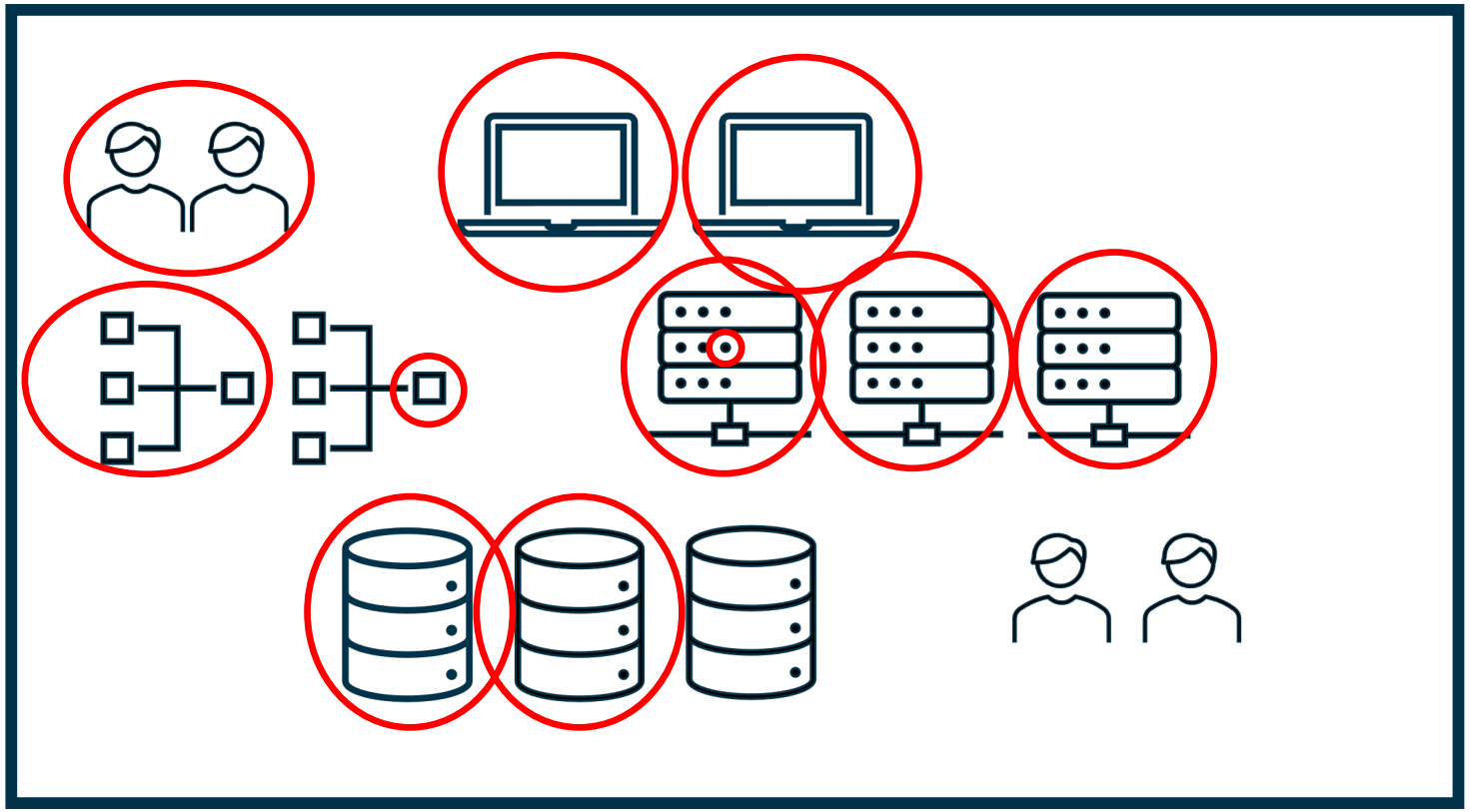

omega
point.



«Secure By Design»

