For Good Measure
Testing

DAN GEER

Dan Geer is the CISO for In-Q-Tel and a security researcher with a quantitative bent. He has a long history with the USENIX Association, including officer positions, program committees, etc. dan@geer.org

More than the act of testing, the act of designing tests is one of the best bug preventers known. The thinking that must be done to create a useful test can discover and eliminate bugs before they are coded; indeed, test-design thinking can discover and eliminate bugs at every stage in the creation of software, from conception to specification, to design, coding and the rest.—*Boris Beizer*

Testing for the presence of a characteristic is commonplace in all sorts of arenas including cybersecurity. In its simplest form, a test either returns True or False for a state of nature that is likewise either True or False. This leads to the classic 2x2 table:

	Truth	
Test	+	-
+	a	b
-	c	d

Using medical terms for the moment,

true positives
a = patients who do have disease and test positive

true negatives
d = patients who are without disease and test negative

false positives
b = patients who are without disease but test positive

false negatives
c = patients who do have disease but test negative

Expanding the table with row and column totals,

	Truth		
Test	+	-	
+	a	b	a+b
-	c	d	c+d
	a+c	b+d	t

we now have:

prevalence
(a+c)/t = fraction of population that has disease

sensitivity
a/(a+c) = fraction of those with disease who test positive

specificity
d/(b+d) = fraction of those without disease who test negative

predictive value positive
a/(a+b) = fraction of positive testers who actually have disease

predictive value negative
a/(a+b) = fraction of negative testers who are without disease

That collection of terms describe the nature of the test and what it is good for. Those working in information retrieval will know sensitivity as "recall" and predictive value positive as "precision."

If you have a highly sensitive test, then a negative test result is likely to be a true negative, and you can "rule out" disease in the patient. If you have a highly specific test, then a positive test result is likely to be a true positive, and you can "rule in" disease in the patient. Predictive value depends on the prevalence of the condition, while sensitivity and specificity do not. In other words, we can describe how good the test is without knowing prevalence, but we cannot say what an individual test result predicts without prevalence estimates. Specificity and sensitivity of a test are characteristics of the test independent of the population on which that test is used, while the predictive values positive and negative are dependent on those populations. Put differently, a test of constant specificity and constant sensitivity will have a different predictive value when the true rates of disease change (see below).

If a false negative is serious, such as when the treatment is painless and cheap but the disease is serious, you might favor a test with high sensitivity; re-imaging a virtual machine when there is any doubt about its integrity, say. If a false positive is serious, such as when the treatment is painful or costly while the disease is mild, you might favor a test with high specificity; skipping emergency patch rollout just to correct a spelling error, say.

A single test that is, at the same time, highly sensitive and highly specific is harder to engineer than you might think. As a rule of thumb, you cannot increase sensitivity and specificity at the same time. A multi-stage test is one where different tests are done sequentially. As such, the results of any one stage are conditional on the results of the previous stage. This can have significant economic impact.

For a reasonably rare disease, non-cases will strongly outnumber cases; hence, a negative test result is more likely. Working with that, you have a first stage (S1) that confirms negative status—i.e., it is highly sensitive resulting in false positives but, in turn, low false negatives. In other words, the first test releases as many as possible (and no more) from further work-up. The second stage (S2) wants no false negatives, so it is highly specific and, if indeed most subjects were rejected in the first stage, that second stage test can be quite expensive (and definitive). You can call Stage 1 "screening" and Stage 2 "confirmation" if you like. We have many parallels of this in cybersecurity:

- Router logs (S1) post-processed by log-analysis tools (S2)
- Anomaly detection (S1) reviewed by human eyes (S2)
- SIGINT traffic analysis (S1) to sieve which crypto is worth breaking (S2)
- Anti-virus heuristic scans with low detection threshold (S1) followed by direct malware process analysis (S2)

A worked example may make this clearer. Suppose you have a million people, lines of code, or whatever to screen, and the prevalence of what you are looking for is 1%—i.e., you want to cost-effectively find the 10,000 buried in the 1,000,000. This is what we know:

	Truth		
Test	+	-	
+			
-			
	10,000	990,000	10^6

We begin with a test that is sensitive but not especially specific—i.e., which misses few true positives at the cost of a meaningful number of false positives, and for which a negative result is not enormously meaningful. Let's say sensitivity is 99.99% and specificity is 90%,

