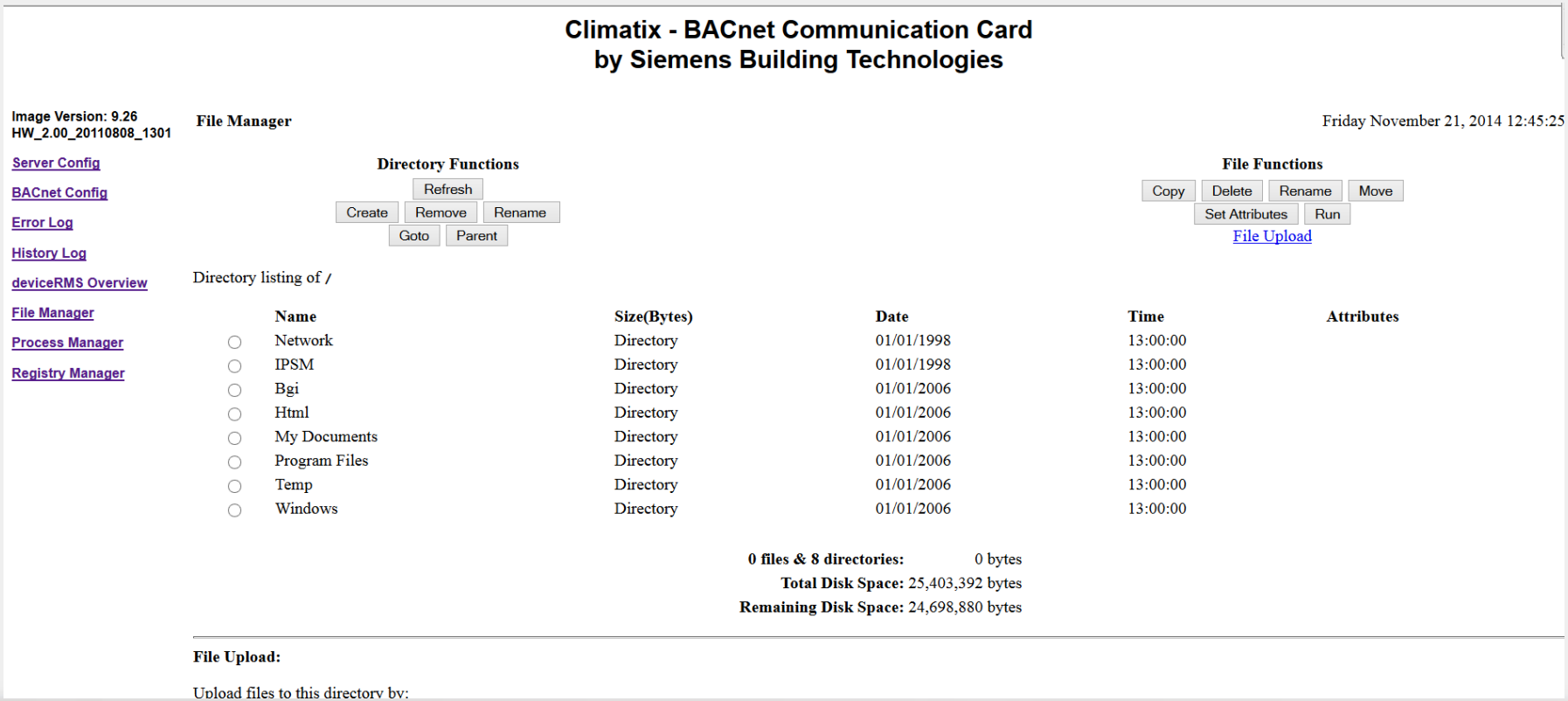
MISSING FUNCTION LEVEL ACCESS CONTROL

Targeting SCADA again – Climatix

Developed by Siemens

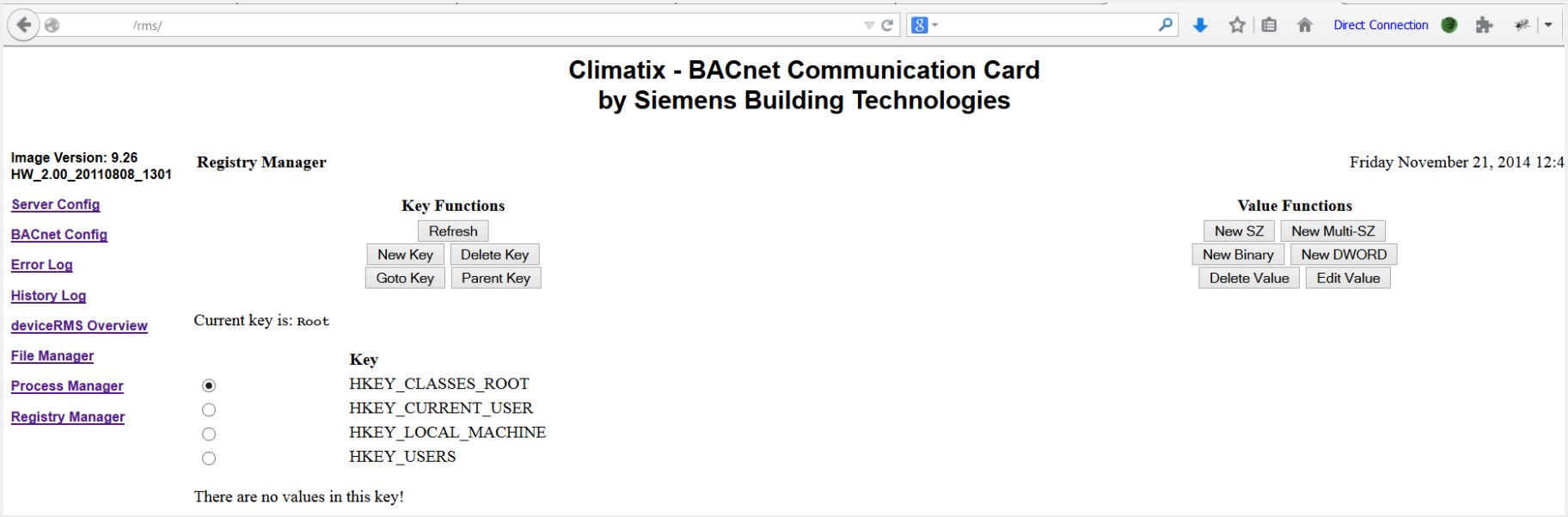
Used by many other vendors

A fellow researcher and I were researching SCADA and…



MISSING FUNCTION LEVEL ACCESS CONTROL

The awesome exploit? Writing /RMS/ after the hostname



MISSING FUNCTION LEVEL ACCESS CONTROL

And if upload and execute does not satisfy the lazy attackers need…

MISSING FUNCTION LEVEL ACCESS CONTROL

DLINK – TELCO

REDACTED AS FIX IS STILL UNDER IMPLEMENTATION

DLINK – TELCO

REDACTED AS FIX IS STILL UNDER IMPLEMENTATION

MISSING FUNCTION LEVEL ACCESS CONTROL

How do I prevent this?

- Build a security model for authorization as a module and invoke this this from the business functions
- Design a practical flow for granting access, ensuring its also easy to to audit
- Do not hard code
- Set to “Default Deny”
- Build the model around roles
- If the function is involved in a workflow, check to make sure the conditions are in the proper state to allow access.
- Hiding links and buttons to unauthorized functions does not provide provide protection

MISSING FUNCTION LEVEL ACCESS CONTROL

More reading

- https://www.owasp.org/index.php/Top_10_2013-A7-Missing_Function_Level_Access_Control

Pause – I'll be back

www.outpost24.com
MJ@OUTPOST24.COM
+46 708 47 43 15

Outpost24

WHOAMI?

Outpost24

Martin Jartelius
CSO

INVALIDATED REDIRECTS AND FORWARDS

“Applications frequently redirect users to other pages, or use internal forwards in a similar manner. Sometimes the target page is specified with an invalidated parameter, allowing attackers to choose the destination page.