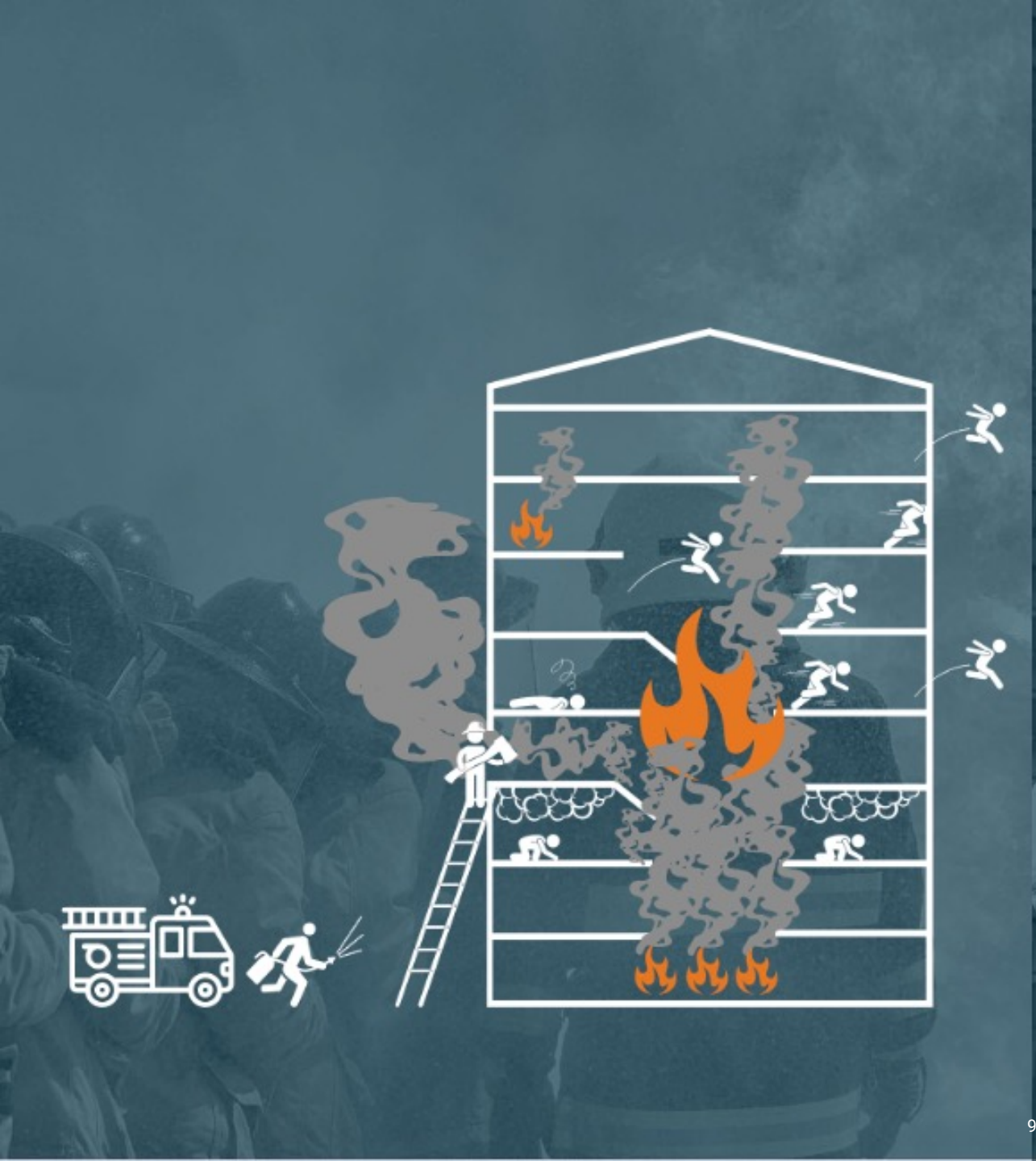
Domain driven security

Cybersecure digitalization

WE NOT ONLY FIGHT FIRE. WE PREVENT IT.

Fire-fighters
fight fire.

Fire engineers
prevent fire.



“History never repeats itself. Man always does.

Voltaire

Anders Kofoed

- Siv.ing./MSc Communication Technology– Information Security(NTNU 2015)
- Partner & Academy Lead at Omegapoint Norge
- System development/design, architecture, leadership and competence development

Eivind Jahr Kirkeby

- Siv.ing./MSc Cybernetics and Robotics (NTNU 2018)
- Senior Expert and Secure Development Advocate at Omegapoint Norge
- System development, AppSec, defensive security

Håkon Anders Strømsodd

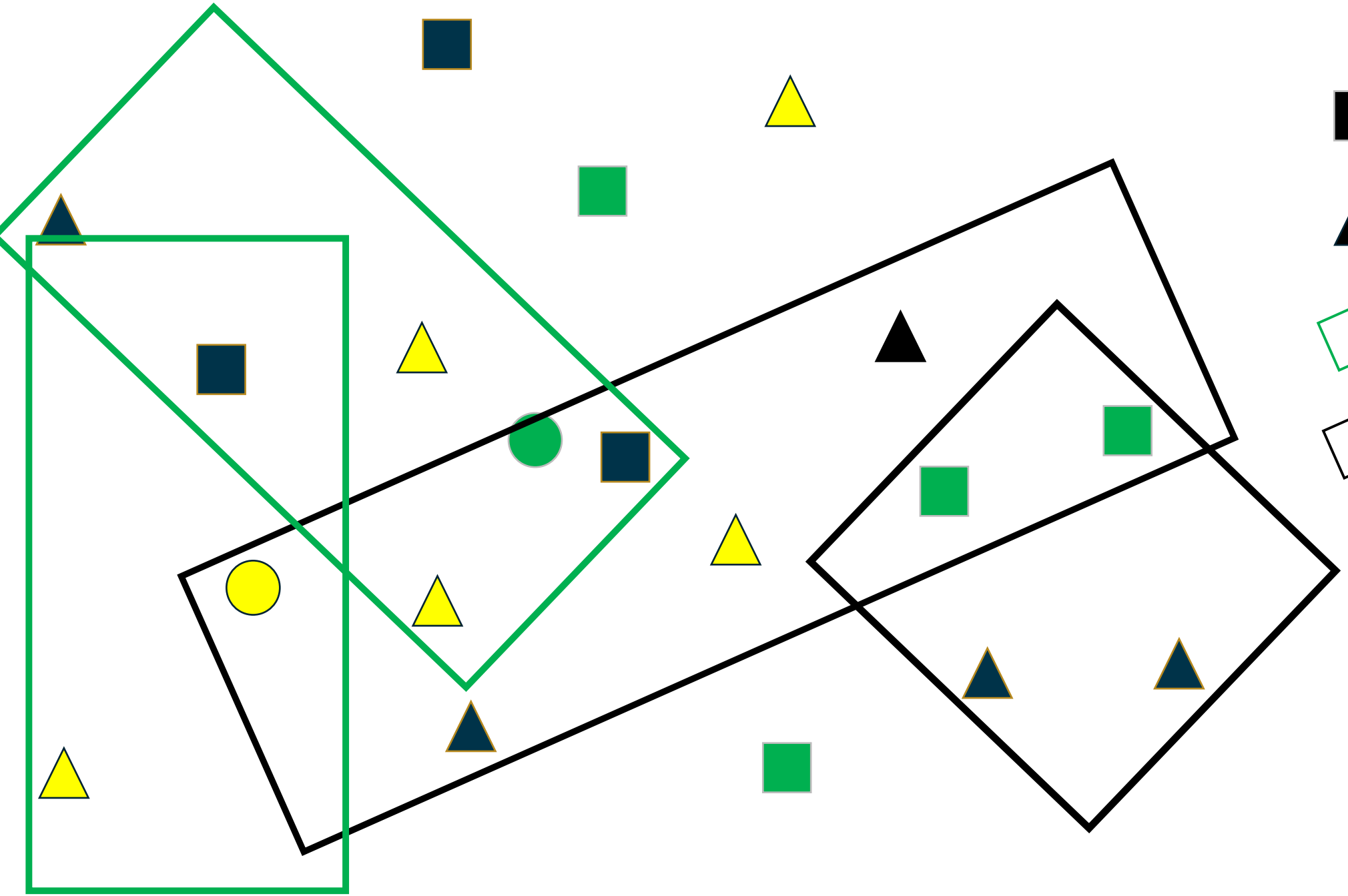
- Siv.ing./MSc Computer Science – Artificial Intelligence (NTNU 2023)
- Consultant at Omegapoint Norge

Secure by Design

Dan Bergh Johnsson
Daniel Deogun
Daniel Sawano
Foreword by Daniel Terhorst-North

MANNING

omega point.



