

# Security - Every Developer's Responsibility

---

Christoph Iserlohn



# About me

---

Senior Consultant @ innoQ

MacPorts Team member

# Agenda



Developers

Microsoft  
Windows

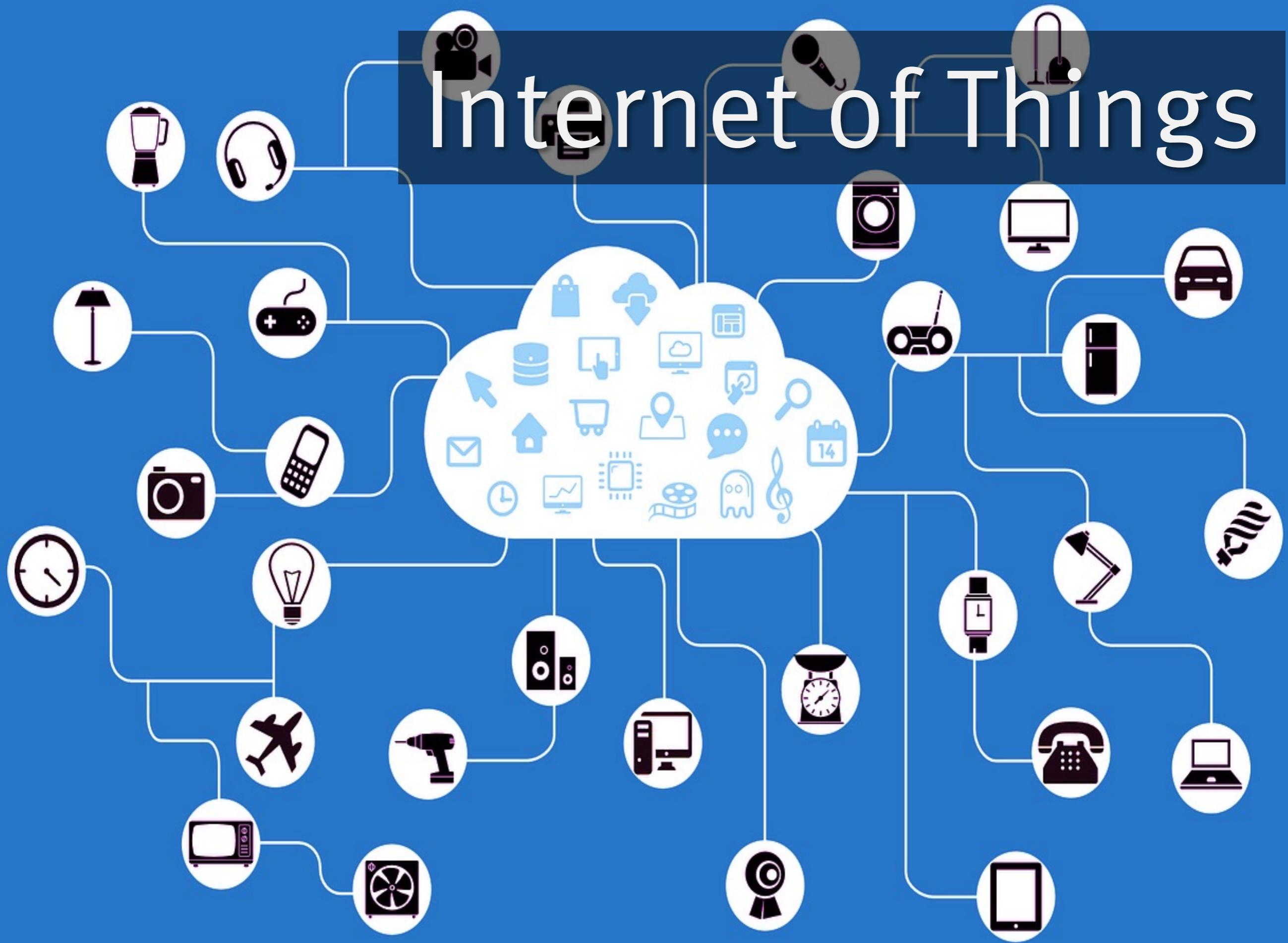
# Security



„Software is  
eating the world“



# Internet of Things





The mess we're in

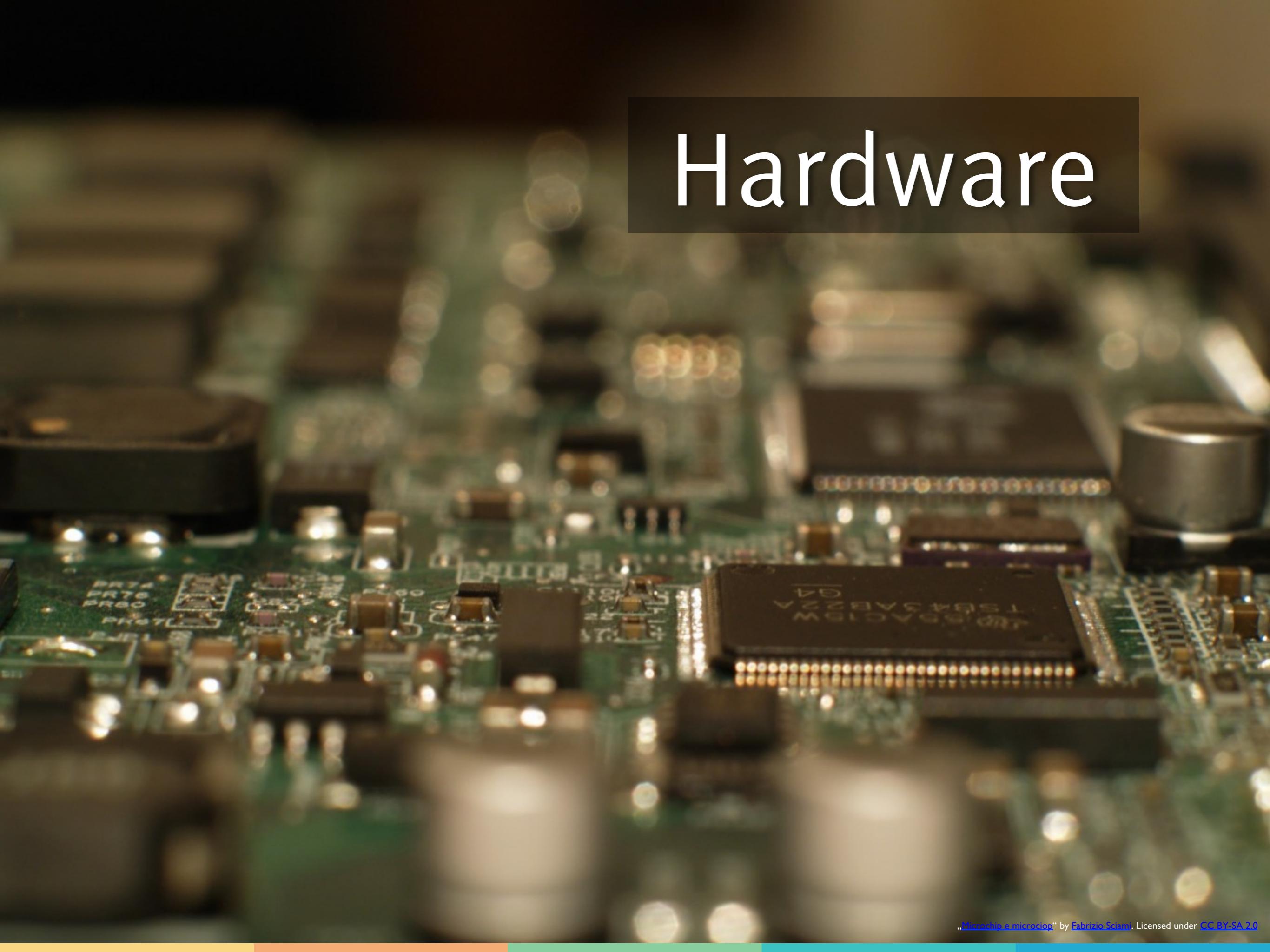
Security is  
not a product





Security dimensions

# Hardware



Current date is Tue 1-01-1980

Enter new date:

Current time is 7:48:27.13

Enter new time:

The IBM Personal Computer DOS

Version 1.10 (C)Copyright IBM Corp 1981, 1982

A>dir/w

COMMAND	COM	FORMAT	COM	CHKDSK	COM	SYS	COM	DISKCOPY	COM
DISKCOMP	COM	COMP	COM	EXE2BIN	EXE	MODE	COM	EDLIN	COM
DEBUG	COM	LINK	EXE	BASIC	COM	BASIC	COM	ART	BAS
SAMPLES	BAS	MORTGAGE	BAS	COLORBAR	BAS	CALNDAR	BAS	MUSIC	BAS
DONKEY	BAS	CIRCLE	BAS	PIECHART	BAS	SPACE	BAS	BALL	BAS
COMM	BAS								

