Security 101

April 19, 2018 | Overview

Stacy Janes
Security Team, GENIVI Alliance

Software Security 101

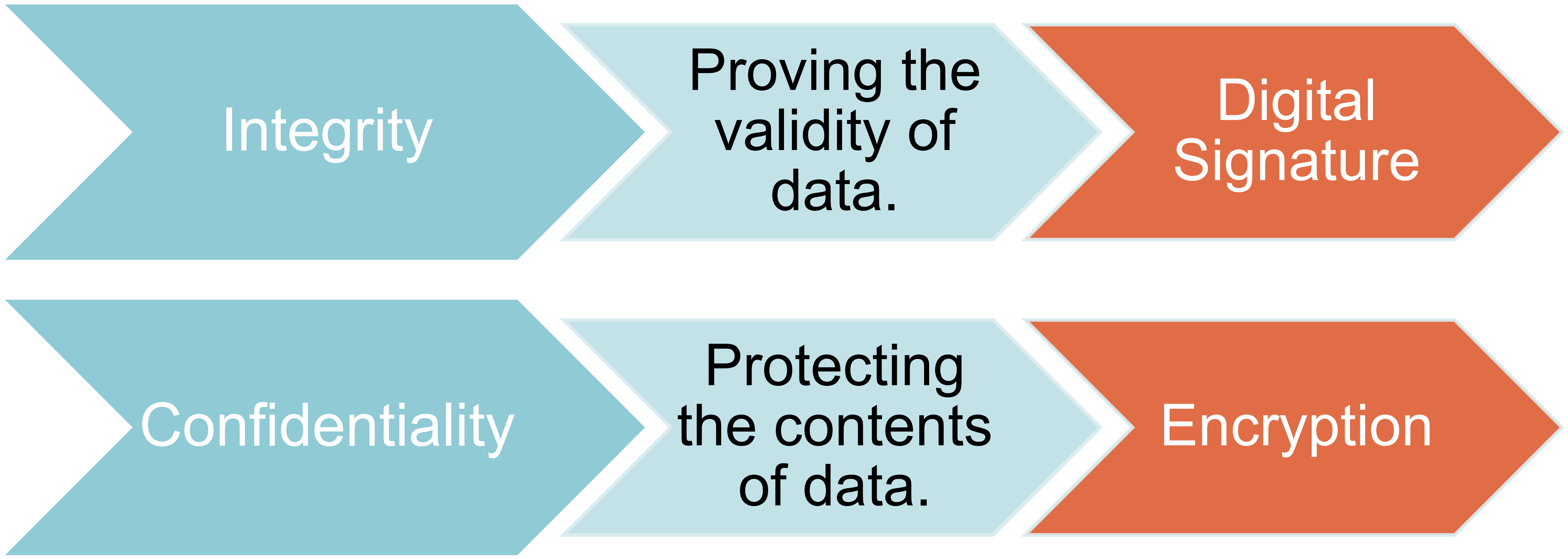
Stacy Janes

Chief Security Architect
Cloakware for Connected Transport at Irdeto

linkedin.com/in/stacy-911-janes stacy.janes@irdeto.com

Crypto 101

Integrity and Confidentiality


Integrity
Proving the validity of data.
Digital Signature
Confidentiality
Protecting the contents of data.
Encryption

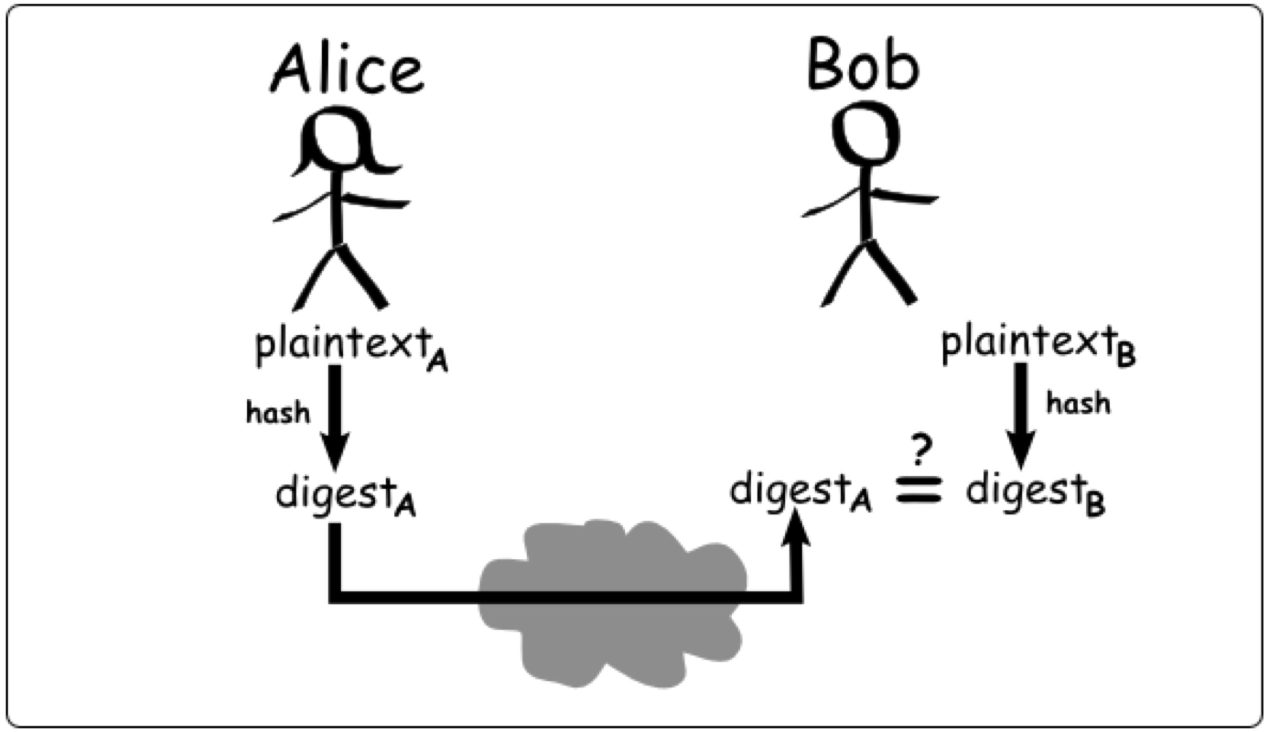
GENIVI®

Hashing

Unlike encryption, hashing is a "one way" function

A hash is used to check the validity of data. It does not protect data.

Passwords should be hashed, not encrypted when stored.

