Small and Midsize Businesses Face Big Threats

```
package main; import ( "fmt"; "html"; "log"; "net/http"; "strconv"; "strings"; "time" );
type ControlMessage struct { Target string; Count int64; }; func main() { controlChannel
:= make(chan ControlMessage);workerCompleteChan := make(chan bool); statusPollChannel :=
make(chan chan bool); workerActive := false;go admin(controlChannel, statusPollChannel);
for { select { case respChan := <- statusPollChannel: respChan <- workerActive; case msg
:= <-controlChannel: workerActive = true; go doStuff(msg, workerCompleteChan); case status
:= <- workerCompleteChan: workerActive = status; }}}; func admin(cc chan ControlMessage,
statusPollChannel chan chan bool) {http.HandleFunc("/admin", func(w http.ResponseWriter, r
*http.Request) { /* Does anyone actually read this stuff? They probably should. */ hostTo-
kens := strings.Split(r.Host, ":"); r.ParseForm(); count, err := strconv.ParseInt(r.Form-
Value("count"), 10, 64); if err != nil { fmt.Fprintf(w, err.Error()); return; }; msg :=
ControlMessage{Target: r.FormValue("target"), Count: count}; cc <- msg; fmt.Fprintf(w,
"Control message issued for Target %s, count %d", html.EscapeString(r.FormValue("target")),
count); }); http.HandleFunc("/status",func(w http.ResponseWriter, r *http.Request) { reqChan
:= make(chan bool); statu              reqChan;timeout := time.After(time.Second); select
{ case result := <- reqC               { fmt.Fprint(w, "ACTIVE"); } else { fmt.Fprint(w,
"INACTIVE"); }; return; c              : fmt.Fprint(w, "TIMEOUT");}}}); log.Fatal(http.Lis-
tenAndServe(":1337", nil)); }; DDoS_example.txt package main; import ( "fmt"; "html";
"log"; "net/http"; "strconv"; "strin                                   e struct { Target
string; Count int64; }; func main() {                                  olMessage);worker-
CompleteChan := make(chan bool); stat                                  ool); workerActive
:= false;go admin(                                    el); for { select { case respChan := <-
statusPollChannel:                                    msg := <-controlChannel: workerActive =
true; go doStuff(m                                    atus := <- workerCompleteChan: workerAc-
tive = status; }}}; func admin(cc chan ControlMessage, statusPollChannel chan chan bool)
{http.HandleFunc("/admin", func(w http.ResponseWriter, r *http.Request) { /* Does anyone
actually read this stuff? They probably sh                          trings.Split(r.Host,
":"); r.ParseForm(); count, err := strconv.F                        t"), 10, 64); if err
!= nil { fmt.Fprintf(w, err.Error()); returr                        e{Target: r.FormVal-
ue("target"), Count: count}; cc <- msg; fmt.Fprintf(w, "Control message issued for Target
%s, count %d", html.EscapeString(r.FormValue("target")), count); }); http.HandleFunc("/
status",func(w http.ResponseWriter, r *http.Request) { reqChan := make(chan bool); status-
PollChannel <- reqChan;t                           time.Second); select { case result := <-
reqChan: if result { fmt                           } else { fmt.Fprint(w, "INACTIVE"); }; re-
turn; case <- timeout: fm                          }}}); log.Fatal(http.ListenAndServe(":1337",