Security bug

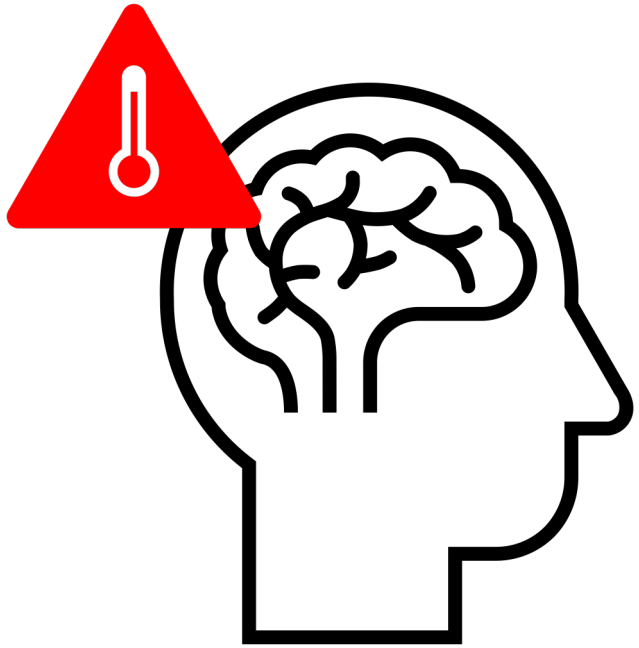
«Normal» bug

Design pattern

«Secure by Design» pattern

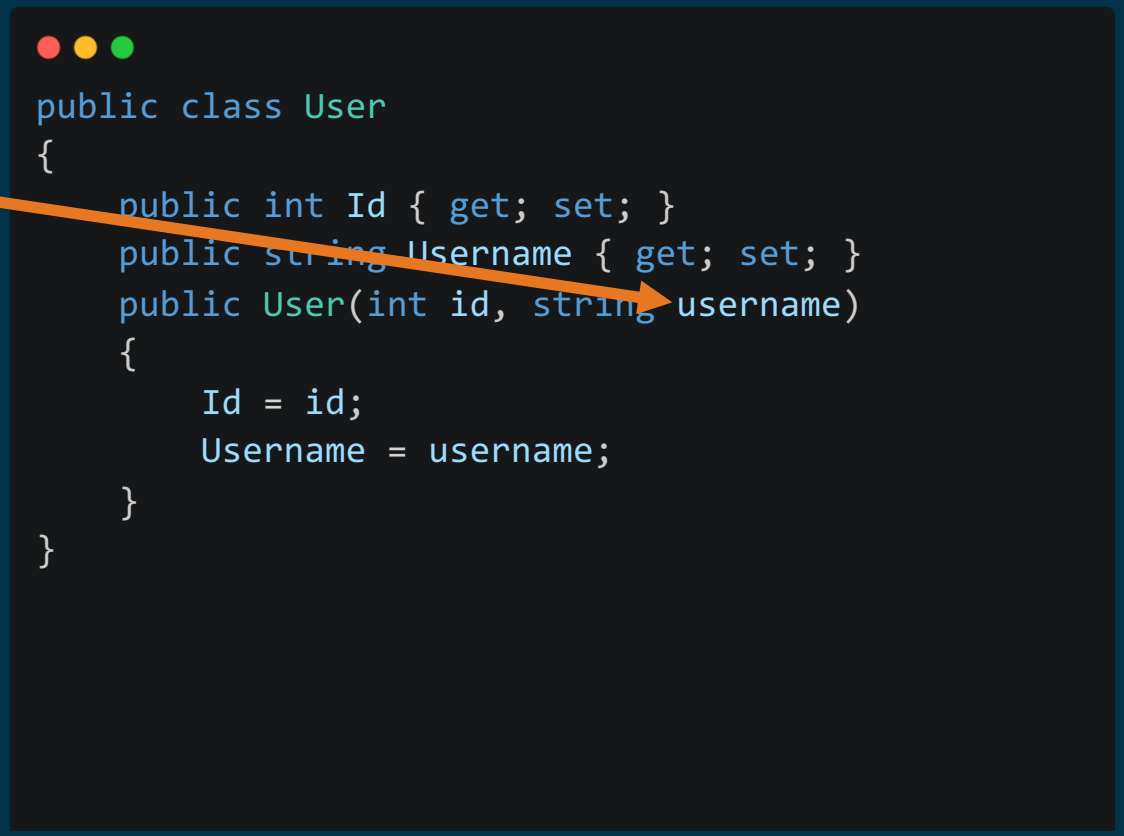
“Traditional” approach

- ✓ OWASP
- ✓ XSS
- ✓ SQL Injection
- ✓ Attack vectors
- ✓ 0-day vulnerabilities
- ✓ …



“Traditional” approach

- Possible XSS and SQL injection
- `<script>alert("Hacked");</script>`



```
public class User
{
    public int Id { get; set; }
    public string Username { get; set; }
    public User(int id, string username)
    {
        Id = id;
        Username = username;
    }
}
```

“Traditional” approach

```
public class User
{
    public int Id { get; set; }
    public string Username { get; set; }
    public User(int id, string username)
    {

        Id = notNull(id);
        Username = validateForXSSAndSQLI(username);
    }
}
```


Secure by Design

```
public class User
{
    private static readonly int USERNAME_MAX_LENGTH = 20;
    private static readonly int USERNAME_MIN_LENGTH = 5;
    private static readonly int USERNAME_ALLOWED_CHARS = "[a-zA-Z0-9_-]+";
    private int Id { get; }
    private string Username { get; }
    public User(int id, string username)
    {
        var trimmedUsername = username.Trim();
        lengthBetween(trimmedUsername, USERNAME_MAX_LENGTH, USERNAME_MIN_LENGTH);
        matches(trimmedUsername, USERNAME_ALLOWED_CHARS);
        Id = notNull(id);
        Username = trimmedUsername;
    }
}
```

