Технически университет – София

Изпитен проект по дисциплината „Семантичен уеб“:

Криптографски методи за защита на информацията

Изготвил:

Исмаил Осама Салех
ФКСТ, Компютърно и софтуерно инженерство, гр. 223, фак. № 121323039;

Ръководител:

доц. д-р инж. Аделина Алексиева

13.01.2024 г.

1. Област и обхват на онтологията

Каква е областта, която ще се обхваща от онтологията?

Областта на онтологията обхваща съставните части на криптографските методи за защита на информация. Онтологията съответно ще включва описание на тези части, както и техни конкретни изграждащи компоненти и взаимовръзки.

За какво ще се използва онтологията?

Онтологията ще се използва като наръчник за основните характеристики и компоненти на криптографията за защита на информация. По този начин се улеснява усвояването на знания, необходими за правилното познаване и прилагане на различните техники.

На какви въпроси отговаря онтологията?

Онтологията дава отговор на следните въпроси:

- Какво е криптографски метод и какви са приложенията му?
- Какви принципи трябва да следва една крипто система?
- Каква е разликата между симетричните и асиметричните криптографски методи?
- Какво е хеш функция и как се използва в криптографията?;
- Какво е алгоритъм за криптиране и как работи?;
- Какво е алгоритъм за декриптиране и как работи?;
- Как се генерират ключове в криптографските методи?;
- Какво е алгоритъм за обмяна на ключове и как работи?;
- Какво е код за удостоверяване на съобщение (MAC) и как се използва в криптографията?;
- Какво е цифров подпис?;
- Какво е блоков шифър и как се различава от поточен шифър? Какви са разликите между тях;
- Каква е връзката между силата на криптографския метод и неговата производителност?;
- Размерът на блока в блоков шифър трябва ли да е равен на размера на ключа, който се използва за криптирането му?;
- Какъв е ефектът от дължината на ключа върху сигурността на един криптографски метод?

Кой ще използва и поддържа онтологията?

Онтологията е подходяща за запознаване на ученици и студенти с основите на криптографските методи за защита на информация. Съответно онтологията служи и като идеална основа за надграждане на базови знания по киберсигурност. Тя притежава публично репо в GitHub, където ентусиасти могат да разглеждат, теглят (клонират или fork-ват) и правят „pull requests (PRs)”. PRs представляват заявки от трети лица към автора/началния разработчик за въвеждане на определени промени в репото. По този начин настоящата разработка се актуализира и от общността освен само от автора.

2. Преизползване на съществуващи онтологии

За направата на текущата онтология не са използвани други цялостни съществуващи онтологии.

3. Ключови термини в онтологията

Класовете, които ще се използват в онтологията и ще изразяват съставните части на криптографските методи за защита на информация, са:

- CryptographicMethod- основен клас на онтологията. Показва криптографския метод като обект, свързан чрез обектни свойства с останалите главни класове в йерархията- AsymmetricCryptographicMethod, SymmetricCryptographicMethod, HashingFunction, MessageAuthenticationCode, EncryptionAlgorithm, DecryptionAlgorithm, KeyGenerationAlgorithm, KeyExchangeAlgorithm.
- AsymmetricCryptographicMethod- Асиметричното криптиране използва математически свързана двойка ключове за шифроване и дешифроване: публичен ключ и частен ключ. Ако публичният ключ се използва за шифроване, тогава свързаният с него частен ключ се използва за дешифроване. Ако частният ключ се използва за шифроване, тогава свързаният с него публичен ключ се използва за дешифроване.
- DigitalSignatureAlgorithm- Подклас на AsymmetricCryptographicMethod. Техниките при цифровите подписи целят осигуряването на автентичност, цялост и неопровержимост за цифровите съобщения или документи.
- DecryptionAlgorithm- Декриптирането приема шифротекст и таен ключ като вход. После криптиранта информация (шифротекстът) се преобразува обратно в нейния оригинален вид.
- EncryptionAlgorithm- Криптирането е математическа процедура или набор от правила, използвани за преобразуване на четлив текст (некриптирана информация) в криптограм (криптирана информация).
- HashingFunction- Функциите тук се използват за преобразуване на данни с произволен размер в такива с фиксиран.
- KeyExchangeAlgorithm- Класът съдържа криптографски методи които осъществяват сигурния обмен на криптографски ключове между две страни. Целта на алгоритъма за обмен на ключове е да позволи на две страни да се споразумеят за споделен секретен ключ.
- KeyGenerationAlgorithm- Алгоритми за генериране на ключове. Сигурността на много криптографски системи разчита на генерирането на силни и непредсказуеми криптографски ключове.
- MessageAuthenticationCode- Включва криптографски техники, използвани за проверка на целостта и автентичността на определено съобщение.
- BlockCipher- Блоков шифър.
- StreamCipher- Поточен шифър.

Класовете, използвани в онтологията, са свързани помежду си чрез обектни свойства (Object properties); Индивидите (Individuals) са описани чрез свойства за данни (Data properties); Индивид може да се свърже с друг посредством едно или няколко обектни свойства.

