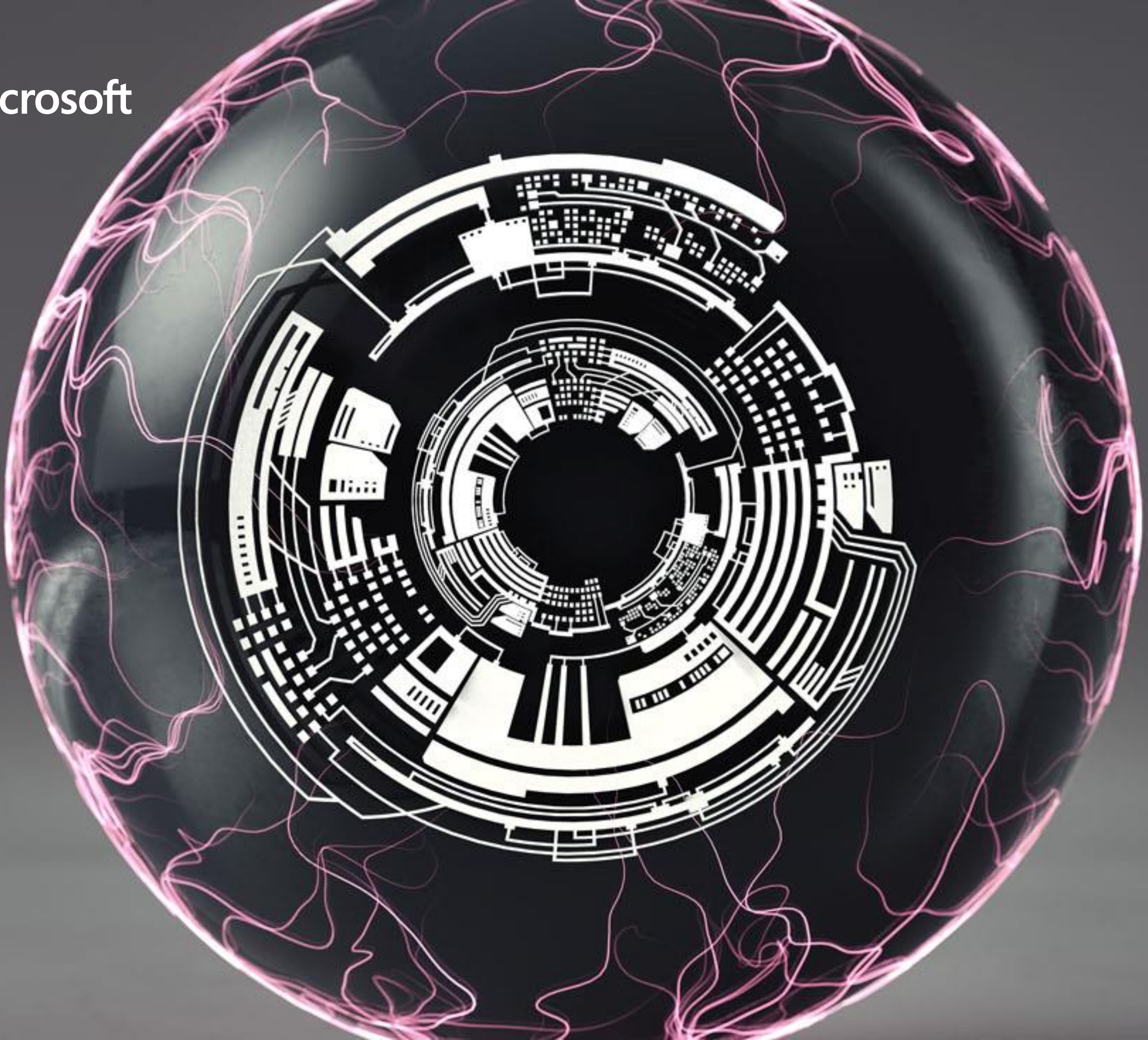
Microsoft

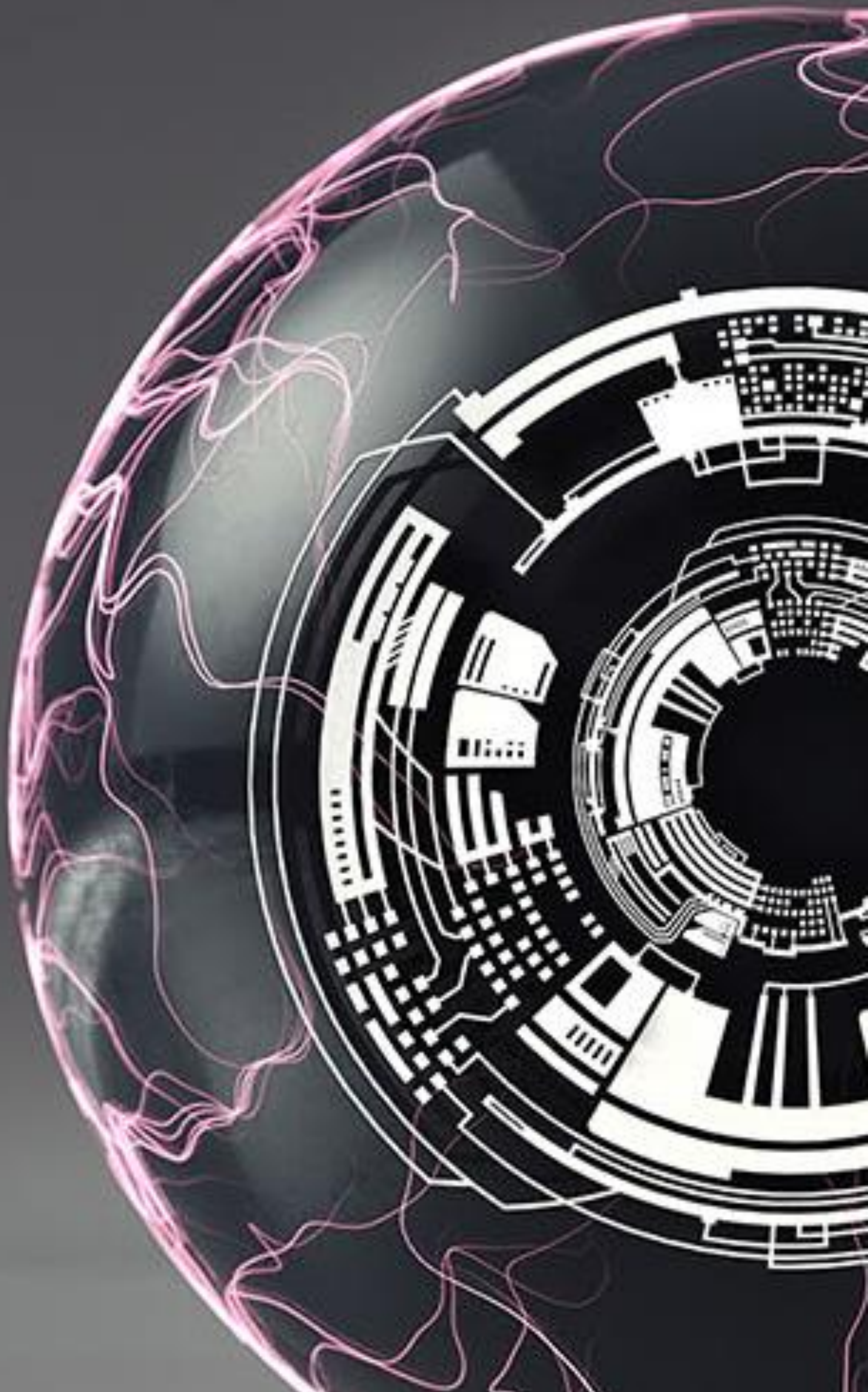


Microsoft Research Faculty Summit 2016

Microsoft

Private Predictions with Homomorphic Encryption

Kim Laine
kim.laine@microsoft.com



Presentation Outline

Privacy in Prediction

SEAL

Examples