“Traditional” approach

```
public class Username
{
    private static readonly int USERNAME_MAX_LENGTH = 20;
    private static readonly int USERNAME_MIN_LENGTH = 5;
    private static readonly int USERNAME_ALLOWED_CHARS = "[a-zA-Z0-9_-]+";
    private string Value { get; }
    public Username(string username)
    {
        var trimmedUsername = username.Trim();
        lengthBetween(trimmedUsername, USERNAME_MAX_LENGTH, USERNAME_MIN_LENGTH)
        matches(trimmedUsername, USERNAME_ALLOWED_CHARS);
        Value = trimmedUsername;
    }
}
```

“Traditional” approach

```
public class User
{
    private int Id { get; }
    private Username Username { get; }
    public User(int id, Username username)
    {
        Id = notNull(id);
        Username = username;
    }
}
```


Domain Primitives/ Value Objects

- Can only exist if the values are valid
- Building blocks modelling reality
- Cannot be changed (immutable)
- Contains rules and operations in addition to validation

Postal Code


posten
Frist for innlevering:
Kl. 16.00 mandag – fredag
posten
Frist for innlevering:
Kl. 16.00 mandag – fredag
posten
Frist for innlevering:
Kl. 16.00 mandag – fredag

CIA + T

Confidentiality – data can only be accessed by authorized users

Availability – data and systems are always available to users

Integrity – data is correct and has not been tampered with

Tracability – who changed or accessed what data

```
User CreatePerson(string name, string address, int postalCode)
{
    var person = new Person(name, address + postalCode);
    db.Save(person);
    return person;
}

CreatePerson("Kirkeveien 26", "Anders Kofoed", 0467);
```


«Postal code is simply an int»

Peano's Axioms

- Zero is a number
- If n is a number, the successor of n is a number
- Zero isn’t the successor of a number
- Two numbers of which the successors are equal are themselves equal
- If a set S of numbers contains zero and also the successor of every number in S, then every number is in S

Abelian Group

- Closure: a + b = integer
- Associativity: a + (b + c) = (a + b) + c
- Commutativity: a + b = b + a
- Identity: a + 0 = a
- Inverse: a + (−a) = 0

```
User CreatePerson(Name name, Address address, PostalCode postalCode)
{
    return new Person(name, address + postalCode);
}

CreatePerson("Anders Kofoed", "Kirkeveien 26", 0467);
```


Money!

```
public class Money
{
    private static readonly string[] _allowedCurrencies =
        new string[] {"NOK", "EUR", "USD", "SEK"};

    private decimal Amount { get; }
    private string Currency { get; }

    public Money(decimal amount, string currency)
    {
        Amount = amount;
        Currency = isOneOf(currency, _allowedCurrencies);
    }
}
```


```
public class Money
{
    //……
    public static Money operator +(Money a, Money b)
    {
        if(a.Currency != b.Currency)
            throw new ArgumentException("Different currencies cannot be added");

        return new Money(a.Amount + b.Amount, a.Currency);
    }

    public static Money operator -(Money a, Money b)
    {
        if(a.Currency != b.Currency)
            throw new ArgumentException("Different currencies cannot be subed");

        return new Money(a.Amount - b.Amount, a.Currency);
    }
}
```


Secure by Design

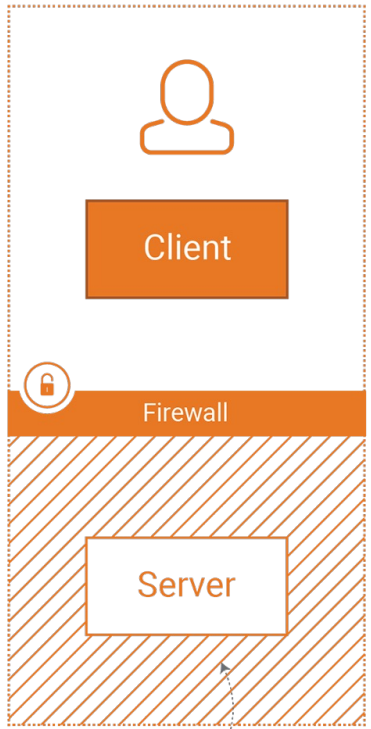
Focus on good design

Less bugs

More secure systems

Defence in Depth

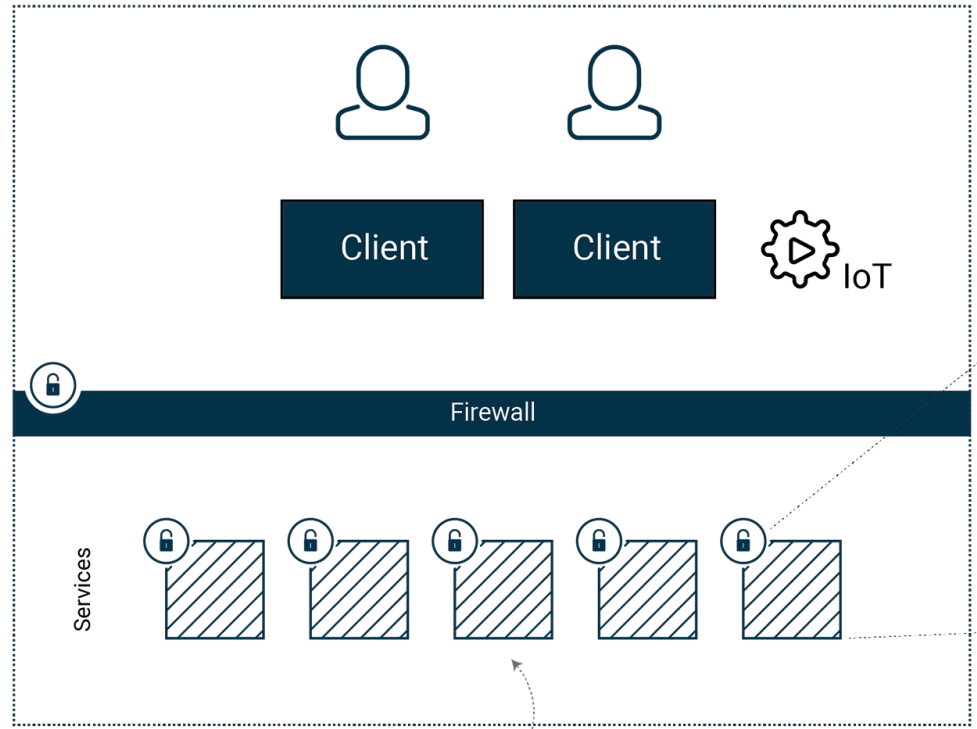
Perimeter defence

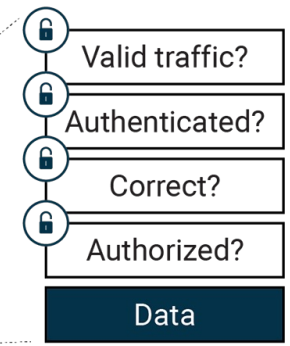


Full Trust (weak design)

