

“Quantum” Classification of Malware

John Seymour (@seymour1)

2015-08-09



whoami

- Ph.D. student at the University of Maryland, Baltimore County (UMBC)
- Actively studying/researching infosec for about three years (mostly academic)
- Currently an intern for CyberPoint International

Outline

- The D-Wave Controversy
- How to play around on a D-Wave
- Building a malware classifier on a D-Wave 2
- What did we find?



▶ **QUANTUM COMPUTERS
AND THE END OF SECURITY**

<https://blog.kaspersky.com/quantum-computers-and-the-end-of-security/>

SON



THOU HAST
LET ME DOWN MIGHTILY

What you might have heard (and why it's wrong)

- FALSE: The D-Wave can solve NP-Complete problems in polynomial time.
- FALSE: The D-Wave is already “better” than classical computing for hard problems.

The Current State of Affairs

Quantum effects are happening...
...but that might not be interesting

We don't know whether the D-Wave uses
quantum effects for computation.

Regardless, it cannot run Shor's/Grovers/QKD.



<http://www.dwavesys.com/>



Kim Stalknecht for The New York Times

<http://www.nytimes.com/>

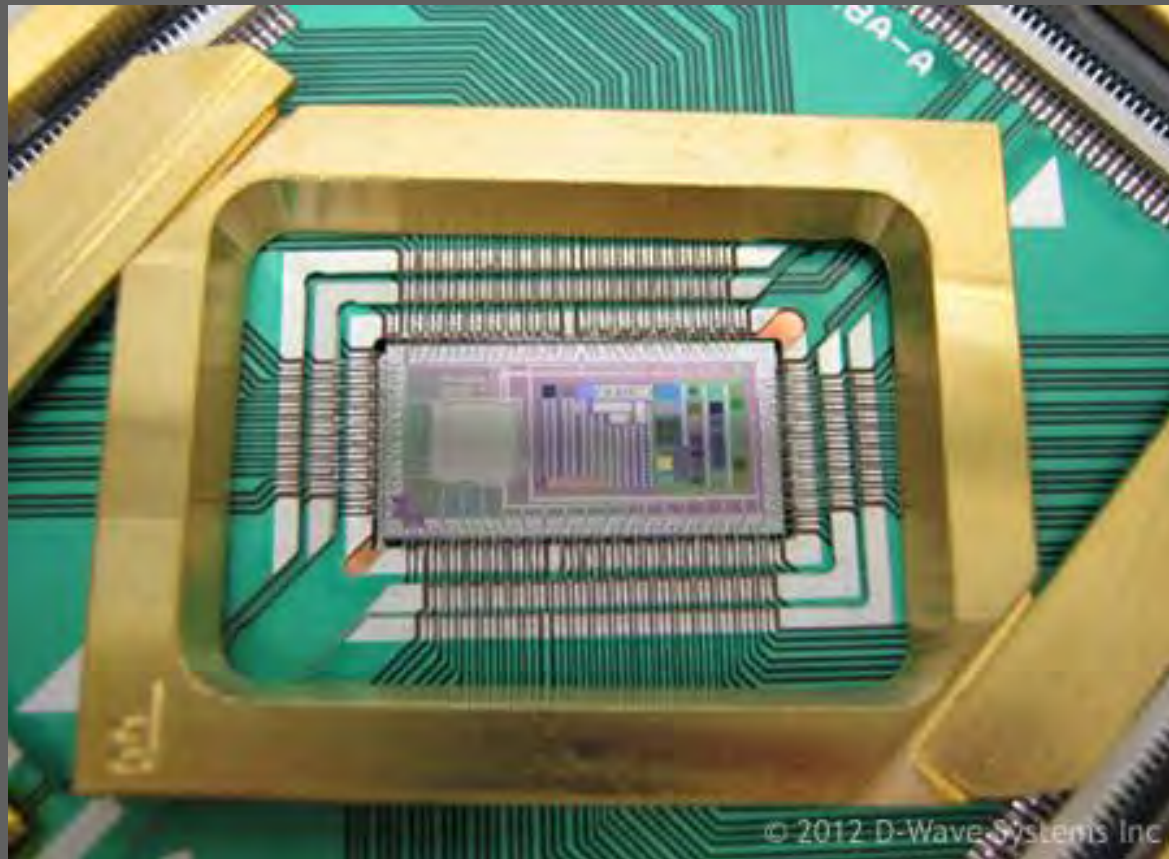


Kim Stalknecht for The New York Times

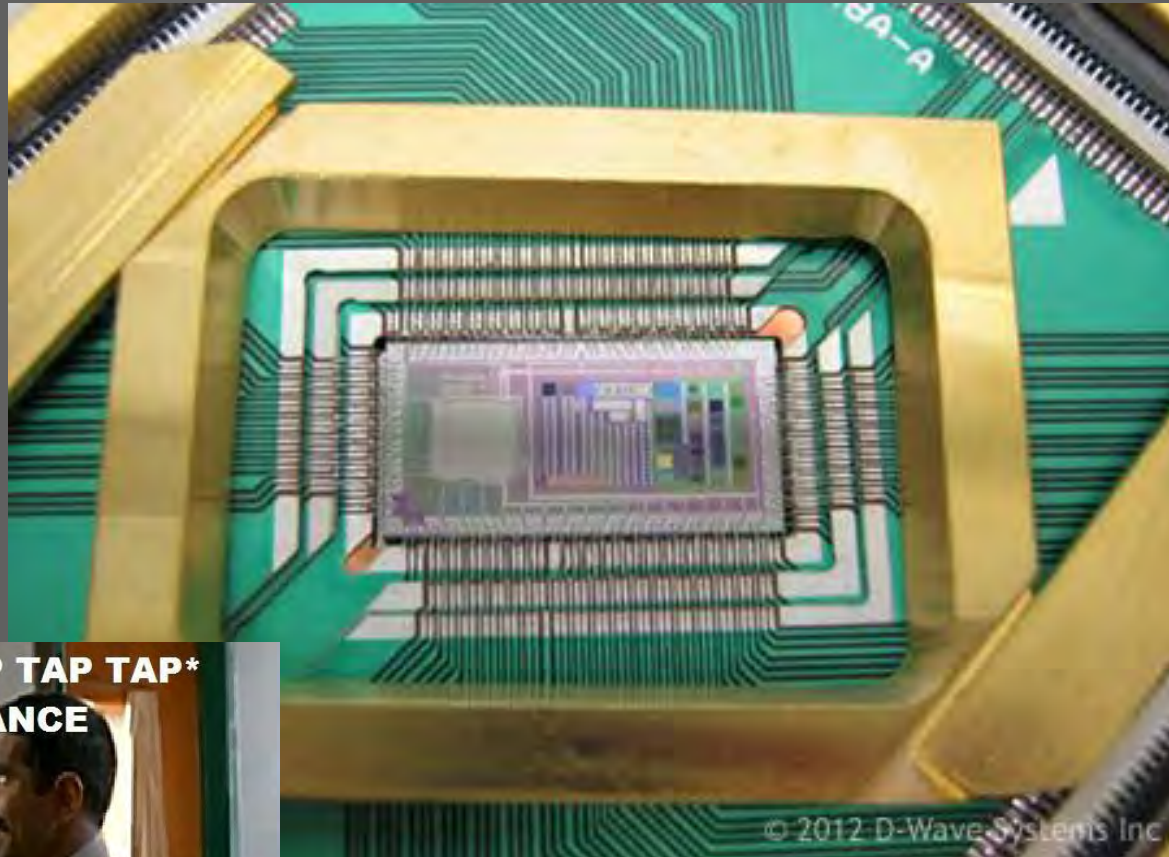