Detecting unchecked redirects is easy. Look for redirects where you can set the full URL. Unchecked forwards are harder, because they target internal pages. “ – OWASP TOP 10 2013

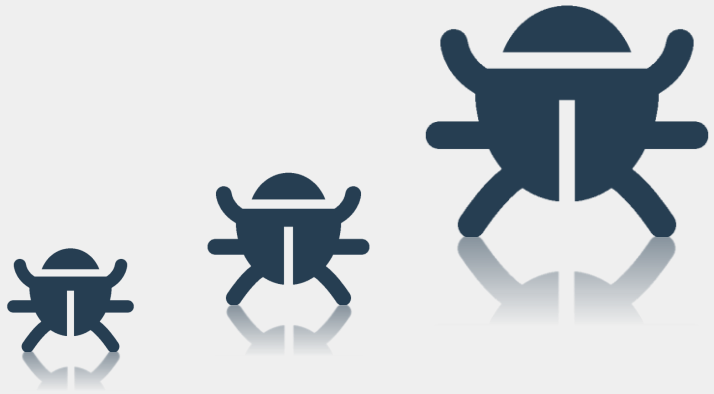
UNVALIDATED REDIRECTS AND FORWARDS

The most common “offensive” vulnerability according to our own statistics

Often considered as with minor impact. And one of my favorites!

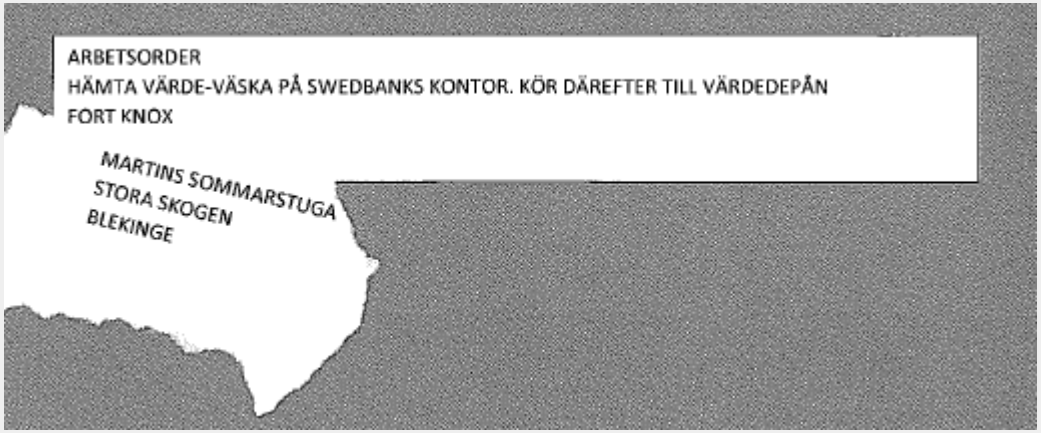
Manifests as (this is just a selection);

- 300-redirects
- IFRAME includes
- FRAME includes



Forwards is different, as they restrict attacks to the same page.

Forwards may help bypass access restrictions, and are also harder to test for.



UNVALIDATED REDIRECTS AND FORWARDS

In real life this is actually NOT that obvious, it also translates poorly

UNVALIDATED REDIRECTS AND FORWARDS

- **Scenario #1:** The application has a page called “redirect.jsp” which takes a single parameter named “url”. The attacker crafts a malicious URL that redirects users to a malicious site that performs phishing and installs malware.
  http://www.example.com/redirect.jsp?url=evil.com

- **Scenario #2:** The application uses forwards to route requests between different parts of the site. To facilitate this, some pages use a parameter to indicate where the user should be sent if a transaction is successful. In this case, the attacker crafts a URL that will pass the application’s access control check and then forwards the attacker to administrative functionality for which the attacker isn’t authorized.
  http://www.example.com/boring.jsp?fwd=admin.jsp

UNVALIDATED REDIRECTS AND FORWARDS

Ease of detection?