Всички обектни свойства са Functional, защото един индивид може да притежава една единствена стойност за тях. Те са следните:

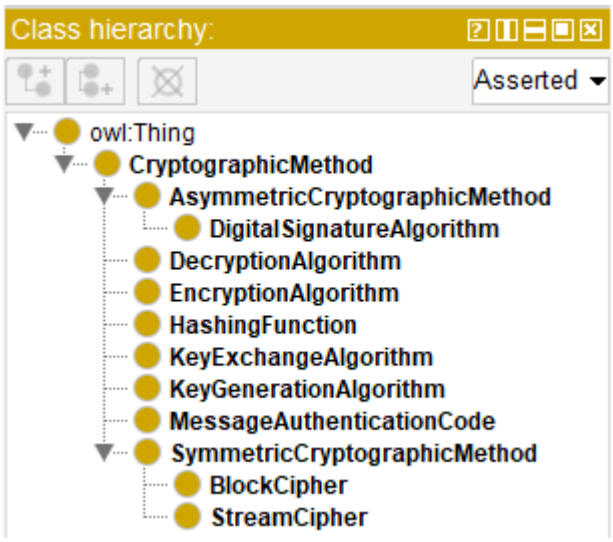
- exchangesKey – свойство, свързващо класовете CryptographicMethod и KeyExchangeAlgorithm.
- generatesKey – свойство, свързващо класовете CryptographicMethod и KeyGenerationAlgorithm.
- isAsymmetric – свойство, свързващо класовете CryptographicMethod и AsymmetricCryptographicMethod. Инверсно на isSymmetric.
- isDecryption – свойство, свързващо класовете CryptographicMethod и DecryptionAlgorithm.
- isEncryption – свойство, свързващо класовете CryptographicMethod и EncryptionAlgorithm.
- isHashing – свойство, свързващо класовете CryptographicMethod и HashingFunction.
- isOfTypeMAC – свойство, свързващо класовете CryptographicMethod и MessageAuthenticationCode.
- isSymmetric – свойство, свързващо класовете CryptographicMethod и SymmetricCryptographicMethod. Инверсно на isAsymmetric.
- signsDigitalSignature – свойство, свързващо класовете AsymmetricCryptographicMethod и DigitalSignatureAlgorithm. Равно е на обкетното свойство verifiesDigitalSignature, за да се покаже на учениците, че криптографските методи, които генерират цифрови подписи, също служат да ги верифицират в една крипто система.
- verifiesDigitalSignature – свойство, свързващо класовете AsymmetricCryptographicMethod и DigitalSignatureAlgorithm. Равно е на обкетното свойство signsDigitalSignature поради гореспоменатата причина.

Свойства за данни (data properties) – използват се, за да опишат определен индивид от даден клас чрез определени характеристики. Отново всички са Functional.

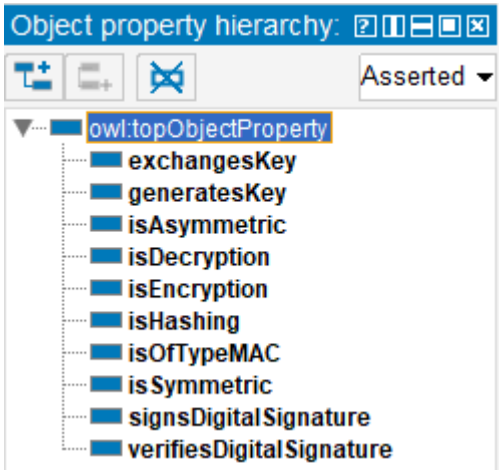
- blockSize (тип: xsd:integer) – размер на блок данни в битове.
- decryptionSpeed (една от следните стойности: {“high”, “medium”, “slow”}) – скорост на декриптиране.
- encryptionSpeed (една от следните стойности: {“high”, “medium”, “slow”}) – скорост на криптиране.
- hashLength (тип: xsd:integer) – размер на хеш в битове.
- isPrinciple (тип: xsd:boolean) – дава информация дали индивид е принцип или „актьор“ в криптографските методи.
- keyLength (тип: xsd:integer) – размер на ключ в битове.
- securityStrength (една от следните стойности: {“weak”, ”medium”, ”strong”, ”very strong”}) – сила на сигурност.
- supportsTweakableBlockEncryption (тип: xsd:boolean) – дали индивид поддържа режим tweakable-block криптиране. В този режим различен ключ за всеки блок-данни се използва, което прави криптиращия процес по-устойчив на атаки от вида диференциален криптоанализ и линеен криптоанализ.

4. Демонстрация чрез графични извадки от онтологията

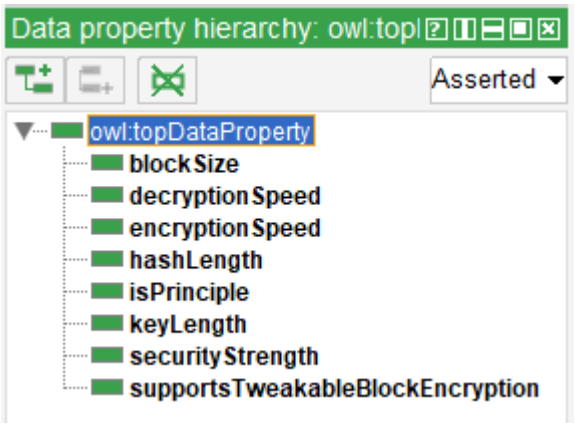
Фиг. 0. Метрики на онтологията.



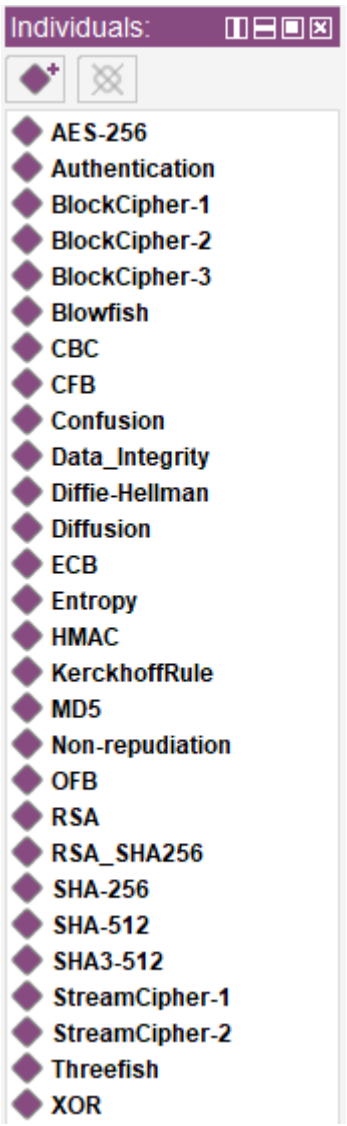
Фиг. 1. Класова йерархия на онтологията.