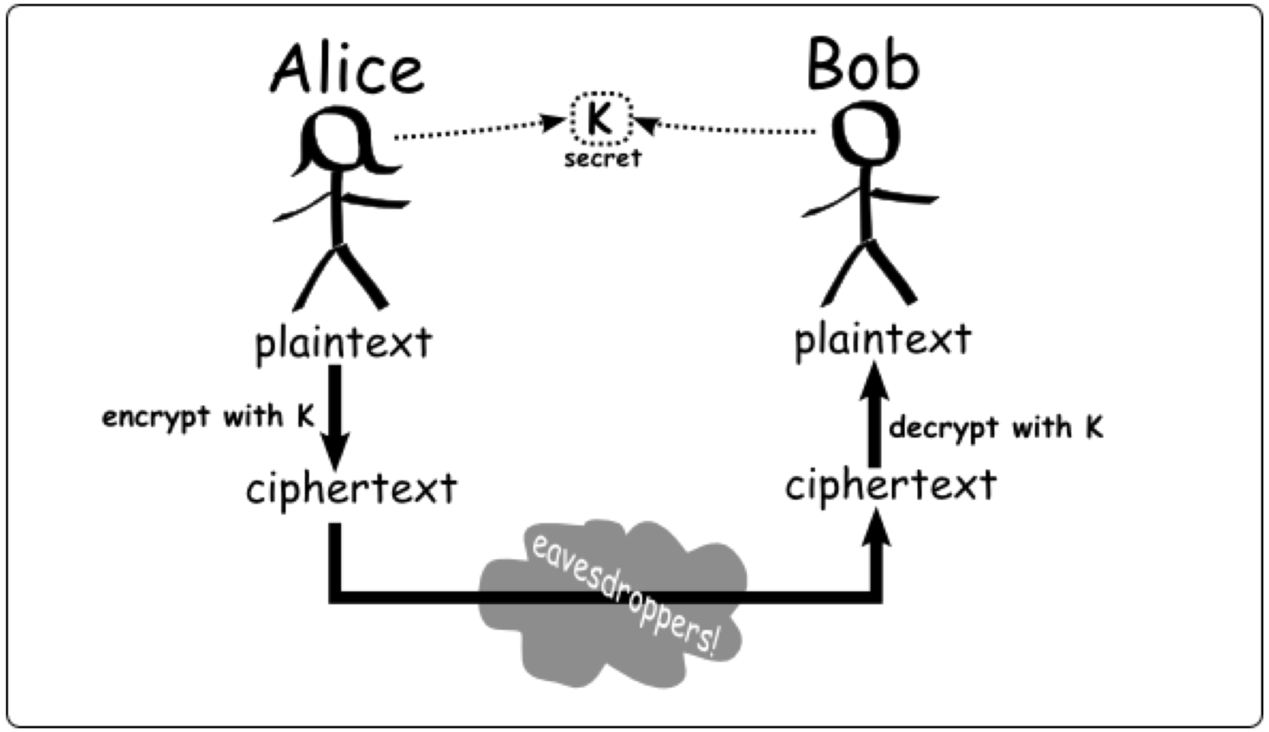


Encryption – Symmetric Key

Encryption and decryption done with the same key

Symmetric cryptography is fast (relative to Asymmetric)

Key management becomes cumbersome beyond a few actors.



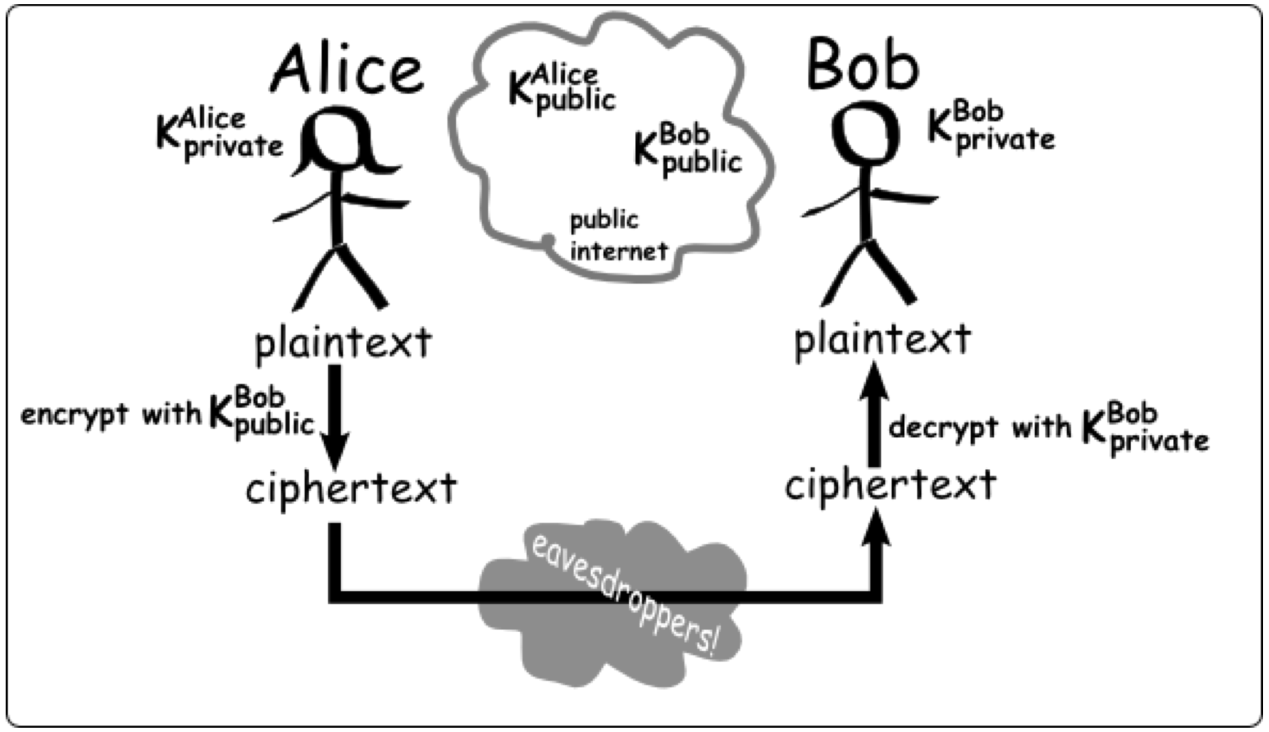
Encryption – Asymmetric Key

- ➢ Encryption with Public Key
- ➢ Decryption with Private Key

Asymmetric cryptography is slow(relative to Symmetric)

Private Keys are not shared

Public Keys can be shared with many actors. PKI enables this.