CAN YOU
ENHANCE THAT

<http://www.nytimes.com/>



<http://www.dwavesys.com/>

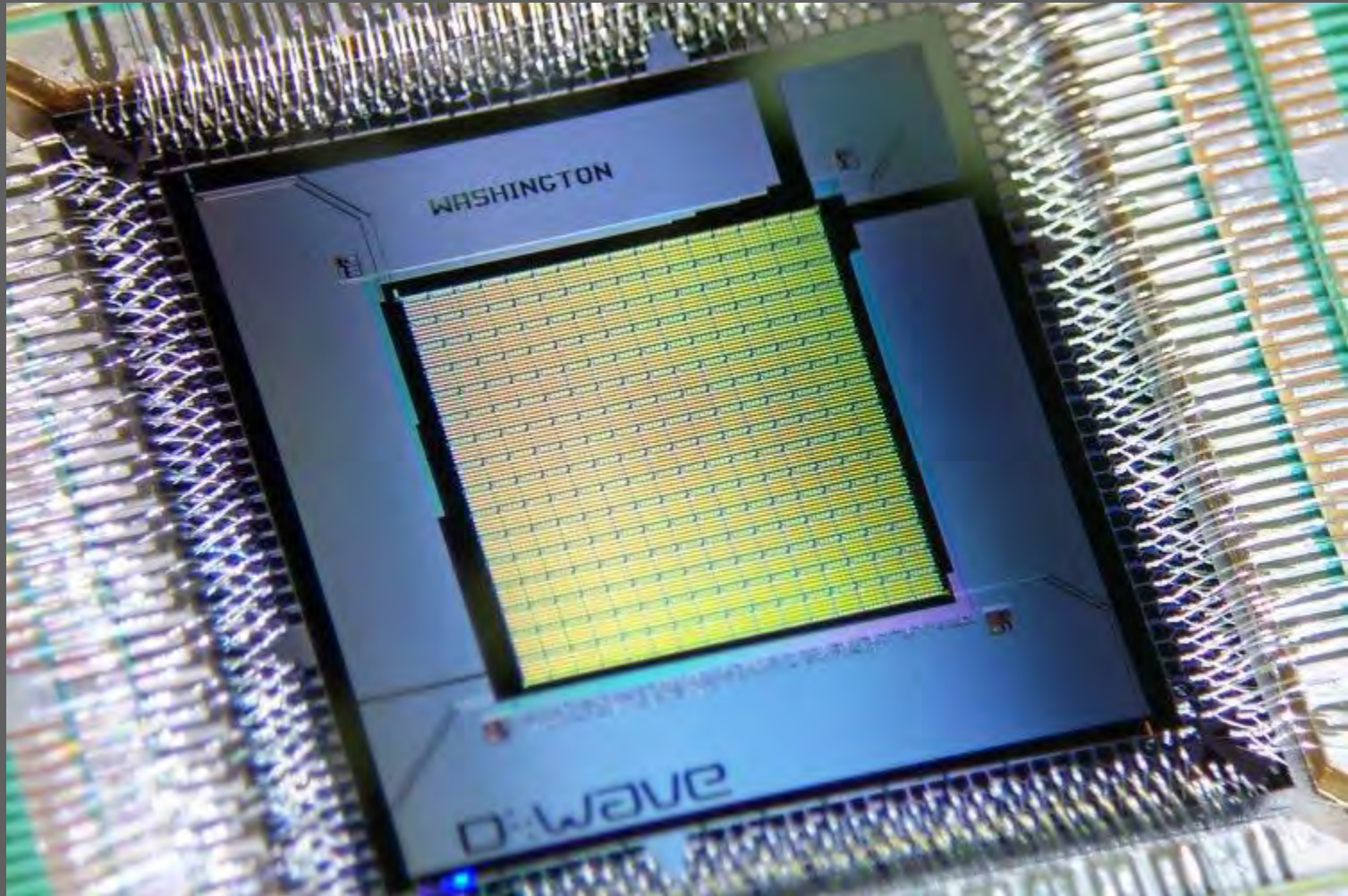


TAP TAP TAP TAP TAP
ENHANCE

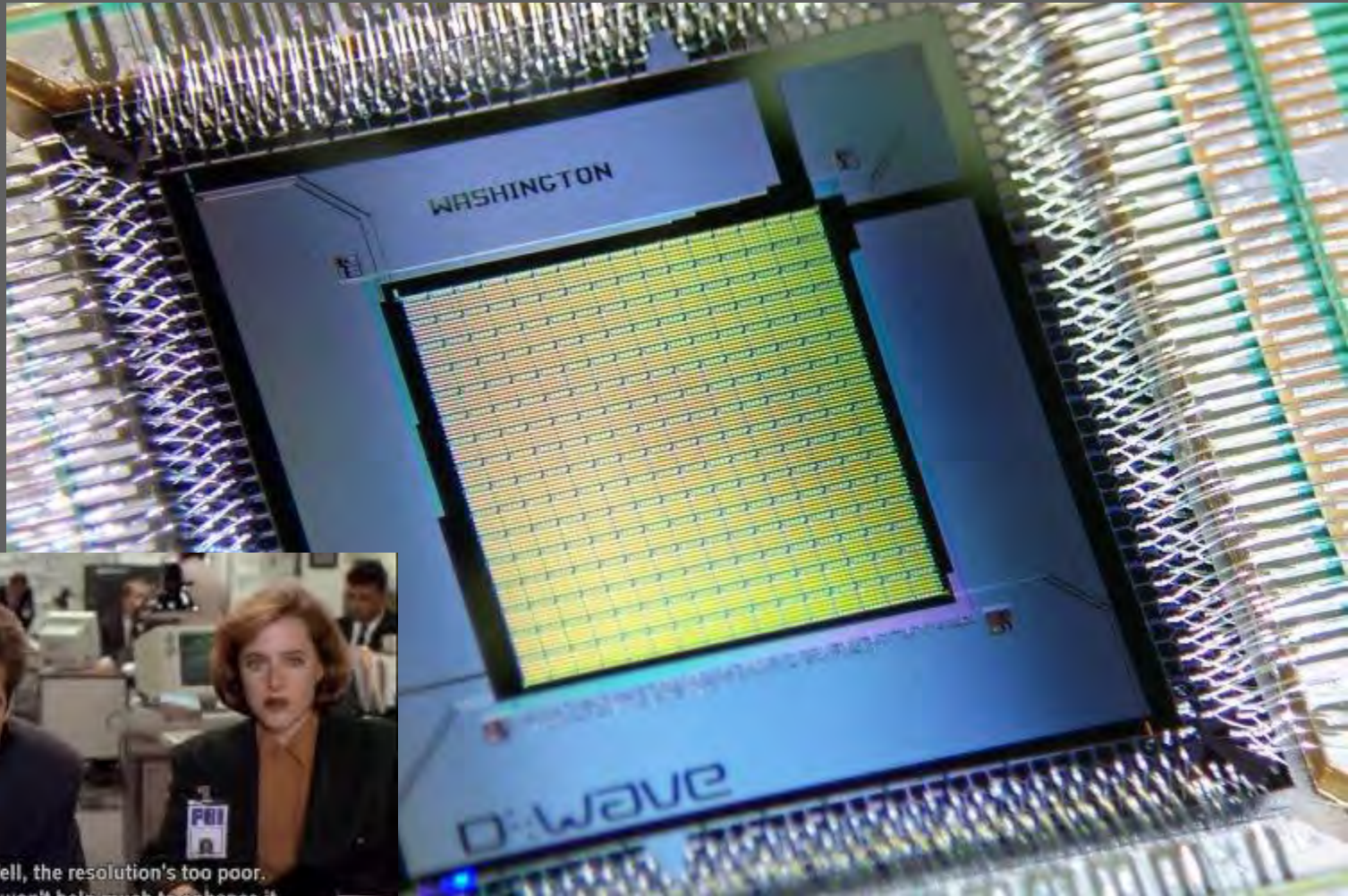


TAP TAP TAP TAP TAP
ENHANCE

<http://www.dwavesys.com/>



<http://www.dwavesys.com/>

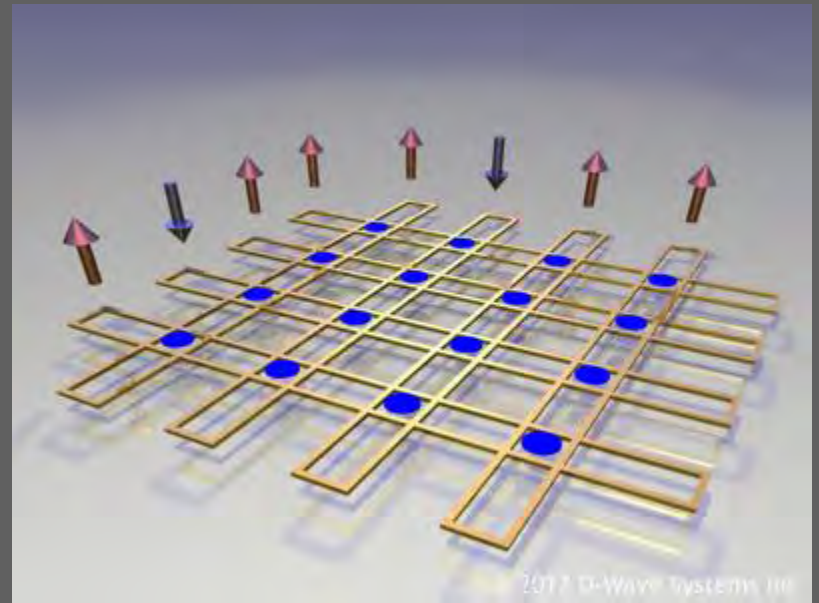


Well, the resolution's too poor.
It won't help much to enhance it.