Фиг. 2. Обектни свойства на онтологията.

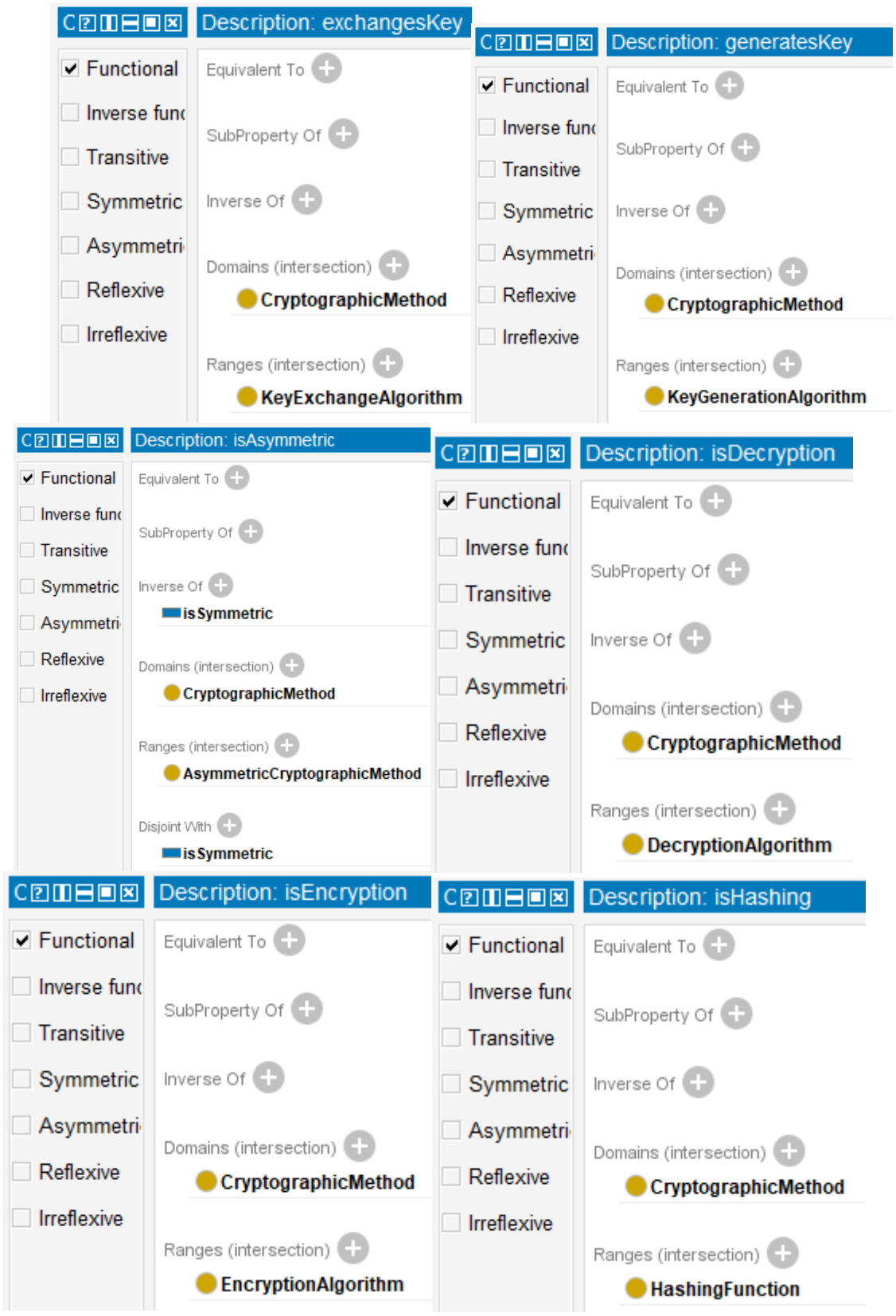


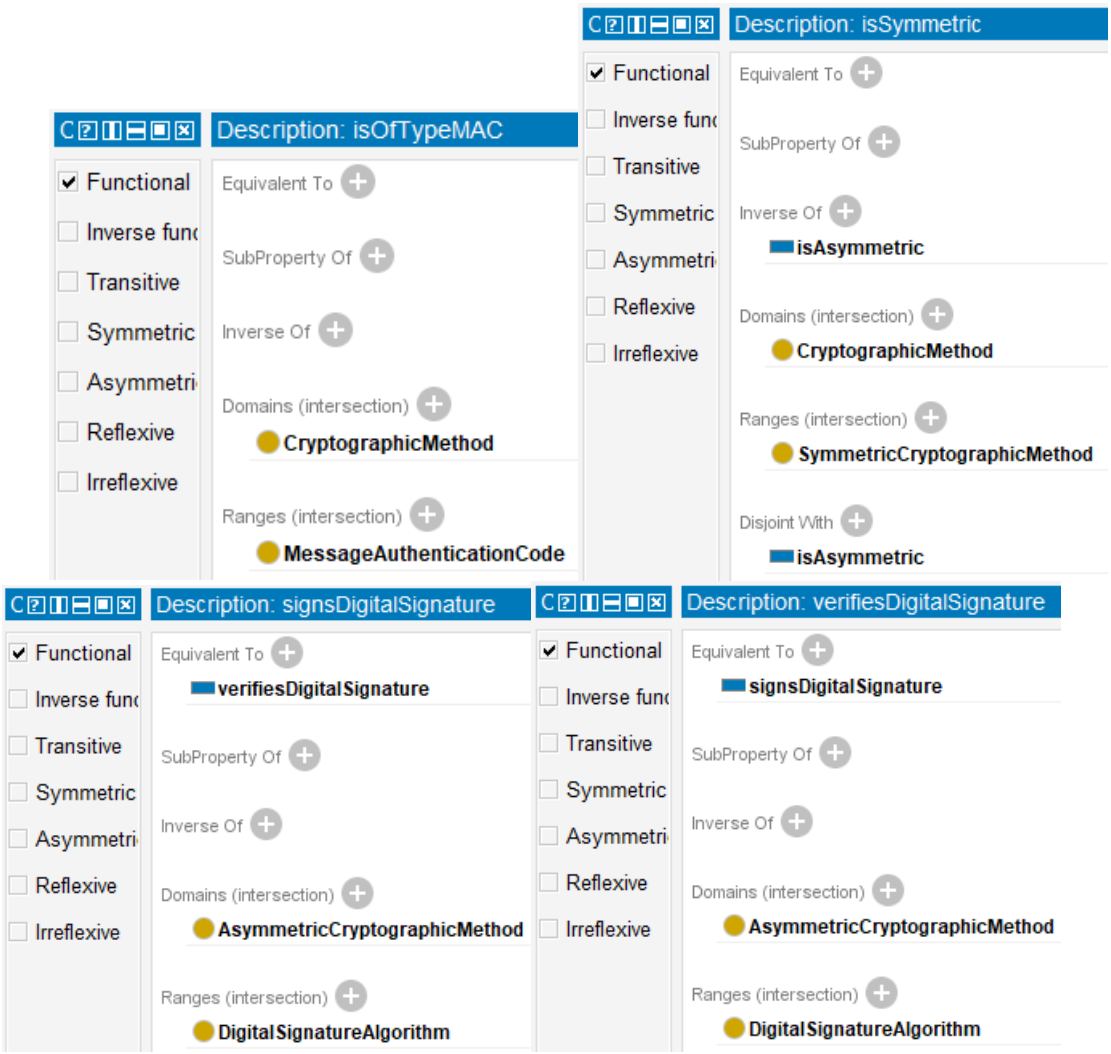
Фиг. 3. Свойства за данни.



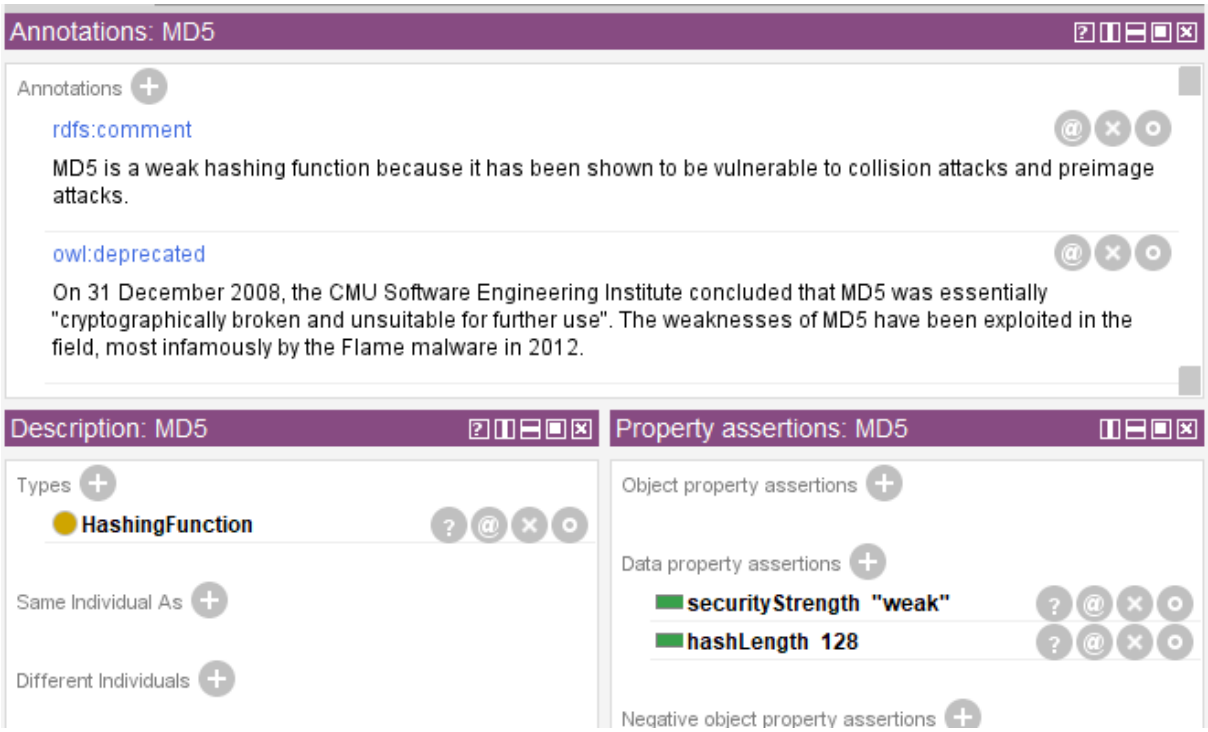
Фиг. 4. Индивиди в онтологията.

Nota bene: Като разглеждате класовете и/или инстанциите им, ще забележите че някои индивиди са зачеркнати. Това е така, защото съответните инстанции притежават „deprecated" анотация. Т.е. технологиите са стари/неподдържани и неподходящи за имплементиране в бизнес решения.