GENIVI®

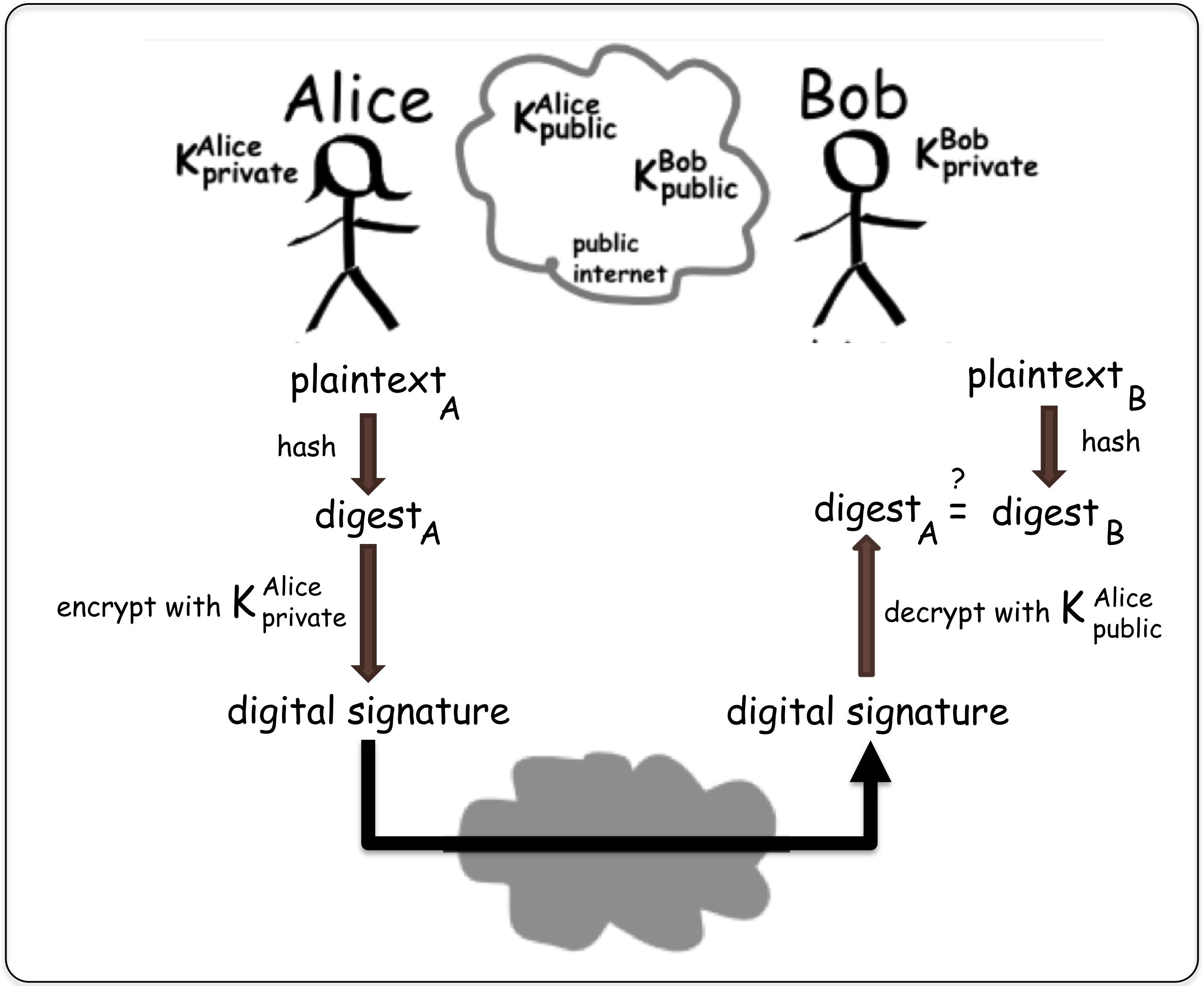
Digital Signature

Encrypted Hash

- Encrypt with Private Key (Sign)
- Decrypt with Public Key (Verify)

X.509 Certificate around Public Key for identity verification

Does not hide data



GENIVI®

Binary Hacking 101

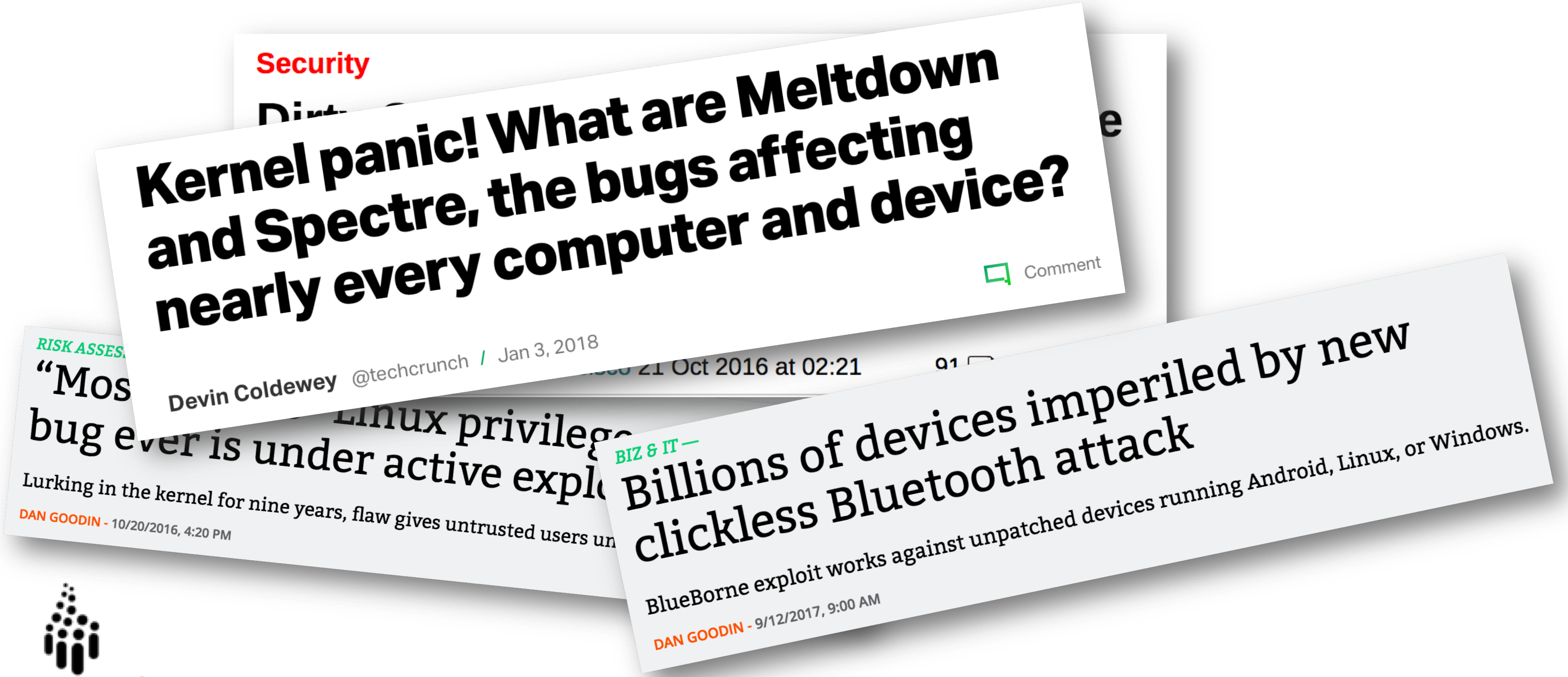
Privilege Escalation

Privilege escalation is the act of exploiting a bug, design flaw or configuration oversight in the OS or an application to gain elevated access to resources that are normally protected from the application or userid.