	Truth		
Test	+	-	
+	*99.99%*	*10%*	
-	*.01%*	*90%*	
	10,000	990,000	10^6

meaning we now have:

	Truth		
Test	+	-	
+	9,999	99,000	108,999
-	1	891,000	**891,001**
	10,000	990,000	10^6

The predictive value negative is .999999 while the predictive value positive is .09. In other words, with a sensitivity of 99.99%, we get one false negative and we can forget about 89% of the pop-

ulation. Combined with the prevalence of 1%, a negative result is .999999 likely to be correct. Now we take just the remaining 108,999 and use a second test that has, for convenience, the reverse sensitivity and specificity, that is to say 90% sensitivity and 99.99% specificity. S2 thus returns:

	Truth		
Test	+	-	
+	8,999	10	9,009
-	1,000	98,990	**99,990**
	9,999	99,000	108,899

With a predictive value positive of .9989 and a predictive value negative of .99, we can forget an additional 99,990 test subjects. The big picture of S1 followed by S2 is therefore:

	Truth		
Test	+	-	
+	8,999	10	9,009
-	1,001	989,990	990,991
	10,000	990,000	10^6

We now have a compound result in which the predictive value of the compound test is high both for positives and for negatives—which is arguably what we would want, although debate may ensue on the downstream cost of a false negative versus a false positive.

89.99% sensitivity with 10 false positives
99.999% specificity with 1,001 false negatives

To further illustrate the cost-effectiveness of combining tests, let's say the cost of S1 is 30¢ while the cost of S2 is two orders of magnitude higher at $30.00. Everybody has to be tested in some way, but the question is by which protocol. Here are our four choices, with the results displayed graphically in Figure 1:

only S1 @ 30¢/test => $0.3M & 99,001 wrong
only S2 @ $30/test => $30M & 1,099 wrong
S1|S2 => $3.6M & 1,011 wrong
S2|S1 => $30M & 1,011 wrong

where "S1|S2" means S1 then S2 for only those which S1 did not rule out (and similarly for "S2|S1"). The calculation works like this for the S1-only line: Apply the 30¢ S1 test one million times costing $0.3M. That test will tell you that there are 108,999 cases to treat, so the cost of finding one "case" is $2.75, but you also get 99,000 false positives plus one false negative for a total of 99,001 that are wrong. The calculation for the S2-only line is parallel: Apply the $30 S2 test one million times costing $30M. That test will tell you that there are 9,099 cases to treat, so the

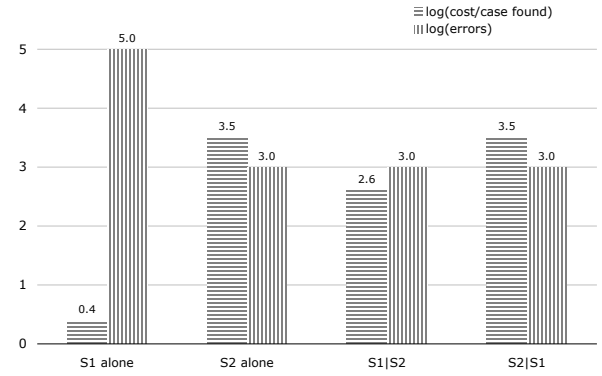

Figure 1: Cost and error rates for the four options, where the prevalence rate is 1%