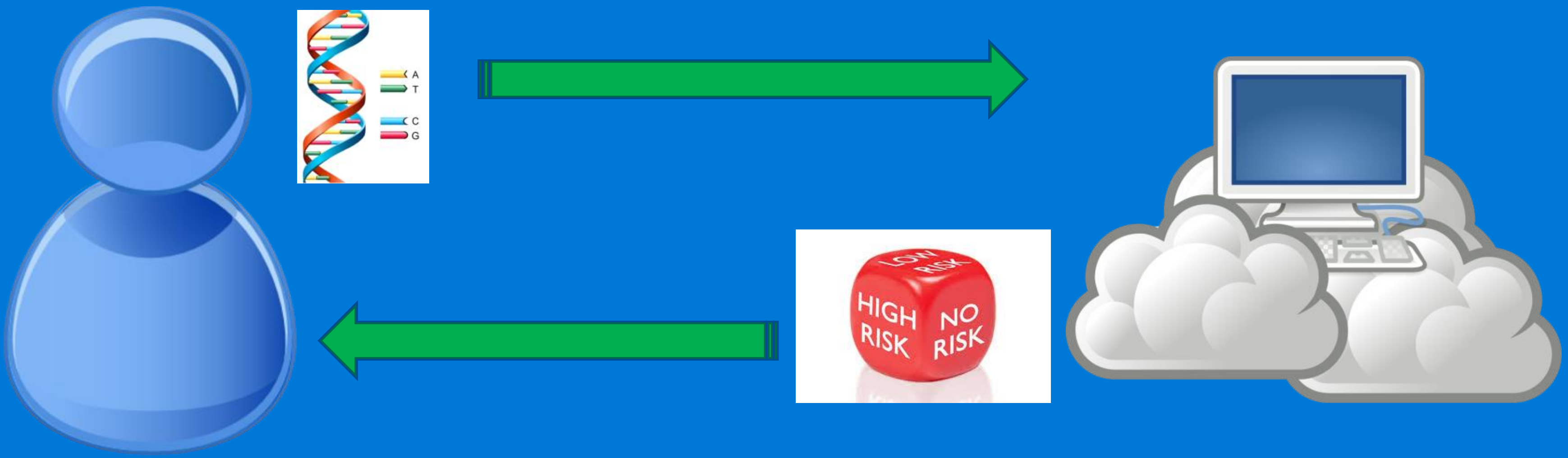
Privacy in Prediction

Medical

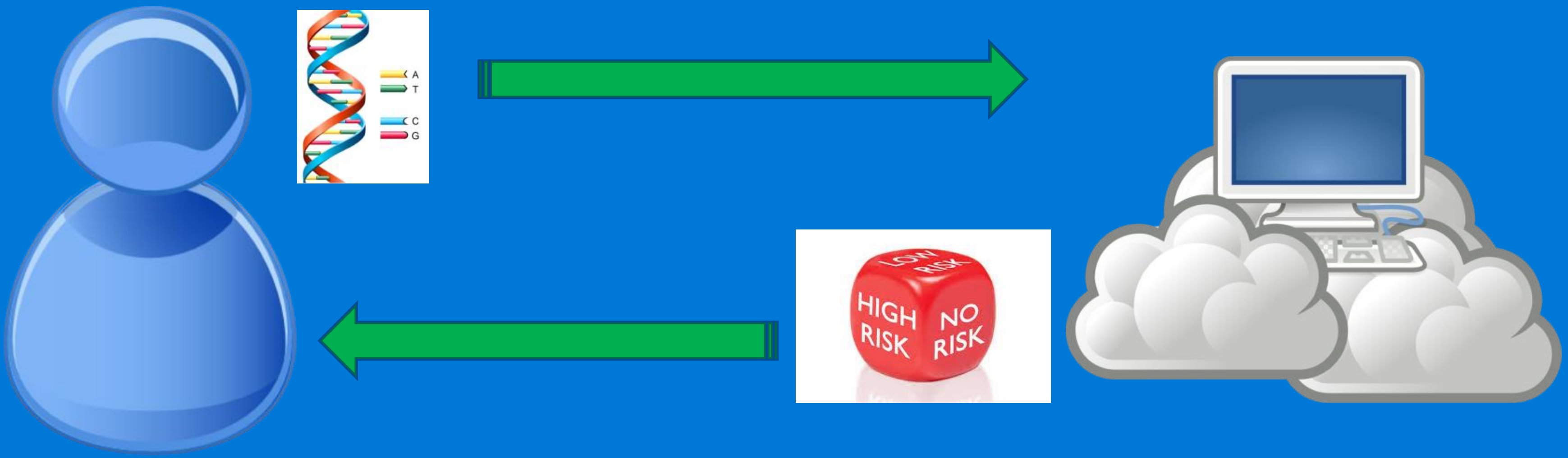
Genomic

Financial

Wait! What about Privacy?

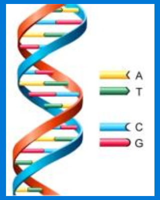

A
T
C
G
LOW RISK
HIGH RISK
NO RISK

Who else is going to see your DNA sequence and the prediction?



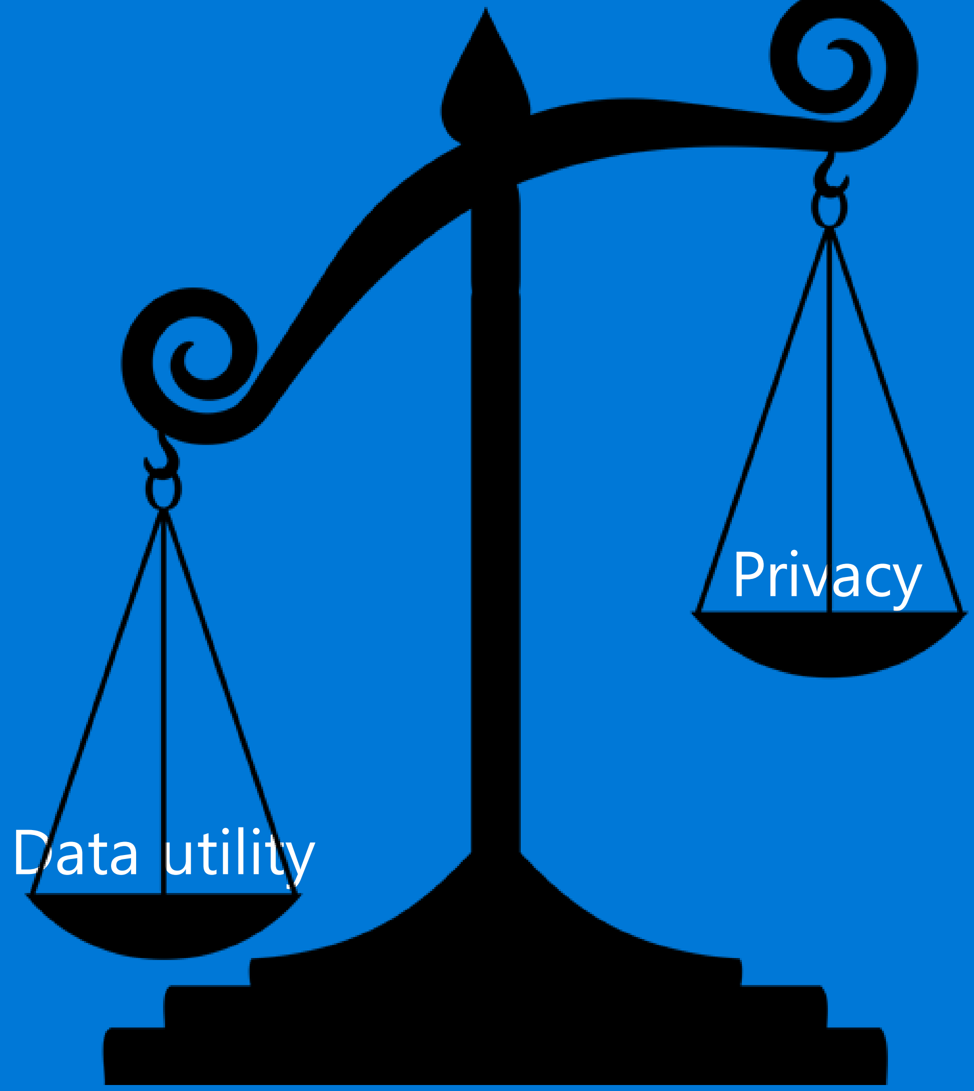
Who else is going to see your DNA sequence and the prediction?

“Sorry, your DNA does not match this job description.”



“Here is an advertisement that according to your DNA you will not be able to resist.”

“We are not giving you this loan because it is not in your DNA to pay it back.”