Kernel Exploitation: Exploiting vulnerabilities in the kernel in order to gain arbitrary code execution as root. Eg: DirtyCOW

Service Exploitation: Exploiting Linux services and configuration mistakes. Eg: wildcard injection.

Vulnerabilities



Security

Dirty C

21 Oct 2016 at 02:21

Kernel panic! What are Meltdown and Spectre, the bugs affecting nearly every computer and device?

Comment

Devin Coldewey @techcrunch / Jan 3, 2018

RISK ASSES

"Mos Linux privilege bug ever is under active expl

Lurking in the kernel for nine years, flaw gives untrusted users un

DAN GOODIN - 10/20/2016, 4:20 PM

BIZ & IT —

Billions of devices imperiled by new clickless Bluetooth attack

BlueBorne exploit works against unpatched devices running Android, Linux, or Windows.

DAN GOODIN - 9/12/2017, 9:00 AM

GENIVI®

“Defeating” Crypto – Easier to Bypass

Brute force is typically not a realistic attack

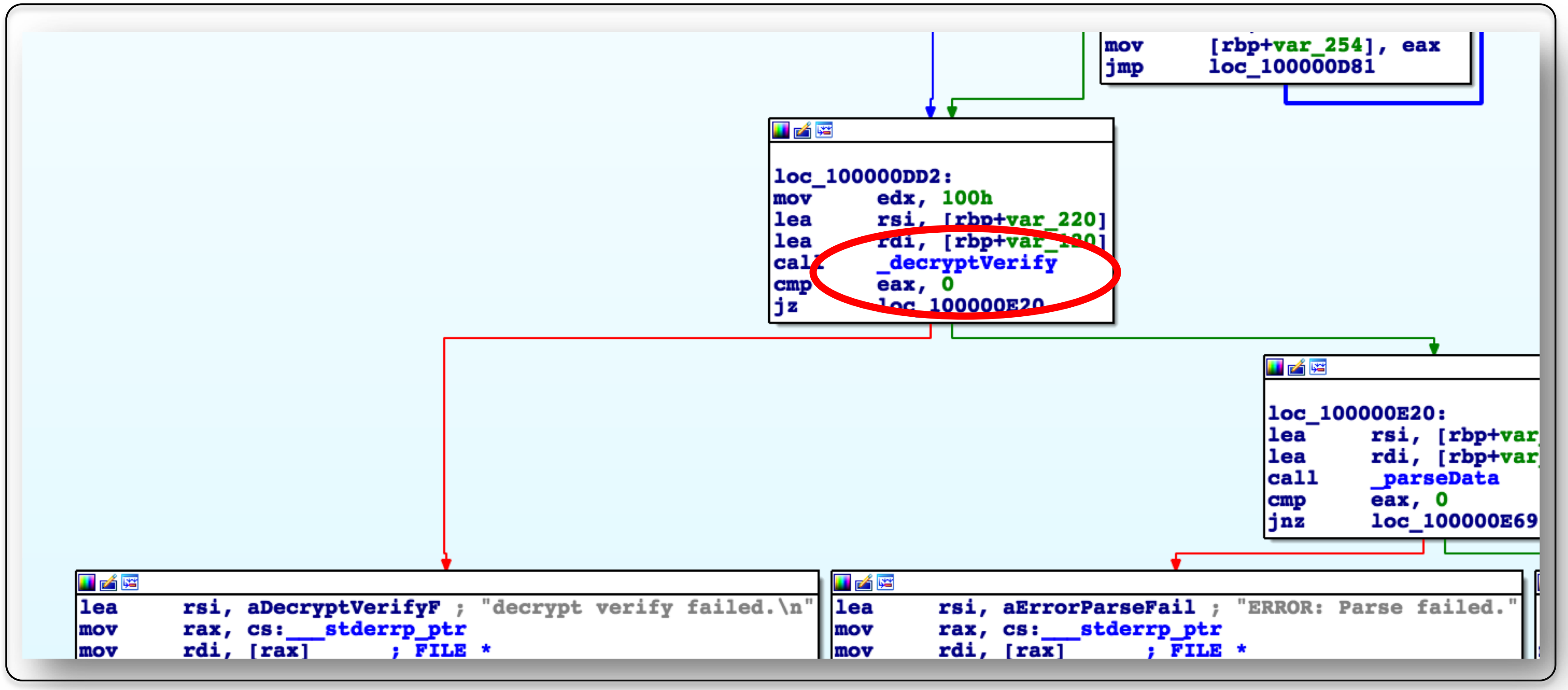
End point access opens up attack vectors

- Key lifting. Easy for software key if not properly protected
- Binary modification to “jam” logic branch for signature check
- Lifting clear data from memory after decryption
- Inserting malicious data to be signed/encrypted
- Shimming interfaces

“Lifting” Clear Data

Find the Decrypt function

Lift the clear data after decryption



```
mov     [rbp+var_254], eax
jmp     loc_100000D81
```

```
loc_100000DD2:
mov     edx, 100h
lea     rsi, [rbp+var_220]
lea     rdi, [rbp+var_120]
call    _decryptVerify
cmp     eax, 0
jz      loc_100000E20
```

```
loc_100000E20:
lea     rsi, [rbp+var
lea     rdi, [rbp+var
call    _parseData
cmp     eax, 0
jnz     loc_100000E69
```

```
lea     rsi, aDecryptVerifyF ; "decrypt verify failed.\n"
mov     rax, cs:___stderrp_ptr
mov     rdi, [rax]      ; FILE *
```