26 File(s)

A>dir command.com

COMMAND	COM	4959	5-07-82	12:00p
---------	-----	------	---------	--------

1 File(s)

A>

# Operating Systems

# Software

```
scope.$watch(attrs.ngSwitch, element, attr, ngSwitchController) {
    var previousElements = attrs.ngSwitch || attr.on,
        selectedTranscludes = [],
        selectedElements = [],
        previousElements = [],
        previousScopes = [];
    scope.$watch(expr, function ngSwitchWatchAction(value) {
        var i, ii;
        for (ii = 0, ii = previousElements.length; i < ii; ++i) {
            previousElements[i].remove();
        }
        previousElements.length = 0;

        for (ii = 0, ii = selectedScopes.length; i < ii; ++i) {
            var selected = selectedElements[i];
            selectedScopes[i].$destroy();
            previousElements[i] = selected;
            $animate.leave(selected, function() {
                previousElements.splice(i, 1);
            });
        }
    });

    selectedElements.length = 0;
    selectedScopes.length = 0;

    if ((selectedTranscludes = ngSwitchController.cases['!' + value] || ngSwitchController.cases[value])) {
        scope.$eval(attr.change);
        forEach(selectedTranscludes, function(selectedTransclude) {
            var selectedScope = scope.$new();
            selectedScopes.push(selectedScope);
            selectedScope.$on('$destroy', function() {
                selectedScopes.pop();
            });
        });
    }
});
```