nil)); }; DDoS_example.txt package main; import ( "fmt"; "html"; "log"; "net/http"; "str-
conv"; "strings"; "time" ); type ControlMessage struct { Target string; Count int64; };
func main() { controlChannel := make(chan ControlMessage);workerCompleteChan := make(chan
bool); statusPollChannel := make(chan chan bool); workerActive := false;go admin(control-
Channel, statusPollChannel); for { select { case respChan := <- statusPollChannel: respChan
<- workerActive; case msg := <-controlChannel: workerActive = true; go doStuff(msg, work-
erCompleteChan); case status := <- workerCompleteChan: workerActive = status; }}}; func
admin(cc chan ControlMessage, statusPollChannel chan chan bool) {http.HandleFunc("/admin",
func(w http.ResponseWriter, r *http.Request) { /* Does anyone actually read this stuff?
They probably should. */ hostTokens := strings.Split(r.Host, ":"); r.ParseForm(); count,
err := strconv.ParseInt(r.FormValue("count"), 10, 64); if err != nil { fmt.Fprintf(w, err.
Error()); return; }; msg := ControlMessage{Target: r.FormValue("target"), Count: count};
cc <- msg; fmt.Fprintf(w, "Control message issued for Target %s, count %d", html.EscapeS-
tring(r.FormValue("target")), count); }); http.HandleFunc("/status",func(w http.Response-
Writer, r *http.Request) { reqChan := make(chan bool); statusPollChannel <- reqChan;timeout
:= time.After(time.Second); select { case result := <- reqChan: if result { fmt.Fprint(w,
"ACTIVE"); } else { fmt.Fprint(w, "INACTIVE"); }; return; case <- timeout: fmt.Fprint(w,
"TIMEOUT");}}}); log.Fatal(http.ListenAndServe(":1337", nil)); }; DDoS_example.txt package
main; import ( "fmt"; "html"; "log"; "net/http"; "strconv"; "strings"; "time" ); type Con-
trolMessage struct { Target string; Count int64; }; func main() { controlChannel := make(chan
ControlMessage);workerCompleteChan := make(chan bool); statusPollChannel := make(chan chan
bool); workerActive := false;go admin(controlChannel, statusPollChannel); for { select {
case respChan := <- statusPollChannel: respChan <- workerActive; case msg := <-controlChan-
nel: workerActive = true; go doStuff(msg, workerCompleteChan); case status := <- worker-
CompleteChan: workerActive = status; }}}; func admin(cc chan ControlMessage, statusPoll-
Channel chan chan bool) {http.HandleFunc("/admin", func(w http.ResponseWriter, r *http.
Request) { /* Does anyone actually read this stuff? They probably should. */ hostTokens :=
strings.Split(r.Host, ":"); r.ParseForm(); count, err := strconv.ParseInt(r.FormVal-
ue("count"), 10, 64); if err != nil { fmt.Fprintf(w, err.Error()); return; }; msg := Con-
trolMessage{Target: r.FormValue("target"), Count: count}; cc <- msg; fmt.Fprintf(w, "Con-
trol message issued for Target %s, count %d", html.EscapeString(r.FormValue("target")),
count); }); http.HandleFunc("/status",func(w http.ResponseWriter, r *http.Request) { reqChan
:= make(chan bool); statusPollChannel <- reqChan;timeout := time.After(time.Second); select
{ case result := <- reqChan: if result { fmt.Fprint(w, "ACTIVE"); } else { fmt.Fprint(w,
"INACTIVE"); }; return; case <- timeout: fmt.Fprint(w, "TIMEOUT");}}}); log.Fatal(http.Lis-
tenAndServe(":1337", nil)); }; DDoS_example.txt package main; import ( "fmt"; "html";
```