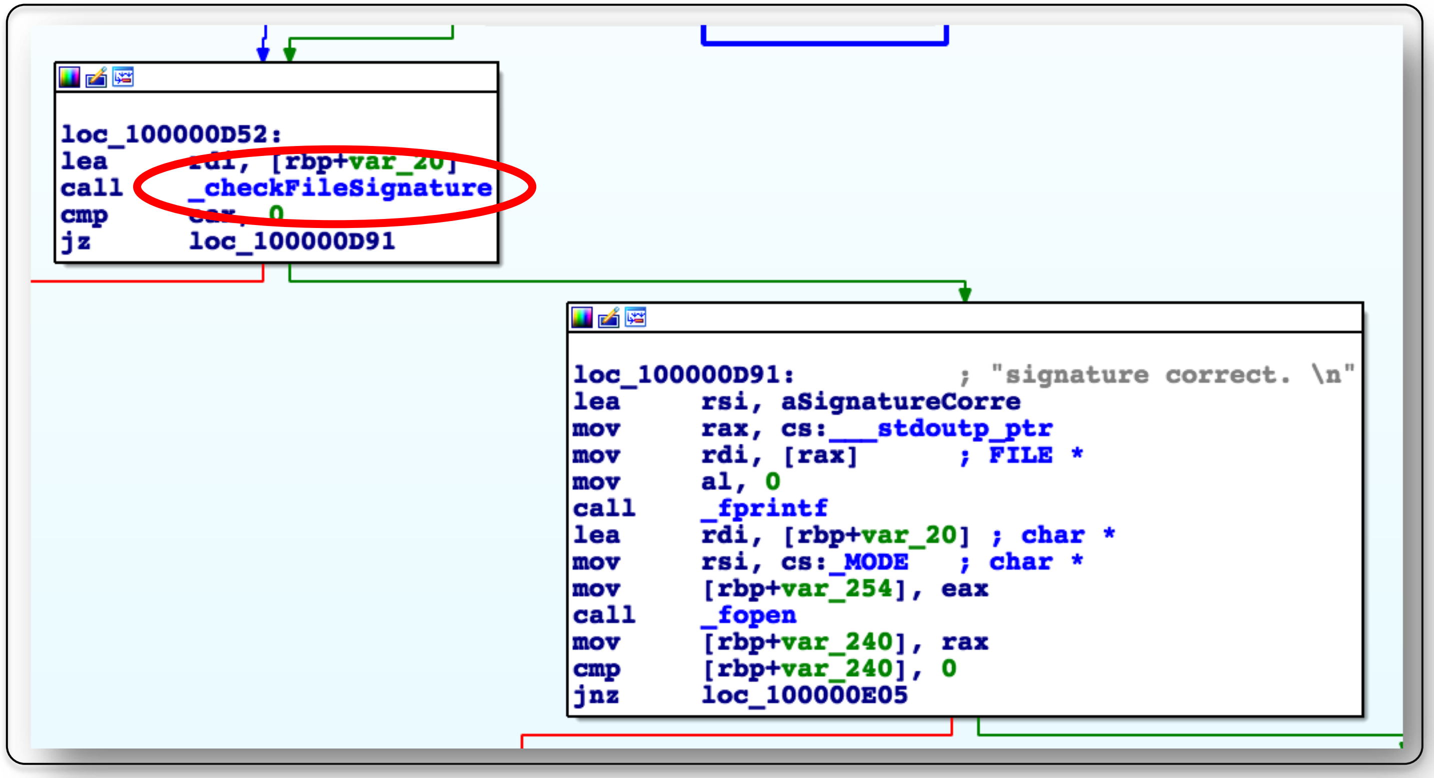
```
lea     rsi, aErrorParseFail ; "ERROR: Parse failed."
mov     rax, cs:___stderrp_ptr
mov     rdi, [rax]      ; FILE *
```

Branch “Jamming”

Let software verify signature

Find branch that checks return code

Reverse comparison opcode to allow invalid signature to pass



```
loc_100000D52:
lea     rdi, [rbp+var_20]
call    _checkFileSignature
cmp     eax, 0
jz      loc_100000D91
```

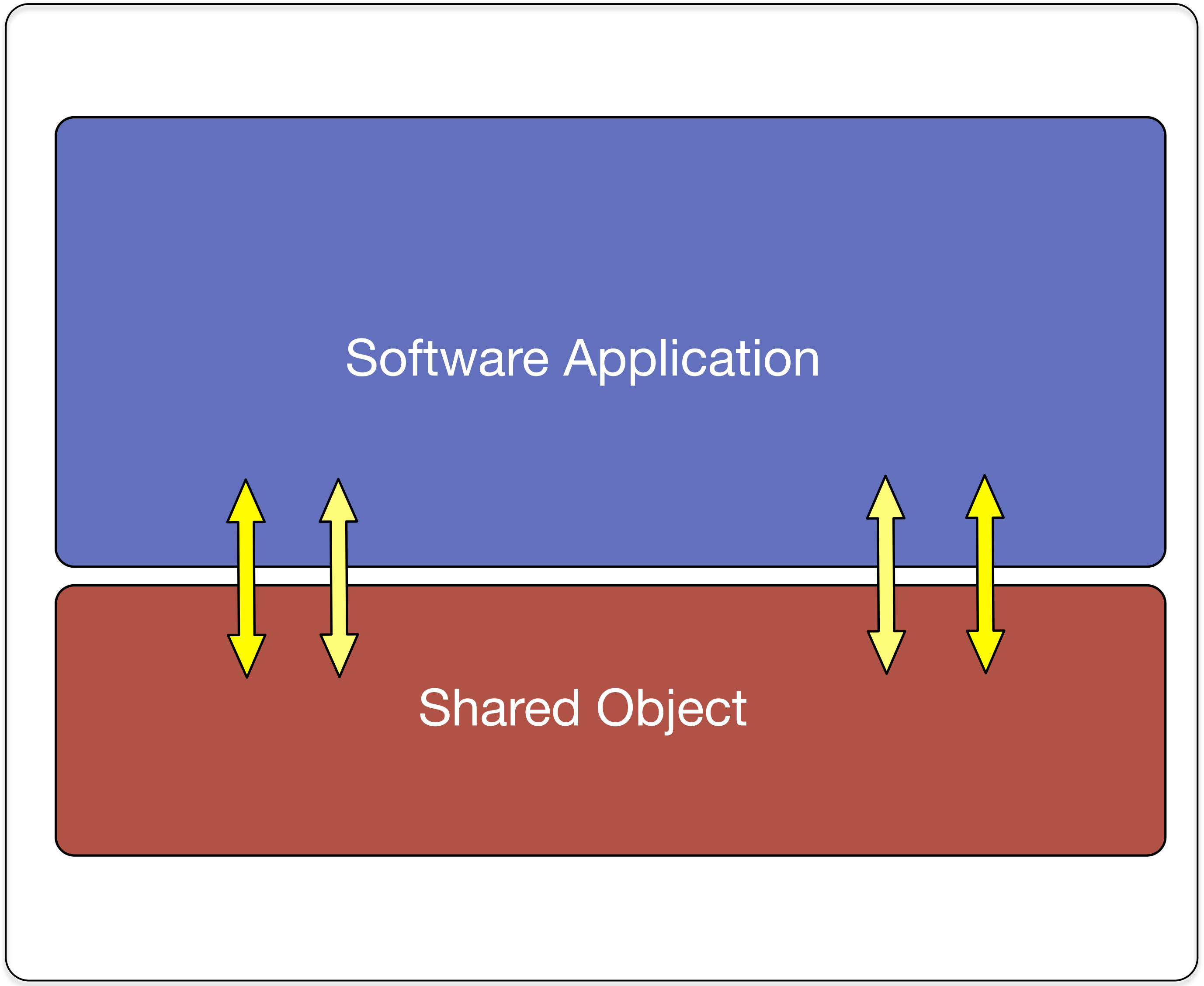
```
loc_100000D91:                  ; "signature correct. \n"
lea     rsi, aSignatureCorre
mov     rax, cs:___stdoutp_ptr
mov     rdi, [rax]          ; FILE *
mov     al, 0
call    _fprintf
lea     rdi, [rbp+var_20] ; char *
mov     rsi, cs:_MODE     ; char *
mov     [rbp+var_254], eax
call    _fopen
mov     [rbp+var_240], rax
cmp     [rbp+var_240], 0
jnz     loc_100000E05
```

“Shimming”

When an application uses a shared object, an attacker can interfere with the boundary.