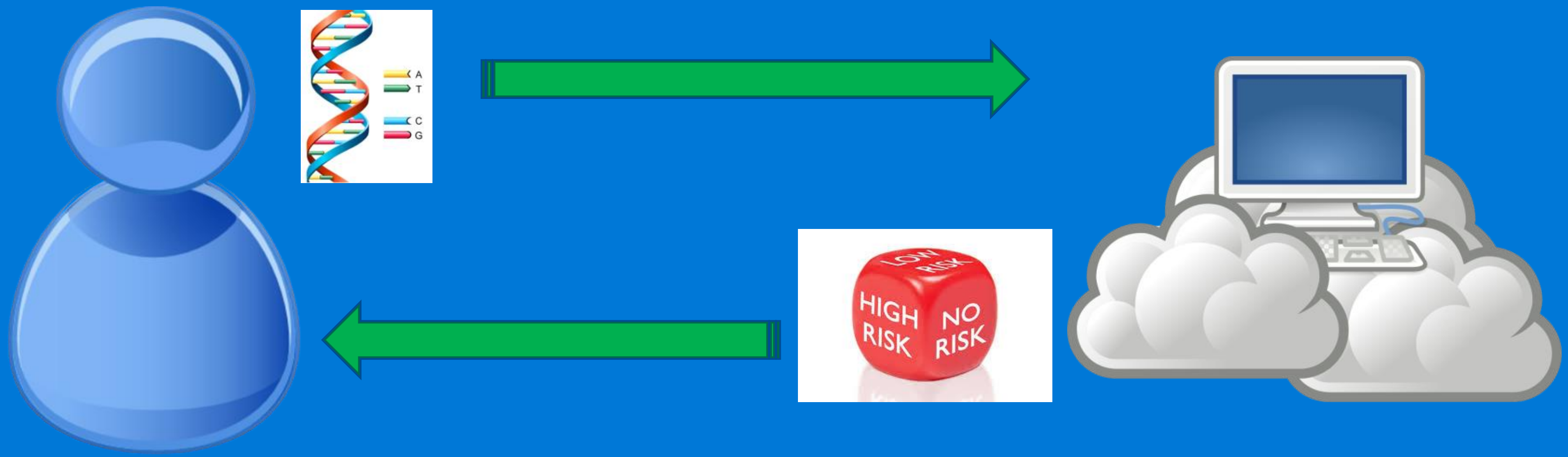

Data utility
Privacy

Can encryption help?

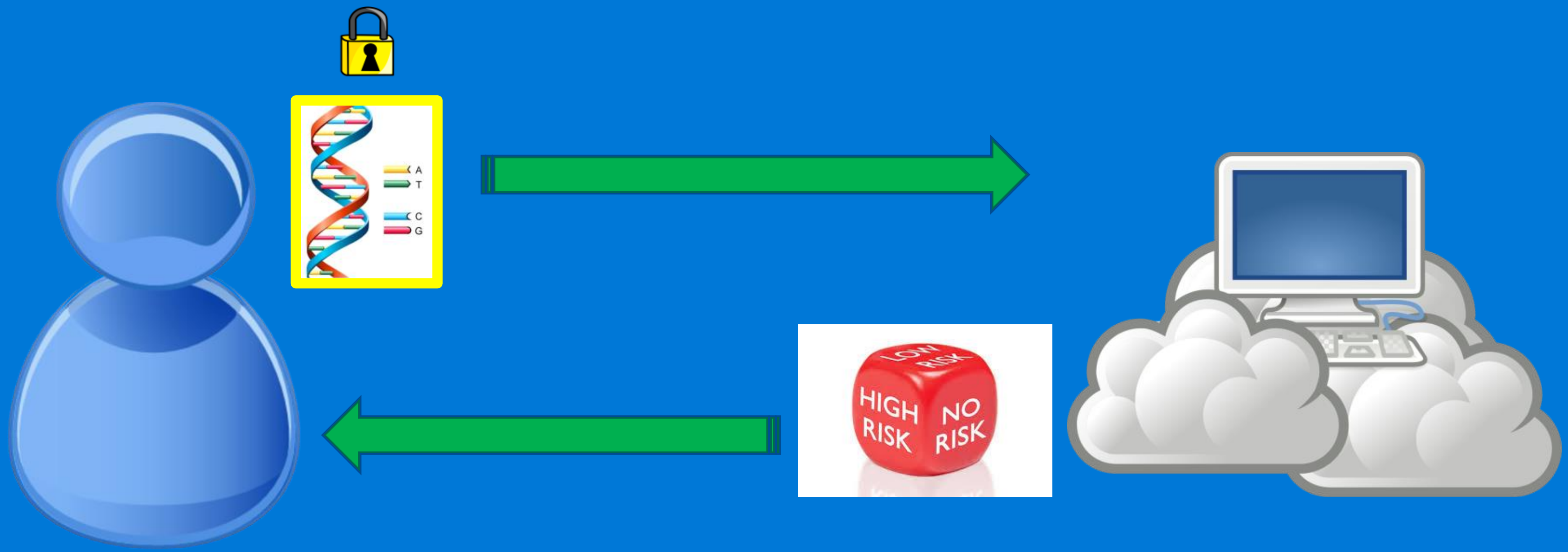
Can encryption help?

Possibly. But need *very* special type of encryption!