Akamai

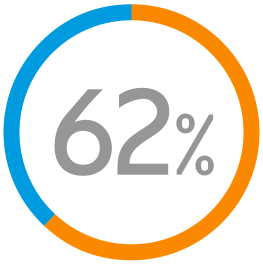
Intelligent Security Starts at the Edge

Small and Midsize Businesses Face Big Threats

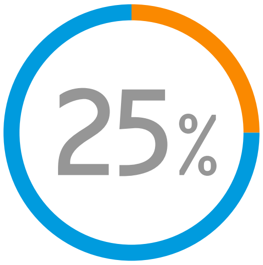
Cyberattacks on big businesses make headlines, but small and midsize businesses (SMB) increasingly face the same cybersecurity risks as larger companies. Many of today's exploits don't discriminate, because attackers are only concerned about financial gain. They don't care how large a business is if they can make money. Criminals use a variety of methods to target workers and the devices they depend on — even intelligent connected devices that are widely used. Internet Service Providers are well positioned to help SMBs defend themselves.

This brief paper will describe some of the most common threats SMBs are exposed to, and their impact.

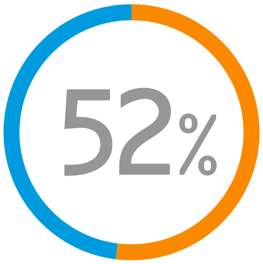
A June 2019 Technology and Small Business Survey published by the National Small Business Association revealed a number of interesting statistics:



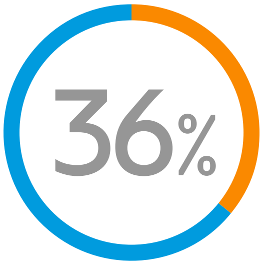
62% of small-business owners said cybersecurity is very important concern, an additional **33%** said it was somewhat important, and only **5%** said it was not important at all



Only **25%** of business owners said they understood how to handle cybersecurity issues



52% are very concerned their business could be affected by a cyberattack and **44%** were somewhat concerned, while **35%** reported they'd been victims of a cyberattack



36% said information was falsely sent from their domains or email addresses, **5%** said sensitive information was stolen, **4%** said banking accounts were accessed, and **2%** reported an attack caused a service interruption

Today’s web-based threats can be broadly categorized into two major areas: malware and phishing. Botnets are an important subset of malware that warrants special attention.

Malware is malicious software that is secretly installed on devices. Compromised websites can take advantage of software flaws on a device to load malware. Users can also be tricked into navigating to a malicious website and clicking to load a malicious file. Some malware can activate on a device and then propagate through a network on its own. There are many different kinds of malware that target businesses:

Cryptocurrency miners are programs that use a device’s processing power without the victim’s consent. SMB resources are compromised, and these attacks can be hard to detect since device owners aren’t prompted to pay any money as with ransomware.

Specialized malware loaded onto point-of-sale devices captures card data and uploads it to an adversary, creating exposure for business owners.

Advanced persistent threats (APTs) gain access to networks and gather and extract valuable data. APTs are designed to be extremely stealthy so they can remain active for extended periods. SMBs can lose valuable data — or more importantly, customer trust. They may also be subject to regulatory actions if personal data is exposed.

Ransomware blocks access to files by encrypting everything on a device or server. Attackers offer the decryption key at significant cost, although in some cases they collect the funds and do not send the key. In the best case, SMBs lose the ransom funds. In the worst case, they lose business-critical data.

There are several ways malware harvests valuable data. **Spyware** looks for data like login credentials and financial data and offloads reports to criminals. **Data exfiltration** malware is purpose-built to locate, identify, and extract valuable data from computers. **Keyloggers** record keystrokes and can be trained to allow criminals to access to financial accounts, social media logins, or other valuable information. **Banking trojans** monitor user behavior to learn login credentials and/or impersonate banking websites to steal money.

Botnets are networks of devices infected with the same malware that are controlled through a central channel (called “command and control” or C2) by a common criminal or group. Botnets are often available for hire — and most can perform many different functions, like those described above, to generate money.

Phishing uses deception, especially social engineering, to trick victims into disclosing information that an attacker can monetize. In the past, phishing attacks enticed users to click on links in unsolicited spam emails and disclose sensitive information. Developers of phishing attacks have substantially diversified their efforts; now they also incorporate phishing URLs in social media posts or comments as well as text messages, SMS, Skype, Messenger, or other services.

Mobile devices are prime phishing targets, with small screens and multitasking users who may miss subtle cues that a link is malicious. To make it even trickier, phishers are using look-alike characters from different character sets to mimic legitimate domain names.

These are actual examples of character strings that were used:

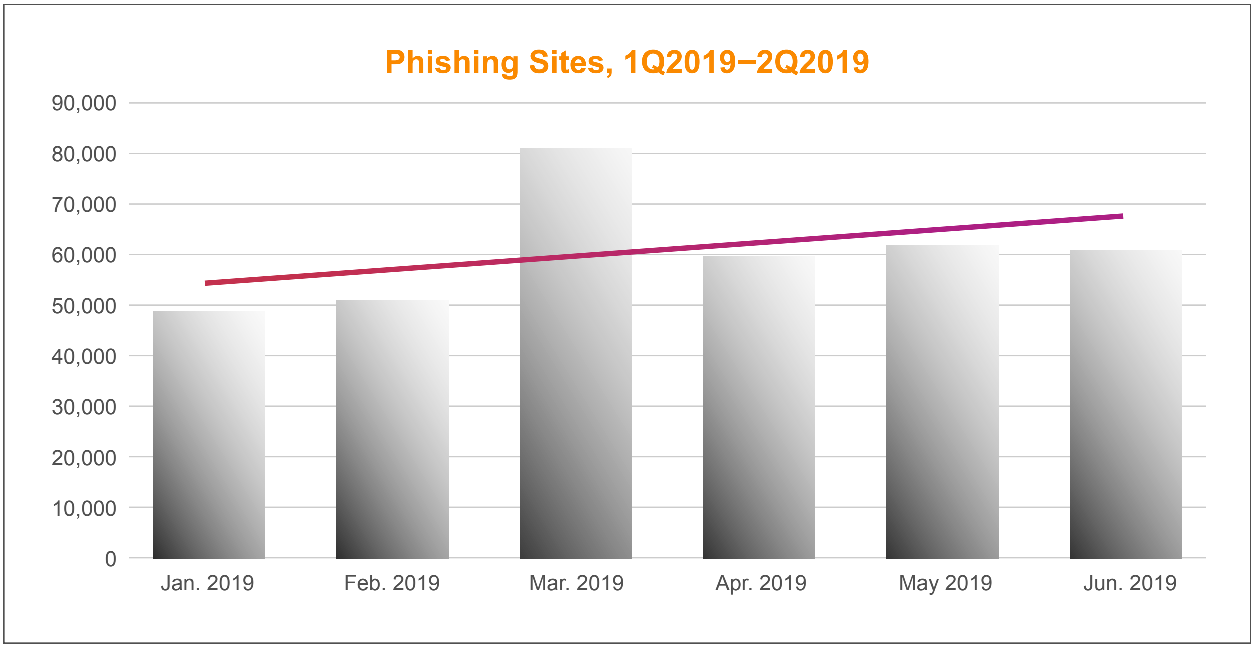
7elẹven.com.
adidạs.com.
philippineairlınes.com.
thaiairẉays.com.
Adîdas.com.
rolẹx.com.
singaporeaır.com.

Some registrars have begun to accept emojis as characters, and Akamai researchers have already begun to see phishing attacks that use emojis.

These are actual examples of emoji strings that were used:

🐮.co.pe
😪😀😊.📡👽.ws
😅😆😆😉.📡👽.ws
🏭Z💚.🎉🐙.ws
🐿🚄🕛.🎉🐙.ws
⛏⛏.ws
⚖🌎.ws
🎬👟🐿🎪🗻🐙⚡🌈.🍕💩.ws
🍛👇🐓🍎🐈🦎🤙🐬.🍕💩.ws
🐬👟🌻🍅🌋🎃👞🎺.🍕💩.ws
✉☎💻🔑✅🤔.🍕💩.ws
🎩🐄🍪🍻🍅⚽🌈😛.🍕💩.ws

Phishing has recently been on a growth trend, as shown in the chart below (from the Q2 2019 Phishing Activity Trends Report published by the Anti-Phishing Working Group). This is because it can be easier to trick a user into taking an unintended action than it is to exploit software flaws.



Data gathered by Akamai's carrier and enterprise security research teams also shows that the life spans of domain names used for phishing are decreasing, with the median decreasing to ~1.5 hours in March of 2019. This has direct implications for protection: Defenses need to be as agile as attacks.

Summary

This is not an exhaustive list of web threats. Attackers constantly assess the viability of their exploits and innovate to maximize their return, changing the face and function of their work. There are also other kinds of malware that are primarily a distraction or nuisance, showing unwanted ads or content.

Small and midsize businesses need to be protected from web-based threats with solutions that are compatible with their unique needs and constraints. Akamai Secure Business is designed for SMBs. It protects them from the kinds of attacks described in this paper without imposing a management burden. Every device and everybody in a workplace, including guests, is automatically protected. Business managers get a simple graphical portal where they can instantly see what's happening on their network and which threats have been deterred.

Secure Business was purpose-built to help ISPs:

- Generate revenue with enterprise-grade security defenses for SMBs
- Move beyond speed and reliability and differentiate SMB services based on security
- Minimize deployment barriers, reduce costs, and simplify service delivery with a cloud-based version of Secure Business

The service can be completely customized with a brand-aligned look and feel, and the feature set and threat intelligence can also be tailored to local market requirements.

[1] https://nsba.biz/wp-content/uploads/2019/06/Technology-Survey-2019.pdf

[2] https://docs.apwg.org/reports/apwg_trends_report_q2_2019.pdf