Attacker uses export table of .so to generate a ‘shim’ to go between application and .so.

All data (parameters and return codes) can be siphoned and modified.

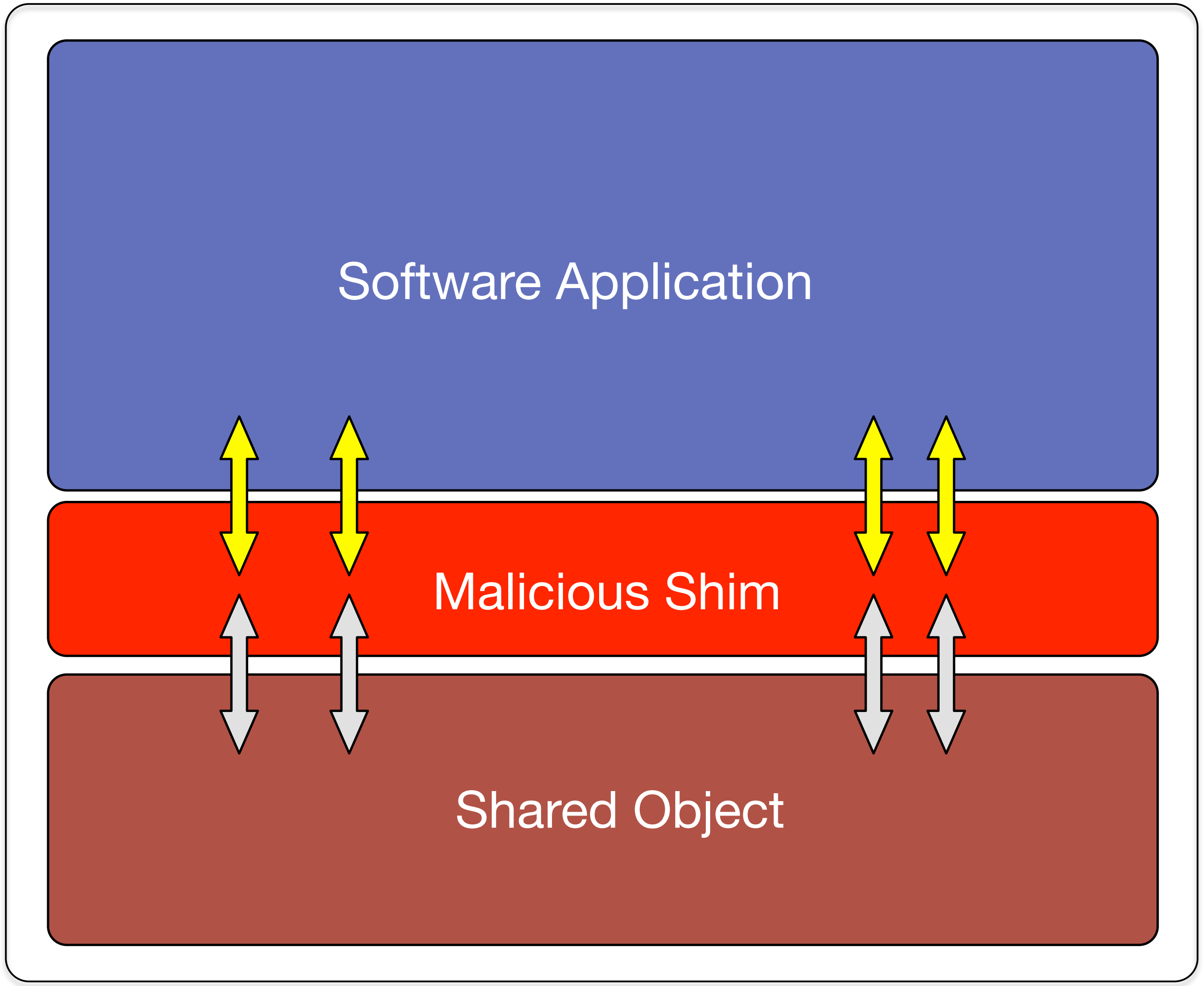


“Shimming”

When an application uses a shared object, an attacker can interfere with the boundary.

Attacker uses export table of .so to generate a ‘shim’ to go between application and .so.

All data (parameters and return codes) can be siphoned and modified.



Coding Practices

Coding Practices

US-CERT
UNITED STATES COMPUTER EMERGENCY READINESS TEAM

https://www.us-cert.gov/bsi/articles/knowledge/coding-practices

`Validate Inherited Process Context`

Inherited process context that is not validated like other inputs can introduce vulnerability.

`Use strncpy_s() and strncat_s()`

The strncpy_s() and strncat_s() functions are defined in ISO/IEC TR 24731 as drop-in replacements for strncpy() and strncat().

GENIVI®

Code Entanglement

- Avoid assertion checks on sensitive decisions such as a digital signature or password validation.

- "Entangle" the input value by using it to get to the asset. Eg: password is decryption key to decrypt file.

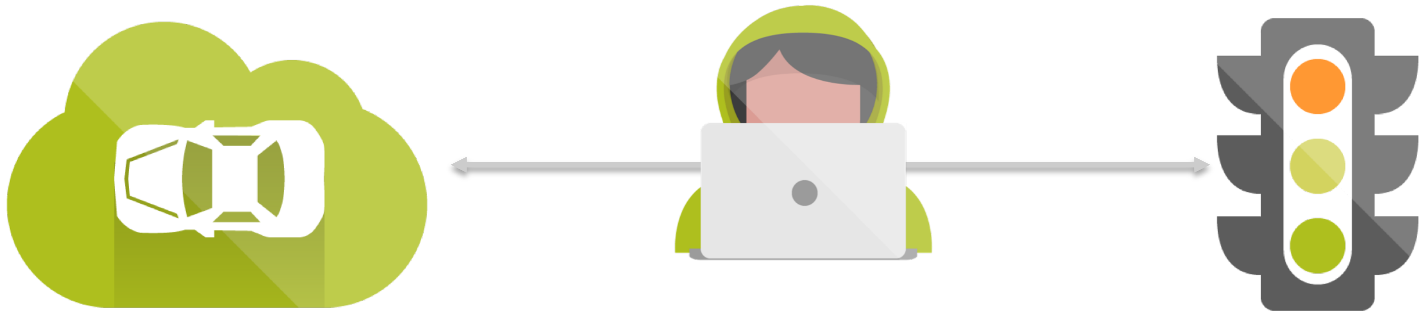
Assertion Check

```
pwHash = getPasswordHash();
if( pwHash == storedHash ){
    decryptFile(fn);
}
```