# Network



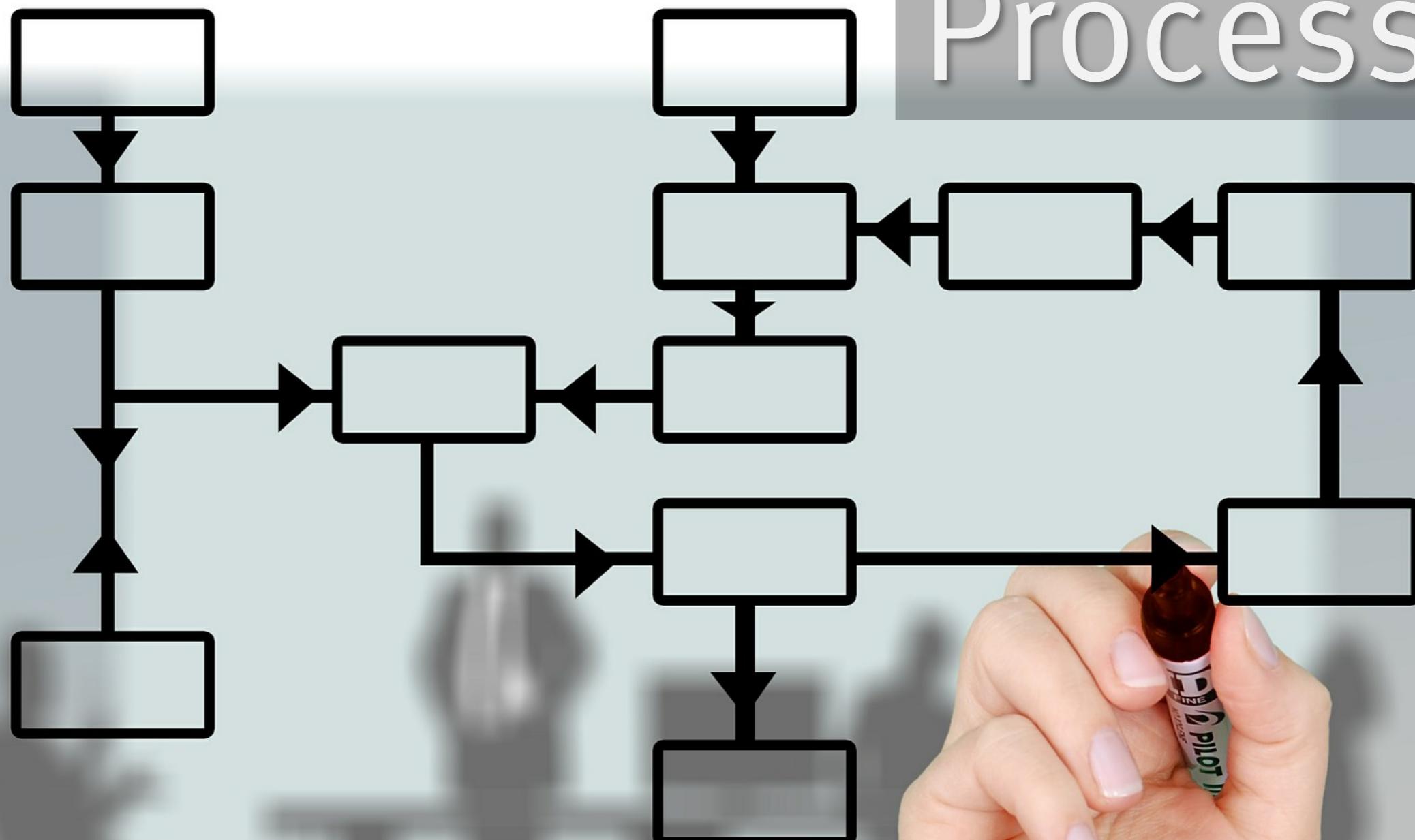
# People





# Organizations

# Processes



An aerial photograph of a dense urban skyline, likely New York City, showing a high concentration of skyscrapers of various heights and architectural styles. The buildings are packed closely together, creating a complex geometric pattern of windows and rooftops. A few taller buildings stand out, such as the MetLife Building with its distinctive teal facade and the One Penn Plaza building with its white grid-like facade. The streets between the buildings are visible as thin lines.

Buildings



Law

Group →	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
↓ Period																		
1	1 H															2 He		
2	3 Li	4 Be														5 B	6 C	7 N
3	11 Na	12 Mg														8 O	9 F	10 Ne
4	19 K	20 Ca	21 Sc	22 Ti	23 V	24 Cr	25 Mn	26 Fe	27 Co	28 Ni	29 Cu	30 Zn	31 Ga	32 Ge	33 As	34 Se	35 Br	36 Kr
5	37 Rb	38 Sr	39 Y	40 Zr	41 Nb	42 Mo	43 Tc	44 Ru	45 Rh	46 Pd	47 Ag	48 Cd	49 In	50 Sn	51 Sb	52 Te	53 I	54 Xe
6	55 Cs	56 Ba		72 Hf	73 Ta	74 W	75 Re	76 Os	77 Ir	78 Pt	79 Au	80 Hg	81 Tl	82 Pb	83 Bi	84 Po	85 At	86 Rn
7	87 Fr	88 Ra		104 Rf	105 Db	106 Sg	107 Bh	108 Hs	109 Mt	110 Ds	111 Rg	112 Cn	113 Uut	114 Fl	115 Uup	116 Lv	117 Uus	118 Uuo
Lanthanides	57 La	58 Ce	59 Pr	60 Nd	61 Pm	62 Sm	63 Eu	64 Gd	65 Tb	66 Dy	67 Ho	68 Er	69 Tm	70 Yb	71 Lu			
Actinides	89 Ac	90 Th	91 Pa	92 U	93 Np	94 Pu	95 Am	96 Cm	97 Bk	98 Cf	99 Es	100 Fm	101 Md	102 No	103 Lr			