<http://www.dwavesys.com/>

D-Wave chips consist of:

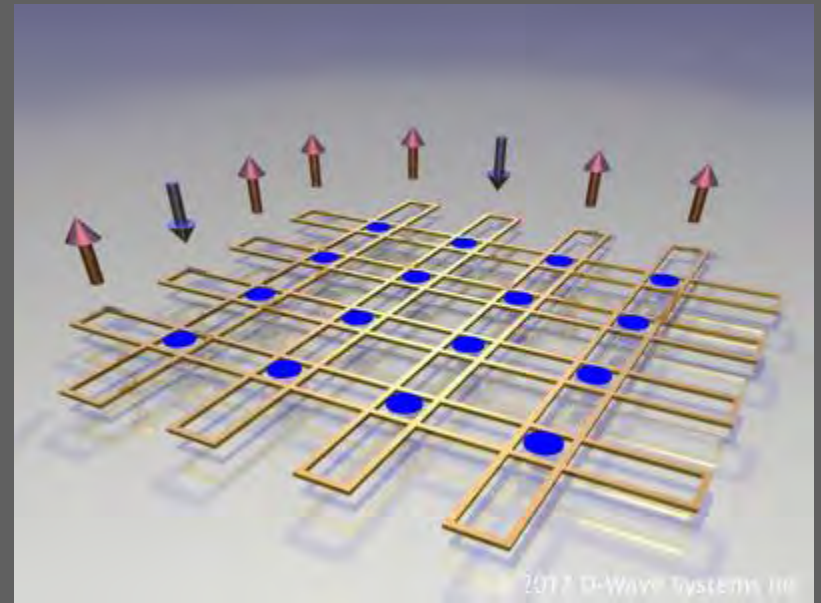
- niobium loops
- couplers



<http://www.dwavesys.com/>

D-Wave chips consist of:

- niobium loops
- couplers



<http://www.dwavesys.com/>

The D-Wave QUBO

The diagram shows the mathematical expression for a Quadratic Unconstrained Binary Optimization (QUBO) problem. The equation is $\sum_i a_i q_i + \sum_{i,j} b_{ij} q_i q_j$. The summation symbols \sum_i and $\sum_{i,j}$ are in white. The coefficients a_i and b_{ij} are in red. The variables q_i and q_j are in blue. A red arrow labeled "Input" points to the coefficients a_i and b_{ij} . A blue arrow labeled "Output" points to the variables q_i and q_j .

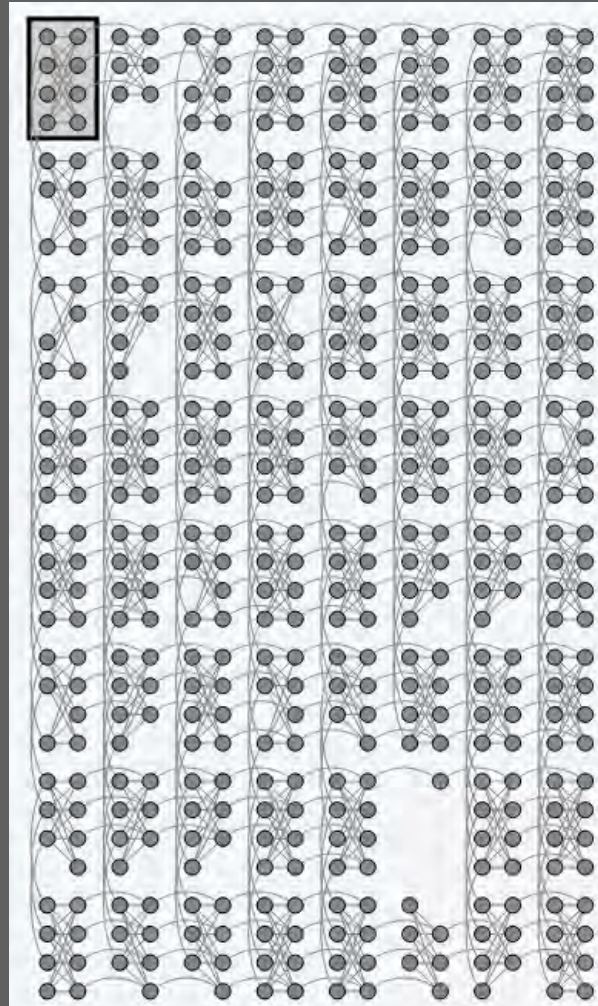
$$\sum_i a_i q_i + \sum_{i,j} b_{ij} q_i q_j$$