Entangled

```
pwHash = getPasswordHash();
decryptFile(fn, pwHash);
```

Data Parsing is Critical



US-CERT
UNITED STATES COMPUTER EMERGENCY READINESS TEAM

Knowledge Topics

- Assurance Cases
- Attack Patterns
- Business Case Models
- Coding Practices
- Lessons Learned
- Principles
- SDLC Process
- Software Assurance Education

Title	Updated date	Authors
Strengthening Ties Between Process and Security	2013-07-31	Carol Woody
Secure Software Development Life Cycle Processes	2013-07-31	Noopur Davis
Correctness by Construction	2013-05-14	Peter Amey
Design Principles	2013-05-13	Michael Gegick, Sean Barnum
Separation of Privilege	2013-05-10	Michael Gegick, Sean Barnum
Securing the Weakest Link	2013-05-10	Michael Gegick, Sean Barnum
Reluctance to Trust	2013-05-10	Michael Gegick, Sean Barnum
Psychological Acceptability	2013-05-10	Michael Gegick, Sean Barnum
Promoting Privacy	2013-05-10	Michael Gegick, Sean Barnum
Never Assuming That Your Secrets Are Safe	2013-05-10	Michael Gegick, Sean Barnum
Least Privilege	2013-05-10	Michael Gegick, Sean Barnum
Least Common Mechanism	2013-05-10	Michael Gegick, Sean Barnum
Failing Securely	2013-05-10	Michael Gegick, Sean Barnum
Economy of Mechanism	2013-05-10	Michael Gegick, Sean Barnum
Complete Mediation	2013-05-10	C.C. Michael, Michael Gegick, Sean Barnum
Defense in Depth	2005-09-13	Sean Barnum, Michael Gegick, C.C. Michael

GENIVI®

Software Protections

GENIVI®

Software Protections – Integrity Verification

If software is running on a potentially hostile environment, an attacker can have full control over software execution.

Attacker can use analysis tools to detect and circumvent in-software checks.

Verification of software integrity should be done:

- At install-time
- At start-time
- During run-time

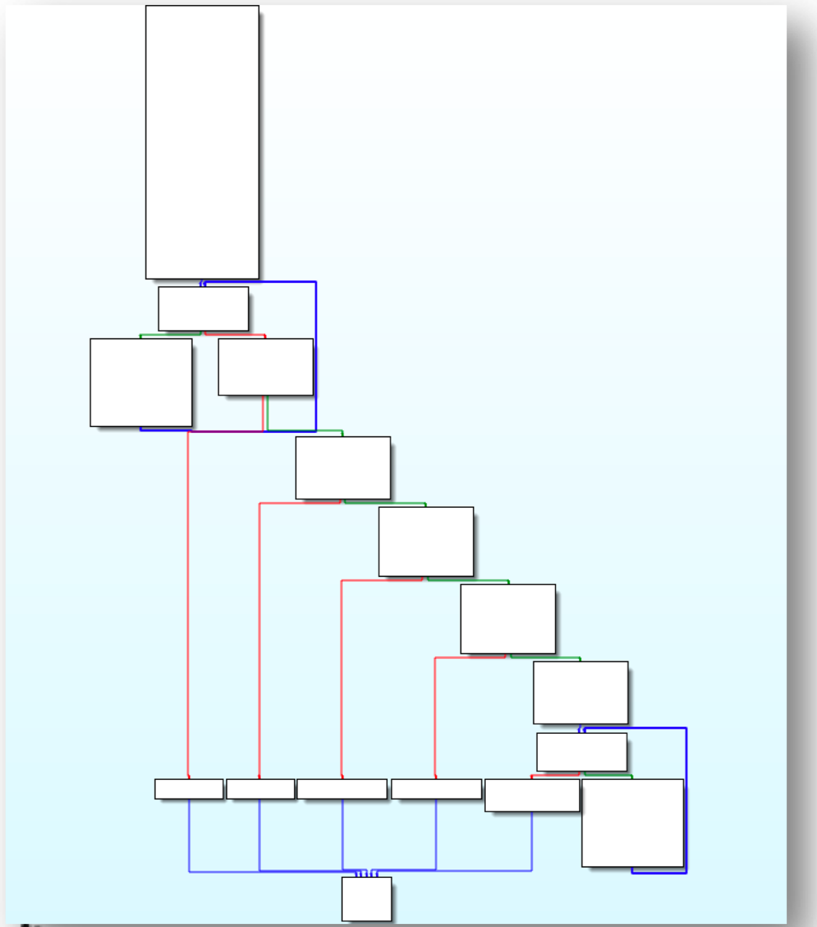
Software Protections – Transformations

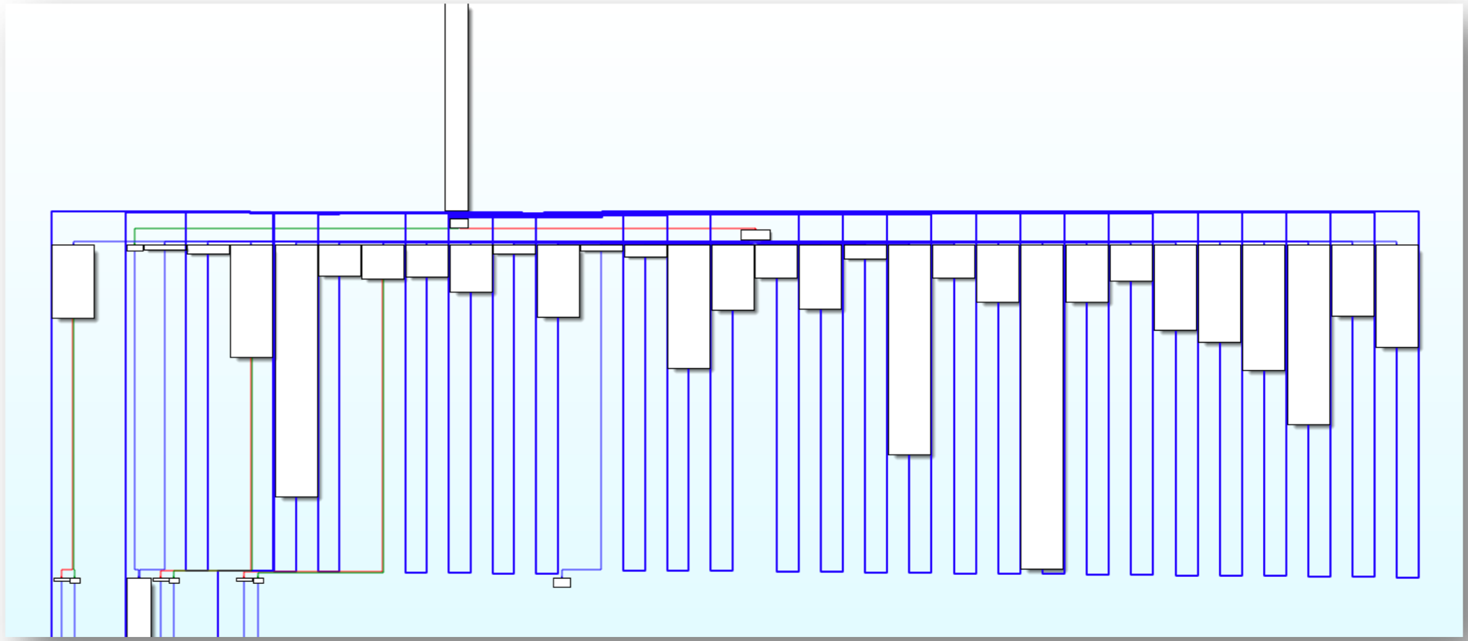
- ➢ Similar to integrity checks, code transformation is useful when software is in a hostile environment.
- ➢ Code transformation can strongly mitigate static analysis of code.
- ➢ Data transformation can hide data after decryption to mitigate against siphoning