# Types of insecurity



# Trusting user input

**TOUCHING WIRES CAUSES  
INSTANT DEATH**

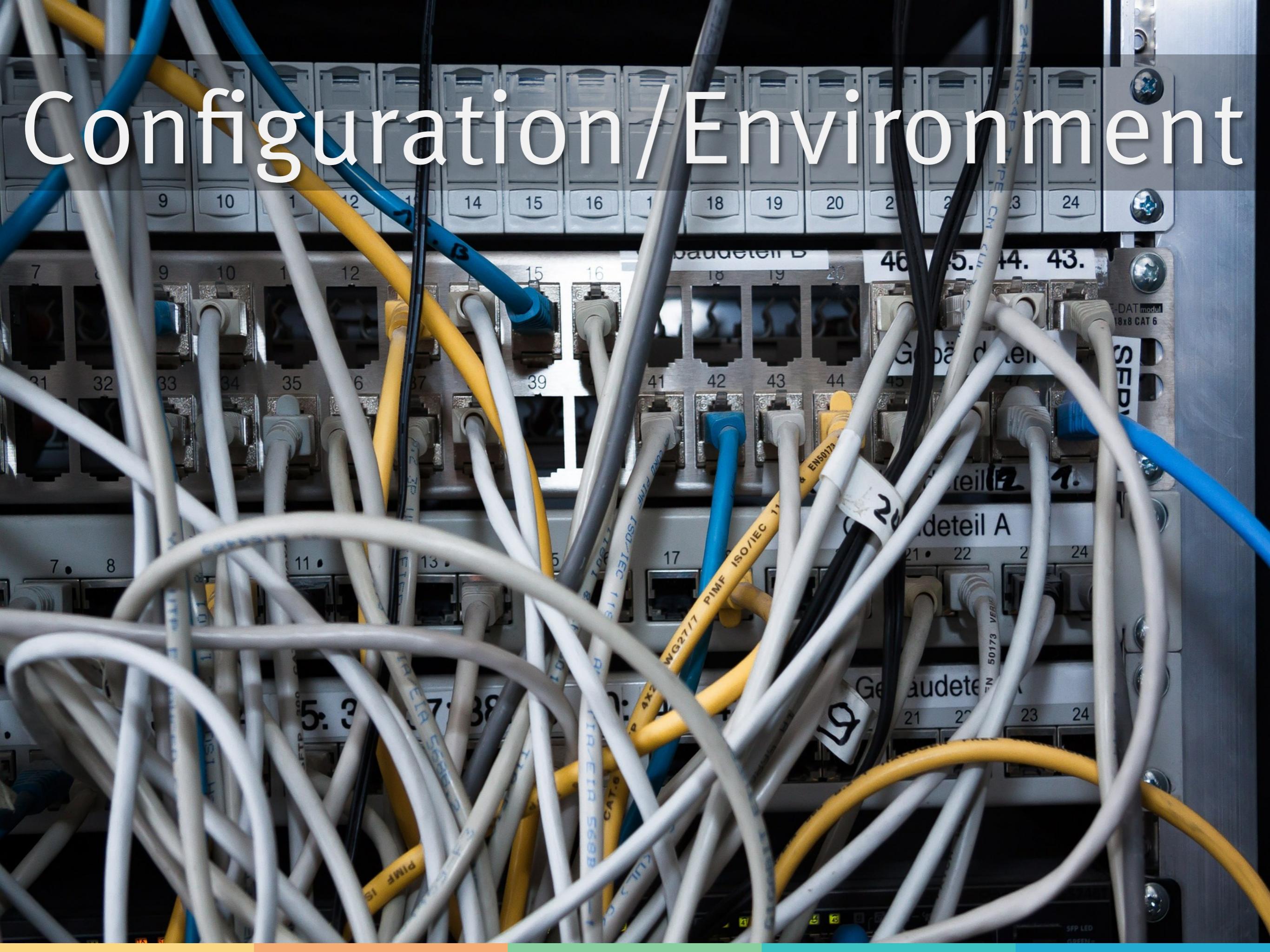


**\$200 FINE**



Logic errors / Design flaws

# Configuration/Environment



# Cryptographic weaknesses

# Bugs



Your PC ran into a problem and needs to restart. We're just collecting some error info, and then we'll restart for you. (0% complete)

If you'd like to know more, you can search online later for this error: HAL\_INITIALIZATION\_FAILED

tab

# Security should be

caps lock

easy

control

A

S

Z

X

alt

option



command

# libsodium

```
#include <sodium.h>

int main(void)
{
    if (sodium_init() == -1) { return 1; }

    const unsigned char message[] = "The quick brown fox jumps over the lazy dog";

    int message_len = sizeof message;
    int encrypted_len= message_len + crypto_secretbox_MACBYTES;

    unsigned char nonce[crypto_secretbox_NONCEBYTES];
    unsigned char key[crypto_secretbox_KEYBYTES];
    unsigned char encrypted[encrypted_len];
    unsigned char plain[message_len];

    randombytes_buf(nonce, sizeof nonce);
    randombytes_buf(key, sizeof key);

    crypto_secretbox_easy(encrypted, message, message_len, nonce, key);

    if (crypto_secretbox_open_easy(plain, encrypted, encrypted_len, nonce, key) != 0) {
        printf("Message has been forged!");
        return 1;
    }

    printf("Message to encrypt: %s\nCiphertext: ", message);

    for(int i = 0; i < ciphertext_len; i++) {
        printf("%02x", ciphertext[i]);
    }

    printf("\nDecrypted message: %s\n", decrypted);
}
```