All internal traffic is trusted

Defence in depth



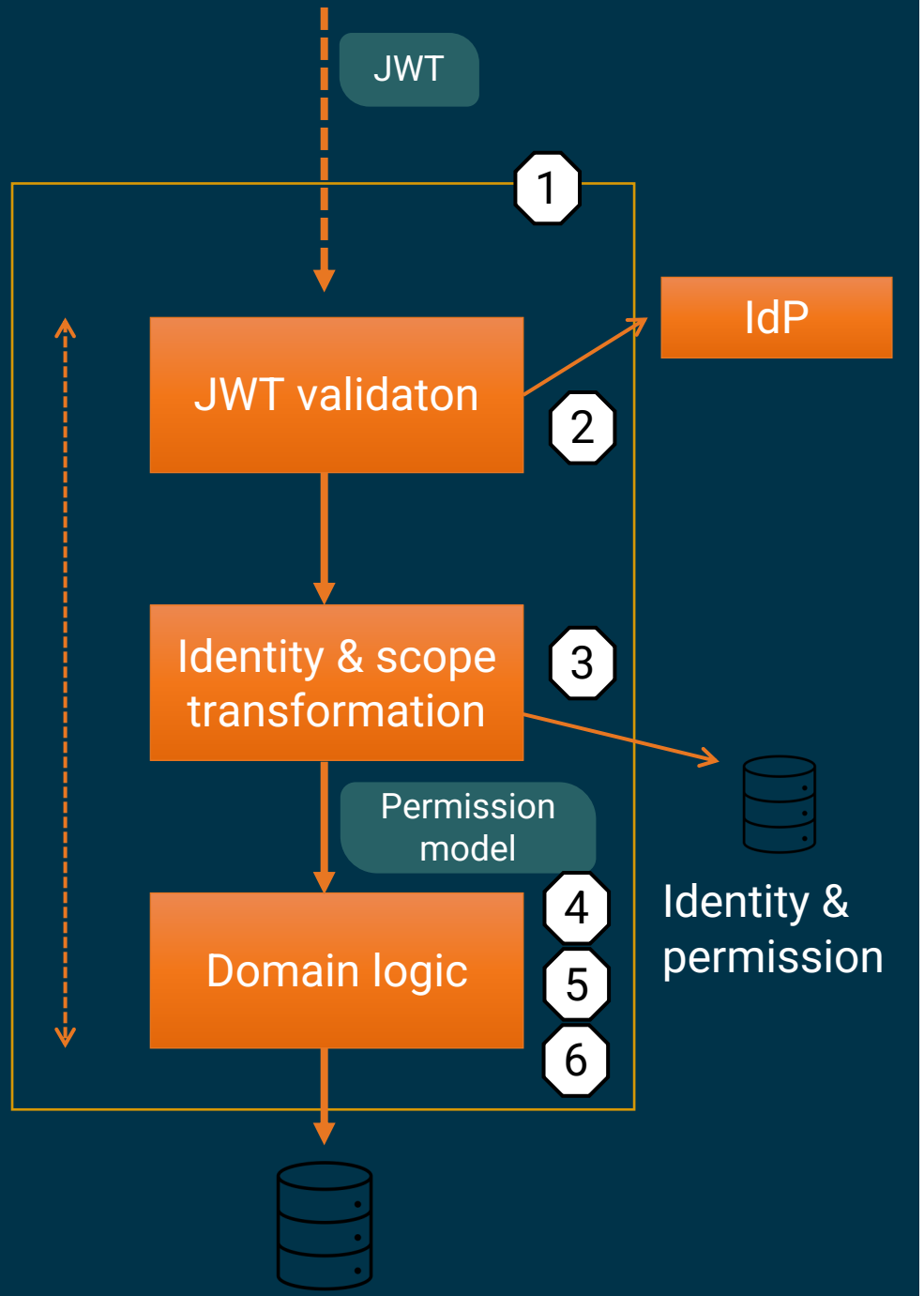


Zero Trust (strong design)

- All networks considered insecure
- All clients and users authenticated
- Access control for every request

Every individual service can be exposed on public networks

Defence in depth, protected by multiple layers



1. Validate that the request is correct
2. Validate that the access token is correct
3. Transform the access token into a permissions model
4. Validate that the data in the request is correct
5. Validate the permission to execute the operation
6. Validate the permission to query or change data