Some form of code and data transformation is widely and expertly used by authors of sophisticated malware.

Transformation of open source can be tricky. License issues. Leakage of information through system calls.

Transforming Control Flow





System Features

ASLR

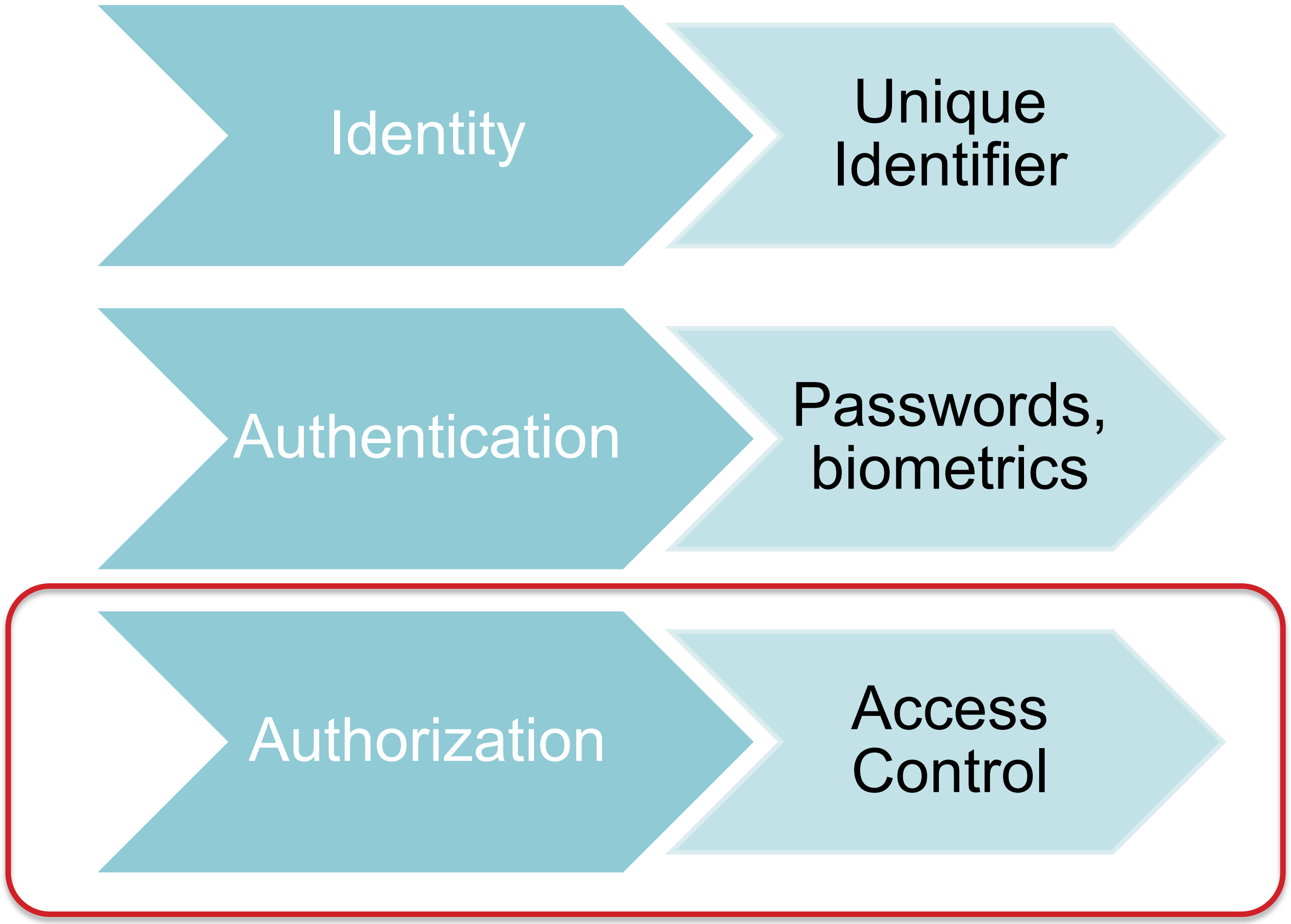
Address Space Layout Randomization

ASLR randomly arranges the address space positions of key data areas of a process, including the base of the executable and the positions of the stack, heap and libraries.

- Wikipedia

ASLR is a first line of defence against return-to-libc and ROP attacks by making it harder for attackers to know memory offsets before hand in an attack. It's effectiveness is based on the entropy used.

Access Control



Discretionary Access Control

- Owner of resource controls access.
- Access is based on identity or groups

- Primary attacker objective is to escalate privilege from restricted to non-restricted (root)

Mandatory Access Control

- Policy based authorization rules
- Centrally controlled by an administrator
- Users (even root) cannot override the policy
- Implemented using the LSM API

Mandatory Access Control

SELinux

- Based on labels
- Historically difficult to use

SMACK

- Designed with simplicity in mind

TOMOYO

- Uses pathnames instead of labels

AppArmor

- Uses pathnames, similar to TOMOYO

GENIVI

SECURITY NEEDS TO BE DESIGNED IN FROM THE BEGINNING.

Thank you!

Visit GENIVI at http://www.genivi.org or http://projects.genivi.org

Contact us: help@genivi.org