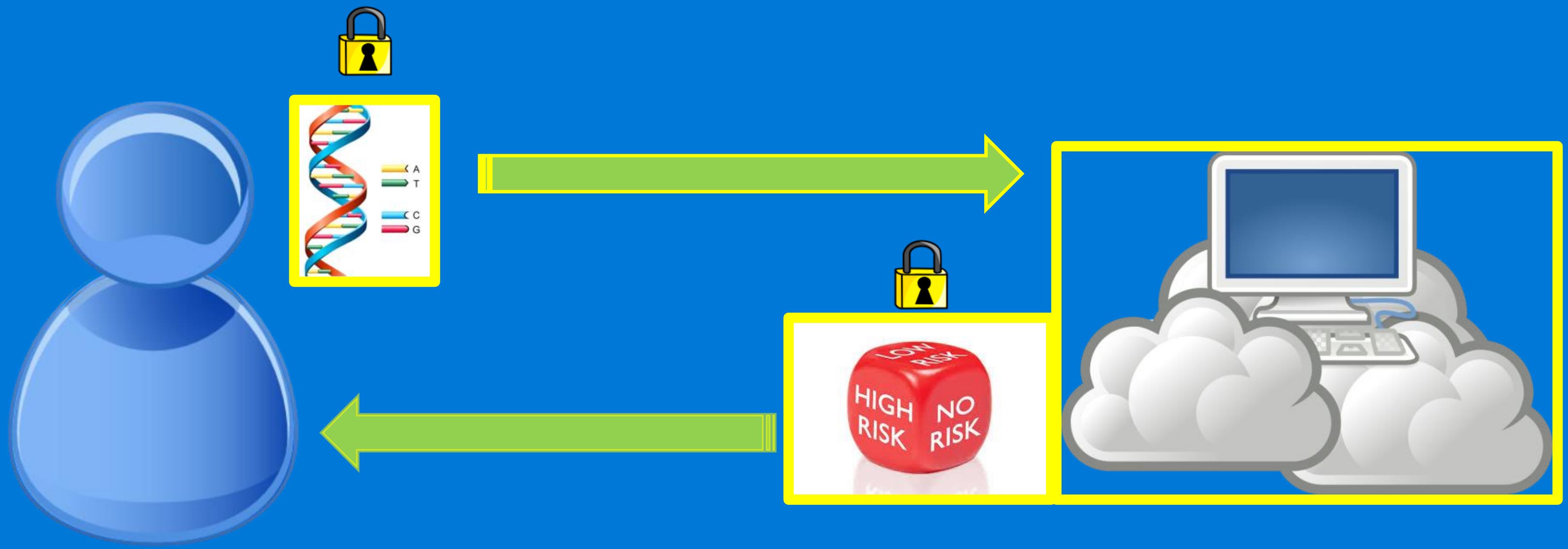
Inference over encrypted data



Inference over encrypted data



Inference over encrypted data



Can encryption help?

Possibly. But need *very* special type of encryption!

Can encryption help?

Possibly. But need *very* special type of encryption!

Yes. Homomorphic encryption.

Fully Homomorphic Encryption Using Ideal Lattices

Craig Gentry
Stanford University and IBM Watson
cgentry@cs.stanford.edu

ABSTRACT

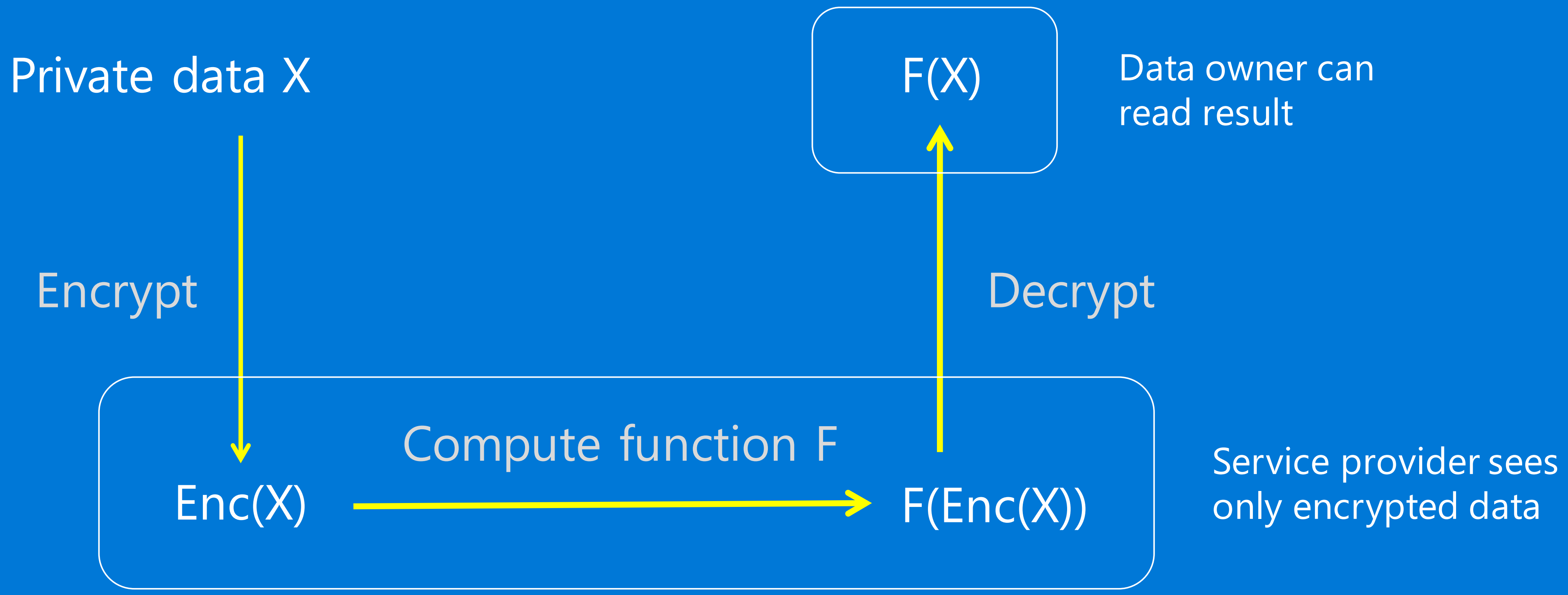
We propose a fully homomorphic encryption scheme – i.e., a scheme that allows one to evaluate circuits over encrypted data without being able to decrypt. Our solution comes in three steps. First, we provide a general result – that, to construct an encryption scheme that permits evaluation of *arbitrary circuits*, it suffices to construct an encryption scheme that can evaluate (slightly augmented versions of) its *own decryption circuit*; we call a scheme that can evaluate its (augmented) decryption circuit *bootstrappable.*