Manual analysis	Easy
Automatic detection	Easy

Testing for this problem

Identify URL-like values passed to the server
Pass URL values, also encoded, as parameters
If partial URLs are passed, attempt call to privileged components
Or use a security scanner to attempt this against every parameter

Risks

Broken access control
Phishing, scams and malware distribution lending trust of the initial landing URL

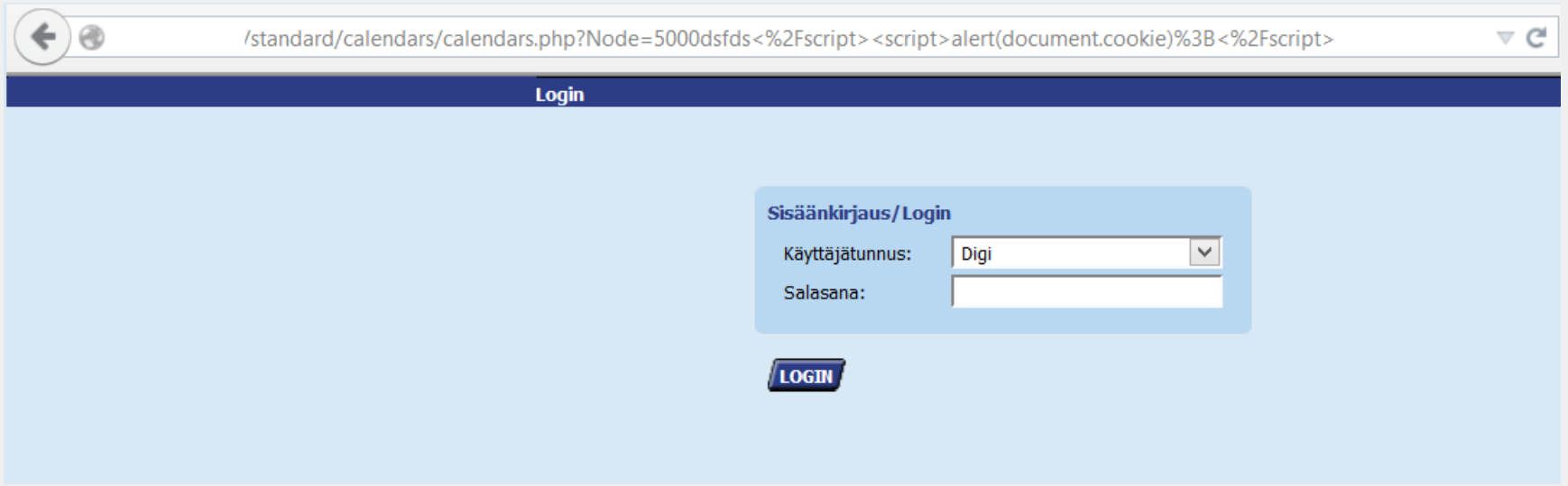
UNVALIDATED REDIRECTS AND FORWARDS

The most dangerous form – similar effect to CSRF

http://example.com/login.cgi?onsuccess=[account.cgi]

http://example.com/login.cgi?onsuccess=[setadmin.cgi?acc=MARTIN]

Main use:
Phishing, often in combination with for example minor XSS vulnerabilities



UNVALIDATED REDIRECTS AND FORWARDS

Well, obviously you need to have a redirect OR forward to be compete!
Honeywell Falcon revisited (a third time? Yes)

http://HOSTNAME/standard/calendars/calendars.php?Node=5000dsfds<%2Fscript><script>alert(docume
.cookie)%3B<%2Fscript>

UNVALIDATED REDIRECTS AND FORWARDS

Where can this be found?

Extremely often in authorization modules, and this is dangerous dangerous

Affects one of our national E-ID providers (edit: fixed for one

UNVALIDATED REDIRECTS AND FORWARDS

How can I prevent this?

- Simply avoid using redirects and forwards
- Don't involve user parameters in calculating the destination. Remember the indirect object references
- If destination parameters can't be avoided, ensure that the supplied value is valid, is valid, and authorized for the user

Applications can use ESAPI to override the sendRedirect() method to make sure all redirect destinations are safe.

Remember that also internal URLs are very dangerous if you accept GET-based parameters.

INVALIDATED REDIRECTS AND FORWARDS

More reading

https://www.owasp.org/index.php/Top_10_2013-A10-A10-Unvalidated_Redirects_and_Forwards

- CWE-601
- WASC-38
- OWASP ESAPI SecurityWrapperResponse sendRedirect() method

THANK YOU FOR LISTENING

mj@outpost24.com

Outpost24