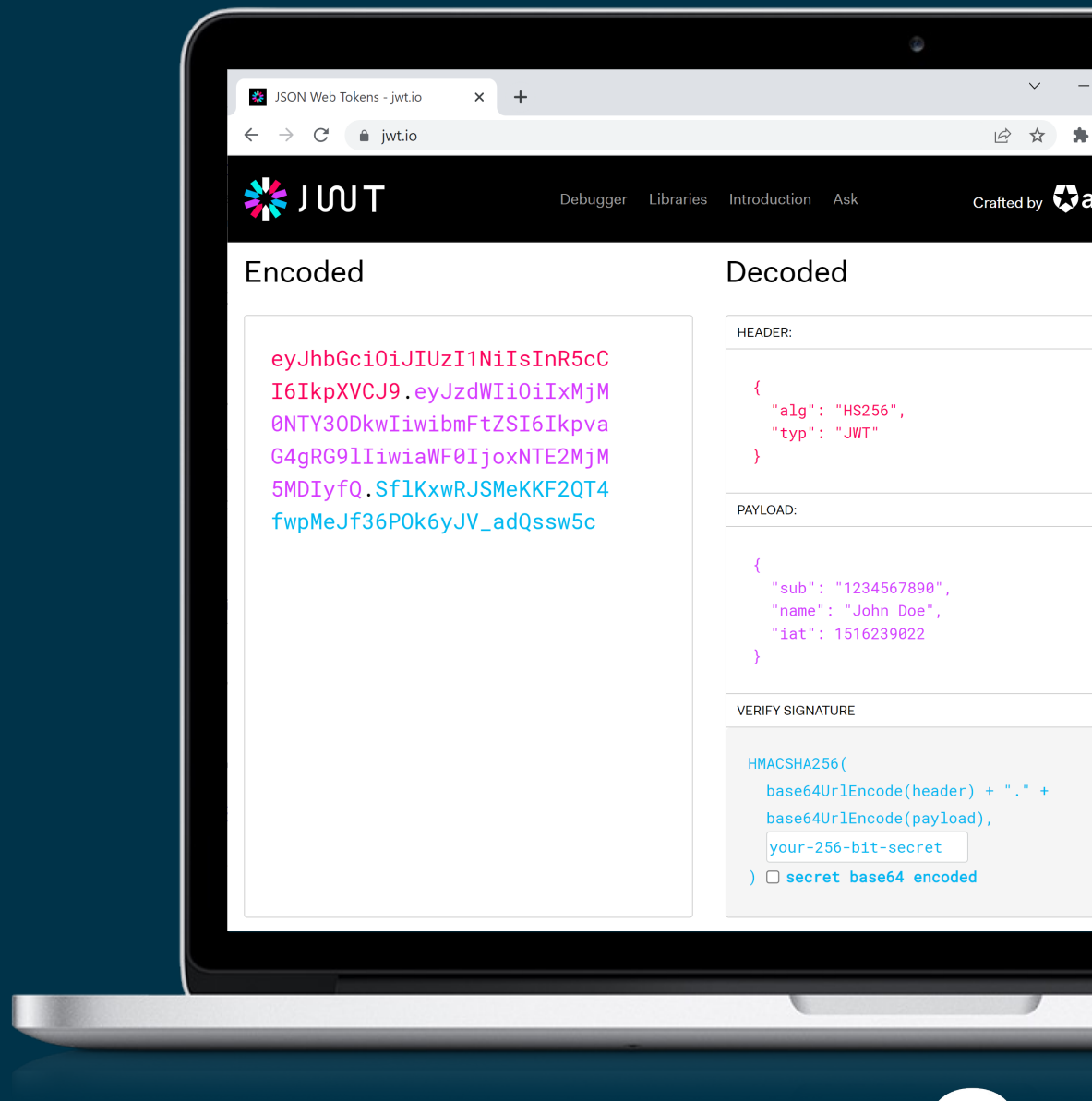
1. Validate that the request is correct

```
GET https://localhost:5000/api/products/1 HTTP/1.1
```

```
HTTP/1.1 200 OK
Connection: close
Content-Type: application/json; charset=utf-8
Date: Mon, 25 Dec 2021 06:00:00 GMT
Server: Kestrel
Transfer-Encoding: chunked

{ "id": "1", "name": "My Product", "market": "se" }
```

2. Validate that the access token is correct

- Issuer
- Audience
- Signature
- Lifetime
- Refuse JOSE header {"alg": none; }



3. Transform the access token into a permissions model

- Transform into permission model
- Lookup permissions from database

4. Validate that the data in the request is correct

```
if (string.IsNullOrEmpty(id) || id.Length > 10 || !id.All(char.IsLetterOrDigit))
{
    return BadRequest("Parameter id is not well formed.");
}
```


5. Validate the permission to execute the operation

```
if (!User.HasClaim("urn:permissions:products:read", "true"))
{
    return Forbid();
}
```


6. Validate the permission to query or change data

```
if (!User.HasClaim("urn:permissions:market", product.Market))
{
    return NotFound();
}
```