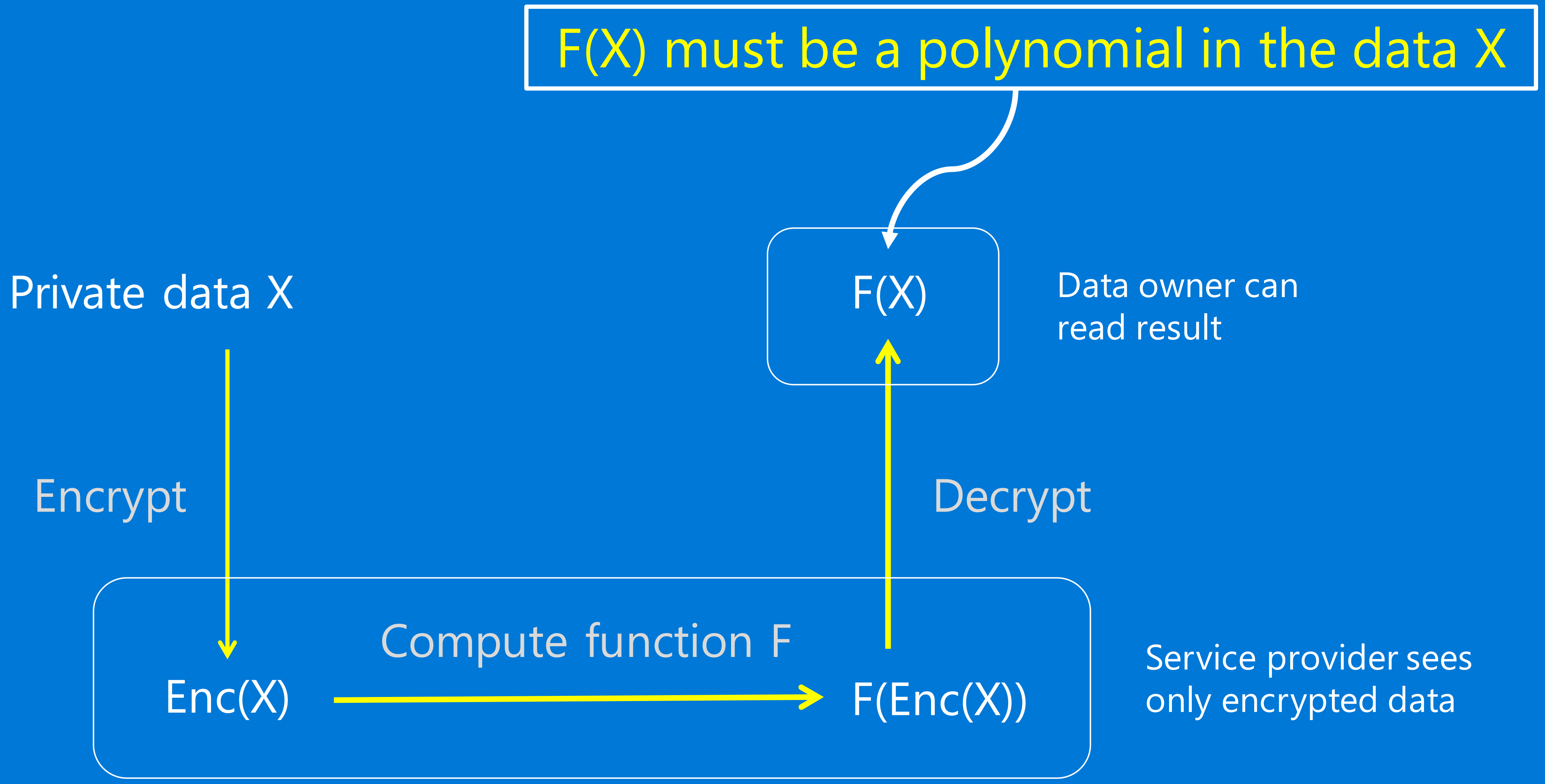
Next, we describe a public key encryption scheme using *ideal lattices* that is *almost* bootstrappable. Lattice-based cryptosystems typically have decryption algorithms with low

duced by Rivest, Adleman and Dertouzos [54] shortly a ter the invention of RSA by Rivest, Adleman and Sham [55]. Basic RSA is a multiplicatively homomorphic encry tion scheme – i.e., given RSA public key pk $= (N, e)$ an ciphertexts $\{\psi_i \leftarrow \pi_i^e \bmod N\}$, one can efficiently comput $\prod_i \psi_i = (\prod_i \pi_i)^e \bmod N$, a ciphertext that encrypts th product of the original plaintexts. Rivest et al. [54] aske a natural question: What can one do with an encryptio scheme that is *fully* homomorphic: a scheme $\mathcal{E}$ with an eff cient algorithm $\mathsf{Evaluate}_{\mathcal{E}}$ that, for any valid public key pl *any* circuit C (not just a circuit consisting of multiplicatio gates), and any ciphertexts $\psi_i \leftarrow \mathsf{Encrypt}_{\mathcal{E}}(\mathrm{pk}, \pi_i)$, output

$$\psi \leftarrow \mathsf{Evaluate}_{\mathcal{E}}(\mathrm{pk}, C, \psi_1, \ldots, \psi_t) ,$$

$$
\begin{tikzcd}
\text{Private data } X \arrow[r, "\text{Compute function } F"] \arrow[d, "\text{Encrypt}"'] & F(X) \\
\mathrm{Enc}(X) \arrow[r, "\text{Compute function } F"] & F(\mathrm{Enc}(X)) \arrow[u, "\text{Decrypt}"']
\end{tikzcd}
$$