"Quadratic Unconstrained Binary Optimization"

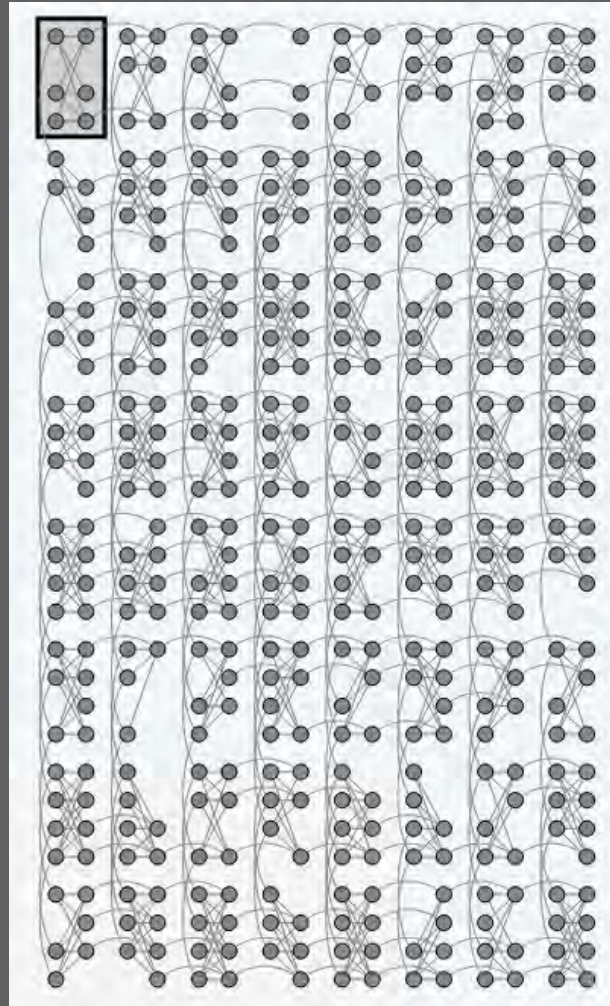
They've got a website. To do stuff on and stuff.

The screenshot shows the D-Wave website interface. At the top, there is a navigation bar with links for Home, Developers, Documentation, FAQ, and Downloads. The user's email, seymour1@umbc.edu, is displayed in the top right corner. The D-Wave logo and tagline, "The Quantum Computing Company," are visible on the left. The main content area features a solver selection dropdown menu set to "c4-sw_sample" with the note "Fully connected Chimera solver in 'sample' mode." To the right, the "Problem Type" is set to "Ising" with a help icon, and the "Constant Term" is 0. There are "Reset" and "Submit Problem" buttons. Below this, a tabbed interface shows the "Parameters" tab selected. The "Problem Parameters" section includes "Answer Mode" (set to "Histogram"), "Number of Reads" (set to "10"), and "Maximum Number of Answers" (empty). The "Timing Parameters" section includes "Annealing Time", "Post Programming Thermalization Time", and "Post Readout Thermalization Time", each with an input field. The "Automatic Scaling" checkbox is checked, with a note: "Automatically scale h and J values in Ising Hamiltonian to use their respective full ranges. If checked, h and J values will not be restricted to their respective ranges." At the bottom, the footer contains the text "Qubist 1.5.6-2-22699; ©2014 D-Wave Systems Inc. Timezone: PDT".