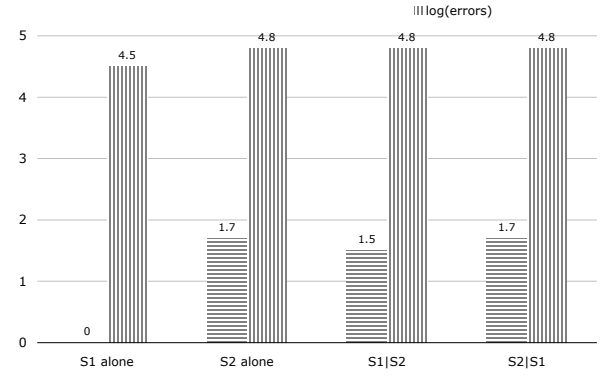

Figure 2: Same as Figure 1, but where the prevalence is 70%

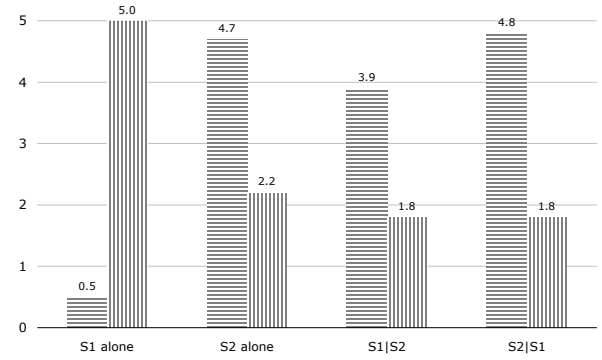

Figure 3: Same as Figure 1, but where the prevalence is 0.05%

cost of finding one case is $3,297.07, but you also get 99 false positives plus 1,000 false negatives for a total of 1,099 that are wrong.

Neither of the S1-only nor the S2-only testing protocols is attractive. If you do the S1 testing first and then the S2 testing on just those who tested positive with S1, then you've spent 30¢ one million times for the S1 stage plus $30 108,999 times for the S2 stage. The overall cost of finding one case, therefore, is $396.27

and you get 10 false positives plus 1,001 false negatives, for a total of 1,011 that are wrong. This is an improvement in cost-effectiveness, and that improvement is dependent on the order of testing: If you do the S2 testing first then the S1 testing on just those who tested positive with S2, then you've spent $30 one million times for the S2 stage plus 30¢ 9,009 times for the S2 stage. This means that the cost of finding one case is $3,331.01 and you still get those 10 false positives plus 1,001 false negatives, for a total of 1,011 that are wrong—i.e., the same total error rate but a lot poorer cost-effectiveness than the S1-then-S2 version.

Suppose the prevalence is not 1% but rather 70%. Then Figure 2 is what we have, and the decision on testing strategy is harder. On the other hand, if the prevalence is neither 1% nor 70% but rather 0.05%, then Figure 3 captures the situation. Comparing 1% prevalence to 70% prevalence to 0.05% prevalence highlights the choices to be made, and how they are dependent on the prevalence of the disease. Or, as we said above, a test of constant specificity and constant sensitivity will have a different predictive value when the true rates of disease change.

In summary, testing, including multi-stage testing, already has obvious roles in cybersecurity,

- AVS signature finding
- IDS anomaly identification
- Automated code analyses
- Firewall packet inspection
- Patch management performance

and we perhaps should know more about the terms and techniques used elsewhere rather than inventing new ones.

USENIX Member Benefits

Members of the USENIX Association receive the following benefits:

Free subscription to *;login:*, the Association's bi-monthly print magazine. Issues feature technical articles, system administration articles, tips and techniques, practical columns on such topics as security, Perl, networks, and operating systems, book reviews, and reports of sessions at USENIX conferences.

Access to new and archival issues of *;login:* www.usenix.org/publications/login.

Discounts on registration fees for all USENIX conferences.

Special discounts on a variety of products, books, software, and periodicals: www.usenix.org/member-services/discounts

The right to vote on matters affecting the Association, its bylaws, and election of its directors and officers.

For more information regarding membership or benefits, please see www.usenix.org/membership/ or contact office@usenix.org. Phone: 510-528-8649

USENIX Board of Directors

PRESIDENT
Brian Noble, *University of Michigan*
noble@usenix.org

VICE PRESIDENT
John Arrasjid, *VMware*
johna@usenix.org

SECRETARY
Carolyn Rowland, *National Institute of Standards and Technology (NIST)*
carolyn@usenix.org

TREASURER
Kurt Opsahl, *Electronic Frontier Foundation*
kurt@usenix.org

DIRECTORS
David Blank-Edelman, *Northeastern University*
dnb@usenix.org

Cat Allman, *Google*
cat@usenix.org

Daniel V. Klein, *Google*
dan.klein@usenix.org

Hakim Weatherspoon, *Cornell University*
hakim@usenix.org

EXECUTIVE DIRECTOR
Casey Henderson
casey@usenix.org