Private data X
F(X)
Data owner can
read result
Encrypt
Decrypt
Compute function F
Enc(X)
F(Enc(X))
Service provider sees
only encrypted data


F(X) must be a polynomial in the data X
Private data X
F(X)
Data owner can read result
Encrypt
Decrypt
Compute function F
Enc(X)
F(Enc(X))
Service provider sees only encrypted data

SEAL

Simple Encrypted Arithmetic Library – SEAL

Easy-to-use homomorphic encryption library

Homomorphic encryption library by MSR Cryptography Research group
Focus on ease-of-use, good API design, good engineering
Written in C++11
Contains .NET wrappers for entire public API
Source code publicly available
Under active development

http://sealcrypto.codeplex.com

Choose encryption parameters

Create public and secret keys

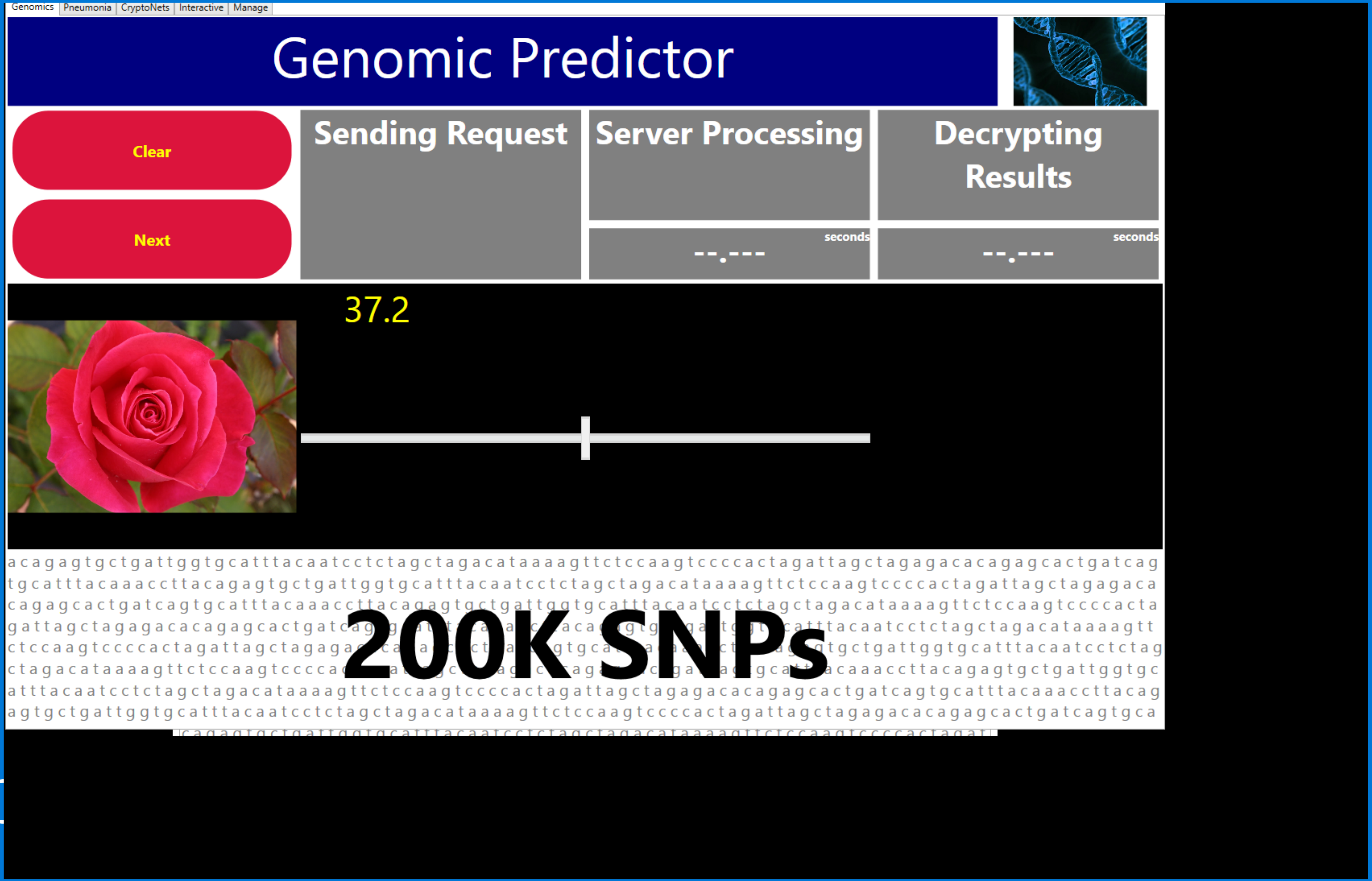
Encrypt some integers (encode + encrypt)

Do homomorphic evaluation

Decrypt the results (decrypt + decode)