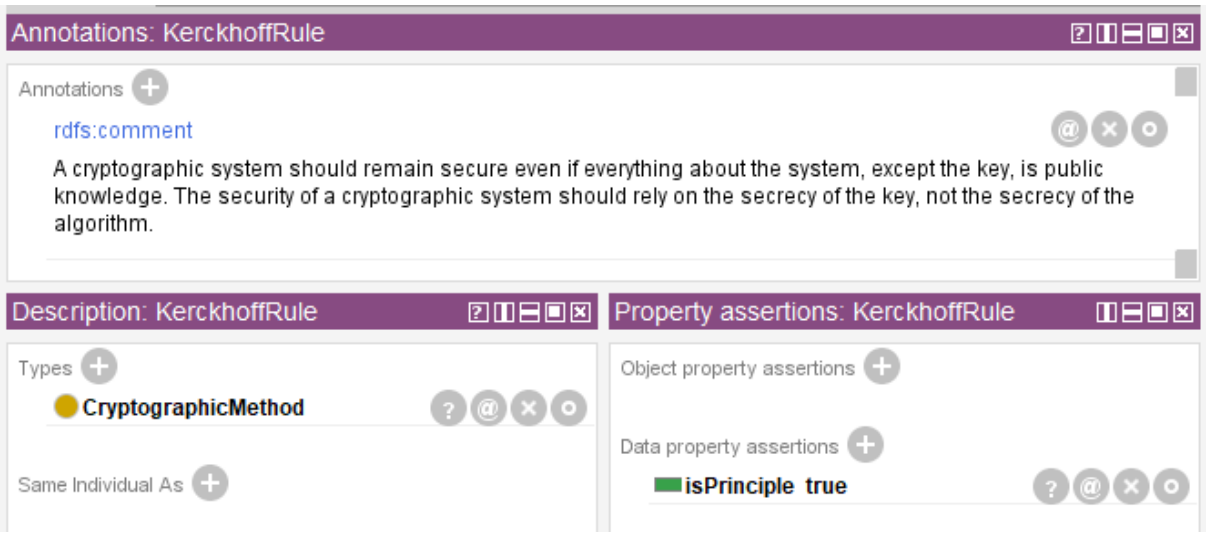

Description: exchangesKey
Functional
Inverse func
Transitive
Symmetric
Asymmetri
Reflexive
Irreflexive
Equivalent To
SubProperty Of
Inverse Of
Domains (intersection)
CryptographicMethod
Ranges (intersection)
KeyExchangeAlgorithm
Description: generatesKey
Functional
Inverse func
Transitive
Symmetric
Asymmetri
Reflexive
Irreflexive
Equivalent To
SubProperty Of
Inverse Of
Domains (intersection)
CryptographicMethod
Ranges (intersection)
KeyGenerationAlgorithm
Description: isAsymmetric
Functional
Inverse func
Transitive
Symmetric
Asymmetri
Reflexive
Irreflexive
Equivalent To
SubProperty Of
Inverse Of
isSymmetric
Domains (intersection)
CryptographicMethod
Ranges (intersection)
AsymmetricCryptographicMethod
Disjoint With
isSymmetric
Description: isDecryption
Functional
Inverse func
Transitive
Symmetric
Asymmetri
Reflexive
Irreflexive
Equivalent To
SubProperty Of
Inverse Of
Domains (intersection)
CryptographicMethod
Ranges (intersection)
DecryptionAlgorithm
Description: isEncryption
Functional
Inverse func
Transitive
Symmetric
Asymmetri
Reflexive
Irreflexive
Equivalent To
SubProperty Of
Inverse Of
Domains (intersection)
CryptographicMethod
Ranges (intersection)
EncryptionAlgorithm
Description: isHashing
Functional
Inverse func
Transitive
Symmetric
Asymmetri
Reflexive
Irreflexive
Equivalent To
SubProperty Of
Inverse Of
Domains (intersection)
CryptographicMethod
Ranges (intersection)
HashingFunction



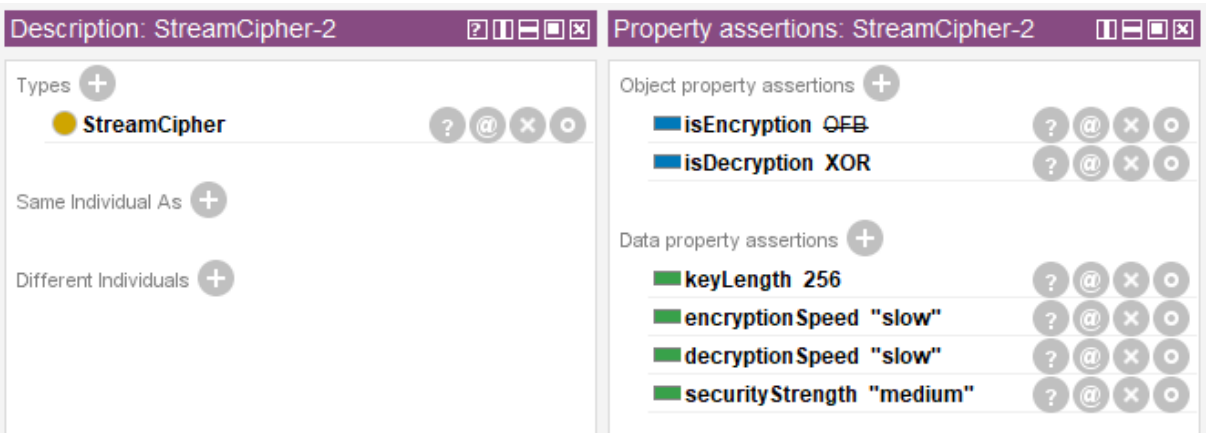
Фиг. 5. Обектните свойства на онтологията – разгърнати.