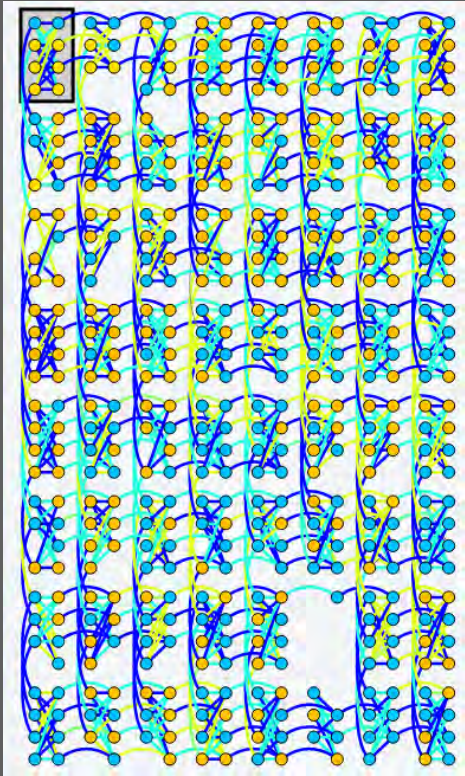
System 6



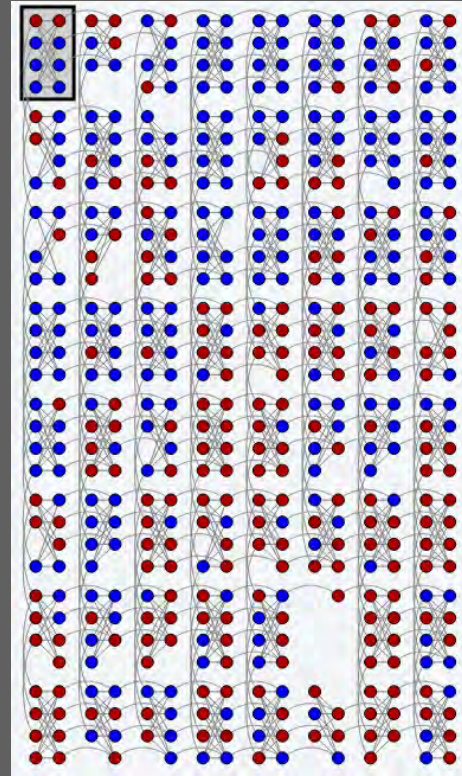
System 13



One D-Wave Run



Input



Output

Blackbox/ToQ

- Software, developed by D-Wave
- Turns arbitrary problems into QUBOs
- Heuristic (problem is NP-Complete)
- Conversation between classical machine and D-Wave

So what can it do?

D-Wave claims applications: classification, protein-folding models, finding close-to-optimal solutions to NPC problems (e.g. Traveling Salesman)

Crash Course in Machine Learning (At least what's relevant to this)

Boosting - using combinations of weak classifiers

- 3 binary classifiers with 70% accuracy

All Correct: $0.7 * 0.7 * 0.7 = 0.3429$	Two Correct: $3 * 0.7 * 0.7 * 0.3 = 0.4409$
Two Wrong: $3 * 0.3 * 0.3 * 0.7 = 0.189$	All Wrong: $0.3 * 0.3 * 0.3 = 0.027$

- Majority vote can increase your accuracy to 0.7838! (Hint: add up the top row)
- Most boosting algorithms also allow weights for these classifiers.

Loss

- Want to minimize:
 - Number of misclassifications
 - Complexity of the model

$$G(w) = \frac{1}{4} \sum_{s=1}^S (\text{sign}[\sum_{j=1}^D w_j F_j(x_s)] - y_s)^2 + \lambda \sum_{j=1}^D w_j$$

N-Grams

- Sliding window over text

DEADBEEF

N-Grams

- Sliding window over text

DEADBEEF

N-Grams

- Sliding window over text

DEADBEEF

N-Grams

- Sliding window over text

*DEAD***BEEF**

N-Grams

- Sliding window over text

DEADBEEF

- Easy to generate
- Used before in malware with good results
- Easy to turn into weak classifiers
- Complex enough to compare classifiers

Building the “Quantum” Malware Classifier

QBoost

- Outperforms Adaboost
- Robust to label noise
 - Will generally still learn even if training data is mislabeled
 - Good for learning malware: ground truth is hard!