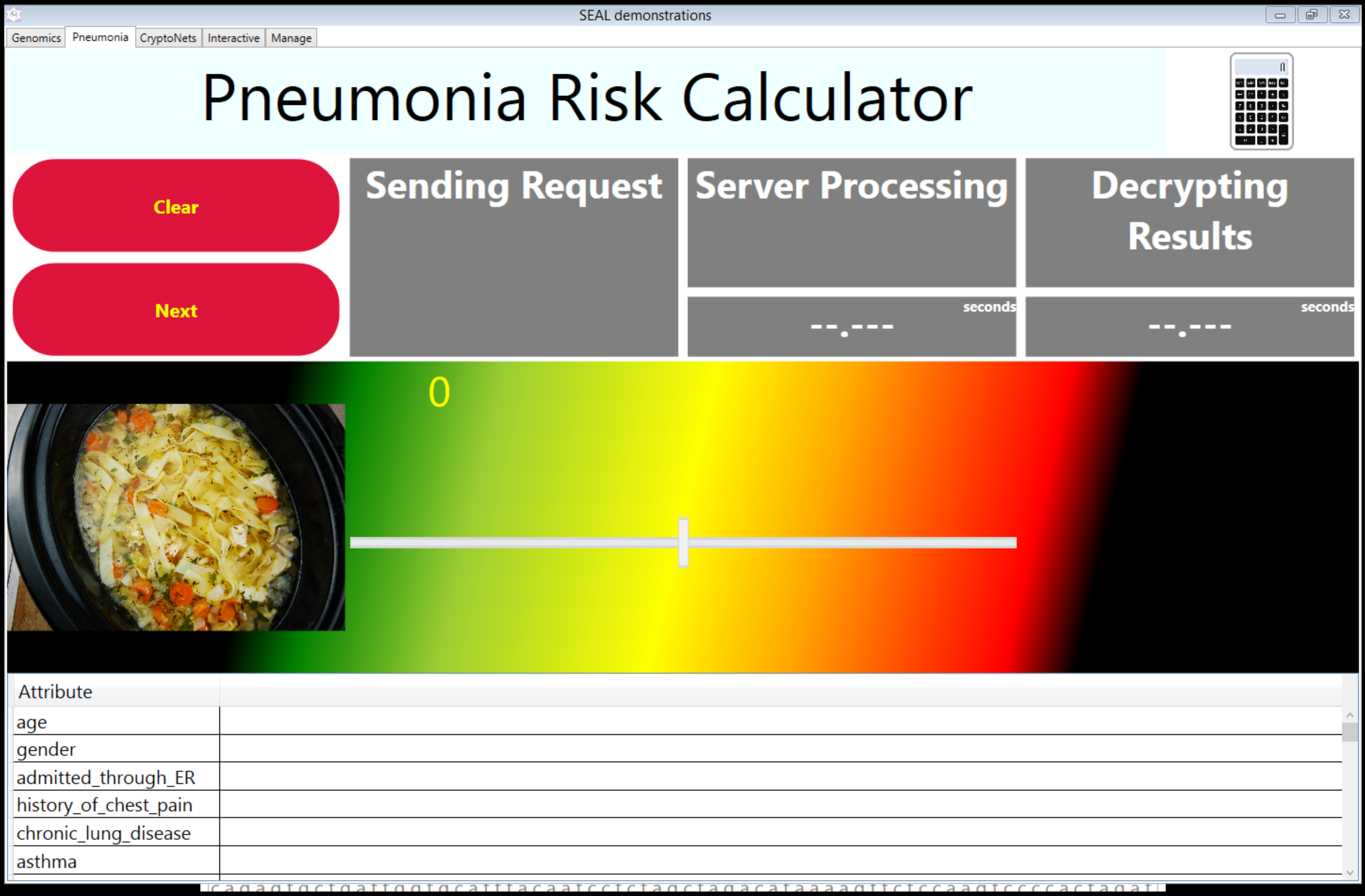
```
void simple_example()
{
    EncryptionParameters parms;
    parms.poly_modulus() = "1x^2048 + 1";
    parms.coeff_modulus() = ChooserEvaluator::default_parameter_options().at(2048);
    parms.plain_modulus() = 1 << 10;

    KeyGenerator keygen(parms);
    keygen.generate();
    auto public_key = keygen.public_key();
    auto secret_key = keygen.secret_key();

    BinaryEncoder encoder(parms.plain_modulus());
    Encryptor encryptor(parms, public_key);
    auto plain1 = encoder.encode(5);
    auto plain2 = encoder.encode(7);
    auto enc1 = encryptor.encrypt(plain1);
    auto enc2 = encryptor.encrypt(plain2);

    Evaluator evaluator(parms);
    auto enc_product = evaluator.multiply(enc1, enc2);
    auto enc_sum = evaluator.add(enc1, enc2);

    Decryptor decryptor(parms, secret_key);
    auto plain_product = decryptor.decrypt(enc_product);
    auto plain_sum = decryptor.decrypt(enc_sum);
    uint64_t product = encoder.decode_uint64(plain_product);
    uint64_t sum = encoder.decode_uint64(plain_sum);

    cout << product << " " << sum << endl;
}
```

Examples



Genomics
Pneumonia
CryptoNets
Interactive
Manage
Genomic Predictor
Clear
Next
Sending Request
Server Processing
seconds
--.---
Decrypting Results
seconds
--.---
37.2
acagagtgctgattggtgcatttacaatcctctagctagacataaaagttctccaagtccccactagattagctagagacacagagcactgatcag
tgcatttacaaaccttacagagtgctgattggtgcatttacaatcctctagctagacataaaagttctccaagtccccactagattagctagagaca
cagagcactgatcagtgcatttacaaaccttacagagtgctgattggtgcatttacaatcctctagctagacataaaagttctccaagtccccacta
atttacaatcctctagctagacataaaagttctccaagtccccactagattagctagagacacagagcactgatcagtgcatttacaaaccttacag
agtgctgattggtgcatttacaatcctctagctagacataaaagttctccaagtccccactagattagctagagacacagagcactgatcagtgca
200K SNPs

Demo



Demo: Pneumonia Risk Prediction

CryptoNets

Thank you!

Contact:
kim.laine@microsoft.com

Microsoft