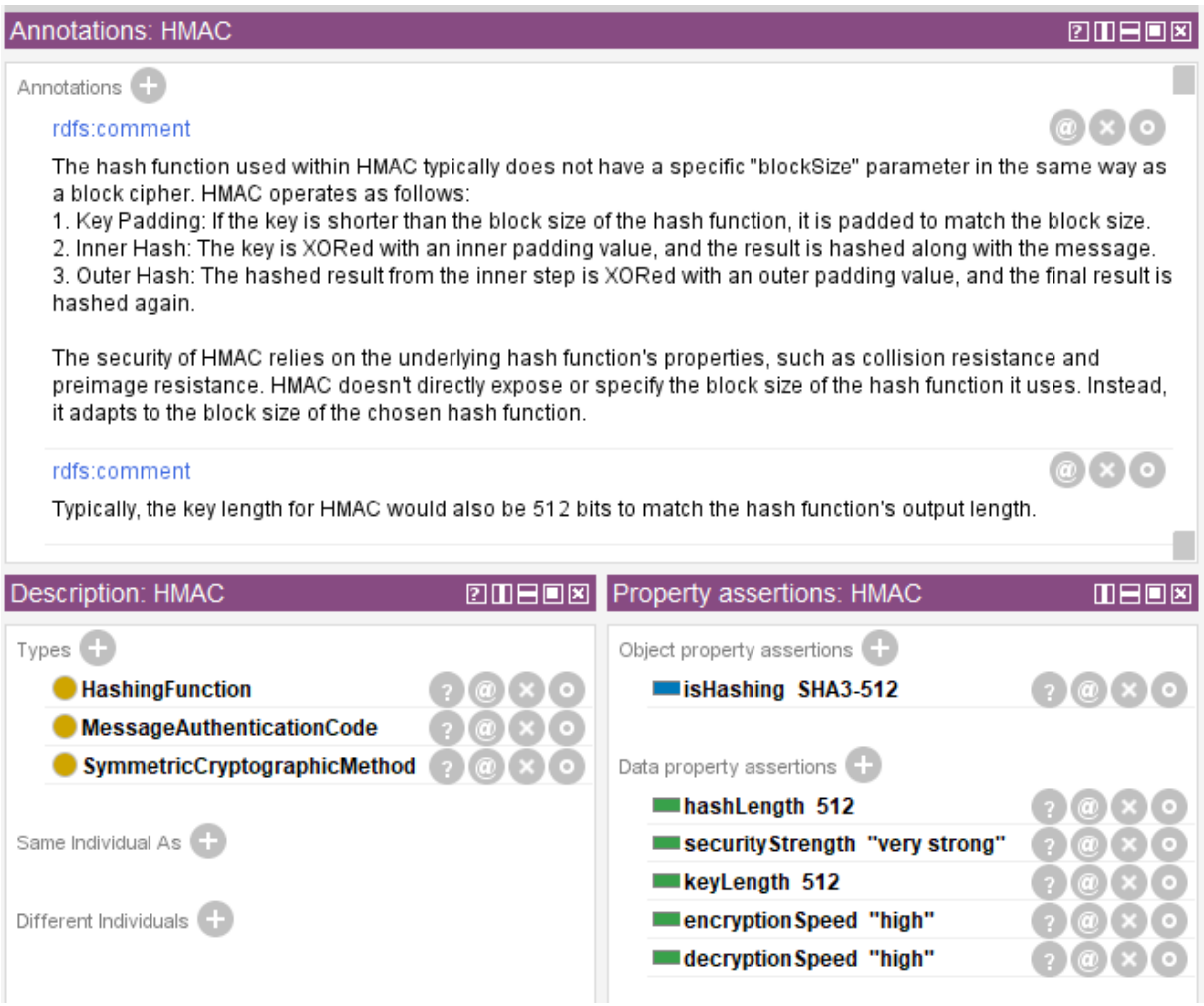
Фиг. 6. Deprecated индивид – MD5.



Фиг. 18. Индивид, представящ принцип в криптографските методи за защита на информацията.



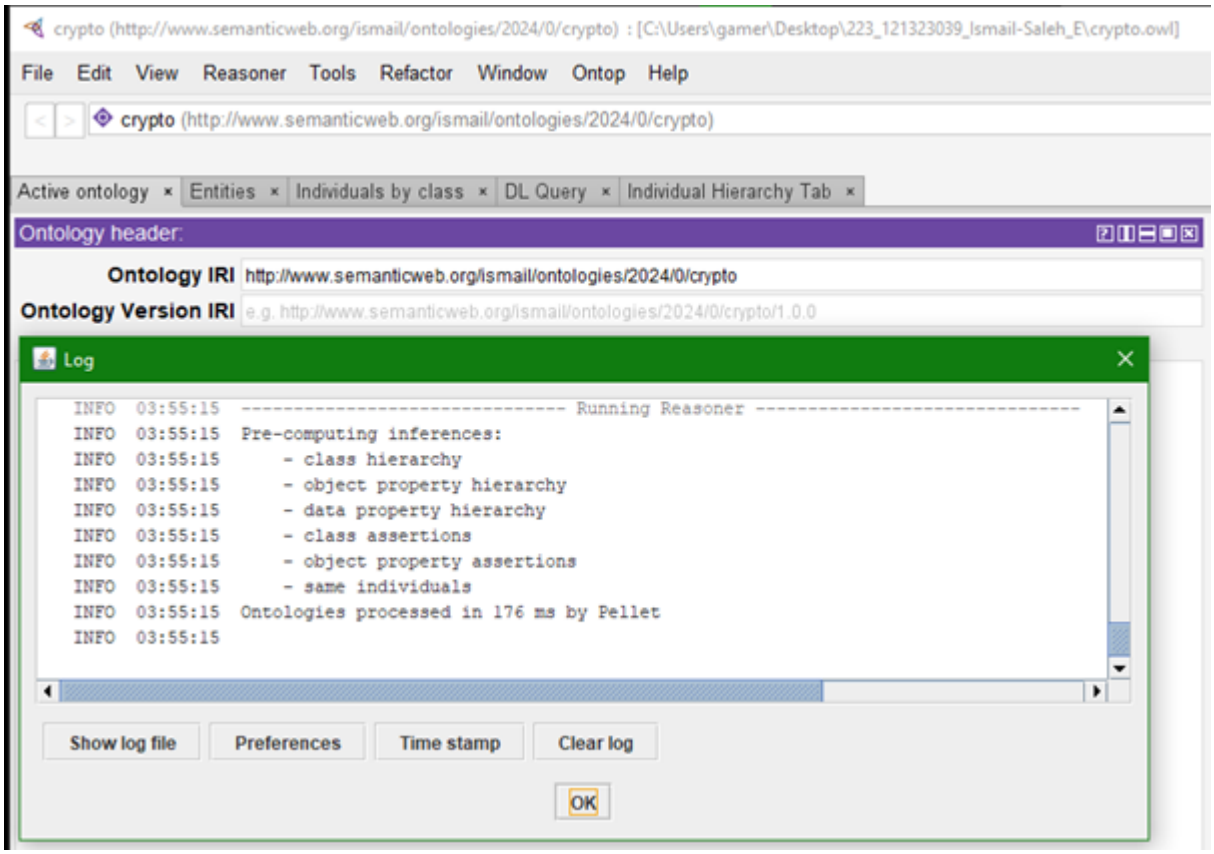
Фиг. 7. Индивид, представящ примерен поточен шифър.