Test Driven Security

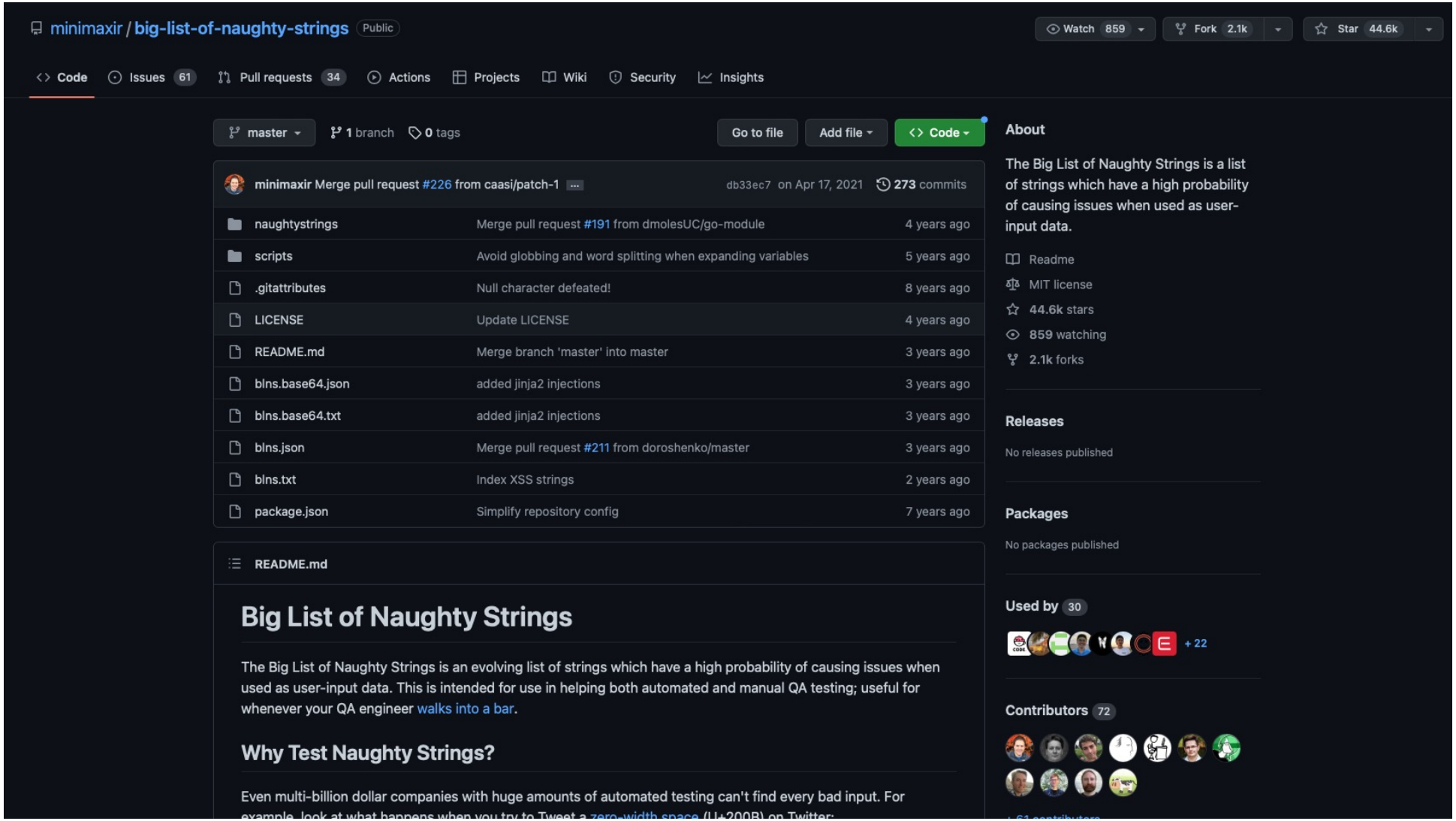
Test Driven Security

Automate the ”real security” in pipelines

Functional bugs are found by users, security bugs by hackers

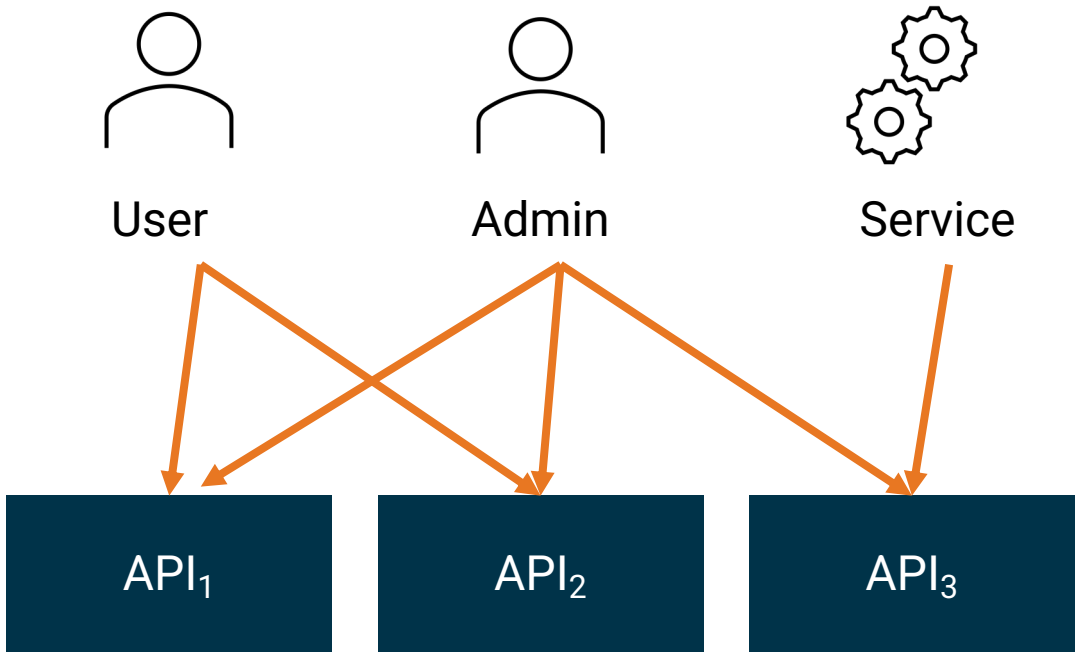
```
#	Unicode Symbols
#
#	Strings which contain common unicode symbols (e.g. smart quotes)

Ω≈ç√∫˜µ≤≥÷
åß∂ƒ©˙∆˚¬…æ
œ∑´®†¥¨ˆøπ“‘
¡™£¢∞§¶•ªº–≠
¸˛Ç◊ı˜Â¯˘¿
ÅÍÎÏ˝ÓÔÒÚÆ☃
Œ„´‰ˇÁ¨ˆØ∏”’
`⁄€‹›ﬁﬂ‡°·‚—±
⅛⅜⅝⅞
ЁЂЃЄЅІЇЈЉЊЋЌЍЎЏАБВГДЕЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯабвгдежзийклмнопрстуфхцчшщъыьэюя
٠١٢٣٤٥٦٧٨٩

#	Unicode Subscript/Superscript/Accents
#
#	Strings which contain unicode subscripts/superscripts; can cause rendering issues

⁰⁴⁵
₀₁₂
⁰⁴⁵₀₁₂
ด้้้้้็็็็็้้้้้็็็็็ ด้้้้้็็็็็้้้้้็็็็็ ด้้้้้็็็็็้้้้้็็็็็

#	Quotation Marks
#
#	Strings which contain misplaced quotation marks; can cause encoding errors

'
"
''
""
'"'
"''''"'"
"'"'"''''"
<foo val=“bar” />
<foo val=“bar” />
<foo val=”bar“ />
<foo val=`bar' />

#	Two-Byte Characters
#
#	Strings which contain two-byte characters: can cause rendering issues or character-length issues

田中さんにあげて下さい
パーティーへ行かないか
和製漢語
部落格
사회과학원 어학연구소
찦차를 타고 온 펲시맨과 쑛다리 똠방각하
社會科學院語學研究所
울란바토르
𠜎𠜱𠝹𠱓𠱸𠲖𠳏
```

```
#	Japanese Emoticons
#
#	Strings which consists of Japanese-style emoticons which are popular on the web

ヽ༼ຈل͜ຈ༽ﾉ ヽ༼ຈل͜ຈ༽ﾉ
(｡◕ ∀ ◕｡)
｀ｨ(´∀｀∩
__ﾛ(,_,*)
・(￣∀￣)・:*:
ﾟ･✿ヾ╲(｡◕‿◕｡)╱✿･ﾟ
,。・:*:・゜’( ☻ ω ☻ )。・:*:・゜’
(╯°□°）╯︵ ┻━┻)
(ﾉಥ益ಥ）ﾉ ┻━┻
┬─┬ノ( º _ ºノ)
( ͡° ͜ʖ ͡°)
¯\_(ツ)_/¯

#	Emoji
#
#	Strings which contain Emoji; should be the same behavior as two-byte characters, but not always

😍
👩🏽
👨‍🦰 👨🏿‍🦰 👨‍🦱 👨🏿‍🦱 🦹🏿‍♂️
👾 🙇 💁 🙅 🙆 🙋 🙎 🙍
🐵 🙈 🙉 🙊
❤️ 💔 💌 💕 💞 💓 💗 💖 💘 💝 💟 💜 💛 💚 💙
✋🏿 💪🏿 👐🏿 🙌🏿 👏🏿 🙏🏿
👨‍👩‍👦 👨‍👩‍👧‍👦 👨‍👨‍👦 👩‍👩‍👧 👨‍👦 👨‍👧‍👦 👩‍👦 👩‍👧‍👦
🚾 🆒 🆓 🆕 🆖 🆗 🆙 🏧
0️⃣ 1️⃣ 2️⃣ 3️⃣ 4️⃣ 5️⃣ 6️⃣ 7️⃣ 8️⃣ 9️⃣ 🔟

#	Regional Indicator Symbols
#
#	Regional Indicator Symbols can be displayed differently across
#	fonts, and have a number of special behaviors

🇺🇸🇷🇺🇸 🇦🇫🇦🇲🇸
🇺🇸🇷🇺🇸🇦🇫🇦🇲
🇺🇸🇷🇺🇸🇦

#	Unicode Numbers
#
#	Strings which contain unicode numbers; if the code is localized, it should see the input as numeric

１２３
١٢٣

#	Right-To-Left Strings
#
#	Strings which contain text that should be rendered RTL if possible (e.g. Arabic, Hebrew)

جزيرتي باستخدام أن دنو. إذ هنا؟ الستار وتنصيب كان. أهّل ايطاليا، بريطانيا-فرنسا قد أخذ. سليمان، إتفاقية بين ما, يذكر الحدود أي بعد, معاملة بولندا، الإطلاق عل إيو
בְּרֵאשִׁית, בָּרָא אֱלֹהִים, אֵת הַשָּׁמַיִם, וְאֵת הָאָרֶץ
הָיְתָהtestالصفحات التّحول
﷽
ﷺ
```