# libcrypto

```
#include <openssl/conf.h>
#include <openssl/evp.h>
#include <openssl/err.h>
#include <string.h>

int main (void) {
    unsigned char *key = (unsigned char *)"01234567890123456789012345678901";
    unsigned char *iv = (unsigned char *)"01234567890123456";
    unsigned char *plain = (unsigned char *)"The quick brown fox jumps over the lazy dog";

    unsigned char ciphertext[128];
    unsigned char decryptedtext[128];

    int decryptedtext_len, ciphertext_len;

    ERR_load_crypto_strings();
    OpenSSL_add_all_algorithms();
    OPENSSL_config(NULL);

    ciphertext_len = encrypt(plain, strlen ((char *)plain), key, iv, ciphertext);

    printf("Ciphertext is:\n");
    BIO_dump_fp (stdout, (const char *)ciphertext, ciphertext_len);

    decryptedtext_len = decrypt(ciphertext, ciphertext_len, key, iv, decryptedtext);

    decryptedtext[decryptedtext_len] = '\0';

    printf("Decrypted text is:\n");
    printf("%s\n", decryptedtext);

    EVP_cleanup();
    ERR_free_strings();
}
```

# libcrypto - continued

---

```
int encrypt(unsigned char *plaintext, int plaintext_len, unsigned char *key,
           unsigned char *iv, unsigned char *ciphertext)
{
    EVP_CIPHER_CTX *ctx;
    int len;
    int ciphertext_len;
    if(!(ctx = EVP_CIPHER_CTX_new())))
        handleErrors();
    if(1 != EVP_EncryptInit_ex(ctx, EVP_aes_256_cbc(), NULL, key, iv))
        handleErrors();
    if(1 != EVP_EncryptUpdate(ctx, ciphertext, &len, plaintext, plaintext_len))
        handleErrors();
    ciphertext_len = len;
    if(1 != EVP_EncryptFinal_ex(ctx, ciphertext + len, &len))
        handleErrors();
    ciphertext_len += len;
    EVP_CIPHER_CTX_free(ctx);
    return ciphertext_len;
}
```

# libcrypto - continued

```
int decrypt(unsigned char *ciphertext, int ciphertext_len, unsigned char *key,
           unsigned char *iv, unsigned char *plaintext)
{
    EVP_CIPHER_CTX *ctx;
    int len;
    int plaintext_len;
    if(!(ctx = EVP_CIPHER_CTX_new())))
        handleErrors();
    if(1 != EVP_DecryptInit_ex(ctx, EVP_aes_256_cbc(), NULL, key, iv))
        handleErrors();
    if(1 != EVP_DecryptUpdate(ctx, plaintext, &len, ciphertext, ciphertext_len))
        handleErrors();
    plaintext_len = len;
    if(1 != EVP_DecryptFinal_ex(ctx, plaintext + len, &len))
        handleErrors();
    plaintext_len += len;
    EVP_CIPHER_CTX_free(ctx);
    return plaintext_len;
}
```

# libcrypto - continued

---

```
void handleErrors(void)
{
    ERR_print_errors_fp(stderr);
    abort();
}
```



```
// Create a trust manager that does not validate certificate chains
TrustManager[] trustAllCerts = new TrustManager[] { new X509TrustManager() {
    public java.security.cert.X509Certificate[] getAcceptedIssuers() {
        return null;
    }
    public void checkClientTrusted(X509Certificate[] certs, String authType) {}
    public void checkServerTrusted(X509Certificate[] certs, String authType) {}
}};

// Install the all-trusting trust manager
final SSLContext sc = SSLContext.getInstance("SSL");
sc.init(null, trustAllCerts, new java.security.SecureRandom());
HttpsURLConnection.setDefaultSSLSocketFactory(sc.getSocketFactory());
// Create all-trusting host name verifier
HostnameVerifier allHostsValid = new HostnameVerifier() {
    public boolean verify(String hostname, SSLSession session) {
        return true;
    }
};

// Install the all-trusting host verifier
HttpsURLConnection.setDefaultHostnameVerifier(allHostsValid);
```

A close-up photograph of a surgeon's face. The surgeon is wearing a blue surgical mask over a yellow and white checkered scrub cap. A black surgical headlight is mounted on their forehead, illuminating their eyes. They are looking down and to the left, focused on a procedure. In the background, the blurred shapes of other medical staff and equipment are visible.

You can't be a  
professional...

...without knowing  
the basics.



# Thank you!

---

- > Questions ?
- > Comments ?

Christoph Iserlohn

[christoph.iserlohn@innoq.com](mailto:christoph.iserlohn@innoq.com)