Фиг. 20. Индивид HMAC.

Фиг. 8. Индивид, където използвам анотацията rdfs:label с цел да коригирам първоначалното възложено име.

5. Валидиране на логиката на онтологията чрез reasoner – Pellet



Фиг. 9. Pellet Reasoner изход.

6. SPARQL заявки

1. Извежда се като резултат всички индивиди, които са принципи (чрез свойството за данни isPrinciple):

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX crypto: <http://www.semanticweb.org/ismail/ontologies/2024/0/crypto#>

SELECT DISTINCT ?Principles
WHERE {
  ?individual rdf:type owl:NamedIndividual .
  ?Principles crypto:isPrinciple true .
}
```

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX crypto: <http://www.semanticweb.org/ismail/ontologies/2024/0/crypto#>

SELECT DISTINCT ?Principles
WHERE {
  ?individual rdf:type owl:NamedIndividual .
  ?Principles crypto:isPrinciple true .
}
```

Principles
KerckhoffRule
Non-repudiation
Authentication
Diffusion
Data_Integrity
Confusion
Entropy

Фиг. 10. Изход на първата заявка през Protégé.

2. Извежда се като изход всички блокови шифри, техните алгоритми за криптиране и декриптиране и сила на сигурност:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX crypto: <http://www.semanticweb.org/ismail/ontologies/2024/0/crypto#>

SELECT ?blockCipher ?encryption ?decryption ?securityStrength
WHERE {
  ?blockCipher rdf:type crypto:BlockCipher .
  ?blockCipher crypto:isEncryption ?encryption .
  ?blockCipher crypto:isDecryption ?decryption .
  ?blockCipher crypto:securityStrength ?securityStrength .
}
```

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX crypto: <http://www.semanticweb.org/ismail/ontologies/2024/0/crypto#>

SELECT ?blockCipher ?encryption ?decryption ?securityStrength
WHERE {
  ?blockCipher rdf:type crypto:BlockCipher .
  ?blockCipher crypto:isEncryption ?encryption .
  ?blockCipher crypto:isDecryption ?decryption .
  ?blockCipher crypto:securityStrength ?securityStrength .
}
```

blockCipher	encryption	decryption	securityStrength
BlockCipher-2	Blowfish	Blowfish	"strong"
BlockCipher-3	Threefish	Threefish	"very strong"
BlockCipher-1	AES-256	AES-256	"strong"

Фиг. 11. Изход на втората заявка в Protégé.

3. Извежда всички хеш функции, дължината на digest-а (изхода), размер на блока и сила на защита:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX crypto: <http://www.semanticweb.org/ismail/ontologies/2024/0/crypto#>

SELECT ?hashingFunction ?hashLength ?blockSize ?securityStrength
WHERE {
  ?hashingFunction rdf:type crypto:HashingFunction .
  ?hashingFunction crypto:hashLength ?hashLength .
  ?hashingFunction crypto:blockSize ?blockSize .
  ?hashingFunction crypto:securityStrength ?securityStrength .
}
```

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX crypto: <http://www.semanticweb.org/ismail/ontologies/2024/0/crypto#>

SELECT ?hashingFunction ?hashLength ?blockSize ?securityStrength
WHERE {
  ?hashingFunction rdf:type crypto:HashingFunction .
  ?hashingFunction crypto:hashLength ?hashLength .
  ?hashingFunction crypto:blockSize ?blockSize .
  ?hashingFunction crypto:securityStrength ?securityStrength .
}
```