minimaxir / big-list-of-naughty-strings
Public
Watch 859
Fork 2.1k
Star 44.6k
Code
Issues 61
Pull requests 34
Actions
Projects
Wiki
Security
Insights
master
1 branch
0 tags
Go to file
Add file
Code
minimaxir Merge pull request #226 from caasi/patch-1
db33ec7 on Apr 17, 2021
273 commits
naughtystrings Merge pull request #191 from dmolesUC/go-module 4 years ago
scripts Avoid globbing and word splitting when expanding variables 5 years ago
.gitattributes Null character defeated! 8 years ago
LICENSE Update LICENSE 4 years ago
README.md Merge branch 'master' into master 3 years ago
blns.base64.json added jinja2 injections 3 years ago
blns.base64.txt added jinja2 injections 3 years ago
blns.json Merge pull request #211 from doroshenko/master 3 years ago
blns.txt Index XSS strings 2 years ago
package.json Simplify repository config 7 years ago
README.md
Big List of Naughty Strings
The Big List of Naughty Strings is an evolving list of strings which have a high probability of causing issues when used as user-input data. This is intended for use in helping both automated and manual QA testing; useful for whenever your QA engineer walks into a bar.
Why Test Naughty Strings?
Even multi-billion dollar companies with huge amounts of automated testing can't find every bad input. For
About
The Big List of Naughty Strings is a list of strings which have a high probability of causing issues when used as user-input data.
Readme
MIT license
44.6k stars
859 watching
2.1k forks
Releases
No releases published
Packages
No packages published
Used by 30
+ 22
Contributors 72

```
// Each line in the file is a test case, and the test method runs once for each line in the file.
public static string[] InjectionStrings => File.ReadAllLines("blns-injection.txt");

[TestCaseSource(nameof(InjectionStrings))]
public void Constructor_Should_Reject_InvalidData(string injectionString)
{
    Assert.Throws<DomainPrimitiveArgumentException<string>>(() => new ProductId(injectionString));
}

[Test]
public void Constructor_Should_Reject_EmptyData()
{
    Assert.Throws<DomainPrimitiveArgumentException<string>>(() => new ProductId(null!));
    Assert.Throws<DomainPrimitiveArgumentException<string>>(() => new ProductId(string.Empty));
}
```

Security requirements and test cases



- Users should *only* be able to consume API_1
  - User should not be able to consume API_2 and API_3
- Users should only be able to alter data in their own tenat/organization
  - Users should not be able to consume data outside their organization

```
[Test]
public async Task GetWith_ReturnsNotFound_IfValidClaimButNotExisting()
{
    var productRepository = Mock.Of<IProductRepository>();
    var permissionService = Mock.Of<IPermissionService>();

    Mock.Get(permissionService).SetupGet(service => service.CanReadProducts).Returns(true);

    var productService = new ProductService(productRepository, permissionService);
    var productId = new ProductId("notfound");

    var (result, product) = await productService.GetWith(productId);

    Assert.That(result, Is.EqualTo(ReadDataResult.NotFound));
    Assert.Null(product);
}
```


```
[Test]
public async Task GetById_ShouldReturn401_WhenAnonymous()
{
    var response = await _client.GetAsync("api/product/123GQWE");

    Assert.That(response.StatusCode, Is.EqualTo(HttpStatusCode.Unauthorized));
}

[Test]
public async Task GetProductById_ShouldReturn401_WhenAuthorizedButHasWrongScope()
{
    await AuthorizeHttpClient(ProductScope.Write);
    // Use a token with wrong scope, GetProductById requires products.read
    var response = await _client.GetAsync("/api/product/123GQWE");

    Assert.That(response.StatusCode, Is.EqualTo(HttpStatusCode.Forbidden));
}
```