Dataset used

- Plenty of malicious datasets to choose from
 - Vx Heaven, VirusShare, scraping the web
 - We used Vx Heaven (fairly standard but old)
- No standard for benign dataset
 - Problematic
 - Windows + Cygwin + Sourceforge
 - No adware was used in the making of this classifier

(Classical) Preprocessing

- Resample corpus to be balanced
- Side-effects: Less time to train, lose information

```
int getRandomNumber()  
{  
    return 4; // chosen by fair dice roll.  
             // guaranteed to be random.  
}
```

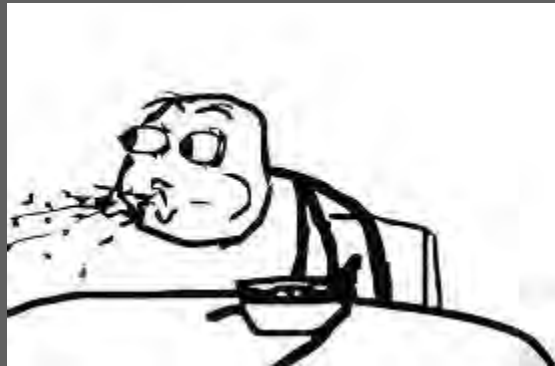
[HTTP://XKCD.COM/221/](http://xkcd.com/221/)

- Extract Features (3-gram bytes)

At first, D-Wave solutions were
no better than random chance

Next question: how long do we
need to let the D-Wave run?

- Previous work says 30 minute timeout



- Pilot Experiment to find how long it should take
 - Even on small problems, it takes 10 minutes to find decent solutions
 - Larger problems require even more time

- Limited to 32 features
 - We used 16 malware and 16 benign n-grams
 - Implemented QBoost with 10-fold cross-validation
 - Both on D-Wave and using a simulator
- Compared to several models from WEKA
 - Adaboost
 - J48 Decision Tree
 - Random Forest

Results

Classifier	Cross-fold Accuracy	Average Time to Build (Seconds)
D-Wave	0.80	536.32
D-Wave Simulator	0.802	451.62
Adaboost	0.768	0.02
J48	0.796	0.03
RandomForest	0.814	0.05

Table 1: Cross-fold accuracy and time to build classifiers.

Results

Classifier	Cross-fold Accuracy	Average Time to Build (Seconds)
D-Wave	0.80	536.32
D-Wave Simulator	0.802	451.62
Adaboost	0.768	0.02
J48	0.796	0.03
RandomForest	0.814	0.05

Table 1: Cross-fold accuracy and time to build classifiers.

Interesting Result 1: takes a LOT of time

Results

Classifier	Cross-fold Accuracy	Average Time to Build (Seconds)
D-Wave	0.80	536.32
D-Wave Simulator	0.802	451.62
Adaboost	0.768	0.02
J48	0.796	0.03
RandomForest	0.814	0.05

Table 1: Cross-fold accuracy and time to build classifiers.

Interesting Result 2: Simulator > actual chip

So, to recap:

- Blackbox/D-Wave CAN learn a malware classifier
- Accuracy comparable to classical algorithms using same features.
- Significant overhead and must restrict problem substantially

Future Work

- How much better is the next D-Wave chip?
- Possible to embed directly onto chip, rather than use Blackbox?
- Better for another task?
 - e.g. feature or instance selection
- Machine Learning standards for Malware Analysis

Thank you!
Questions?

References

Binary classification using a d-wave one system.

<http://www.dwavesys.com/en/dev-tutorial-qbc.html>. Accessed: 2013-06-13.

V. S. Denchev. Binary classification with adiabatic quantum optimization. PhD thesis, Purdue University, 2013.

References

C. C. McGeoch and C. Wang. Experimental evaluation of an adiabatic quantum system for combinatorial optimization. In Proceedings of the ACM International Conference on Computing Frontiers, CF '13, pages 23:1-23:11, New York, NY, USA, 2013. ACM.

References

- J. Seymour and C. Nicholas. Overgeneralization in feature set selection for classification of malware. Technical report, UMBC CSEE Technical Report, TR-CS-14-06, September, 2014, 2014.
- J. Seymour and C. Nicholas. Quantum classification of malware. Master's thesis, University of Maryland, Baltimore County, 2014.