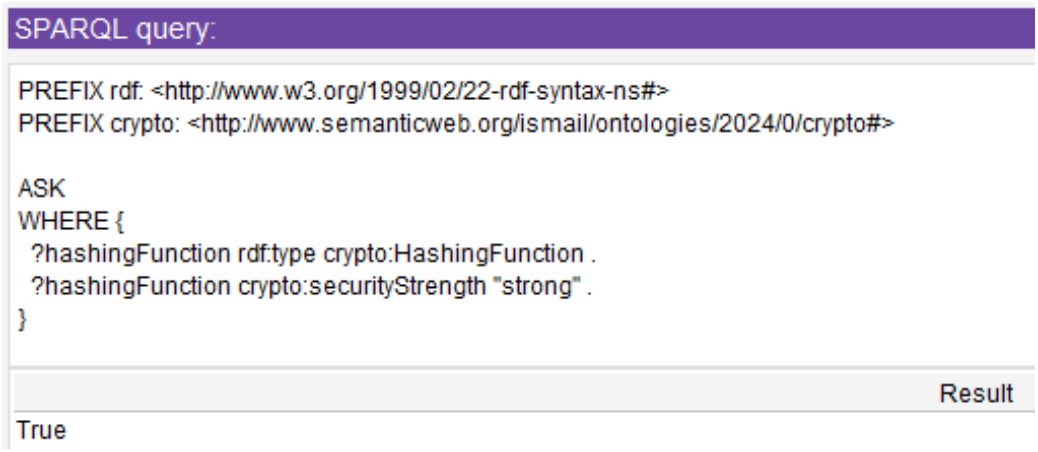
hashingFunction	hashLength	blockSize	securityStrength
SHA-256	"256"^^<http://www.w3.org/2001/XMLSchema#integer>	"512"^^<http://www.w3.org/2001/XMLSchema#integer>	"strong"
~~MD5~~	"128"^^<http://www.w3.org/2001/XMLSchema#integer>	"512"^^<http://www.w3.org/2001/XMLSchema#integer>	"weak"
SHA3-512	"512"^^<http://www.w3.org/2001/XMLSchema#integer>	"1088"^^<http://www.w3.org/2001/XMLSchema#integer>	"very strong"
SHA-512	"512"^^<http://www.w3.org/2001/XMLSchema#integer>	"1024"^^<http://www.w3.org/2001/XMLSchema#integer>	"very strong"

Фиг. 12. Изход на третата заявка в Protégé.

4. Извежда дали съществуват индивиди от клас HashingFunction, които притежават securityStrength = strong:

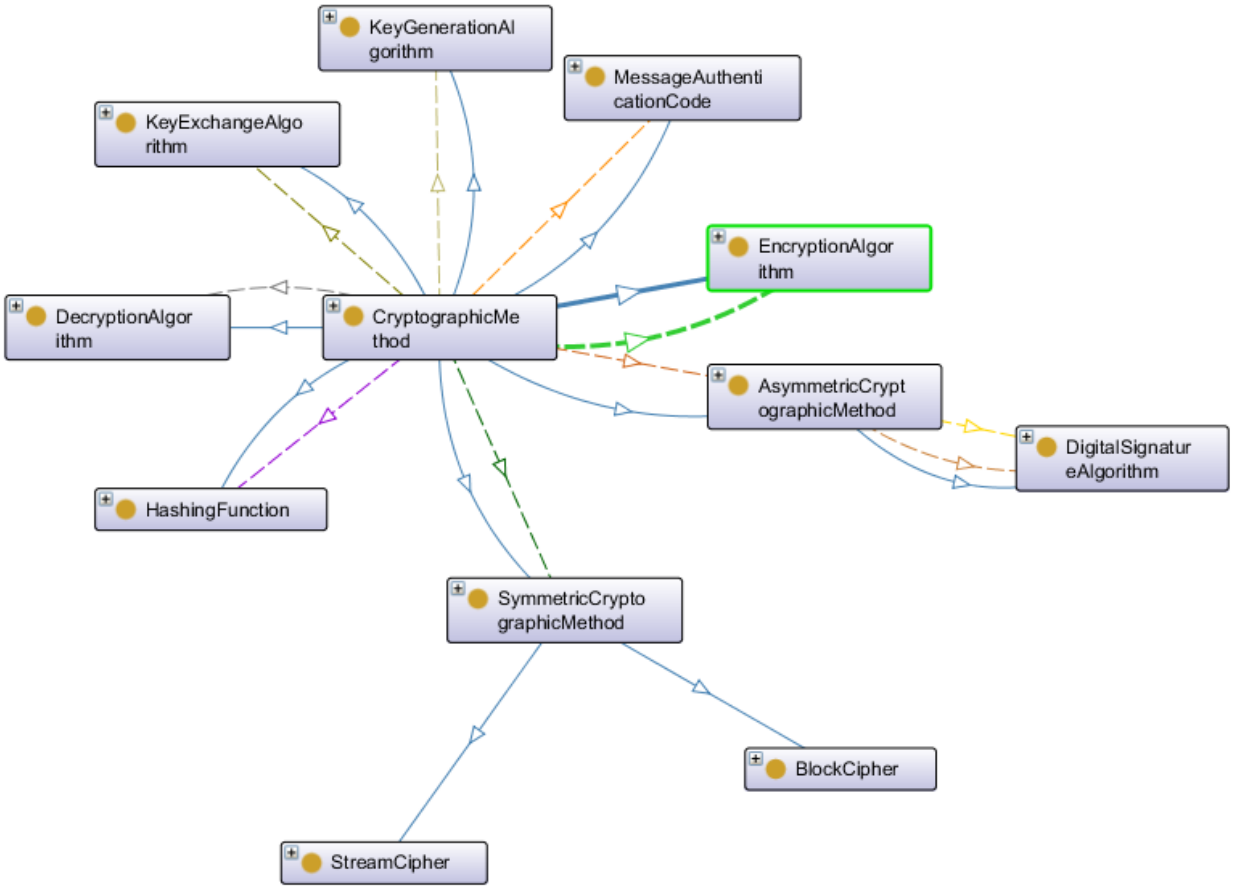
```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX crypto: <http://www.semanticweb.org/ismail/ontologies/2024/0/crypto#>


ASK
WHERE {
  ?hashingFunction rdf:type crypto:HashingFunction .
  ?hashingFunction crypto:securityStrength "strong" .
}
```



Фиг. 13. Изход на четвъртата заявка.

7. Графичен модел на онтологията



Фиг. 14. Графичен модел на онтологията.